This project demonstrates how you can create VPC that can be used server in production environment. This will help improve resilience, by deploying servers in two Available Zones, by using Auto Scaling group and an Application Load Balancer. To strengthen the security, we will deploy the servers in a private subnet.

The servers receive requests through the load balancer, and they also connect to the internet by using NAT Gateway. To improve resiliency, we will deploy NAT Gateway in both Availability zones. An application will be deployed on the ec2 instance in the private subnet. User will access the application through the load balancer. The load balancer will then forward the request to application on the ec2 instance.

The main services for this project are: EC2 instance, Auto scaling group, Load balancers and Target group.

A screenshot of a computer

Description automatically generated

Let first dive into the project.

1. We create VPC that will contain our resources like the ec2, subnet, load balancers, auto scaling group, security groups, jum server.
2. Below is the VPC configuration. We choose VPC and more to automatically create the subnets, the NAT gateway and the IG gateway.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

1. We created the Auto scaling group. But we need to first create a lunch template because we will need it in the auto scaling group creation.
   1. Below is the configuration of the lunch template.

A screenshot of a launch page

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

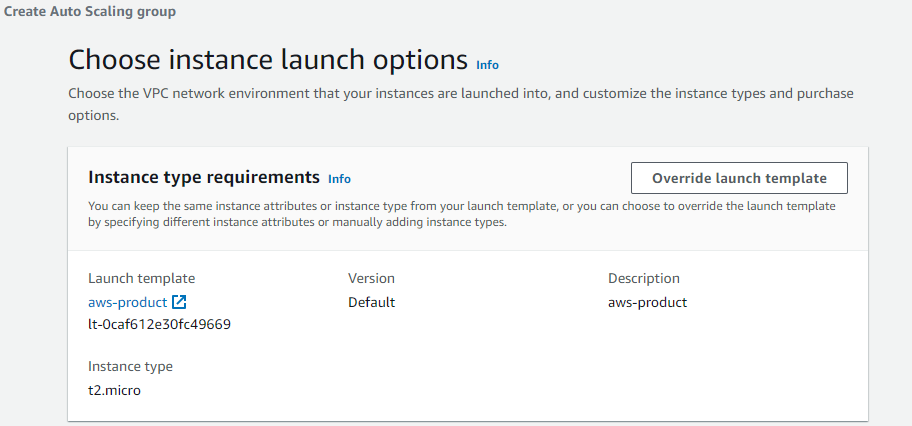
At all the above is configured correctly, you click on the create launch template button to create the launch template.

Below is the configuration of the Auto scaling group.

A screenshot of a computer

Description automatically generated

click on the next button to move on to next page.



A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a health check

Description automatically generated

click on the next button to move on to next page.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

click on the next button to move on to next page.

A screenshot of a computer

Description automatically generated

click on the next button to move on to next page.

A screenshot of a computer

Description automatically generated

click on the next button to move on to next page.

On the next page you review all the configuration you did and click on create Auto scaling group and confirm if it’s created successfully.

A screenshot of a computer

Description automatically generated

1. We now move on to create the jump server that we will use to login to the ec2-instance created by the auto scaling group. We move to the ec2 dashboard to create the instance. Below is the configuration of the jump server.

A screenshot of a computer

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

click on the launch instance button to create the jump server.

Now lets confirm if the auto scaling group has created the ec2-instances and the jump server by navigating to the dashboard.

A screenshot of a computer

Description automatically generated

1. Let’s go ahead and login to the jump server and connect to the ec2 instances in each private subnet. In my case am using mobaxterm terminal client.
   1. Below command is what I used to login to jump server
      1. ssh -i ish\_xxx.pem [ubuntu@34.79.90xxx](mailto:ubuntu@34.79.90xxx)

A screenshot of a computer screen

Description automatically generated

* 1. We proceed to login to the ec2 instance in each availability zone using the same key\_pair file.

A screenshot of a computer program

Description automatically generated

1. We now deploy the application on each ec2 instance in each availability zone and run python http server on each ec2 instance as well.
   1. Deploy the application on each instance.
      1. Create an html file using vim editor and put in your html blocks of code in it and save it.
         1. Vim index.html

A screen shot of a computer program

Description automatically generated

The above is done on each ec2 instance.

* 1. Let’s run python http server on port 5000 on each ec2 instance by using the below command.
     1. python3 -m http.server 5000

A screen shot of a computer

Description automatically generated

1. Now that the applications are deployed on each ec2 instance. We can go ahead to create the load balancer that will distribute the traffic to each ec2 instance.
   1. Before we create the load balancer, we need to create the target group that will be used by the load balancer. Below is the configuration of the target group.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

click on the next button to move on to next page.

A screenshot of a computer

Description automatically generated

click on the include as pending below button to add the select instance to the targets and click on the create target group button to create the target group.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Now that the target is created, let go ahead to create the load balancer.

1. We select the Application Load Balancer. Below is the configuration.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Make sure the target group you created is selected on the required place before you click on the create load balancer.

Now that all the configurations and the resource are created, let’s test the application we deploy on the ec2 instances. The application can be tested by copying the ARN of the load balancer and pasting it on the browser to check if you can access the application.

A screenshot of a computer

Description automatically generated

Below shows that my applications on each ec2 instance are working perfectly.

A screenshot of a computer

Description automatically generated

A screenshot of a web page

Description automatically generated

If you are able to reach this step, then well done. You have done a great job. Hurray.