



St. JOSEPH'S
GROUP OF INSTITUTIONS
OMR, CHENNAI - 119



Placement Empowerment Program

Cloud Computing and DevOps Centre

Set Up a Load Balancer in the Cloud Configure a load balancer to distribute traffic across multiple VMs hosting your web application.

Name: Samiya Afreen J

Department: CSE



St. JOSEPH'S
COLLEGE OF ENGINEERING



St. JOSEPH'S
INSTITUTE OF TECHNOLOGY

AUTONOMOUS INSTITUTIONS, AFFILIATED TO ANNA UNIVERSITY

Introduction

In this Proof of Concept (POC), the focus is on setting up a cloud-based Load Balancer using AWS to distribute traffic across multiple virtual machines (EC2 instances). Load Balancers play a crucial role in modern cloud architectures by ensuring high availability, fault tolerance, and scalability for web applications. This POC demonstrates the basic setup of an AWS Load Balancer, allowing traffic to be distributed between two EC2 instances running simple web servers.

Overview

The POC covers the following:

- 1. Creating EC2 Instances:** Setting up two virtual machines (WebServer1 and WebServer2) in the AWS Free Tier.
- 2. Configuring Web Servers:** Installing and configuring Apache HTTP Server on each instance to host simple HTML web pages.
- 3. Setting Up a Load Balancer:** Creating an Application Load Balancer (ALB) to distribute incoming traffic evenly between the two EC2 instances.
- 4. Testing the Load Balancer:** Verifying that the Load Balancer works by checking the DNS name and ensuring it alternates traffic between the two servers.

Objectives

1. To understand the process of creating and configuring EC2 instances in AWS.
2. To install and configure a web server (Apache HTTP Server) on Linux-based EC2 instances.
3. To set up an Application Load Balancer to distribute traffic across multiple servers.
4. To validate that the Load Balancer works as intended by testing it with unique responses from each server.
5. To build a foundational understanding of cloud-based load balancing for real-world use cases.

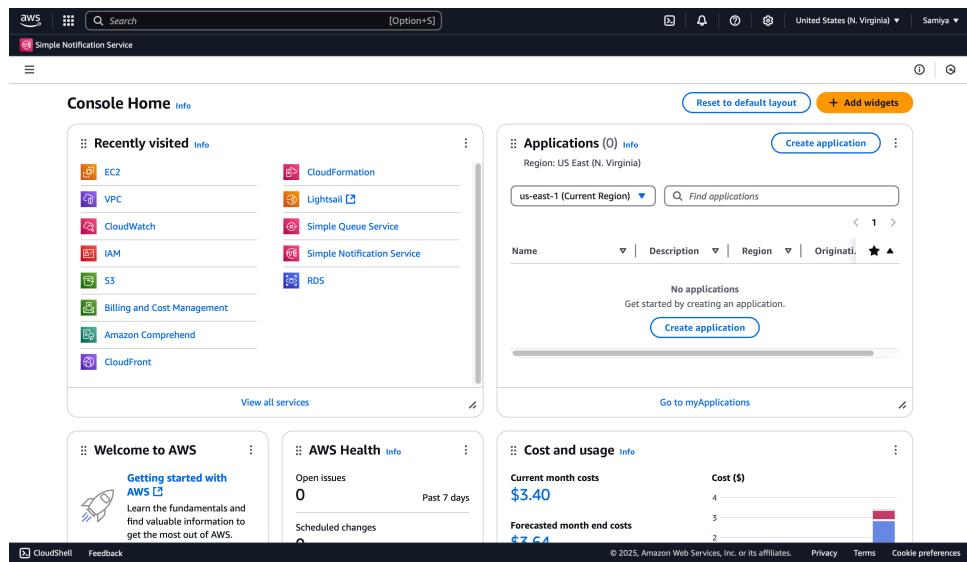
Importance

- 1. Scalability:** Demonstrates how load balancing allows scaling applications by adding or removing servers as traffic demands change.
- 2. Fault Tolerance:** Ensures that if one server goes down, the Load Balancer redirects traffic to the healthy server, improving reliability.
- 3. Cost Efficiency:** Explores how to leverage AWS Free Tier services to test and deploy cloud-based solutions with minimal cost.
- 4. Hands-On Experience:** Provides practical experience in configuring essential AWS services, an important skill for cloud computing professionals.
- 5. Foundation for Advanced Concepts:** Sets the stage for more complex setups, such as auto-scaling, secure traffic distribution, and monitoring solutions.

Step-by-Step Overview

Step 1:

1. Go to [AWS Management Console](#).
2. Enter your username and password to log in.



Step 2:

To create your instances, click **Launch Instance** and fill in the details: name the first instance "WebServer1," select **Amazon Linux 2 AMI (Free Tier eligible)** as the OS, and choose the **t2.micro** instance type. For the Key Pair, either select an existing one or create a new key pair to use for SSH access. Under **Network Settings**, click "Edit" and ensure "Allow HTTP traffic from the internet" is checked to enable web traffic. Keep the storage size at the default 8 GB, then click **Launch Instance**. Repeat the same steps for the second instance, naming it "WebServer2."

The screenshot shows the AWS EC2 Instances page. On the left, there's a navigation sidebar with sections like Dashboard, EC2 Global View, Events, Instances (with sub-options like Instances Types, Launch Templates, etc.), Images, Elastic Block Store, and Network & Security. The main area displays a table of instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
Samiya Web S...	i-0cc6b993921f9df46	Shutting-d...	t2.micro	-	View alarms +	us-east-1c
WebServer1	i-002987f235505ddf8	Running	t2.micro	Initializing	View alarms +	us-east-1c
WebServer2	i-0564ca590fe6949d4	Running	t2.micro	Initializing	View alarms +	us-east-1c
	i-040db95e48b4e4cc	Shutting-d...	t2.micro	-	View alarms +	us-east-1a

Below the table, a modal window titled "Select an instance" is open, showing a single item: "Instances".

Step 3:

Click on **WebServer1**, then click **Connect**.

Use the instructions under **SSH client** to connect to your instance via terminal.

The screenshot shows the "Connect to instance" dialog for the instance i-002987f235505ddf8 (WebServer1). The "SSH client" tab is selected. It displays the following information:

Connect to instance [Info](#)
Connect to your instance i-002987f235505ddf8 (WebServer1) using any of these options

EC2 Instance Connect **Session Manager** **SSH client** **EC2 serial console**

Instance ID: [i-002987f235505ddf8 \(WebServer1\)](#)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is web1.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
 chmod 400 "web1.pem"
4. Connect to your instance using its Public DNS:
[ssh -i "web1.pem" ec2-user@ec2-3-95-169-38.compute-1.amazonaws.com](ssh -i \)

Command copied

[ssh -i "web1.pem" ec2-user@ec2-3-95-169-38.compute-1.amazonaws.com](#)

Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

[Cancel](#)

Step 3:

Run the following commands to install and start a web server

```
PS /Users/samiyaafreen/Downloads> cd ~/Downloads  
PS /Users/samiyaafreen/Downloads> chmod 400 web1.pem  
PS /Users/samiyaafreen/Downloads> ssh -i "web1.pem" ec2-user@ec2-3-95-169-38.compute-1.amazonaws.com
```

```
[ec2-user@ip-172-31-91-192 ~]$ sudo yum update -y
```

```
[ec2-user@ip-172-31-91-192 ~]$ sudo yum install httpd -y
```

```
[ec2-user@ip-172-31-91-192 ~]$ sudo systemctl start httpd  
[ec2-user@ip-172-31-91-192 ~]$ sudo systemctl enable httpd
```

```
[ec2-user@ip-172-31-91-192 ~]$ echo 'Hello from WebServer1!' | sudo tee /var/www/html/index.html  
Hello from WebServer1!
```

```
[ec2-user@ip-172-31-91-192 ~]$ exit
```

Step 4:

Repeat these steps for **WebServer2** but change the message in the last command to:

```
PS /Users/samiyaafreen/Downloads> ssh -i "web2.pem" ec2-user@ec2-44-211-176-224.compute-1.amazonaws.com  
[ec2-user@ip-172-31-91-192 ~]$ echo 'Hello from WebServer1!' | sudo tee /var/www/html/index.html  
Hello from WebServer1!
```

Step 5:

1. In the **AWS Management Console**, go to the **EC2 Dashboard**.
2. Scroll down and click on **Target Groups** under "Load Balancing."
3. Click **Create Target Group**.

The screenshot shows the AWS EC2 Target groups page. On the left, there's a navigation sidebar with sections like Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. The main area is titled "Target groups Info" and has a search bar. It displays a message: "No target groups" and "You don't have any target groups in us-east-1". A prominent blue "Create target group" button is at the bottom. The status bar at the bottom right includes links for CloudShell, Feedback, Privacy, Terms, and Cookie preferences.

Step 6:

To create a target group, select **Instances** as the target type, name it (e.g., "MyTargetGroup"), set the **Protocol** to HTTP and **Port** to 80, and choose the same VPC as your EC2 instances (usually the default VPC). Keep the **Health Check Path** as / to verify the web server's status. Click **Next**, select both WebServer1 and WebServer2 under "Register Targets," click **Include as pending below**, and then create the target group.

This screenshot shows the "Register targets" step of the "Create target group" wizard. On the left, a sidebar shows "Step 1: Specify group details" (selected) and "Step 2: Register targets" (next). The main area is titled "Register targets" with a sub-instruction: "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." Below this is a table titled "Available instances (2/3)" showing two instances: "WebServer2" and "WebServer1", both of which are selected (indicated by checked checkboxes). A summary at the bottom says "2 selected". Below that is a section for "Ports for the selected instances" with a text input containing "80, 1-65535 (separate multiple ports with commas)". A blue "Include as pending below" button is at the bottom. At the very bottom is a "Review targets" button.

Target group name: MyTargetGroup

Protocol : Port: HTTP, Port: 80

IP address type: IPv4

VPC: Vpc-06260ec9a9ae0a454

Protocol version: HTTP

Details		Protocol : Port		Protocol version	
Target type	Instance	HTTP: 80	HTTP1	VPC	vpc-06260ec9a9ae0a454
IP address type	IPv4	Load balancer	None associated		
Total targets	2	0 Healthy	0 Unhealthy	2 Unused	0 Initial
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.					
Targets Monitoring Health checks Attributes Tags					
Registered targets (2) Info Anomaly mitigation: Not applicable Deregister Register targets					
<small>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.</small>					

Step 7:

In the EC2 Dashboard, go to **Load Balancers** under "Load Balancing" and click **Create Load Balancer**. Select **Application Load Balancer (free tier eligible)** and configure it: name it (e.g., "MyALB"), set the **Scheme** to Internet-facing, **IP Address Type** to IPv4, and ensure the listener is HTTP on port 80. Select the VPC and at least two subnets for high availability. Skip the security settings since this is HTTP. On the **Security Groups** page, choose or create a security group that allows HTTP traffic. On the **Routing** page, select

the previously created target group (e.g., "MyTargetGroup") and click **Create Load Balancer**.

The screenshot shows the AWS EC2 Load Balancers page. On the left, there is a navigation sidebar with sections like Images, Elastic Block Store, Network & Security, Load Balancing, and Auto Scaling. Under Load Balancing, the 'Load Balancers' option is selected. The main content area is titled 'Load balancers' and contains a message: 'Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.' Below this is a search bar labeled 'Filter load balancers' and a table header with columns: Name, DNS name, State, VPC ID, Availability Zones, Type, and Date. A message at the top right says 'No load balancers' and 'You don't have any load balancers in us-east-1'. At the bottom right of the table area is a blue button labeled 'Create load balancer'. The status bar at the bottom indicates '0 load balancers selected'.

The screenshot shows the AWS EC2 Load Balancers page with a success message: 'Successfully created load balancer: MyALB. It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.' The main content area is titled 'MyALB' and displays its details. The 'Details' section includes fields for Load balancer type (Application), Status (Provisioning), VPC (vpc-06260ec9a9ae0a454), Hosted zone (Z355XDOTRQ7X7K), Availability Zones (subnet-0fce724e20759806 and subnet-0e4333751635b9ed6), Load balancer IP address type (IPv4), and Date created (February 28, 2025, 01:27 (UTC+05:30)). Below this is a 'Listeners and rules' tab with 1 rule, a 'Manage rules' button, and an 'Add listener' button. The status bar at the bottom indicates '1 load balancer selected'.

Step 8:

To verify the functionality of your Load Balancer:

1. Go to the **Load Balancers** section in the AWS Management Console.
2. Select your Load Balancer and find its **DNS name** under the **Description** tab.
3. Copy the DNS name and open it in your browser.
4. Refresh the page to confirm that traffic is being alternated between the two EC2 instances. You should see the messages "**Hello from WebServer1!**" and "**Hello from WebServer2!**" displayed alternately.

This confirms that the Load Balancer is correctly distributing traffic and ensuring high availability.

Outcome

By completing this POC of setting up an Application Load Balancer in AWS, you will:

1. Launch and configure two EC2 instances with Amazon Linux 2, each hosting a simple web server with unique content.
2. Create and configure an Application Load Balancer to distribute incoming traffic between the two EC2 instances.
3. Verify the functionality of the Load Balancer by accessing the DNS name and observing traffic alternation between the two web servers.
4. Understand the importance of Load Balancers in ensuring high availability and fault tolerance for web applications.