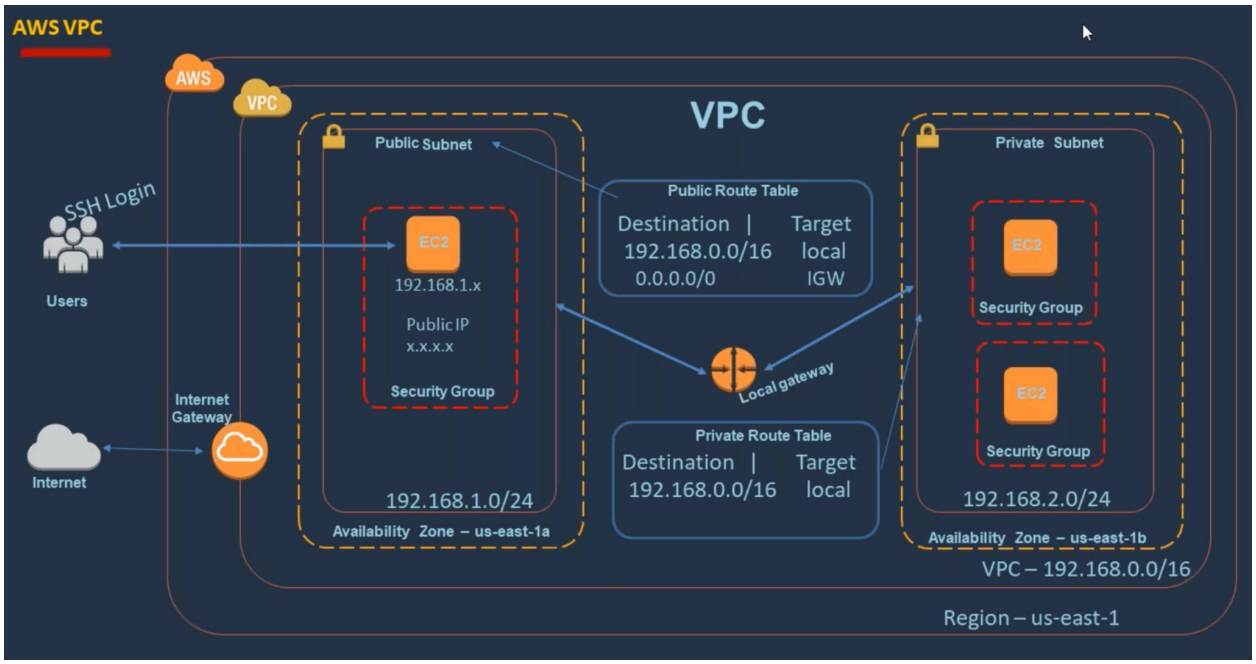
AWS VPC Setup Guide

Introduction

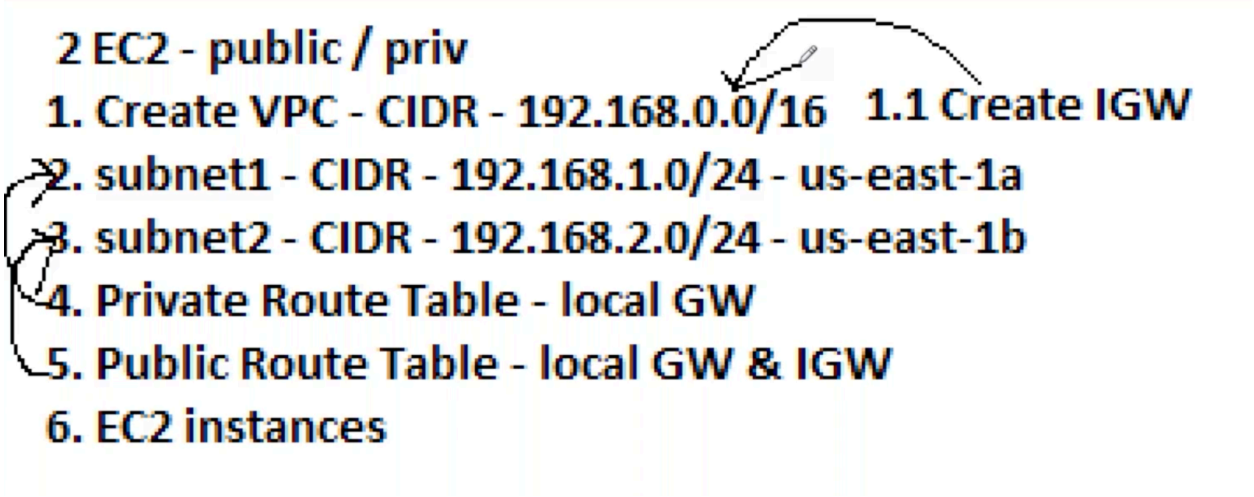
This guide outlines the process of setting up a custom Virtual Private Cloud (VPC) in Amazon Web Services (AWS). It covers creating a VPC, subnets, Internet Gateway (IGW), route tables, and NAT Gateway, as well as launching EC2 instances within the VPC.

Prerequisites

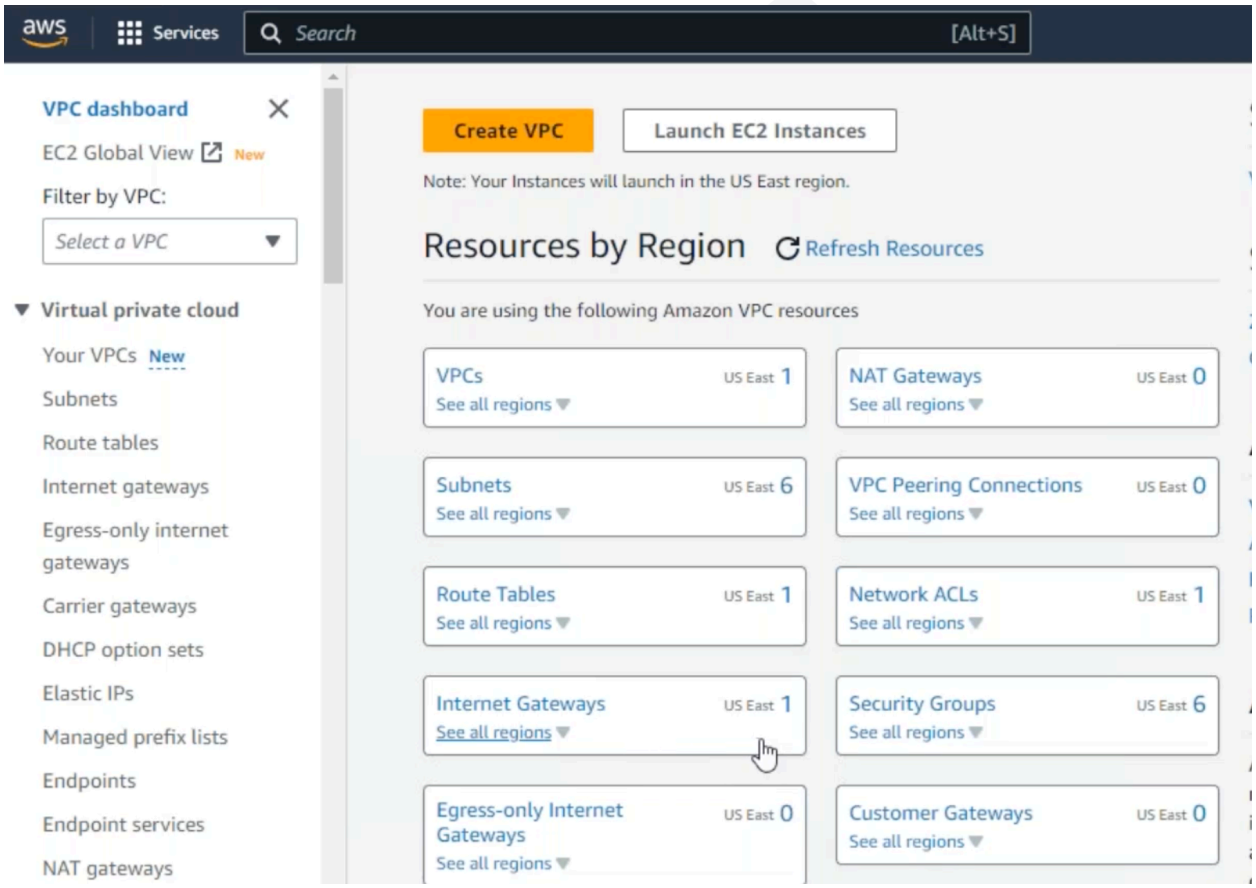
- AWS account
- Access to AWS Management Console



1 VPC can have only 1 IGW

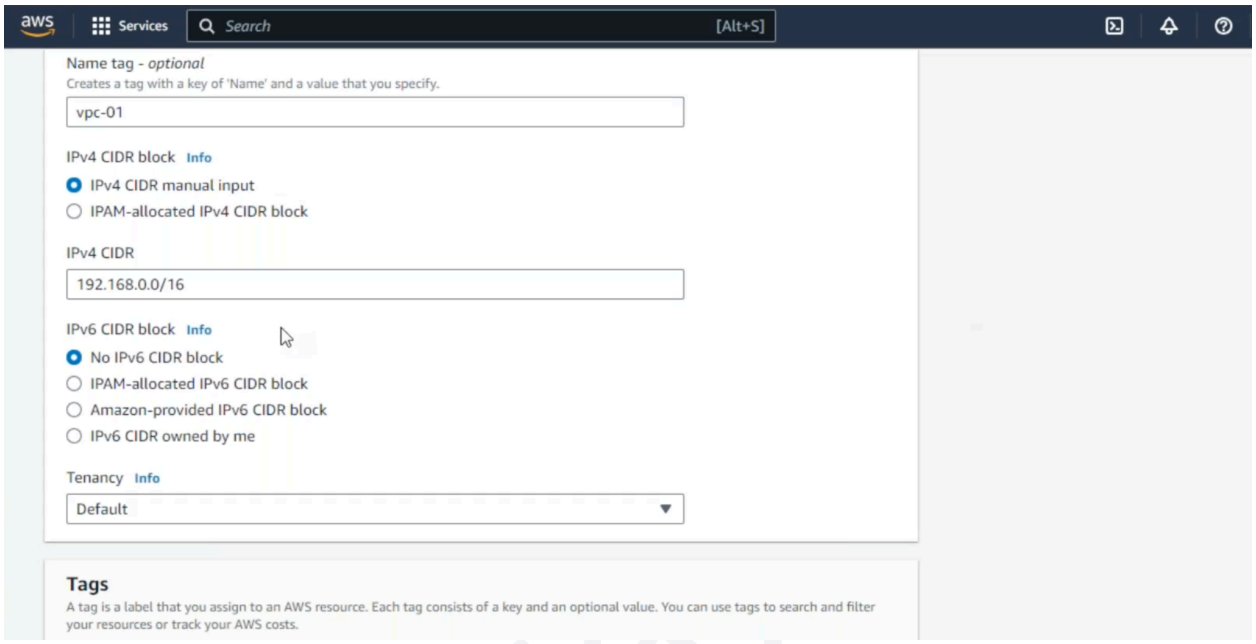


Go to AWS Console->VPC. AWS is maintaining default VPC, Subnets, Route table, IGW, Security groups. While deploying EC2, if no custom VPC is specified, this default VPC will be used by AWS.

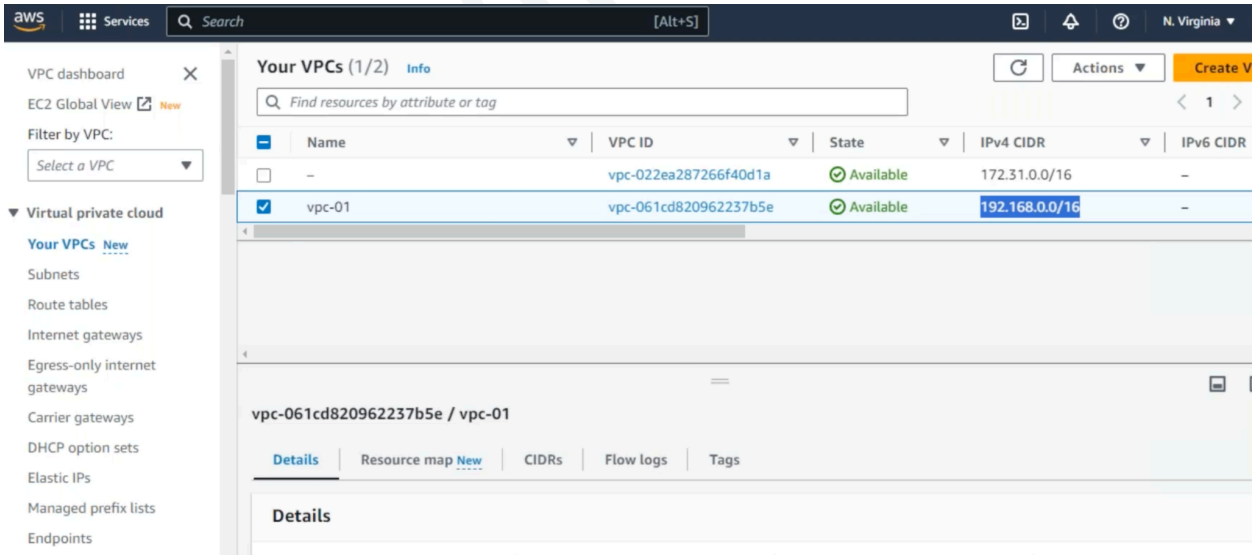


1. Create a VPC

1. Navigate to AWS Console > VPC
2. Click "Create VPC"
3. Provide a name and CIDR range (e.g., 192.168.0.0/16)
4. Choose Tenancy (Default or Dedicated)



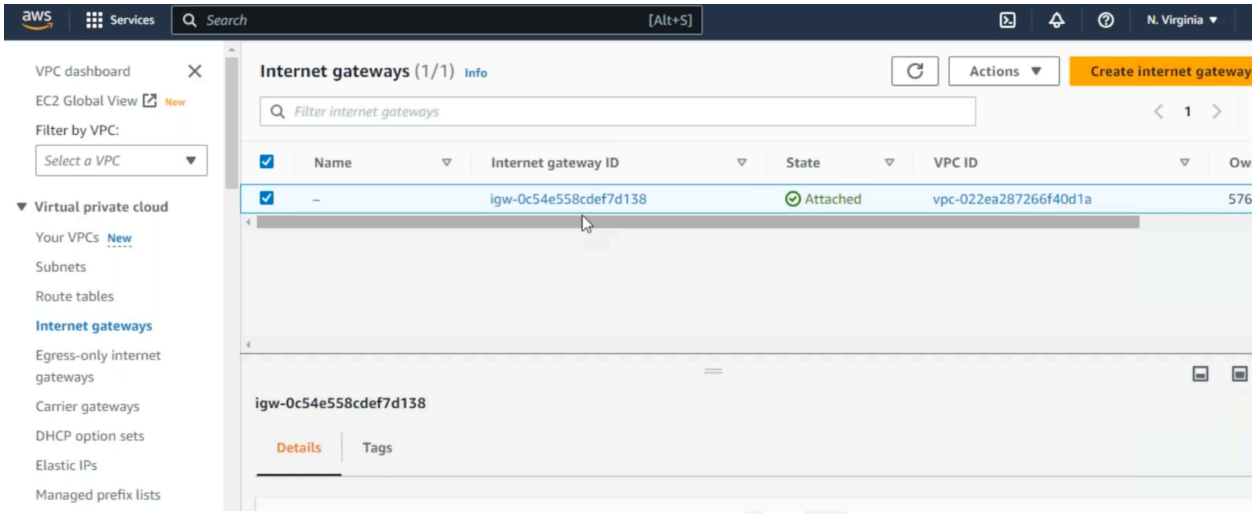
Tenancy values - Default, dedicated. Default is in this VPC Ec2 instances are shared on demand. Dedicated is used for the Dedicated hosts.

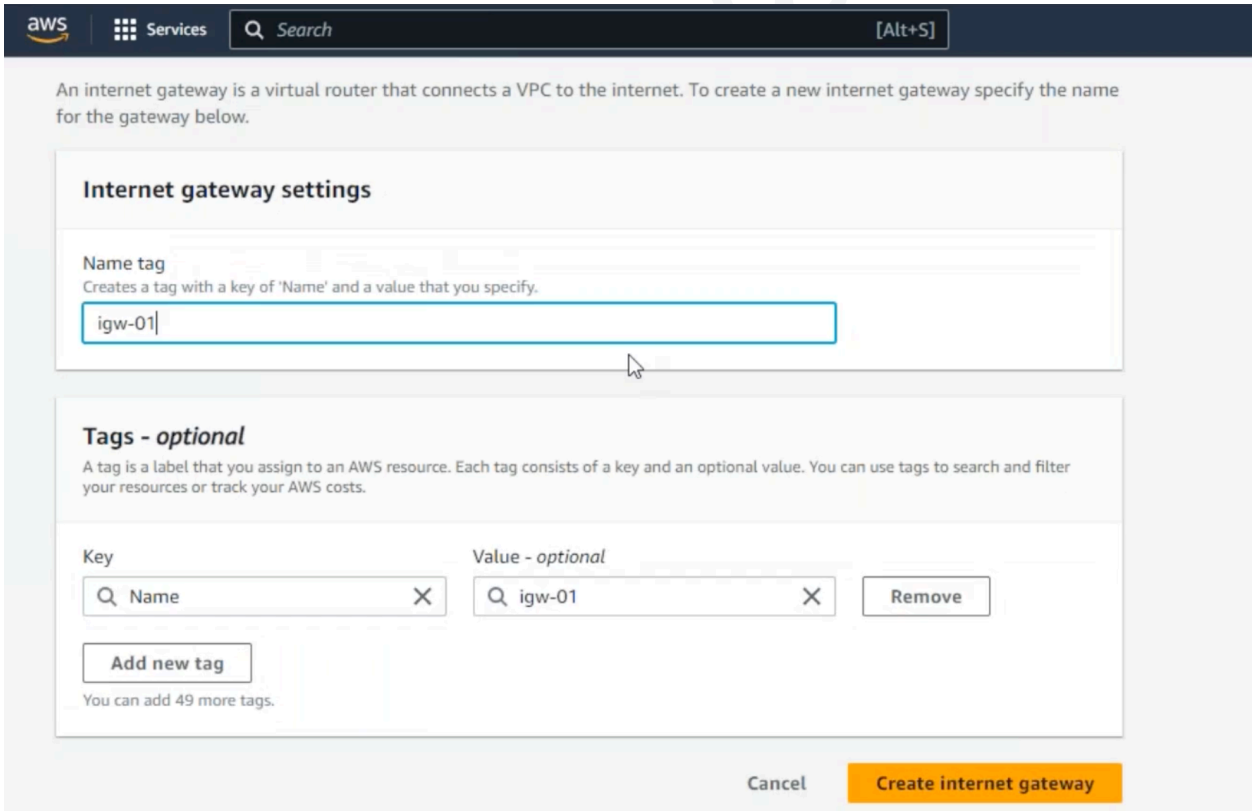


Step1 is completed. Next step is to create IGW. There is default IGW already which is used by default VPC. We need to have our own for VPC-01.

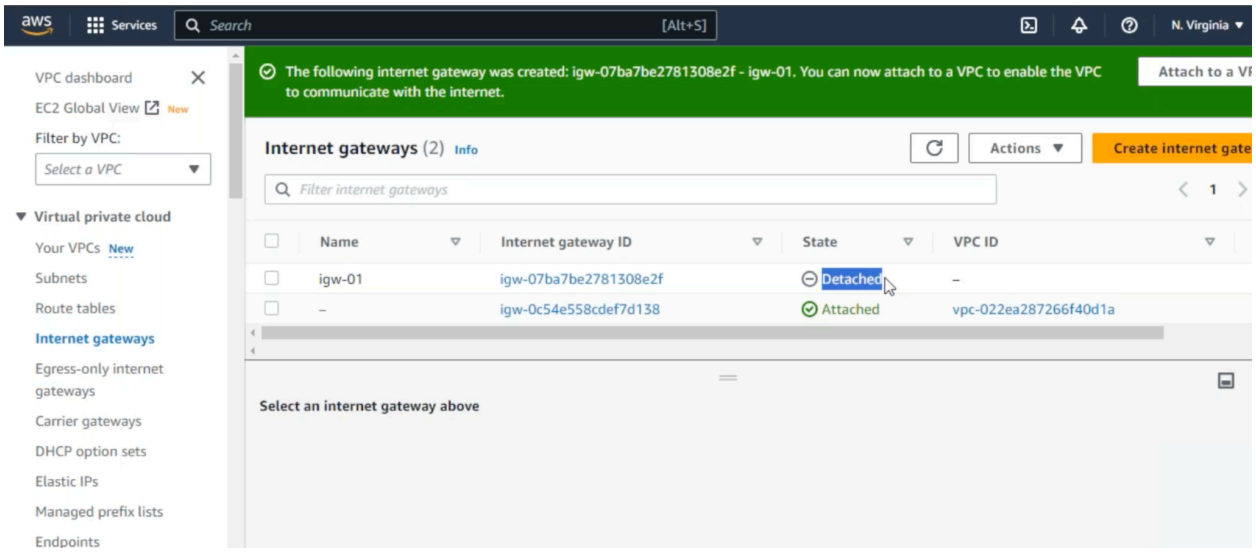
2. Create an Internet Gateway (IGW)

1. From the left menu, select "Internet Gateways"
2. Click "Create Internet Gateway"

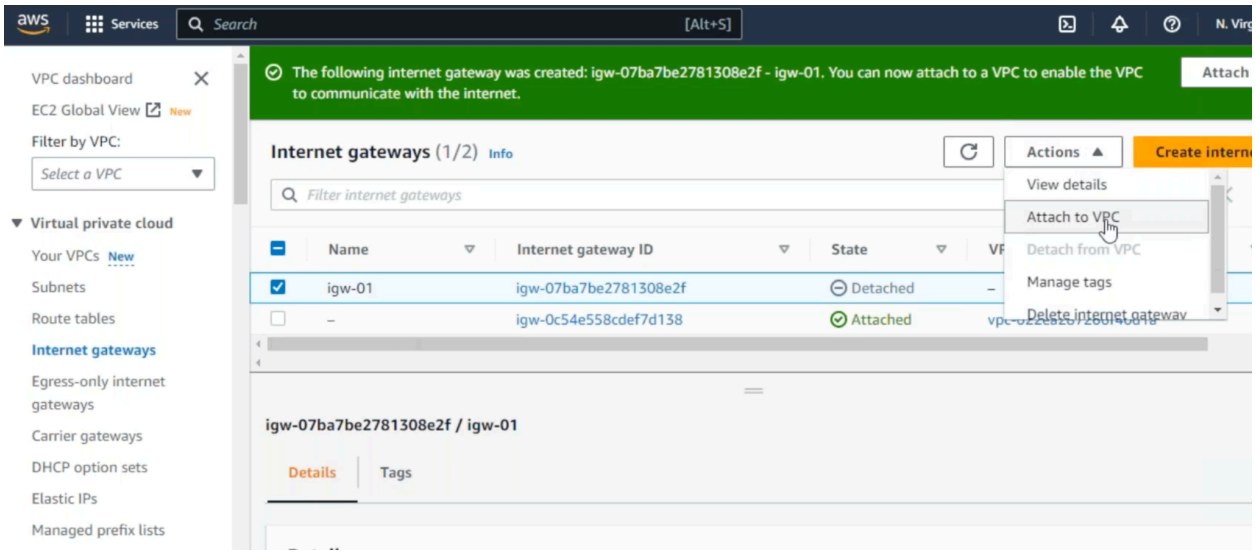




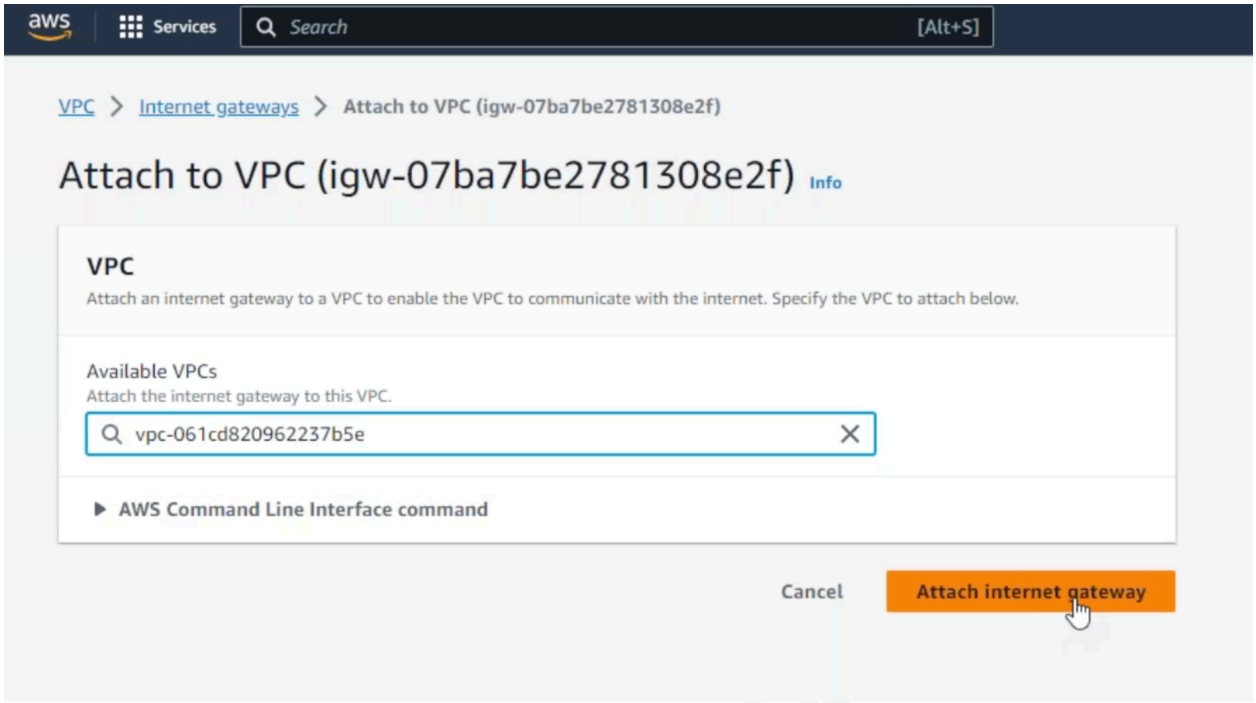
IGW is created and in Detached state.



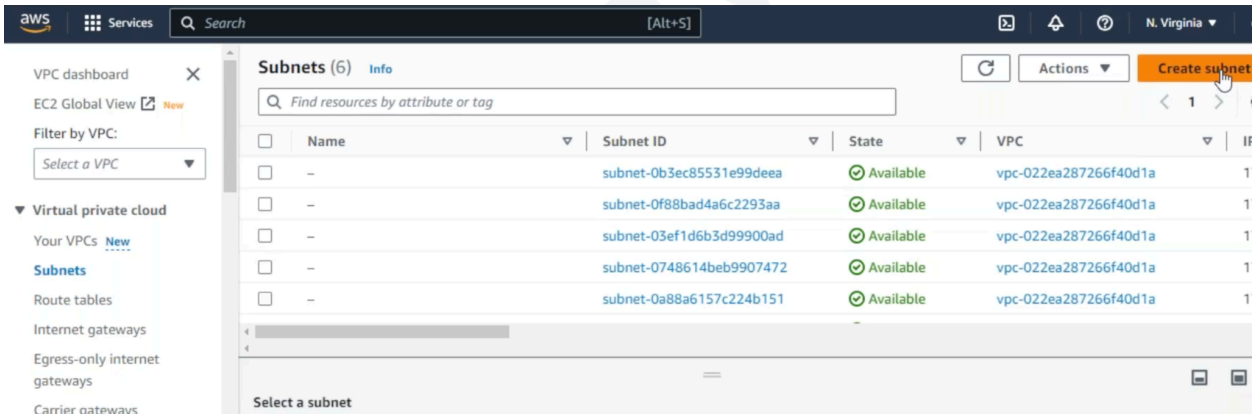
This IGW needs to be attached to a VPC. Select the IGW, click on Actions-> Attach to VPC



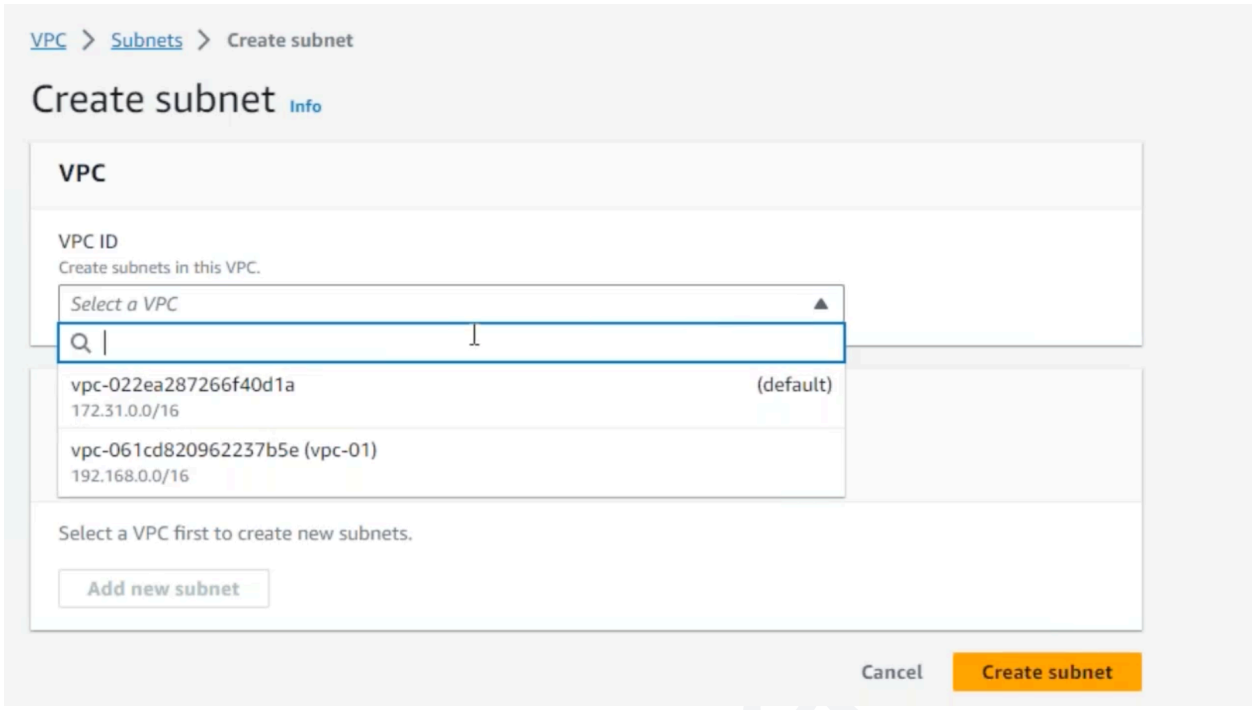
Select VPC from drop down and Attach Internet gateway. Any servers in this VPC can use this IGW.

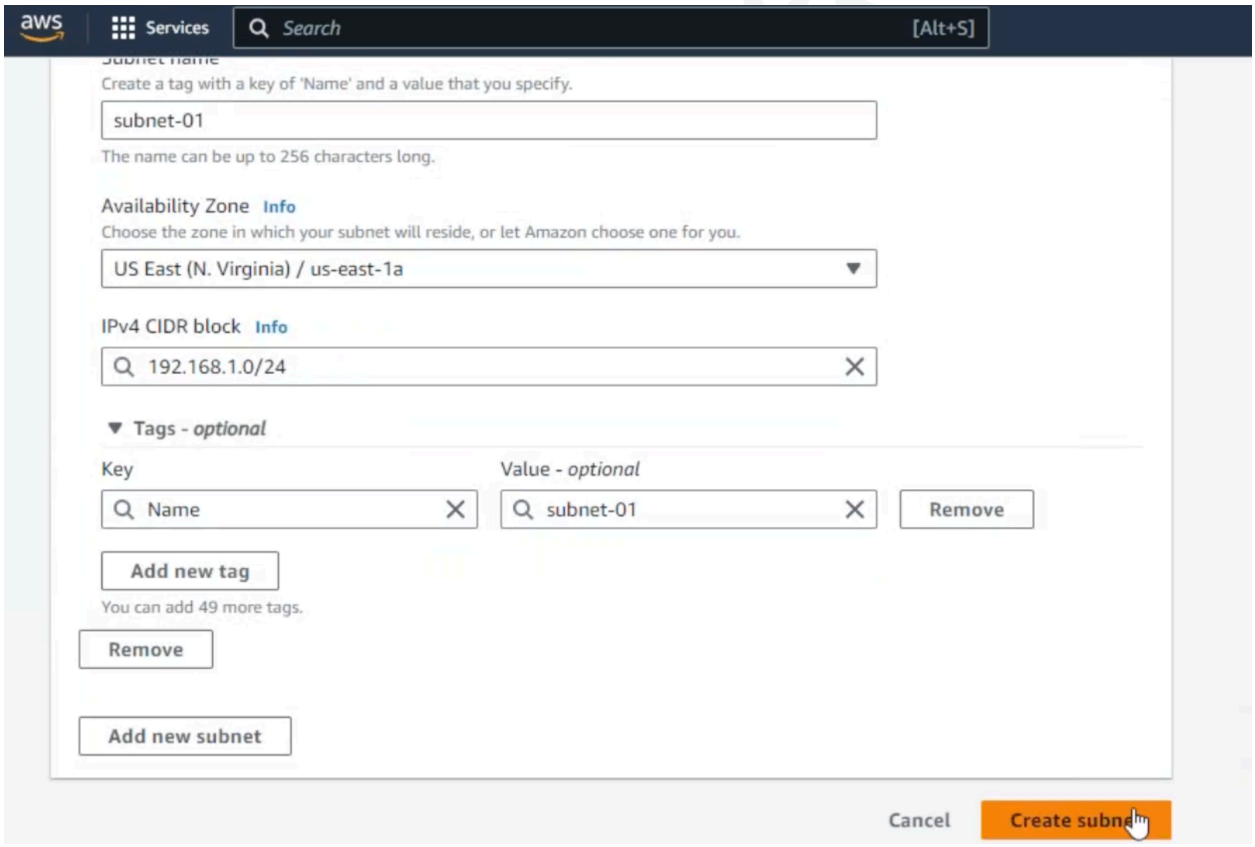


Step 1.1 is completed.
Let's create subnets. Click on Subnets from the left menu. There are default subnets listed. We need to create our own subnets. Click on the Create subnet button.

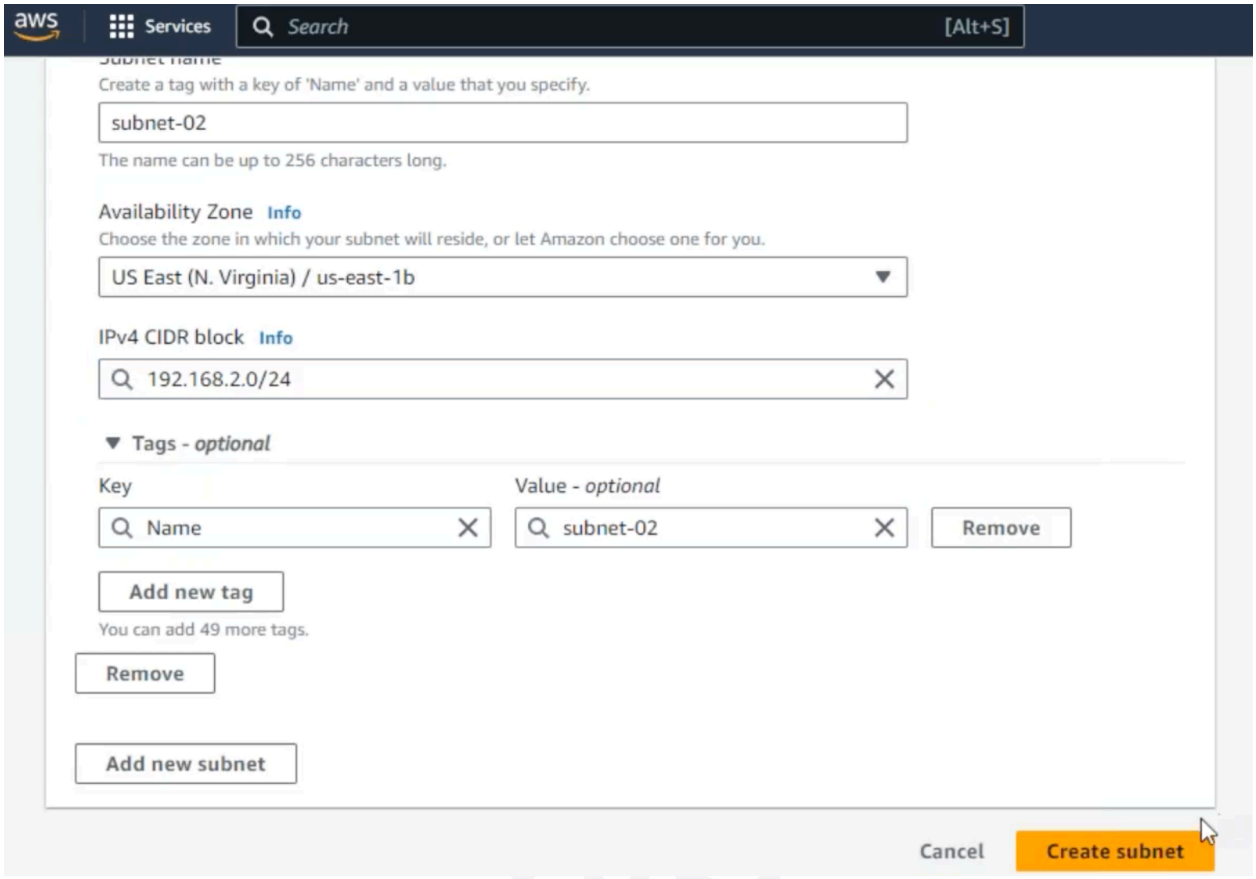


Select the VPC created in step1. Provide the AZ, CIDR details and create a subnet.

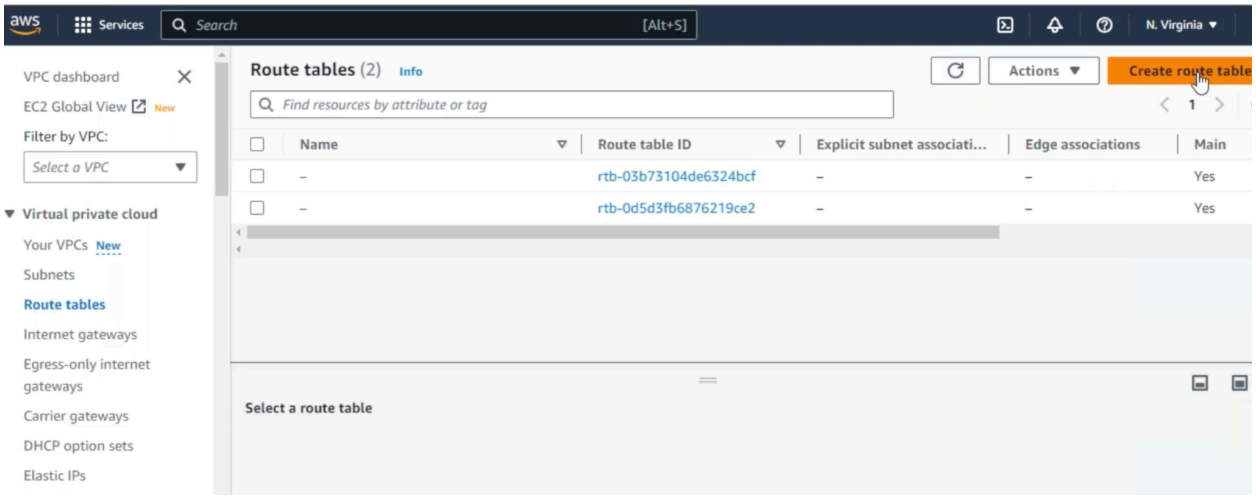


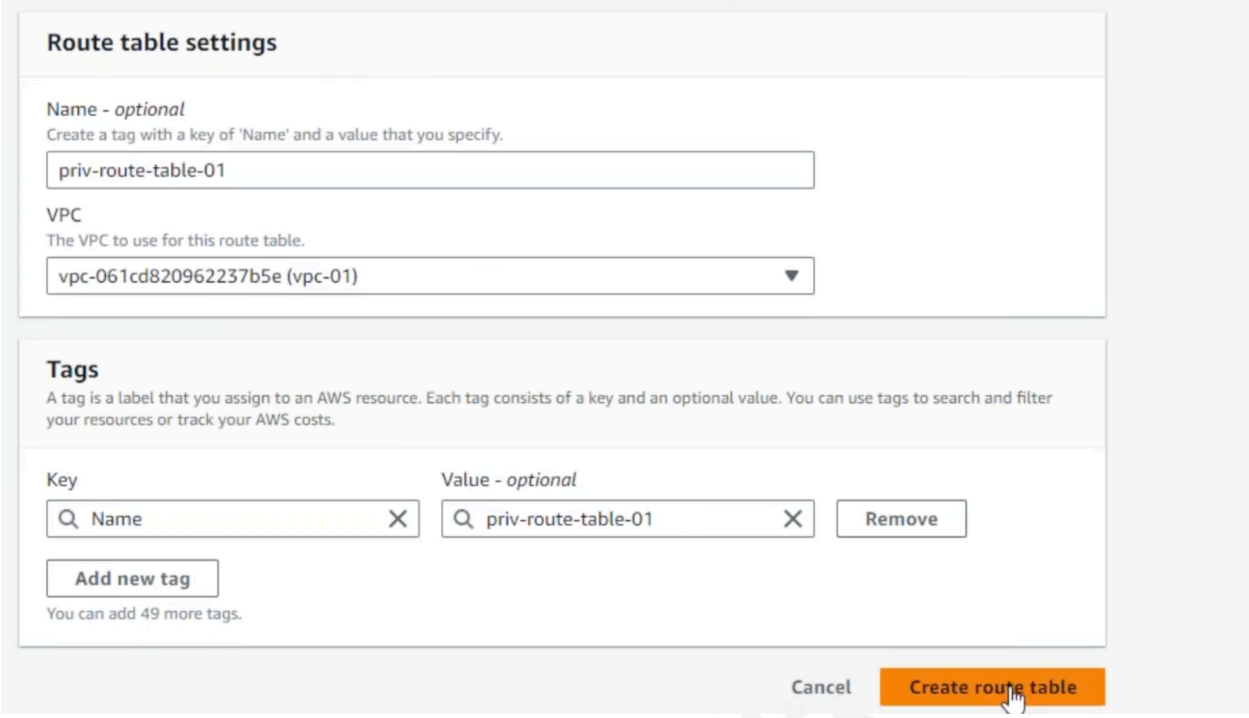


As per our plan, let's create one more subnet.

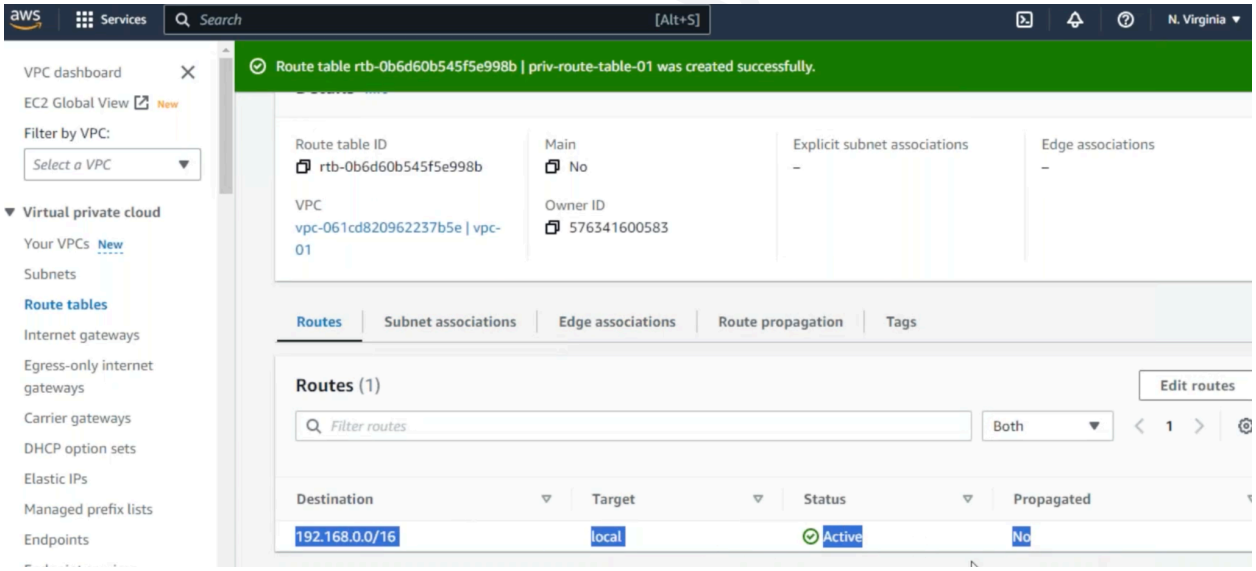


Step2 and Step3 are done.
Let us create a route table. Route table is mandatory for every subnet. If a custom route table is not created, AWS will provide a default route table.
Select route table from left menu and click on Create route table button.

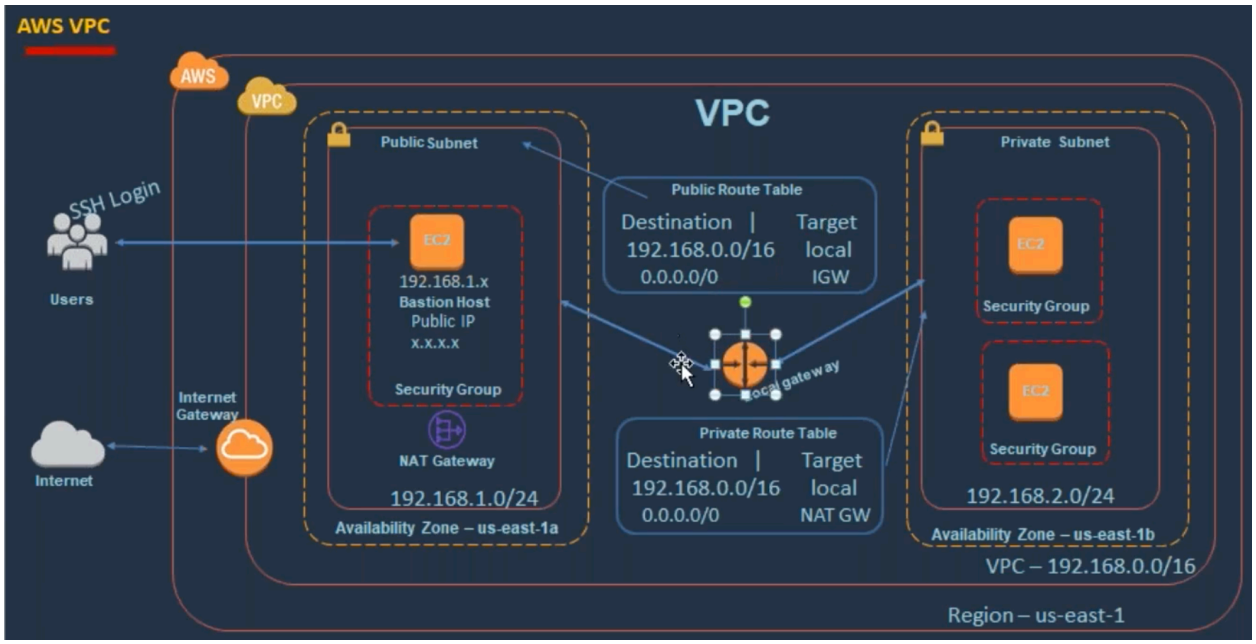




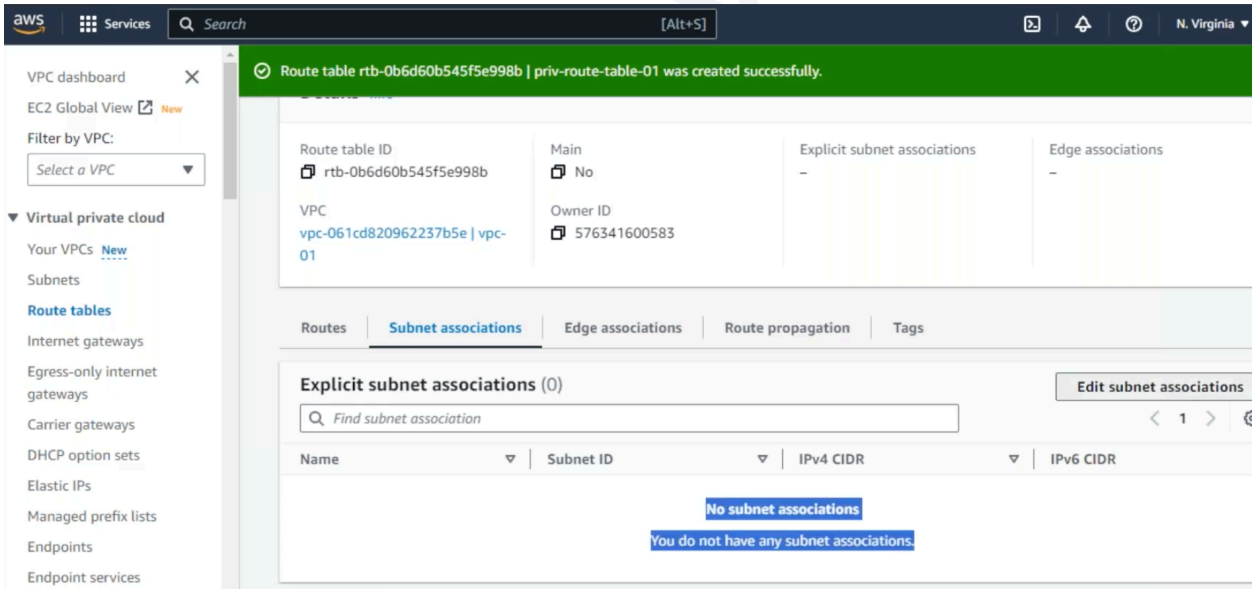
For the route table created, AWS is adding the default route. Its routing the traffic to destination via local gateway.



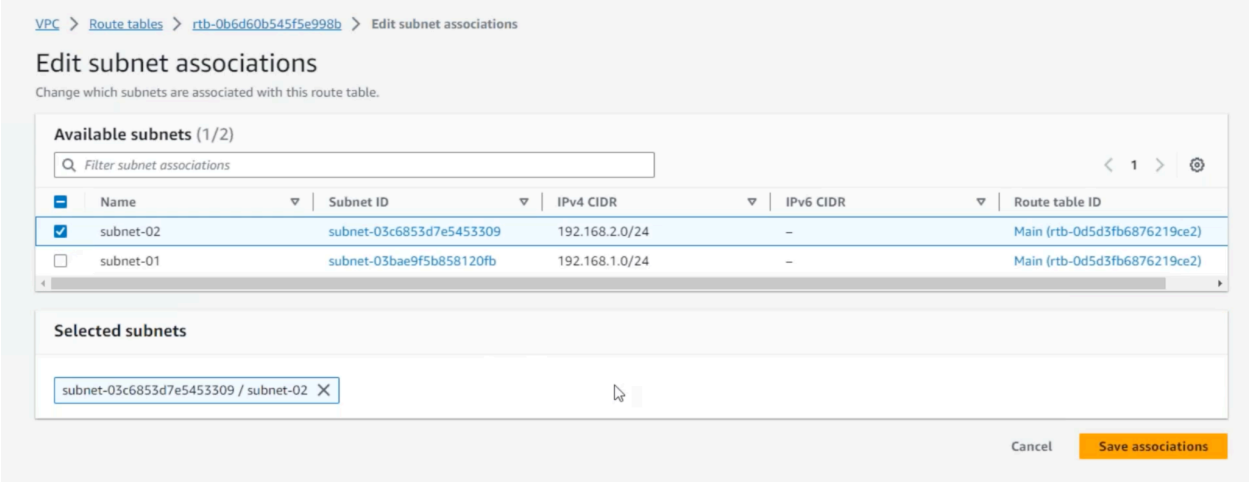
From our diagram, you can see the local gateway between the subnets. Creating local GW, adding routes everything is taken care of by AWS internally.



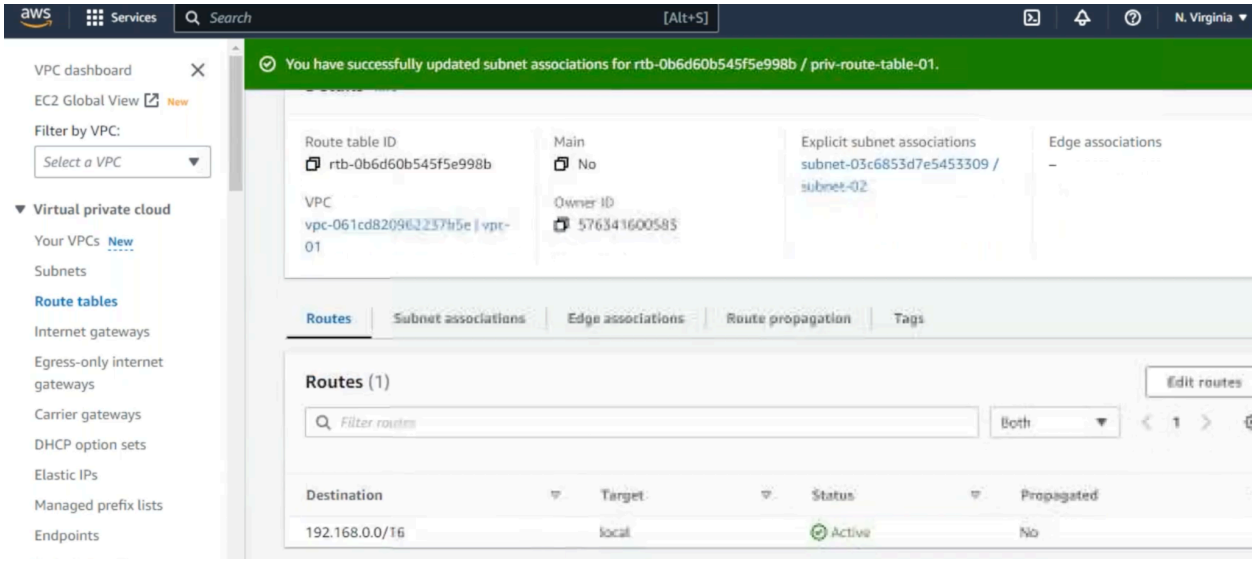
We need to associate our custom route table with a subnet. When it is created, the route table is not associated with any subnets. Select the route table, click on Subnet associations->Edit subnet associations.



As per our example, we want subnet-2 as a private subnet. Select the subnet. Current association is with the Main route table which is the default route table created by AWS. we want to change that to our custom route table. Save associations. One subnet can have 1 route table associated. Main route table will be replaced by custom route table after association.

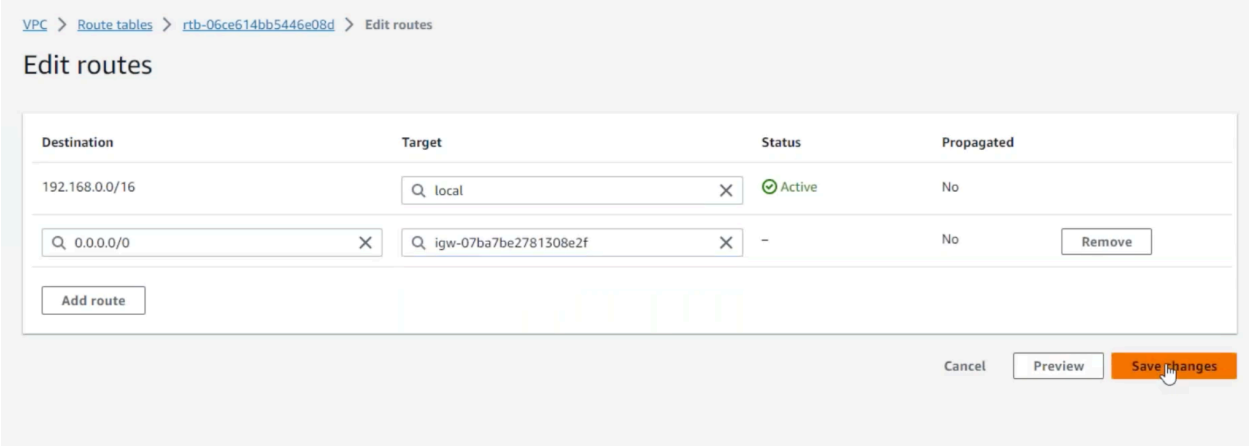


All the EC2 servers in this subnet, follow these routes to route the traffic.

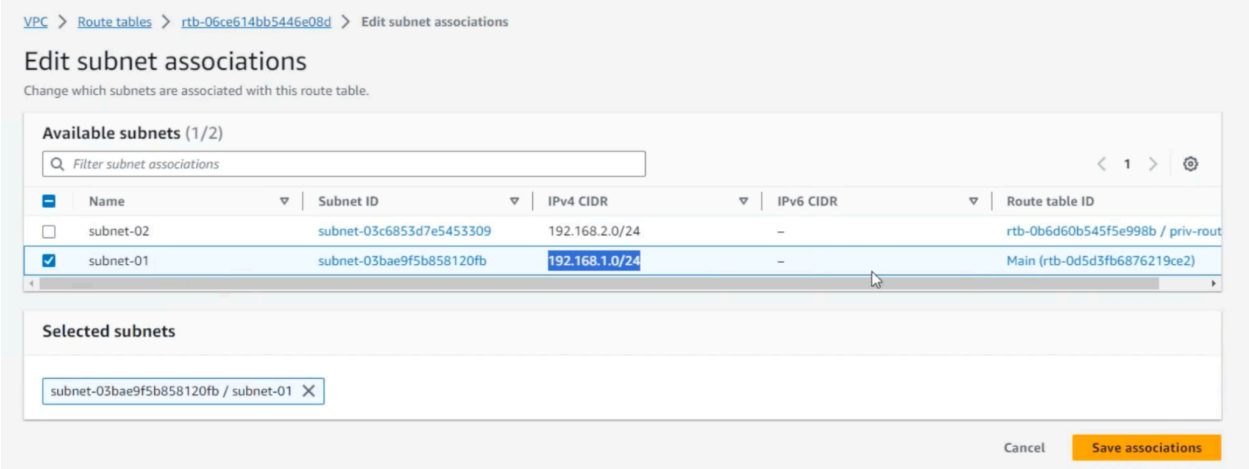


Step 4 is completed.

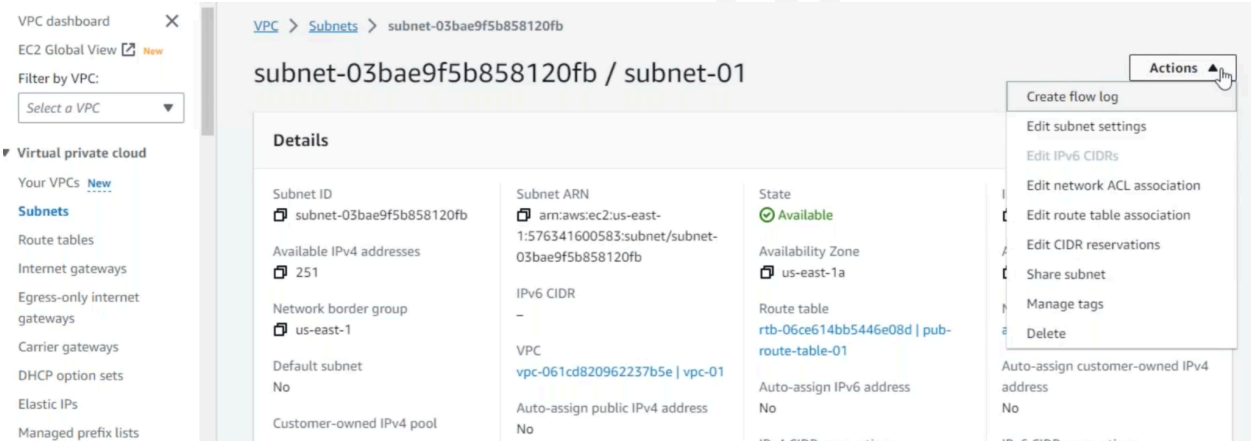
Let us create one more route table to make it public. Add IGW route to make it public route table.



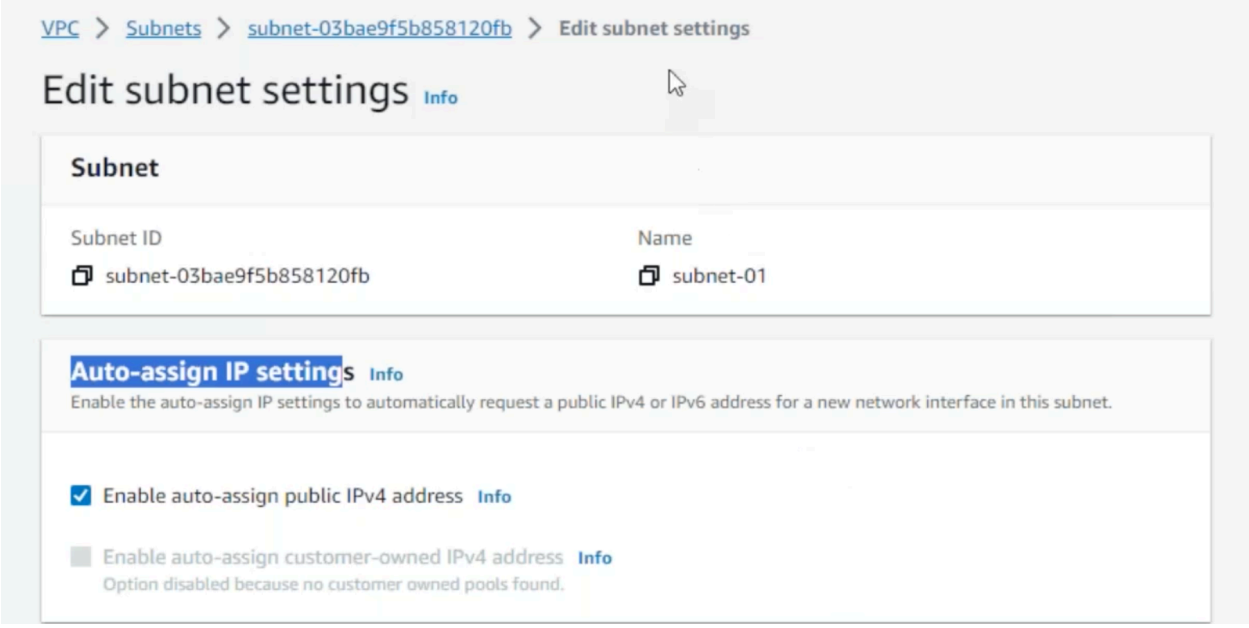
Associate subnet to use public route table.



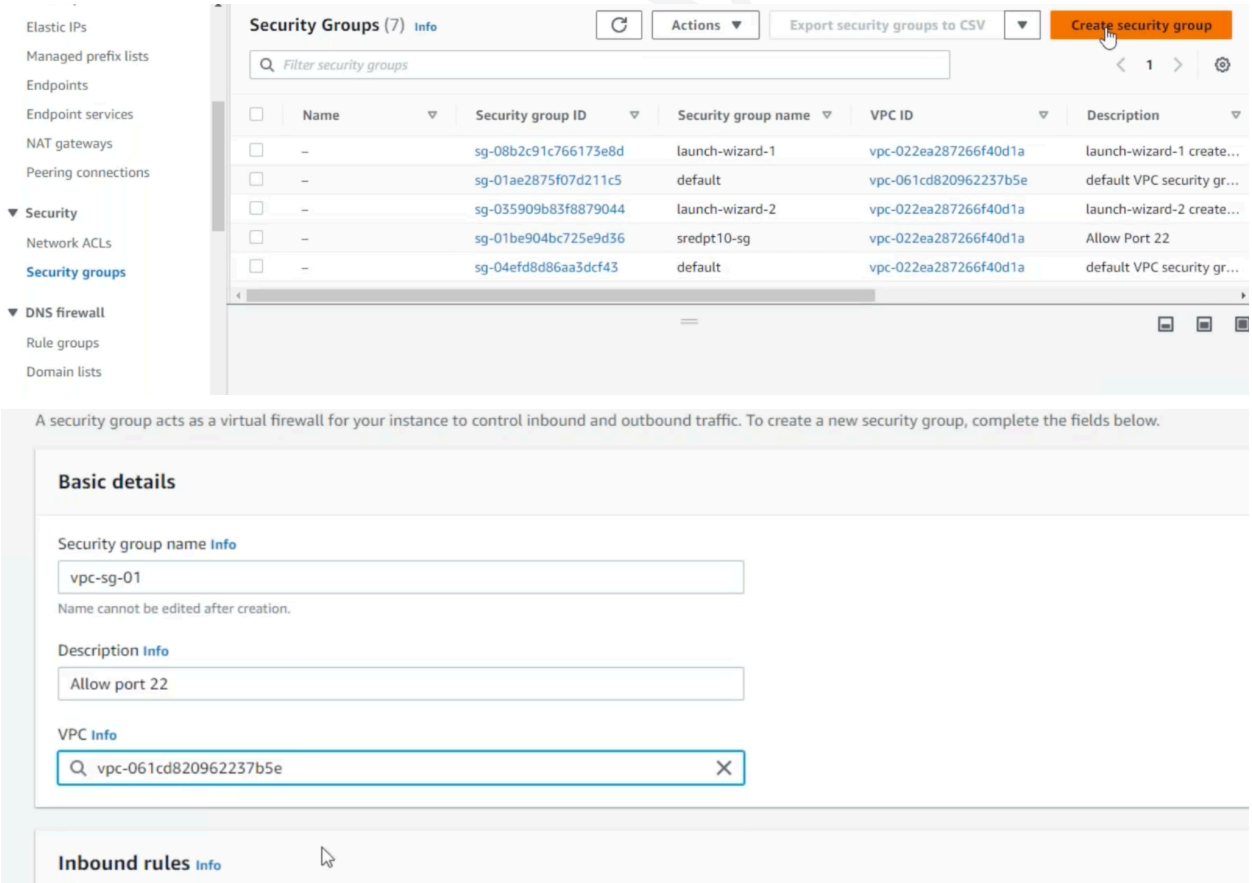
Step5 is completed.
Subnet-01 is a public subnet and Subnet-02 is a private subnet.
If the server needs to be accessible from the outside world, it should have public IP and the associated route table should have an entry with IGW. We can enable an option to automatically assign public IP for all the instances created under subnet-01. Select subnet->Actions->Edit subnet settings.

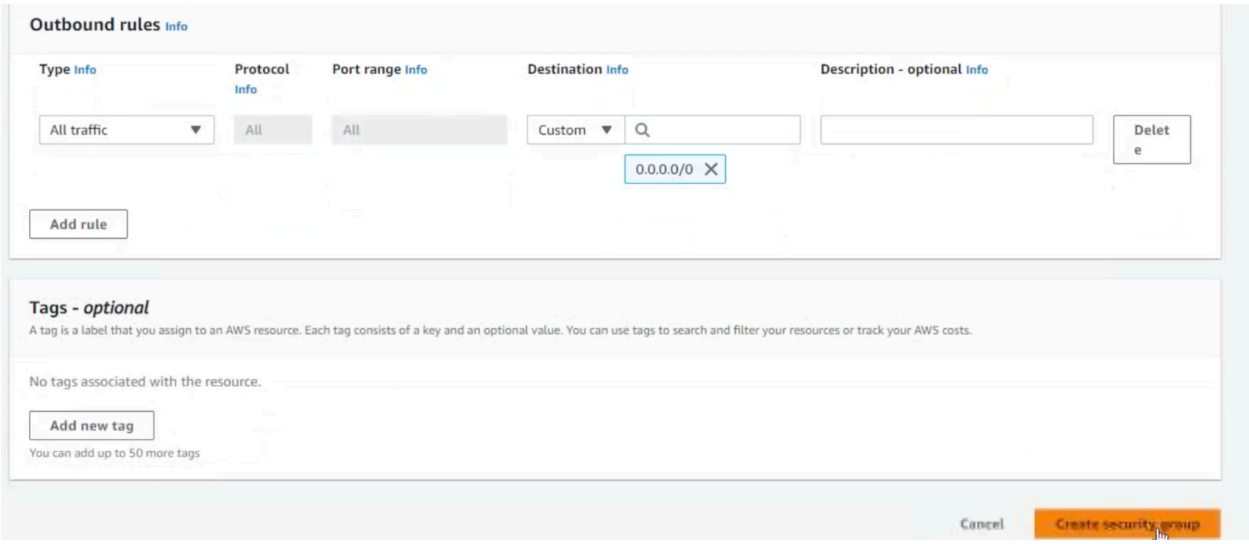


Enable auto-assign IP4 addresses.

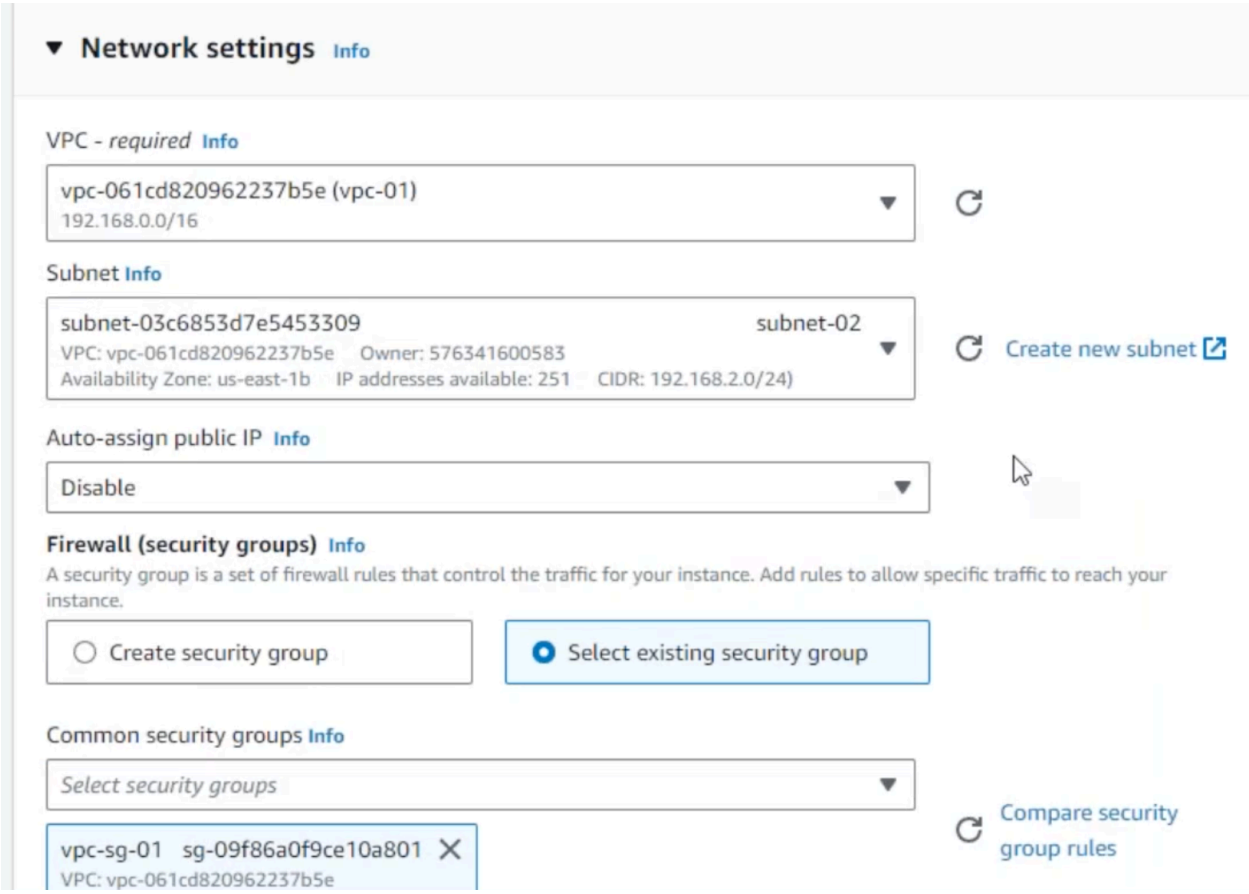


We can start creating EC2 servers in this custom VPC. Create a security group and use it while creating EC2 instances.

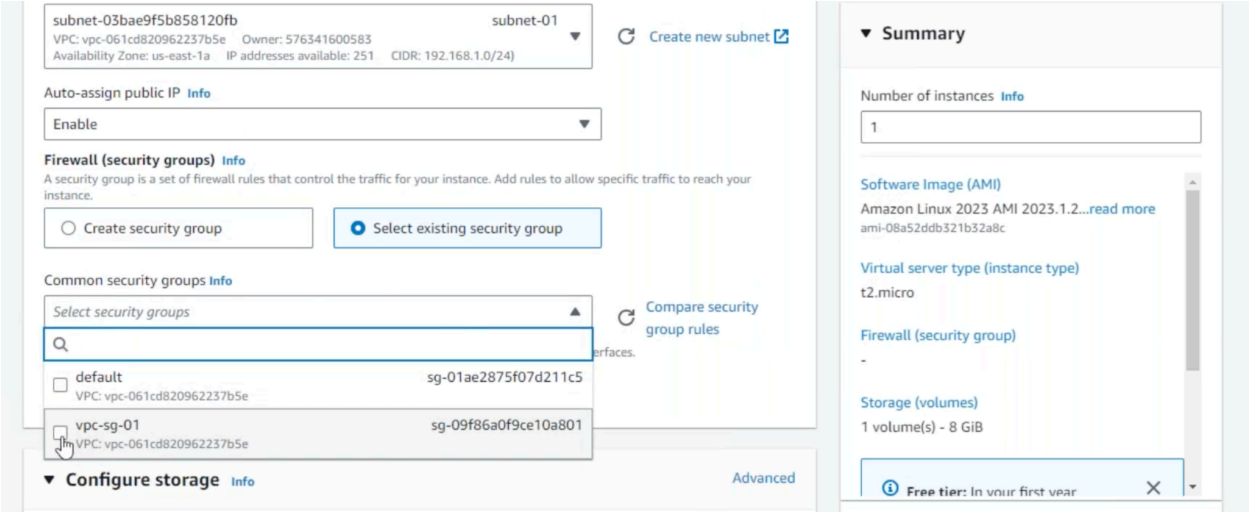




Create Ec2 server. Click Launch instance. Select name, AMI etc. In the Network settings, click Edit.



Create a public EC2 instance

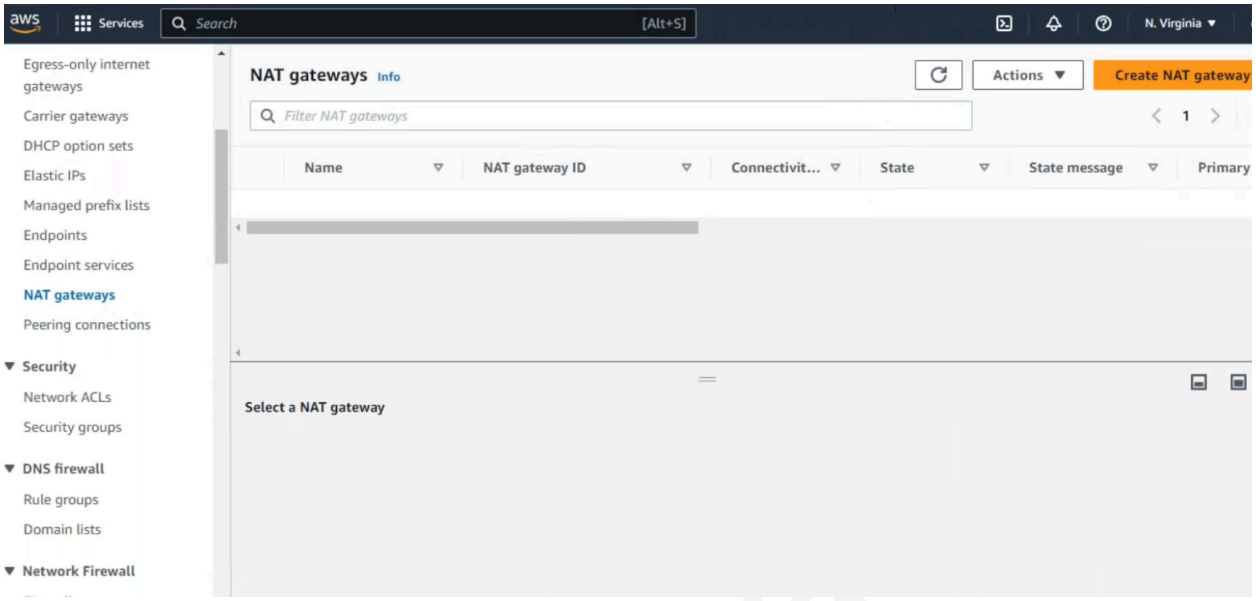


SSH to the public EC2 server. From there you can SSH to private EC2. As the local GW takes care of routing.

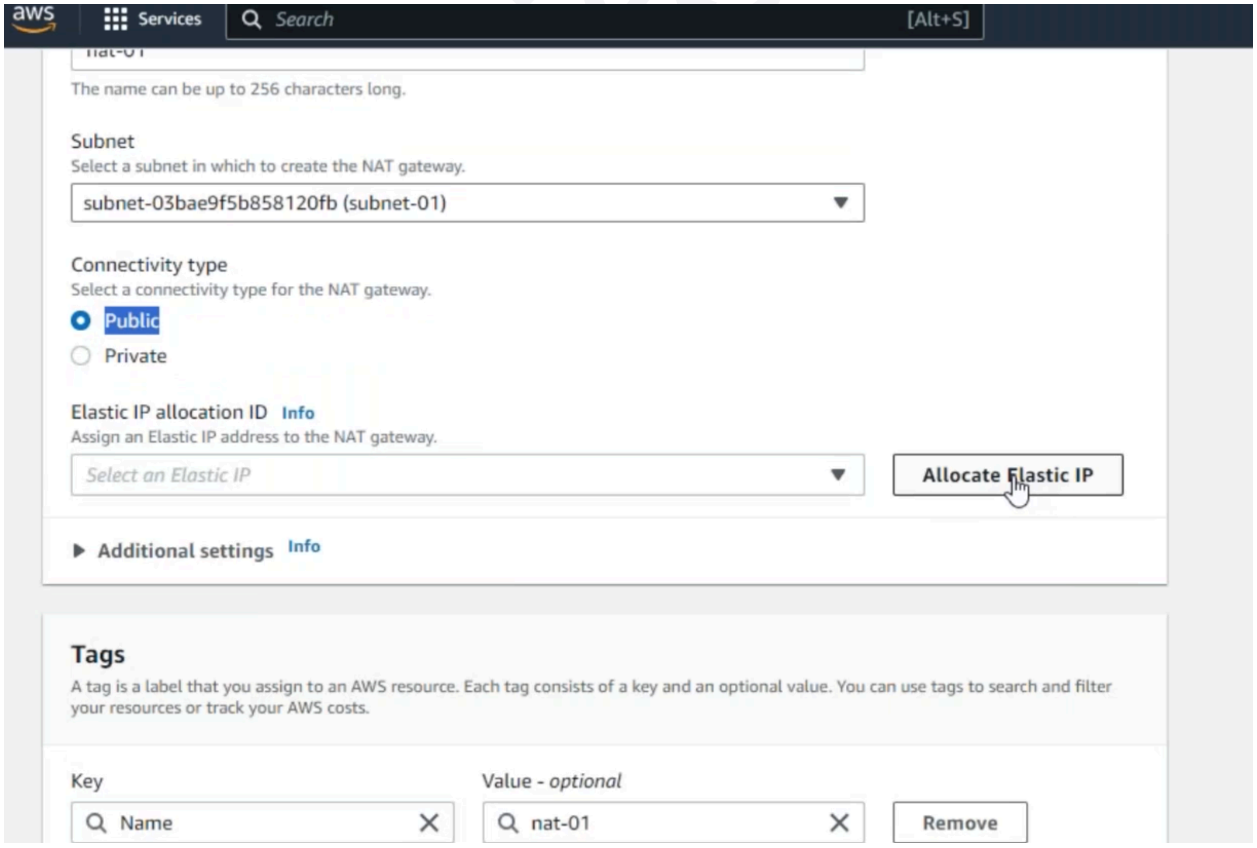
```
[ec2-user@ip-192-168-1-203 ~]$
[ec2-user@ip-192-168-1-203 ~]$
[ec2-user@ip-192-168-1-203 ~]$ ssh ec2-user@192.168.2.39
The authenticity of host '192.168.2.39 (192.168.2.39)' can't be established.
ED25519 key fingerprint is SHA256:s6g+/tra42zhAG/b4uP15x87uAEq39bqwtBbNqs28SM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.2.39' (ED25519) to the list of known hosts.
ec2-user@192.168.2.39: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-192-168-1-203 ~]$ vi sre.pem
[ec2-user@ip-192-168-1-203 ~]$ chmod 600 sre.pem
[ec2-user@ip-192-168-1-203 ~]$ ssh -i sre.pem ec2-user@192.168.2.39
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-192-168-2-39 ~]$
```

To host your application in this private server, we need to install required packages. For which we need internet access from a private server. Private servers should have outbound service to install required software. People from the internet should not be able to connect to private servers. We use NAT gateway for this use case. NAT gateway should be deployed in a public subnet. Which then can use the IGW to connect to the internet. NAT gateway can connect to private servers using local GW. Private server will send the internet request to NAT gateway, and it uses IGW to connect to the internet and reply back with the required packages to the private server. In the private route table, we need to add a route saying that, if the destination is not within VPC, use NAT gateway

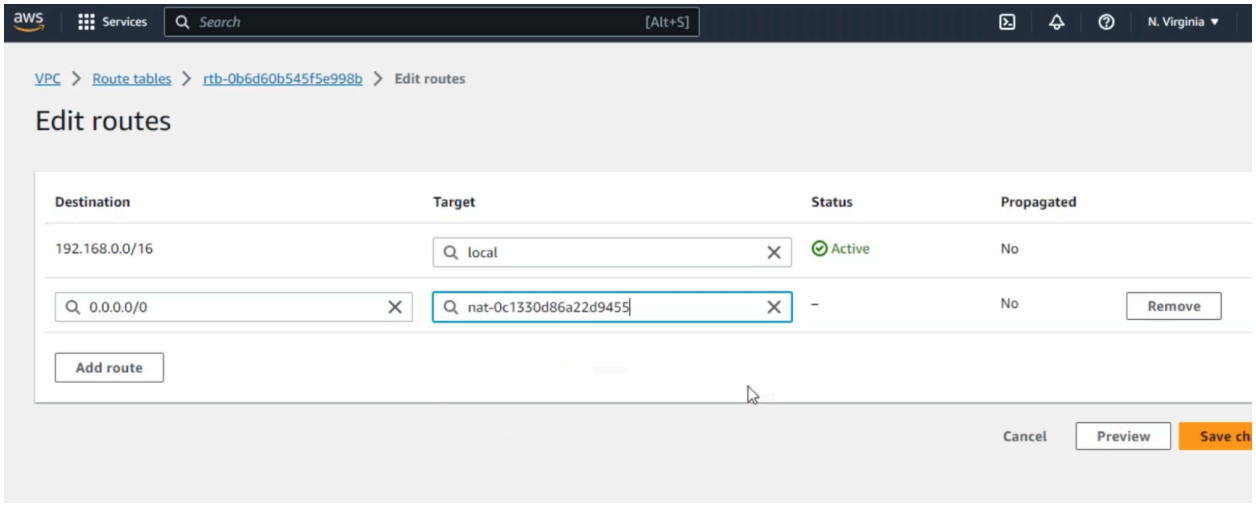
EX: Destination 0.0.0.0/0 Target NAT GW
From VPC section left menu, select NAT gateways->Create NAT gateway



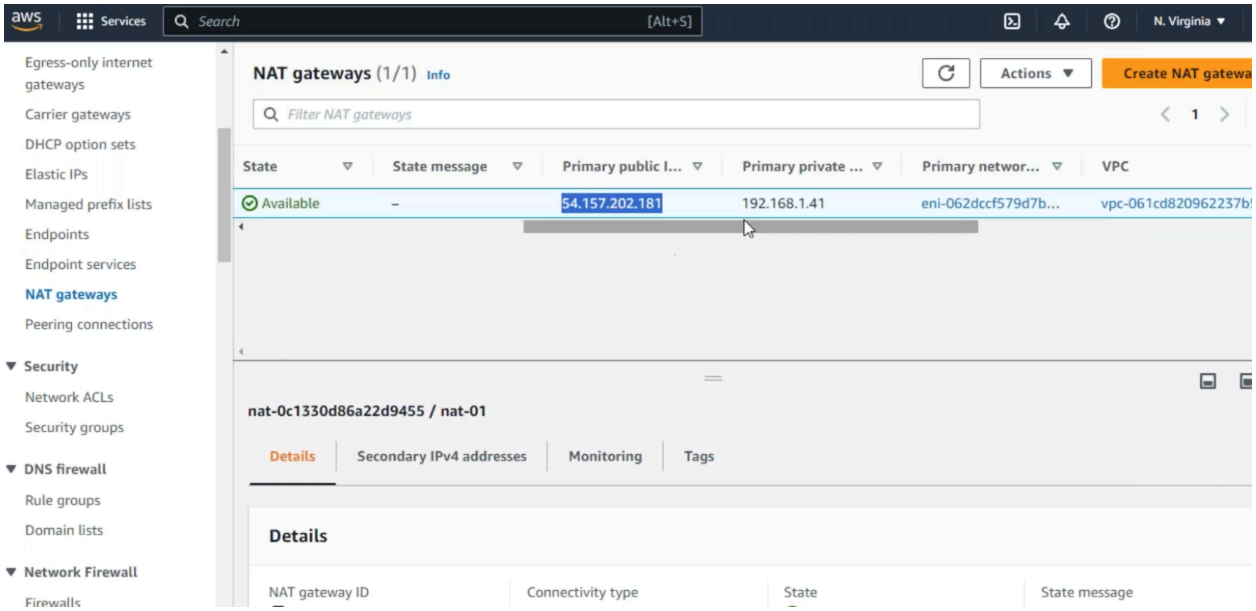
Select public subnet. NAT gateway requires public IP to connect to the internet. Select allocate elastic IP.



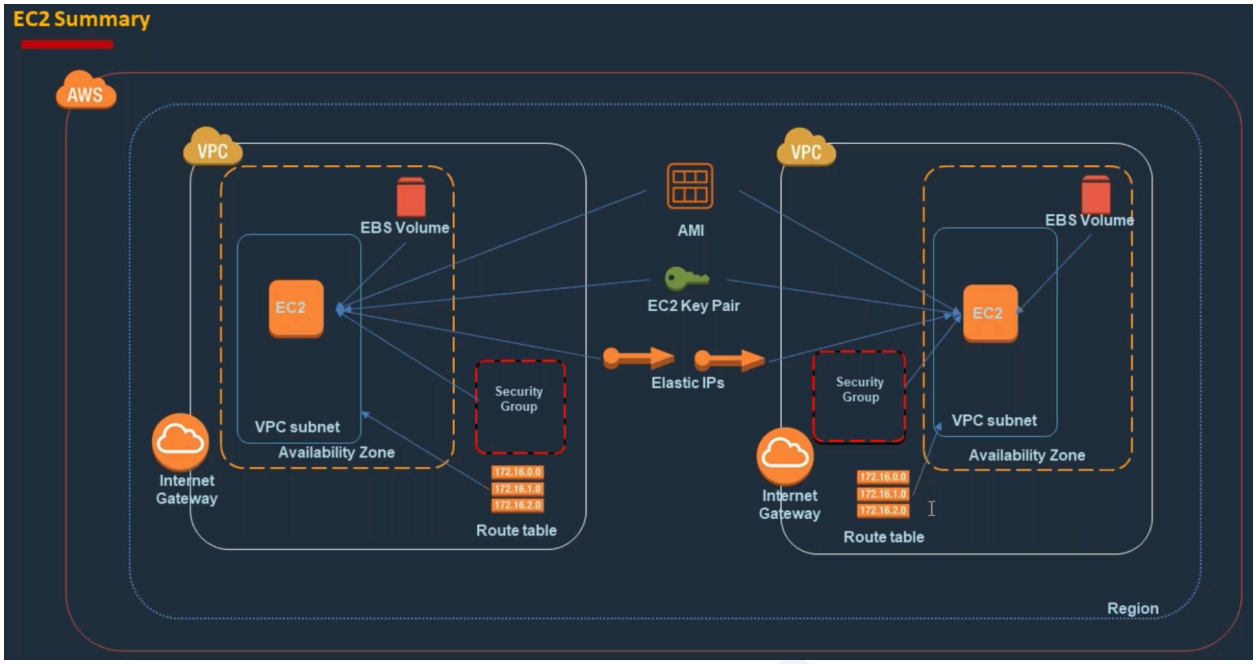
Add an entry in the private subnet route table for NAT gateway



NAT gateway is created with both public IP and private IP



Try to install packages on a private server. It should work now.



Summary

- VPCs are region-specific and can contain multiple subnets
- Subnets are associated with specific Availability Zones
- EC2 instances are deployed into specific subnets
- EBS volumes are zone-specific and must be in the same zone as their EC2 instance
- AMIs are region-specific
- Key pairs and Elastic IPs are region-level resources
- Security Groups are VPC-specific
- Route tables are VPC-specific
- One VPC can have only one IGW

Remember to consider costs associated with running AWS resources, especially for components like NAT Gateways which incur hourly charges.