Deploying a Highly Available Web Application and Bastion Host in AWS

Introduction

Bastion Host

- **A bastion host is a system** that is exposed to the internet.
- In terms of security, Bastion is the only server that is exposed to the internet and should be highly protective to malicious attacks.
- **A Bastion host** is also **known as a Jump Box**. It is a computer that acts like a proxy server and that allows the client machine to connect to the remote server.
- It usually resides outside the firewall.
- The Bastion server filters the incoming traffic and prevents unwanted connections entering the network thus acting as a gateway to maintain the security of bastion hosts, all unnecessary software, daemons.

High Availability

- Consider your application is running on a single EC2 instance. If the traffic to your application increases and you need further resources, we can launch multiple EC2 instances from an already running server and then **use Elastic Load Balancing** to distribute the traffic to your application among the newly-created servers.
- We can also **eliminate the Fault tolerance** in your application by placing the servers (EC2 instances) across different availability zones.
- In the event of **Failure of one Availability zone**, your application will serve or handle the **traffic from another availability zone.**
- High **Availability and fault tolerance** can be **achieved using Elastic Load balancers.**

Task Details

1. Launch a **Bastion Host instance** along with **two web application instances, two web application instances should be launched in the private subnet.**
2. Set up a **Load Balancer** and associate the two web instances to the Load Balancer.
3. **SSH** into the web servers **via** the **Bastion server**.
4. Publish a test **index.html** on both of the web servers.
5. Access the webpage using the load balancer's DNS endpoint.

6. Check the responses to see the **Load Distribution** between the 2 servers.
7. Stop or Terminate one of the web servers.
8. Check the responses to see how to Load Distribution changes

Steps to create Bastion Server

1. Make sure you are in the **N.Virginia** Region.

2. Navigate to EC2 by clicking on the menu at the top, then click on in the section.

3. Click on
4. Choose the first Amazon Machine Image (AMI)**: Amazon Linux 2 AMI (HVM), SSD Volume Type** click on the **Select** button.

5. Instance Type : **t2.micro**

6. In the **Configure Instance Details**, leave all fields with the default values and then click

   on

7. No need to change anything in this step, click on

8. **Add Tags:** Click on

- Key : ***Name***
- Value : **Bastion-Server**

- Click on :

9. **Configure Security Group:**

- Assign a security group: Choose to **Create a new security group**
- Security group name: **Bastion-SG**
- Description: **Security group for Bastion-server**
- To add **SSH:**

  - Choose Type :
  - Source : **Custom(Allow specific IP address) - 0.0.0.0/0**

10. **Review and Launch:** Review all settings and click on .

11. **Key Pair:** Create a new key pair named **bastionkey,** click on and then click on .
12. Navigate to **Instances** and wait for 1-2 minutes (until the Bastion-server's status changes from pending to running).
    - **Bastion-server Public IP:** 18.207.152.159 *(example)*

Creating a Security Group for the Load Balancer

1. Navigate to the Ec2 Dashboard, scroll down to in left menu and click on
2. **Configure the security** group as follows:

- Security group name: **LoadBalancer-SG**
- Description: **Security group for the Load balancer**
- VPC: **Leave as the default**

- Click on and add the port as follows:
- Type: **HTTP**
- Make sure, Protocol is **TCP** and **Port range** is **80**
- Source: Custom and enter **0.0.0.0/0**

3. Leave everything by default in **Outbound rules** and **Tags - optional**

4. Click on .
5. The security group for the load balancer will be created.

Steps to create Web-servers

Note: As part of AWS best practices, the **web servers should reside in private subnets**. We have created a private subnet and NAT gateway. The private subnet is attached to a route table to route traffic via NAT gateway to the internet. Please select the private subnet while launching web servers in the next section.

1. Click on .

2. Choose the first Amazon Machine Image (AMI)**: Amazon Linux 2 AMI (HVM), SSD Volume Type** click on the **Select** button.

3. Instance Type : **t2.micro**
4. In Configure Instance Details scroll down to Subnet and choose the Private subnet as shown below:

?

- Leave other fields default and click on

6. **Add Storage:** No need to change anything in this step, click on

7. **Add Tags:** Click on

- Key : ***Name***
- Value : **Web-server-1**

- Click on :

8. **Configure security group:**

- Name**: web-server-SG**
- Description: **Security group for web servers**
- On **port 22**, we choose the **Bastion-SG security group as its source to allow SSH connection to web servers from only the bastion server** by restricting the public SSH connection.Type bastion in source and select the **Bastion-SG.**
- On port 80, choose the **LoadBalancer-SG as its source** to serve the traffic coming through the load balancer. Type Load in source and select **LoadBalancer-SG.**
- To add **SSH:**
    - Choose Type **:**
    - Source : **Bastion-SG**

- To add **HTTP:**
    - Choose Type **: HTTP**
    - Source : **LoadBalancer-SG**

9. After that, click on .

10. **Key Pair :** Create a new key pair named **web-serverkey,** click on and the **key will be downloaded to your local system. A**fter that click on .
11. After a few minutes, you will see the new instance named **web-server-1** running along with the **Bastion-server** created in the earlier step.

12. Repeat the above steps from Step1 to create **Web-server-2.** You need 2 instances for this lab.
13. On **Add Tags** page, Click on **Add Tag** button, and enter below details.
    Name: **Web-server-2**

14. In, Security Group section, by selecting existing security group **web-server-SG.**
15. Click on **Review and Launch** and then Click on **Launch** and select a key pair, **web-serverkey,** and **Launch**.
16. Now you will see three servers running namely **Bastion-server**, **Web-server-1, and Web-server-2.**

- **Web-server-1 Private IP :** 172.31.101.237
- **Web-server-2 Private IP :** 172.31.16.17

Creating a load balancer

1. In the EC2 console, navigate to in the left side panel.

2. Click on at the top left to create a new load balancer for our web servers.

3. On the next screen, choose since we are testing the high availability of the web app.
4. In **configure the load balancer** enter the details below:

- Name : Enter **Web-application-LB**
- Scheme : Select **Internet-facing**
- Ip address type : Choose **ipv4**

- Listener : **Default** (HTTP:80)
- **Availability Zones**
  - VPC : Choose **Default**
  - Availability Zones : Select **All Availability Zones ,**
  - Make sure you selecte the **Public Subnet in the Availability zone us-east-1a.**

Note: we must specify the availability zones in which your load balancer needs to be enabled, making it routing the traffic only to the targets launched in those availability zones. You must include **subnets from a minimum of two Availability zones** to make our Load balancer **Highly Available.**

5. After filling in all the details above, click on .

6. On the next page, ignore the warning and click on .
7. **Configure Security Settings:**

- Select an **existing** security group and chose the security group **LoadBalancer-SG** (we created this one in the step above)

8. **Configure Routing**

- Target Group: Select New target group (default)
  - Name : Enter **web-app-TG**
  - Target Type: Select **Instance**
  - Protocol : Choose **HTTP**
  - Port : Enter **80**

  Note: The target group is used to **route requests** to one or more registered targets
- Health check:
  - Protocol : **HTTP**
  - Path : **/index.html**

  Note: The load balancer periodically sends pings, attempts connections, or sends requests to test the EC2 instances. These tests are called health checks.

  - In the upcoming steps, we will create an **index.html** in the root directory of the Apache web servers (/var/www/html) to pass this health check.

9. **Registering Targets**

- Choose the two web instances and then click on and click on .

10. Once you reviewed the settings, click on .
11. You have **successfully created** the **Application Load balancer.** Please wait for 3-4 minutes to make this **ALB into Active** state.

Connecting to web server via Bastion(Optional, if you have pasted the commands in User data)

1. SSH into the Bastion server using the Bastion PEM key: **bastionkey.pem**
2. To SSH into web servers via Bastion server, we need the web server key that we used to launch the privious web servers (web-serverkey).
3. Open the **web-serverkey** file on your local system and then **copy the text content**.
4. Navigate to the Bastion server and create a file named **web-serverkey.pem** using below command:

- **vi web-serverkey.pem**

5. Paste the content and save it by pressing **shift+colon followed by :wq!** and then enter to save your private key.
6. Make sure you have changed the **permission of the key file to 400**. You can change the permission using below command:

- **chmod 400 web-serverkey.pem**

7. Now **you can log into the web servers** using the private key copied to the bastion server with the help of below commands.

- **Note:** You **don't have a public IPs** for the web servers since we them in a private **subnet.**
- Syntax **: ssh -i web-serverkey.pem ec2-user@<**Web-server-1 private IP**>**
- Example: **ssh -i web-serverkey.pem ec2-user@172.31.101.237**

8. Now **install the apache service** using the below commands and **create a test index.html file,** which will be **used for a health check.**

- **Installing Apache:**

- **sudo su**
- **yum update -y**
- yum install httpd -y
- systemctl start httpd
- systemctl enable httpd

- **Creating the example homepage :**
  - **echo " REQUEST HANDLING BY SERVER 1" > index.html**
- **?Exit from webserver to Bastion server**
  - **?**To come out of 2nd instance, type **exit** command for coming out of root user, and **exit** command again for coming out of the instance.

9. Repeat steps 7 & 8 for web server 2 **with its respective private IP** , making sure to change the content of index.html to "**REQUEST HANDLING BY SERVER 2"**
10. To come out of 1st instance, type **exit** command for coming out of root user, and **exit** command again for coming out of the instance.

Checking the health of the load balancer

1. Navigate to under the
2. Select the target group you created and then click on to see the **Status** of the attached targets.
3. It should show **Healthy** for the Load Balancer to work properly. You may need to wait for 2-5 minutes before the load balancer's status updates to "Healthy"
4. Now navigate to and select the **load balancer** that you created earlier. Click on

   , copy the and paste it into the browser.
   - DNS URL: Web-application-LB-1853289169.us-east-1.elb.amazonaws.com
5. Refresh the browser a couple of times to see the requests being served from both servers. Seeing output similar to **REQUEST HANDLING BY SERVER 1 & REQUEST HANDLING BY SERVER 2** implies that load is shared between the two web servers via Application Load Balancer.

6. Now we have successfully created a **bastion server, two web servers and an Application Load balancer,** registered the targets to the load balancer and tested the working of Load Balancer.

Test case for High Availability

1. To check for high availability, we will make one of the instances unhealthy and test whether we get response from the other server.
2. If your instance is shown as Unhealthy then it's status would be one of the following:

- stopping
- stopped
- terminating
- Terminated

3. Navigate to the EC2 dashboard and select **Web-server-1.** Click on , select and then click on **stop.**

4. Navigate to and click on **targets.** Here you will find the status of Web-server-1 (which should be unhealthy because it is unused).

5. Navigate to **Load balancers-->Description-->DNS name.** Copy the DNS name and paste it into your browser. You should see the response "REQUEST HANDLING BY SERVER 2" FROM WEB-SERVER-2**.**
6. If you refresh a few times, you will continue to see the response only from Web-server-2
7. Repeat step 3 by stopping *Web-server-2* and starting Web-server-1 back up. This time you should see the response "REQUEST HANDLING BY SERVER" from Web-server-1.

Completion and Conclusion

1. We have launched a Bastion server and two web-servers. We were able to SSH into the servers via Bastion Server successfully.
2. We launched an Application Load Balancer and associated our web servers with the load balancer.
3. We tested the load sharing between web servers.
4. We successfully tested the high availability of the web application by making one of the web servers unhealthy.