Contents

[Intended Architecture 1](#_Toc38075442)

[Important Details 2](#_Toc38075443)

[Steps / Screenshots 3](#_Toc38075444)

[Difference between a Route Table, A NACL & A Security Group 36](#_Toc38075445)

# Intended Architecture

**VPC**

Private Route Table 2

SGs

IGW

Private Route Table 1

Public Route Table

**Pri-sub-2b**

SGs

App Server 2

NACL

NACL

NACL

NACL

Bastion Host

NAT GW 2

NAT GW 1

**Region-1**

**Az-b**

**Az-a**

**Pub-sub-1b**

**Pri-sub-2a**

**Pub-sub-1a**

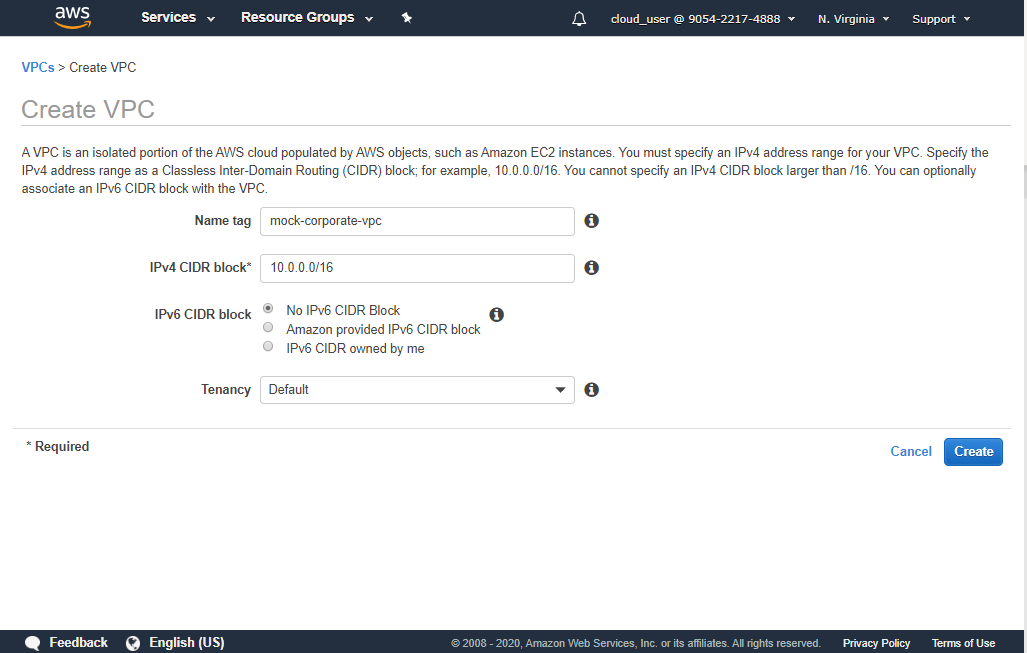
# Important Details

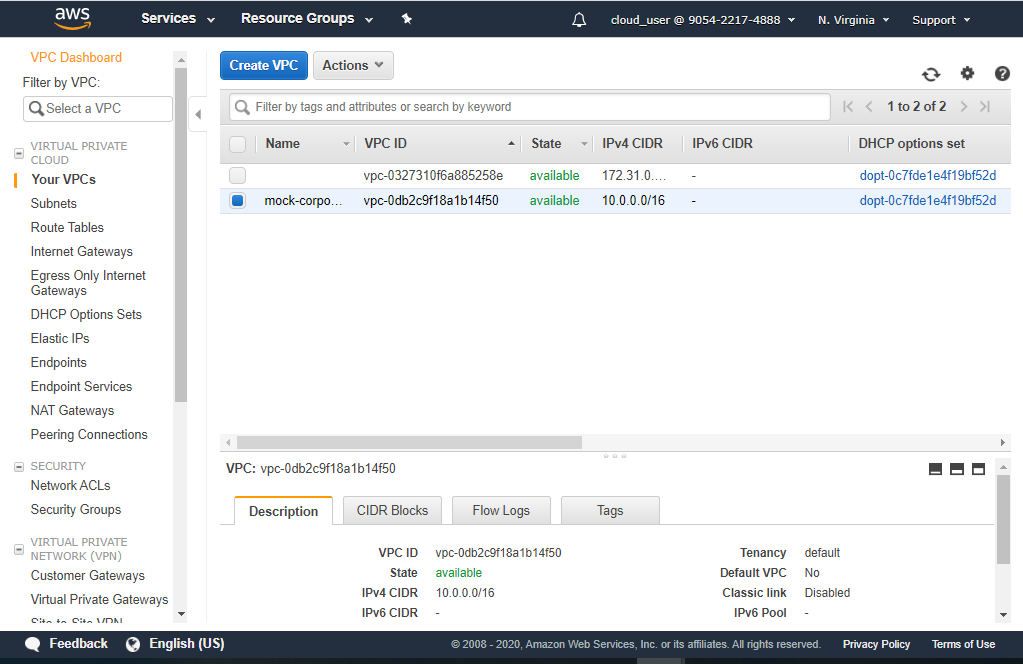
|  |  |  |
| --- | --- | --- |
| Resource Type | Resource Name | Additional Details |
| VPC | mock-corporate-vpc | IPv6 CIDR : 10.0.0.0/16 |
| Subnet | pub-sub-1a | VPC : mock-corporate-vpc  AZ : us-east-1a  IPv4 CIDR : 10.0.1.0/24 |
| Subnet | pub-sub-1b | VPC : mock-corporate-vpc  AZ : us-east-1b  IPv4 CIDR : 10.0.2.0/24 |
| Subnet | pri-sub-2a | VPC : mock-corporate-vpc  AZ : us-east-1a  IPv4 : 10.0.3.0/24 |
| Subnet | Pri-sub-2b | VPC : mock-corporate-vpc  AZ : us-east-1b  IPv4 : 10.0.4.0/24 |
| IGW | mock-corporate-igw | Associated VPC : mock-corporate-vpc |
| Route Table | mock-corporate-vpc-pub-rt | Associated VPC : mock-corporate-vpc  Associated Subnets : pub-sub-1a, pub-sub-1b |
| Route Table | mock-corporate-vpc-pri-rt-1 | Associated VPC : mock-corporate-vpc  Associated Subnets : pri-sub-2a |
| Route Table | mock-corporate-vpc-pri-rt-2 | Associated VPC : mock-corporate-vpc  Associated Subnets : pri-sub-2b |
| NAT GW | NAT GW #1 | Associated Subnet : pub-sub-1a |
| NAT GW | NAT GW #2 | Associated Subnet : pub-sub-1b |
| EC2 | Bastion-host | AMI : Amazon Linux 2 AMI  Family : t2.micro  VPC : mock-corporate-vpc  Subnet : pub-subnet-1a  Auto assign Public IP : Enable |
| EC2 | App-server | AMI : Amazon Linux 2 AMIFamily : t2.micro  VPC : mock-corporate-vpc  Subnet : pri-subnet-2b  Auto assign Public IP : Disable |

# 

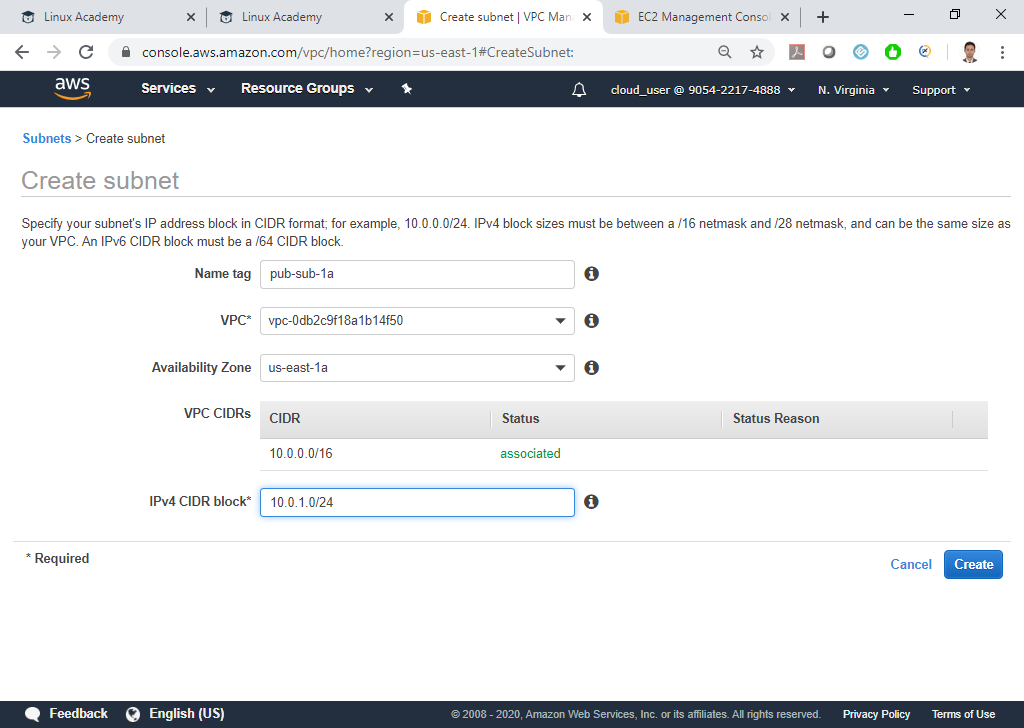
# Steps / Screenshots

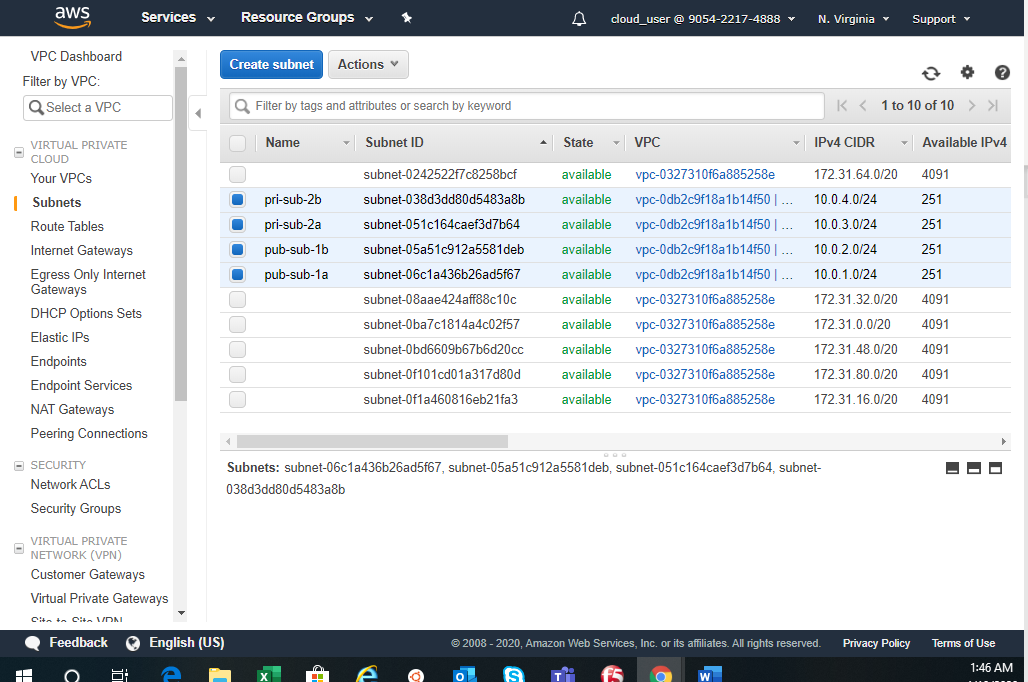
1. Create a VPC



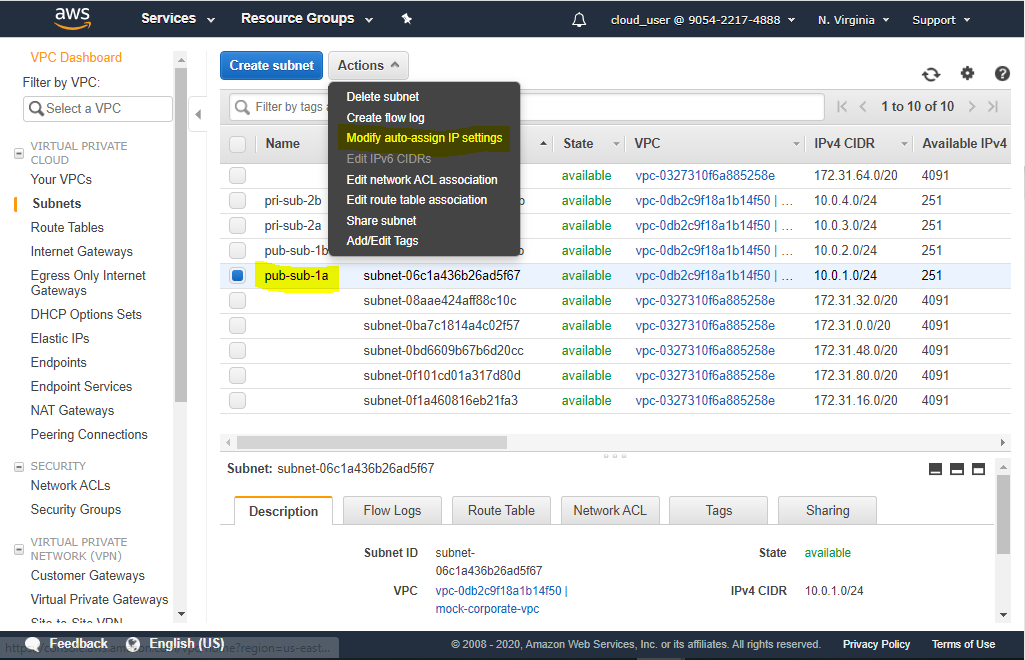


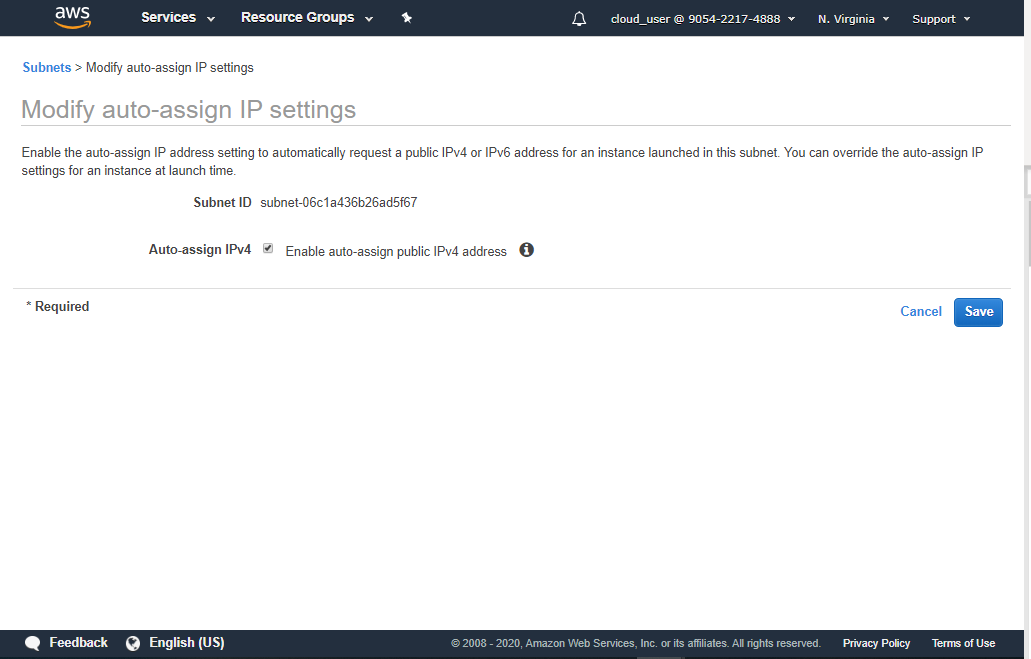
1. Create 4 subnets



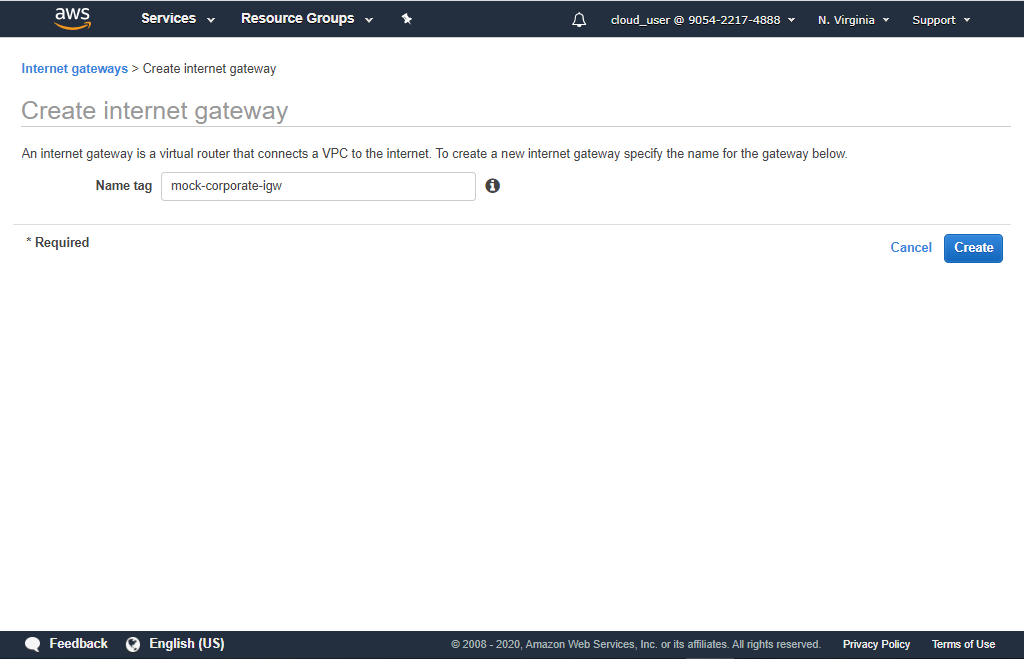


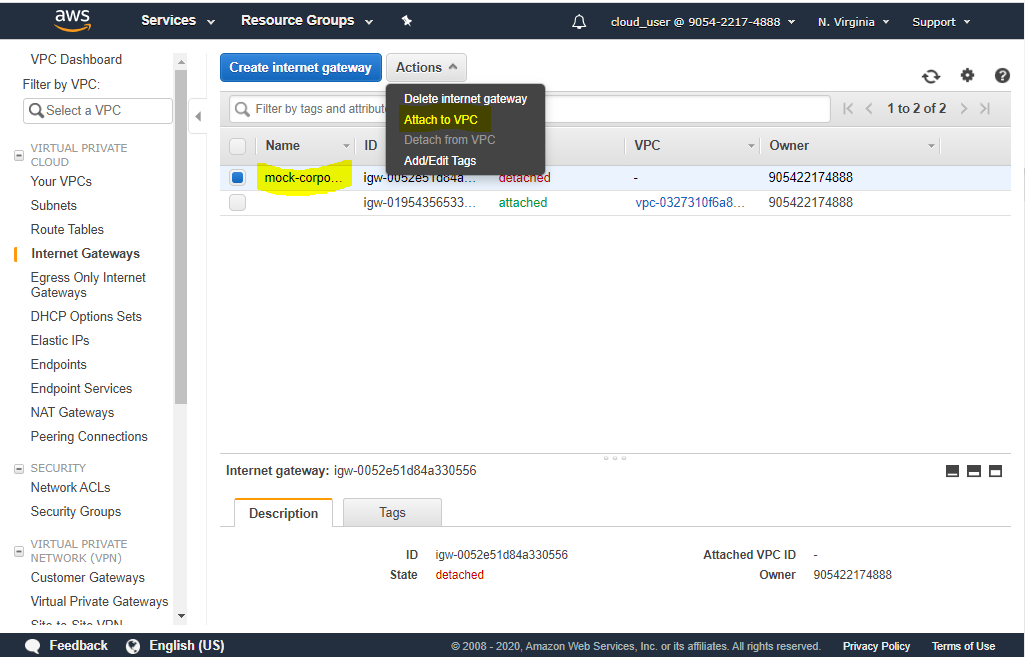
1. For **public subnets**, enable auto assign IP settings

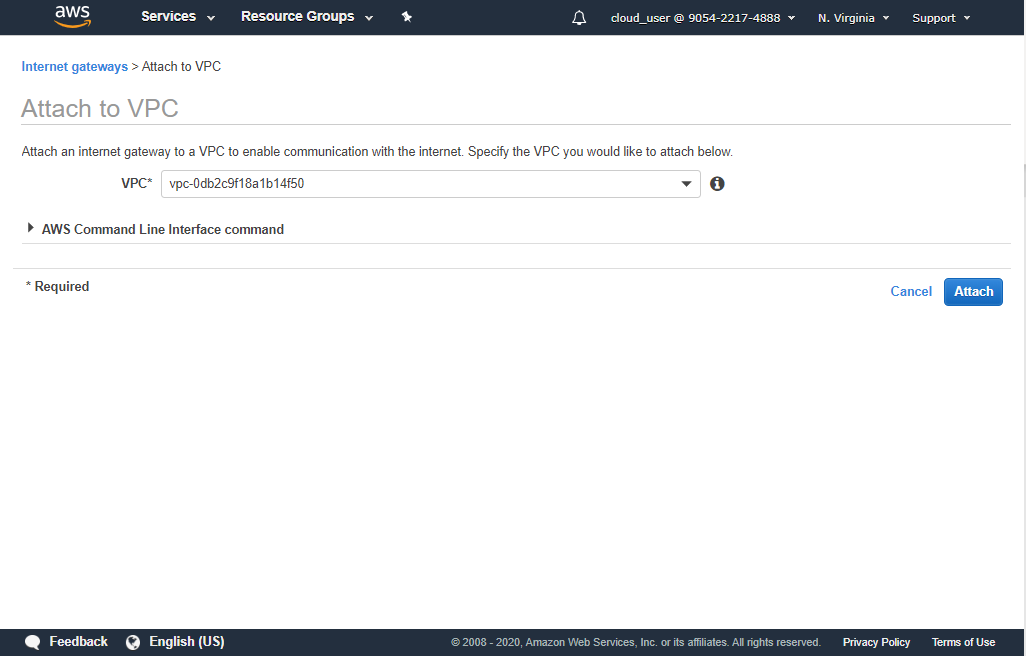




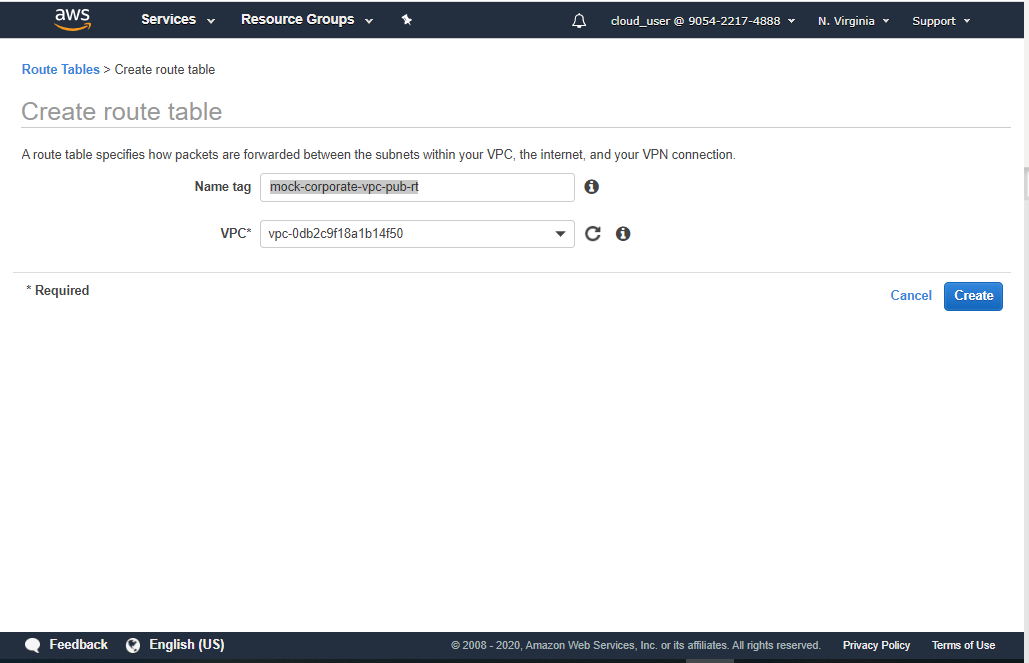
1. Create an Internet Gateway for the VPC to route traffic to internet & attach the same to our VPC

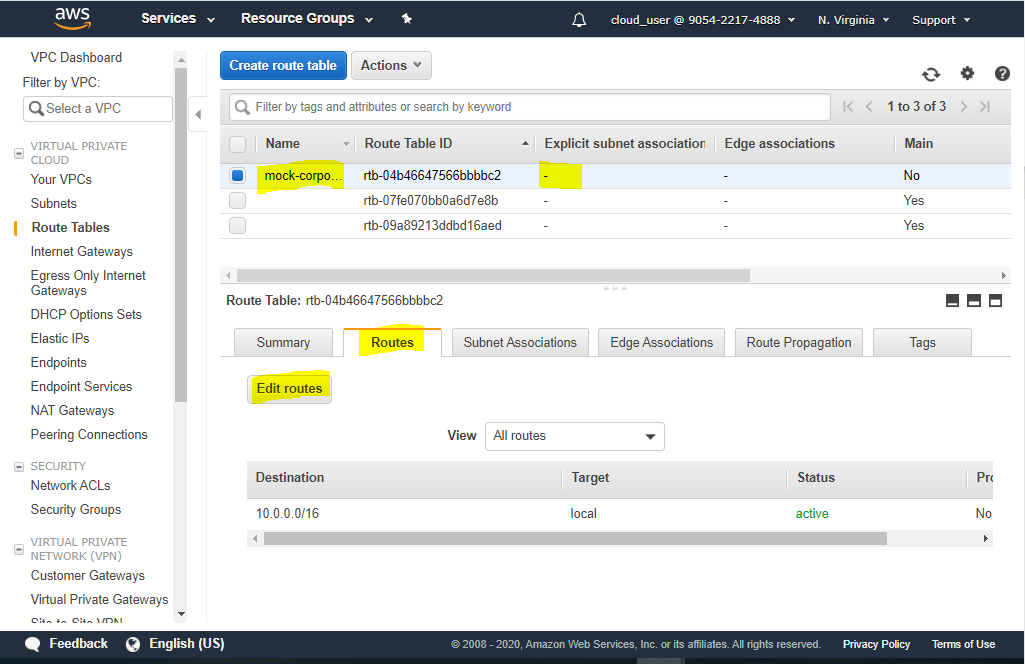


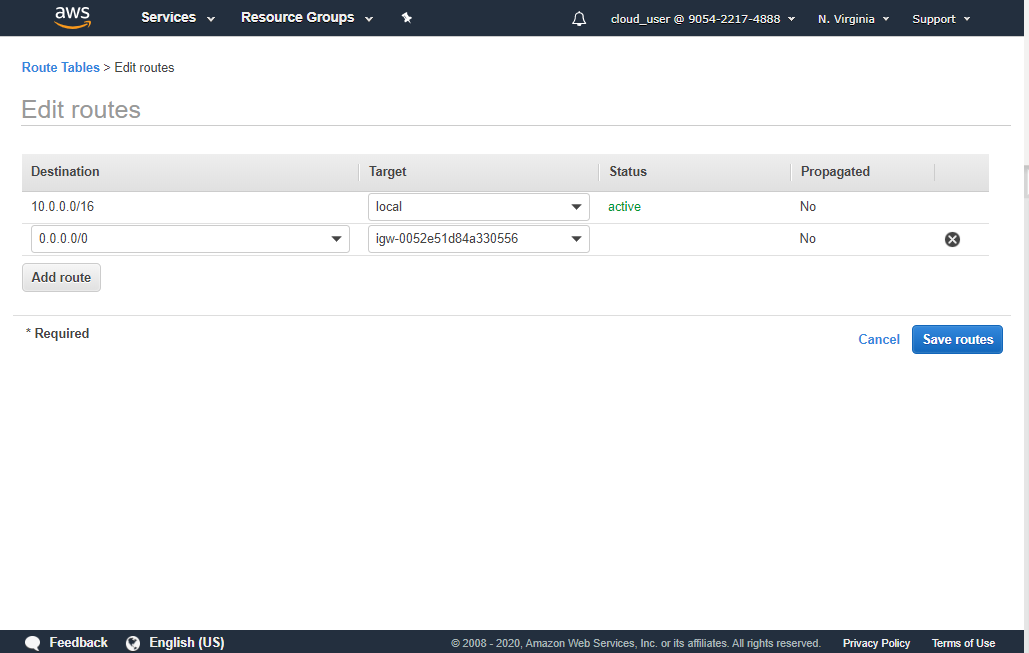


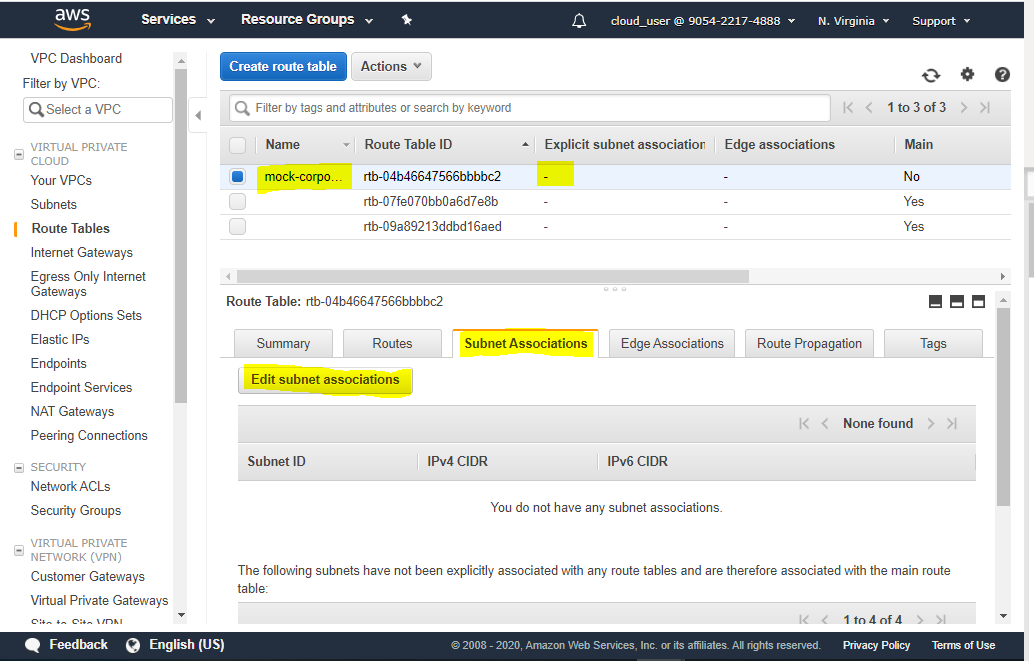


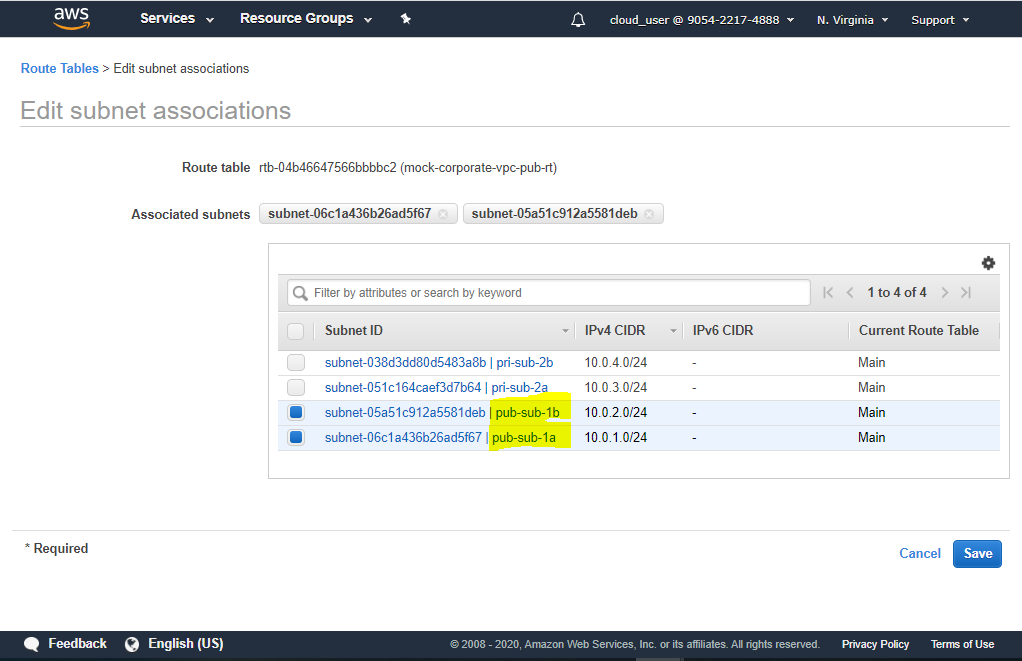
1. Next, create **a public route table** associated with the VPC 🡪 Modify the route table to direct all traffic not intended for VPC to the Internet Gateway 🡪 Attach public subnets to this route table

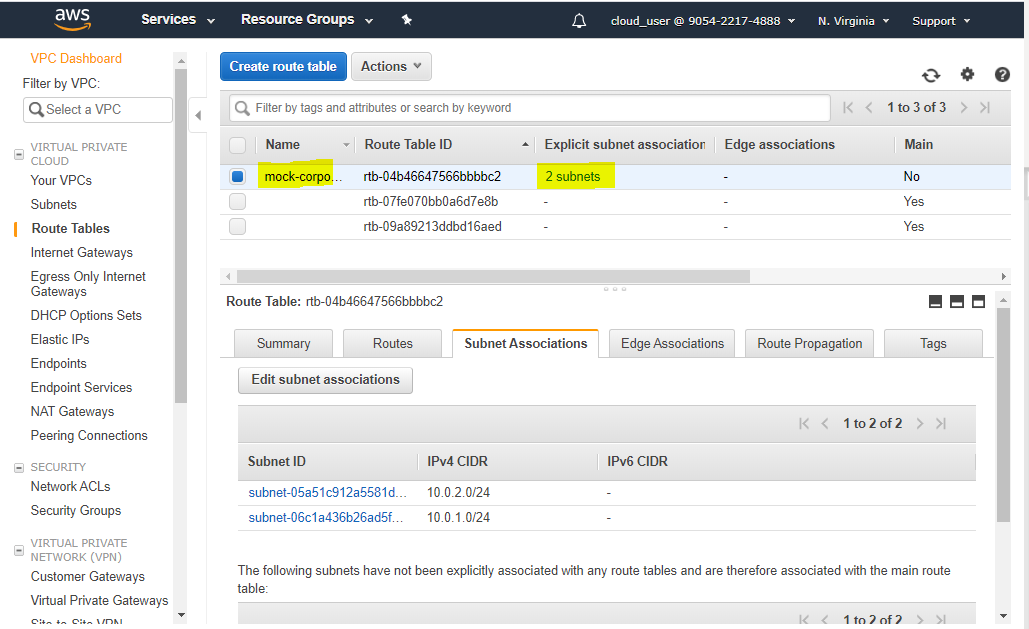




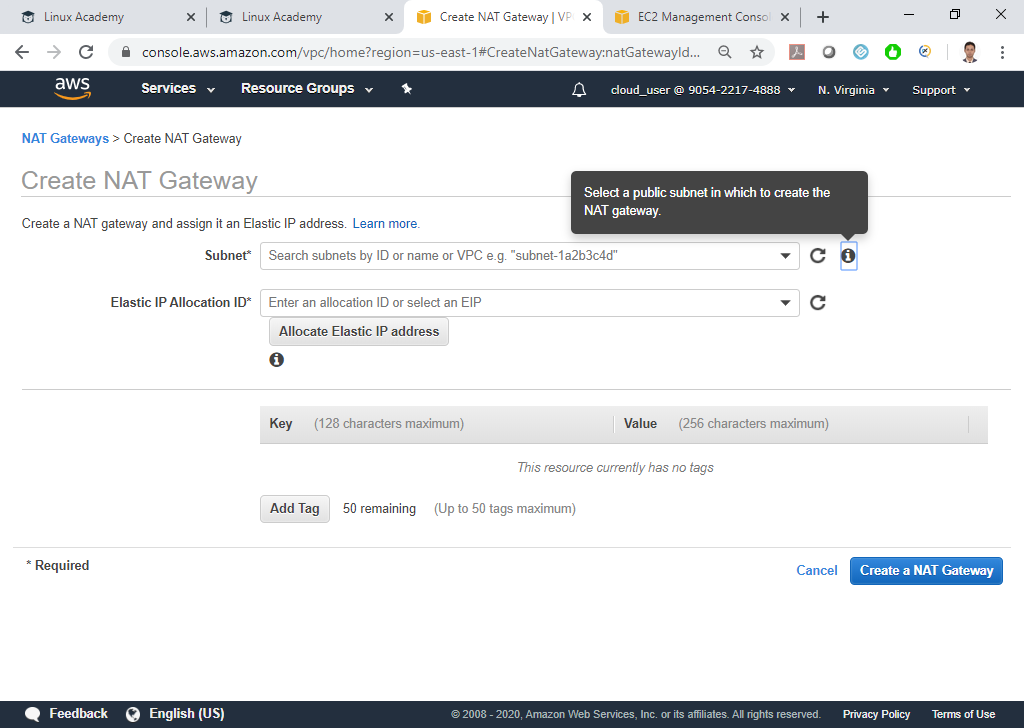


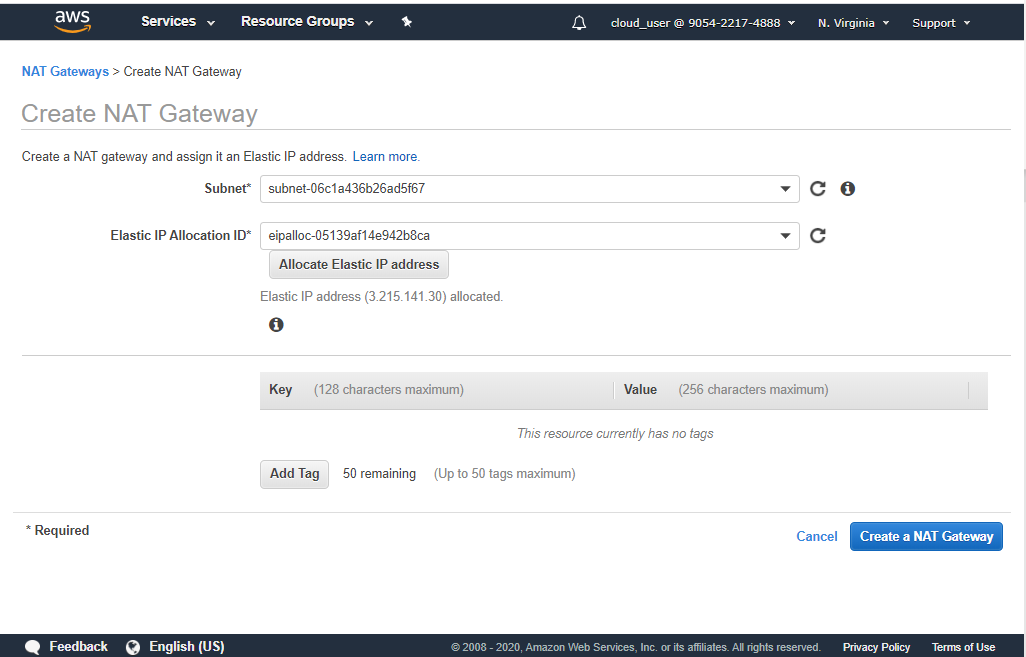


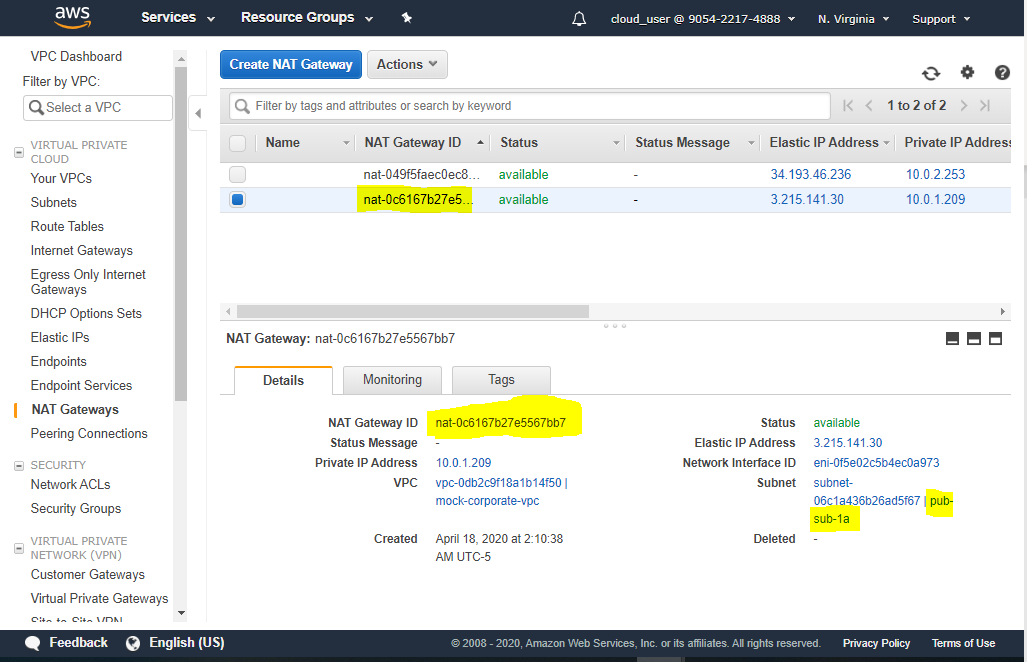


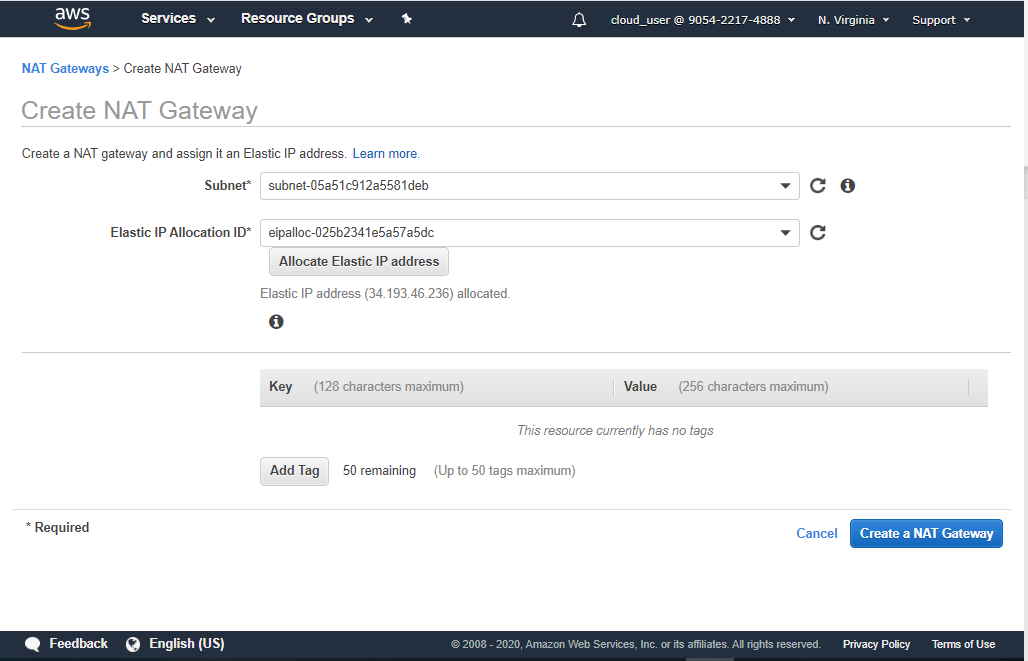


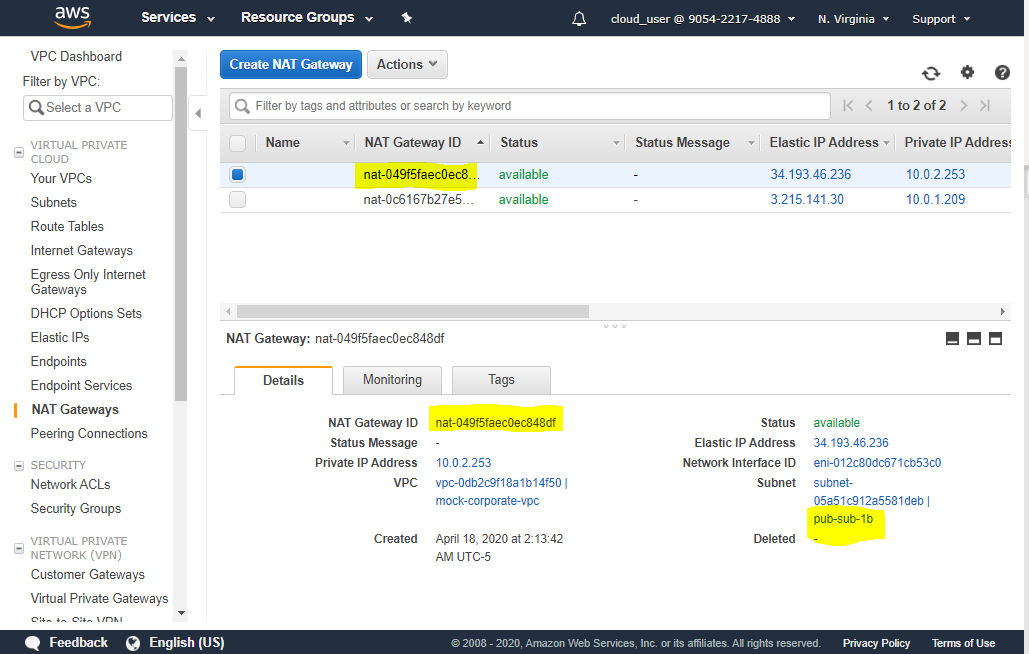
1. Create a **NAT Gateway inside each public subnets**, so that we can **access internet from the EC2 instances in private subnets**



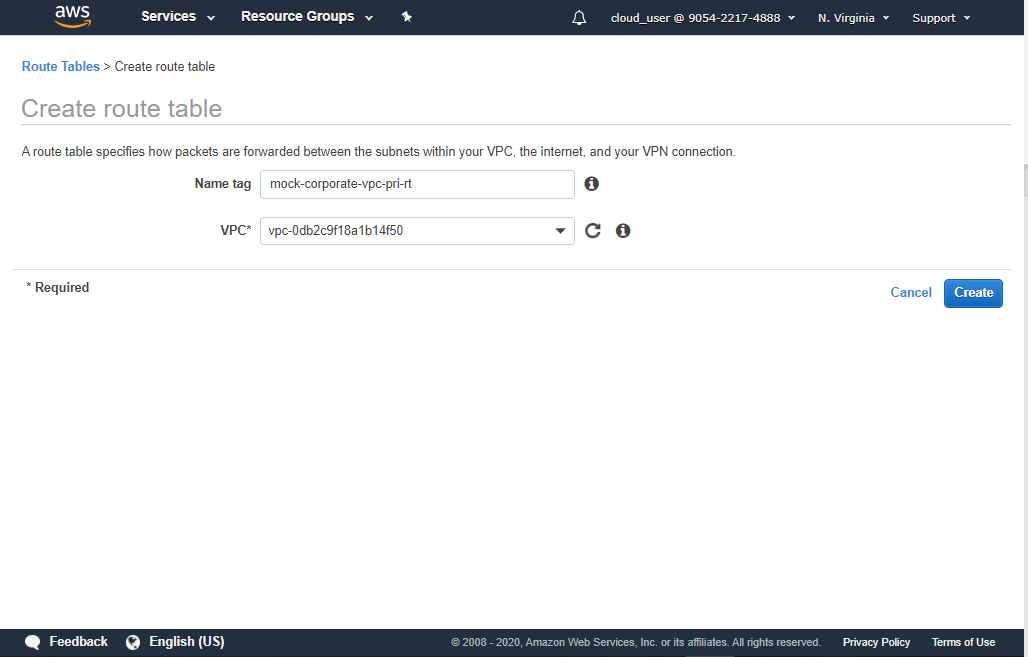


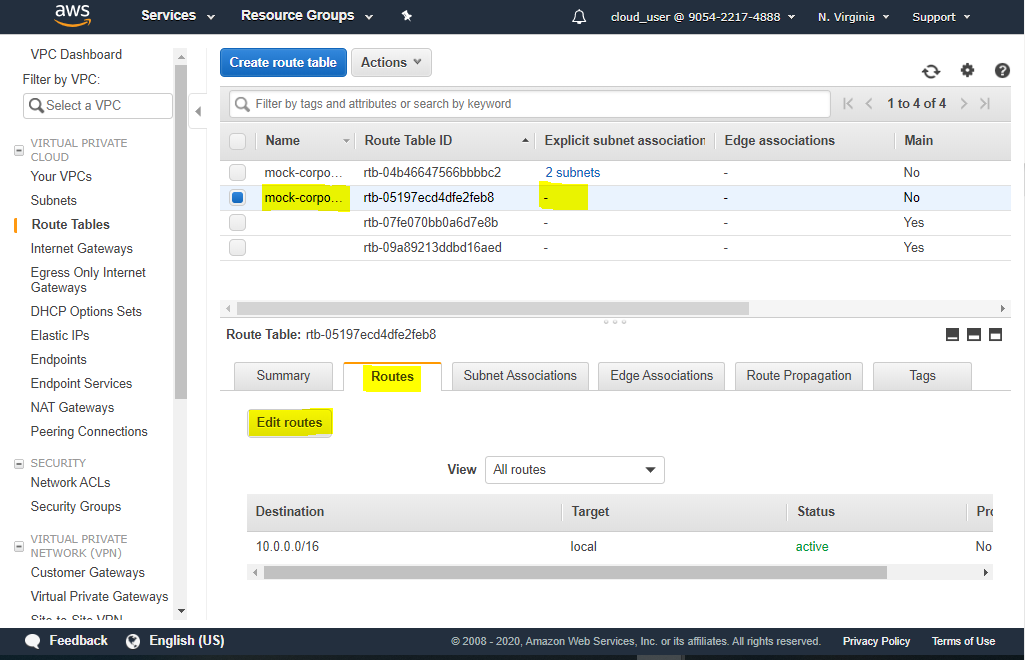


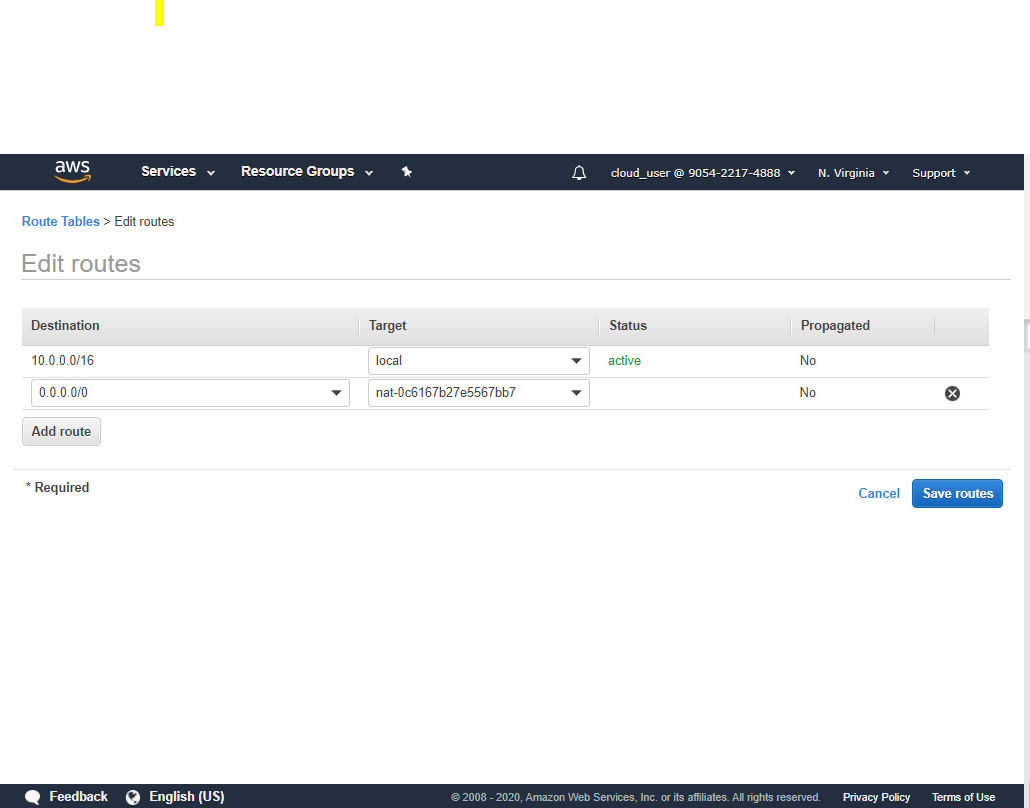


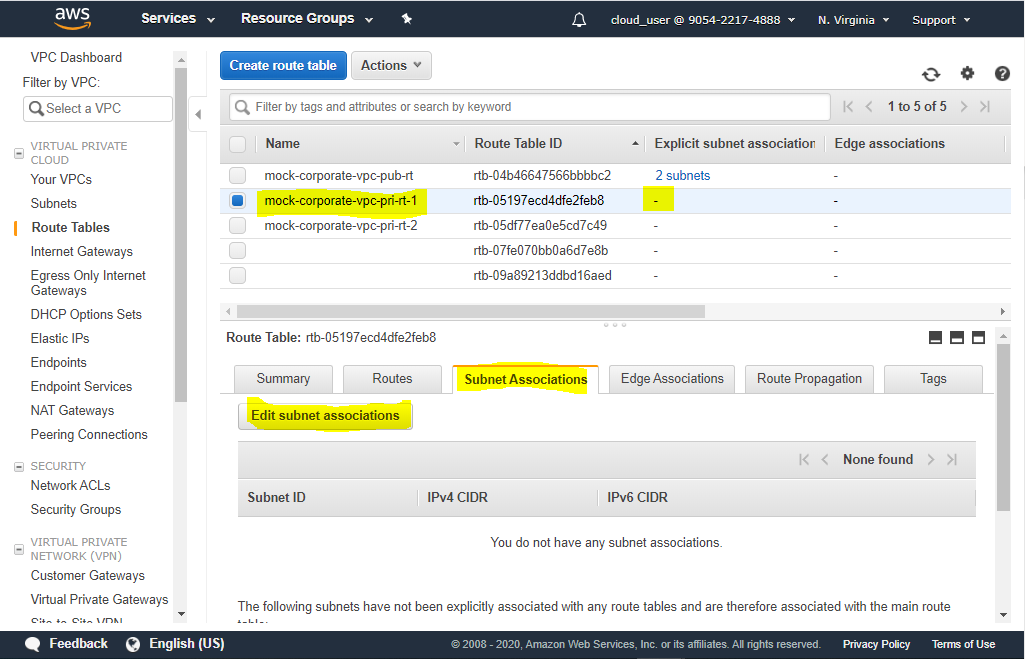


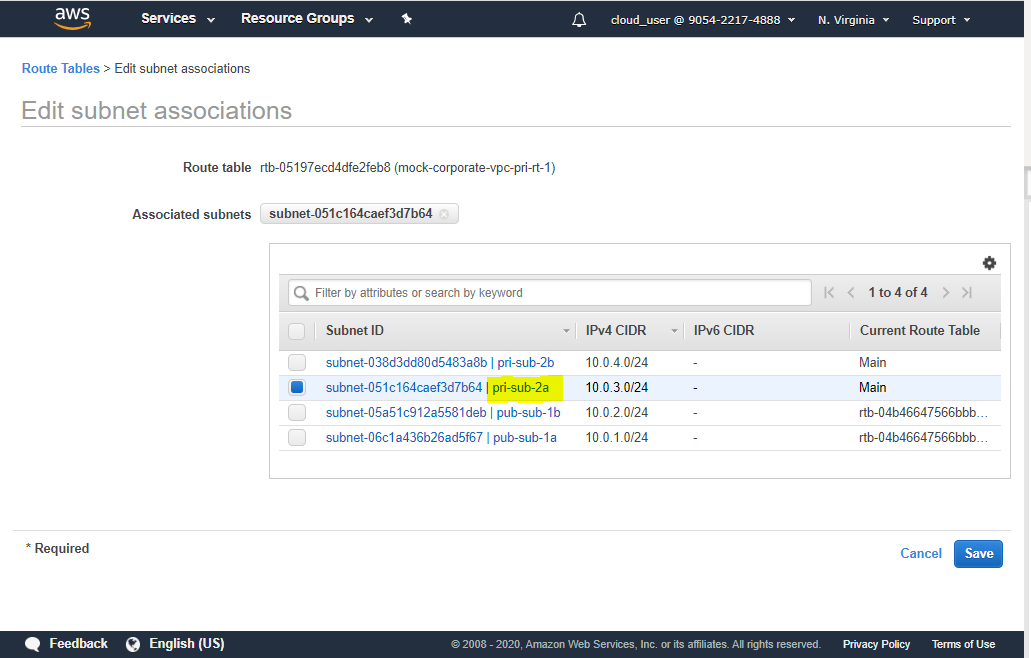
1. Create a route table to route traffic not intended for VPC to NAT GW 🡪 Associate it with one of the private subnets



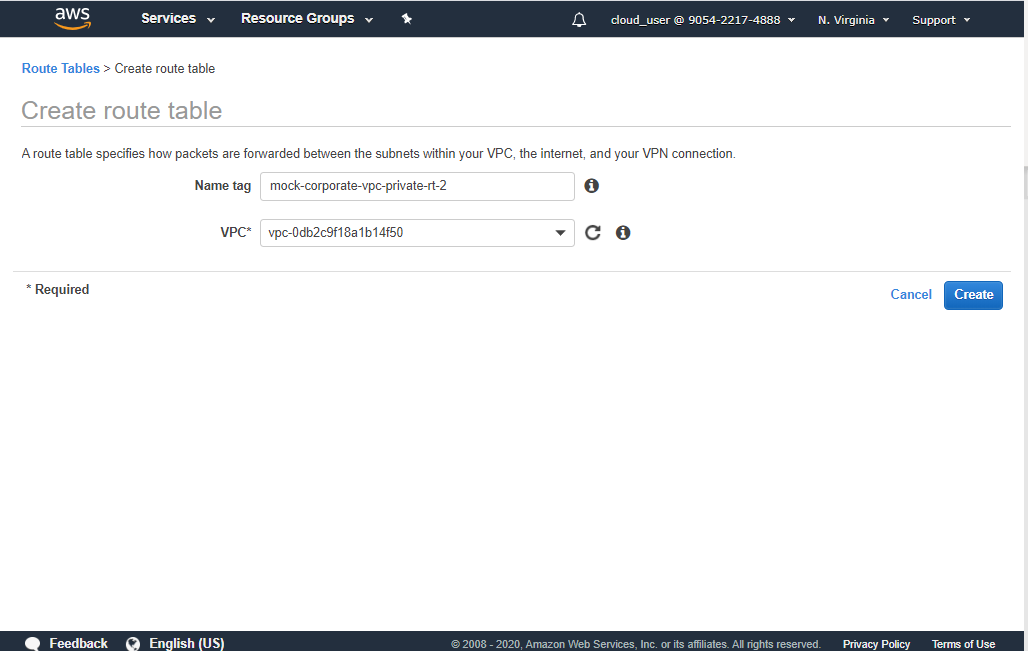


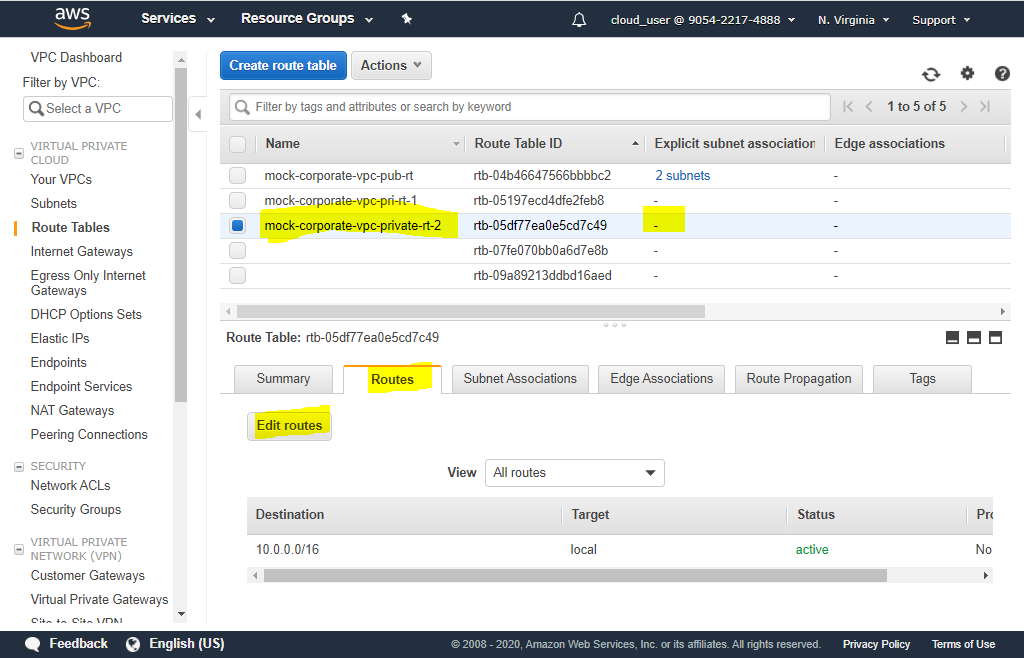


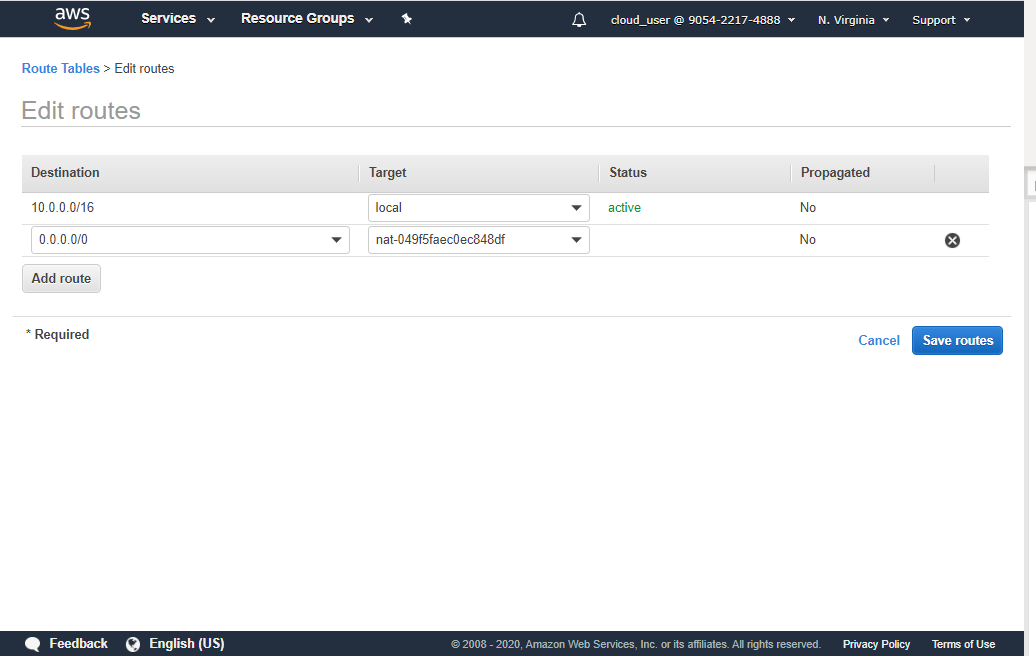


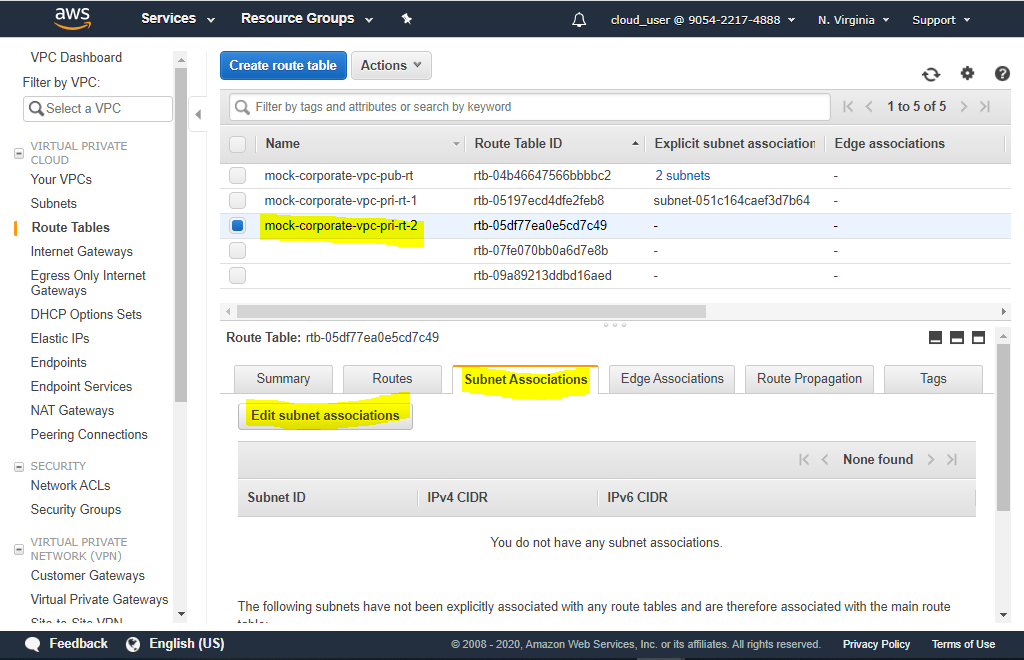


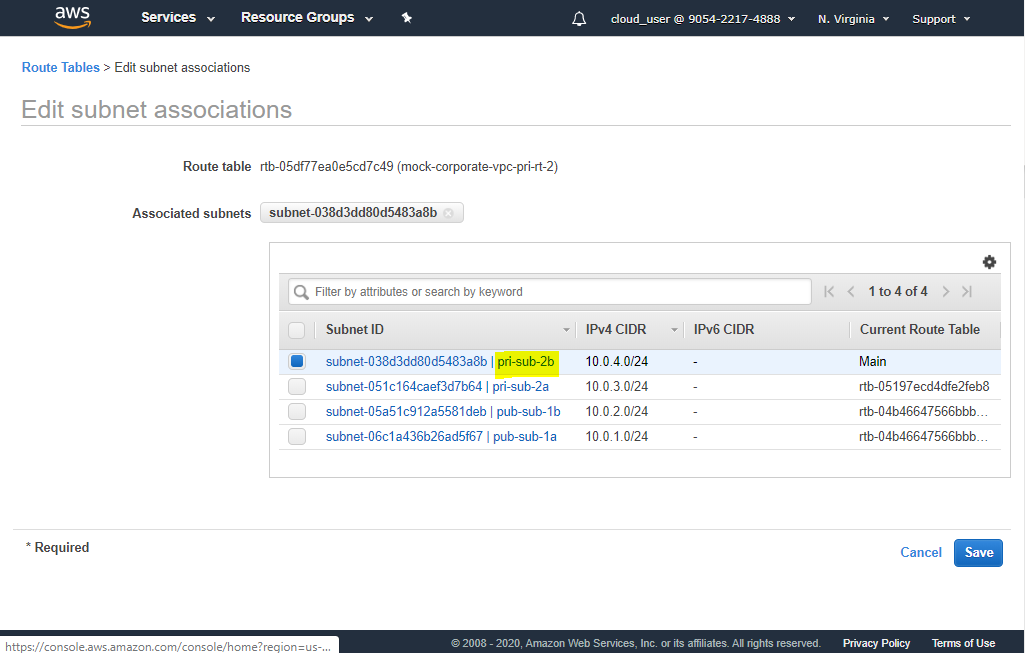
1. Repeat the same for the other private subnet



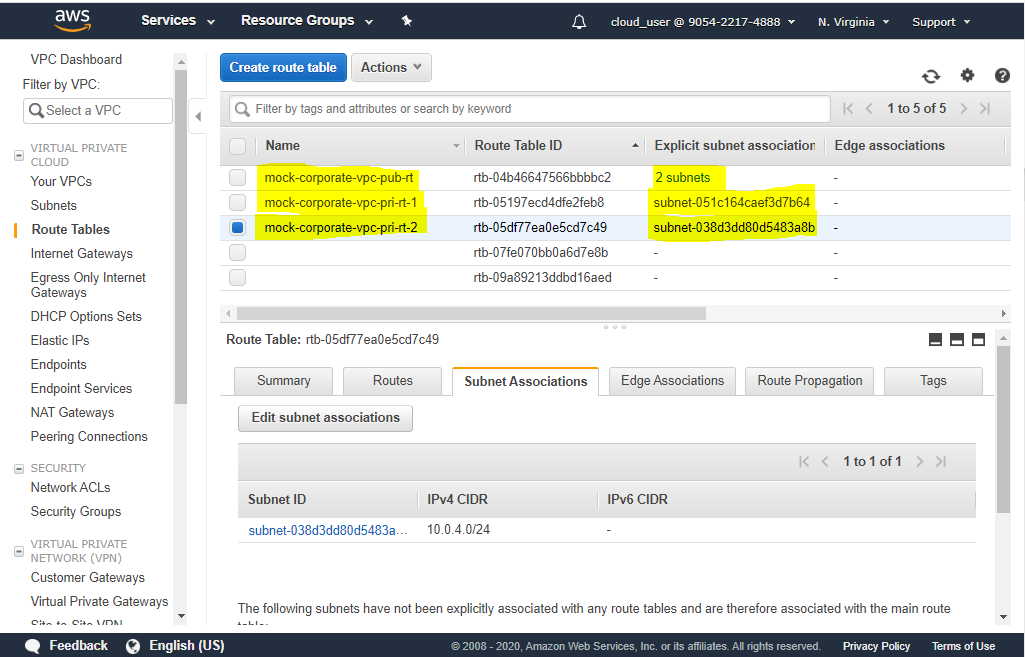




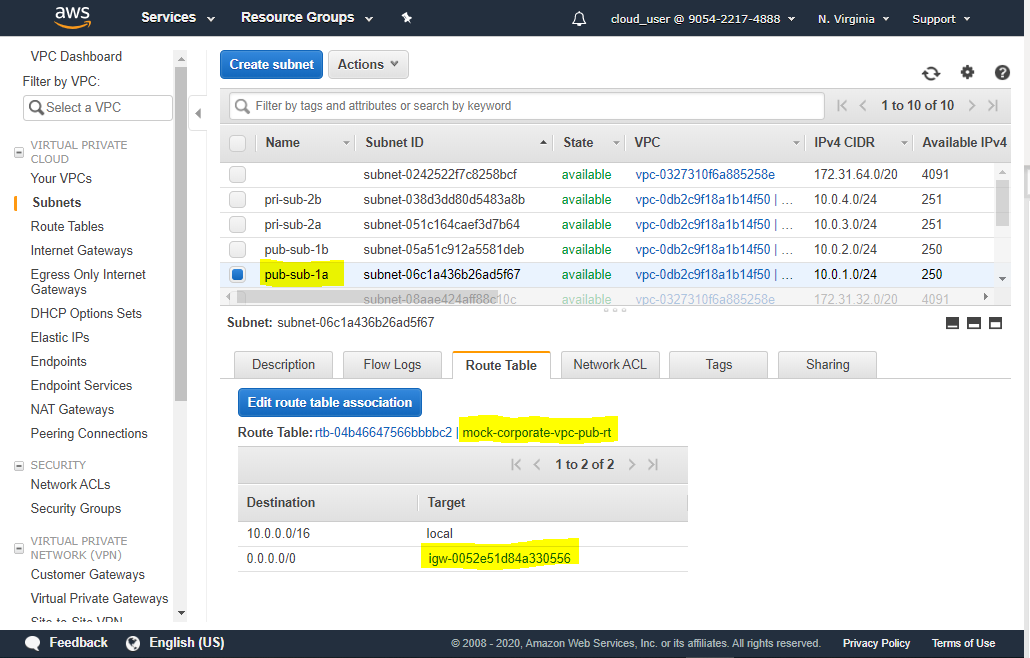


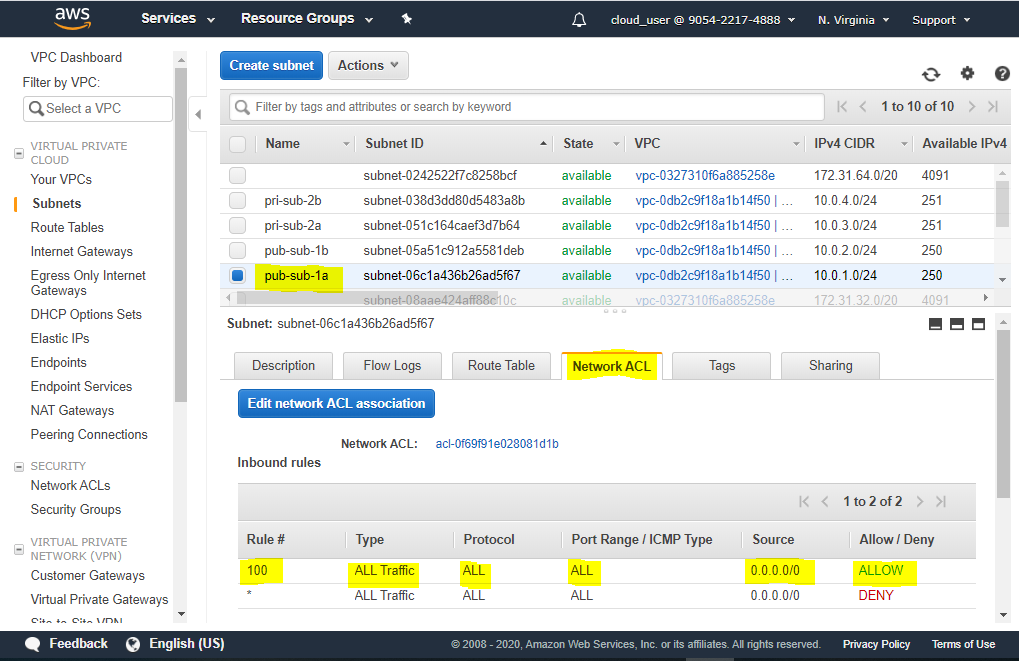


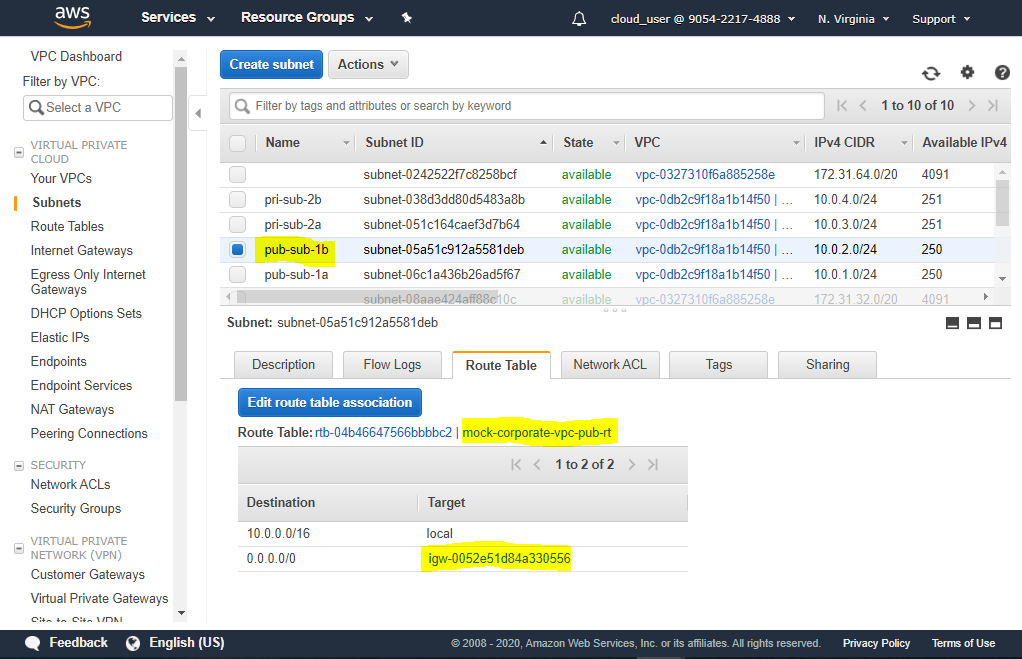
1. At this stage, the routes should look like below



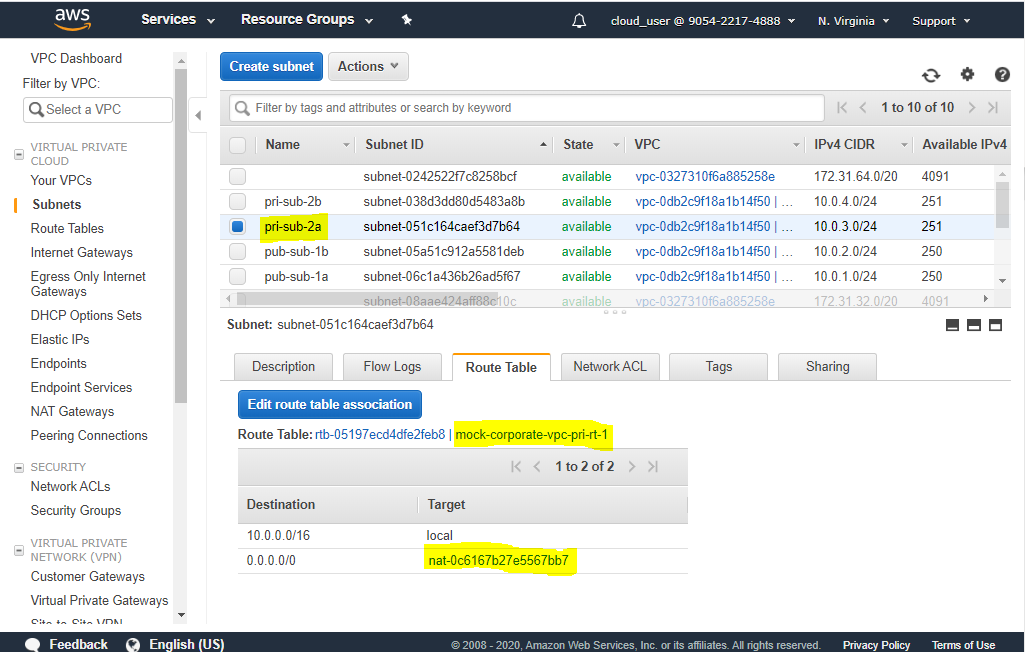
1. And looking from each of the **public** subnets, it would look like following screenshots

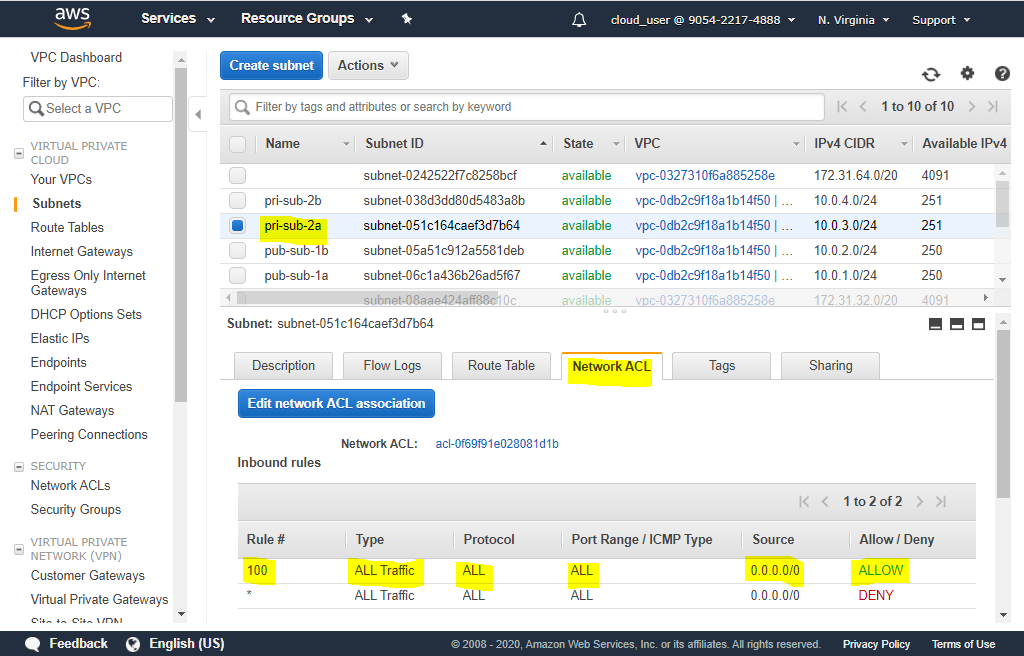


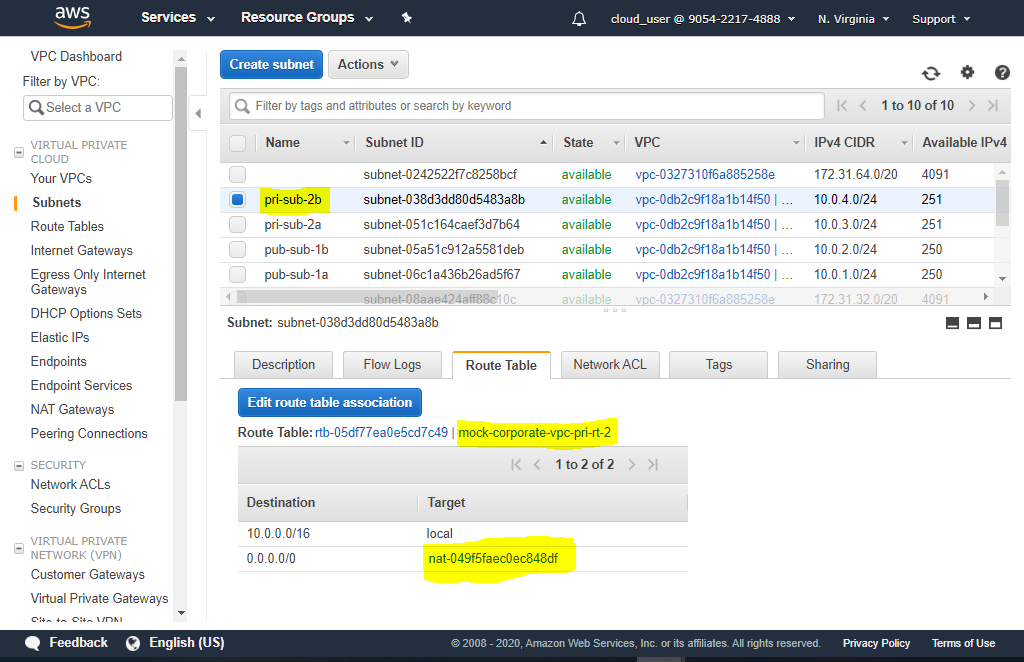




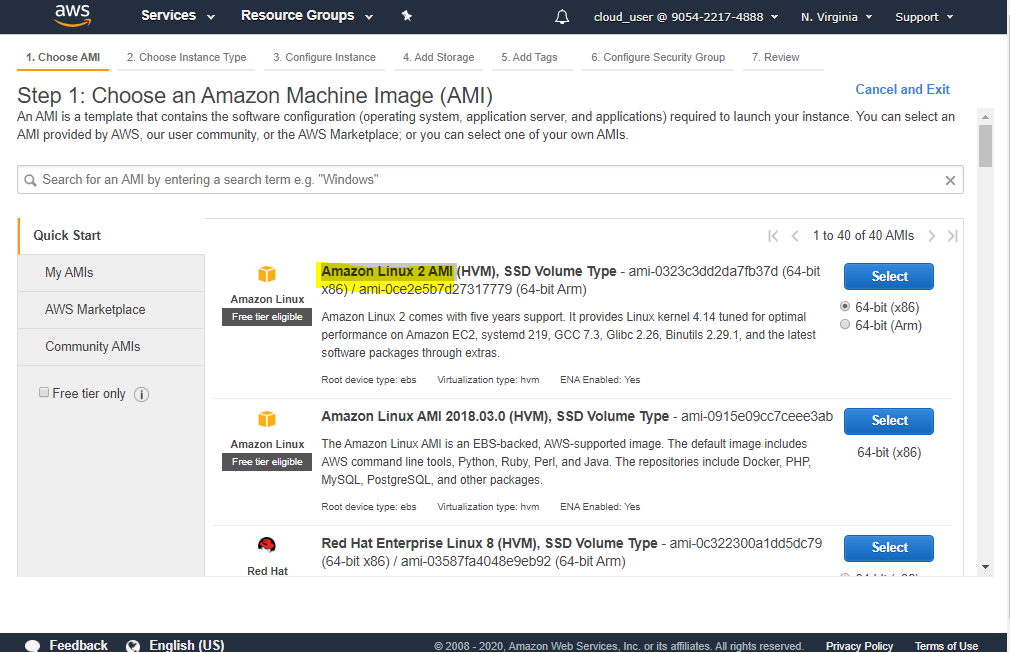
1. And looking from each of the **private** subnets, it would look like following screenshots

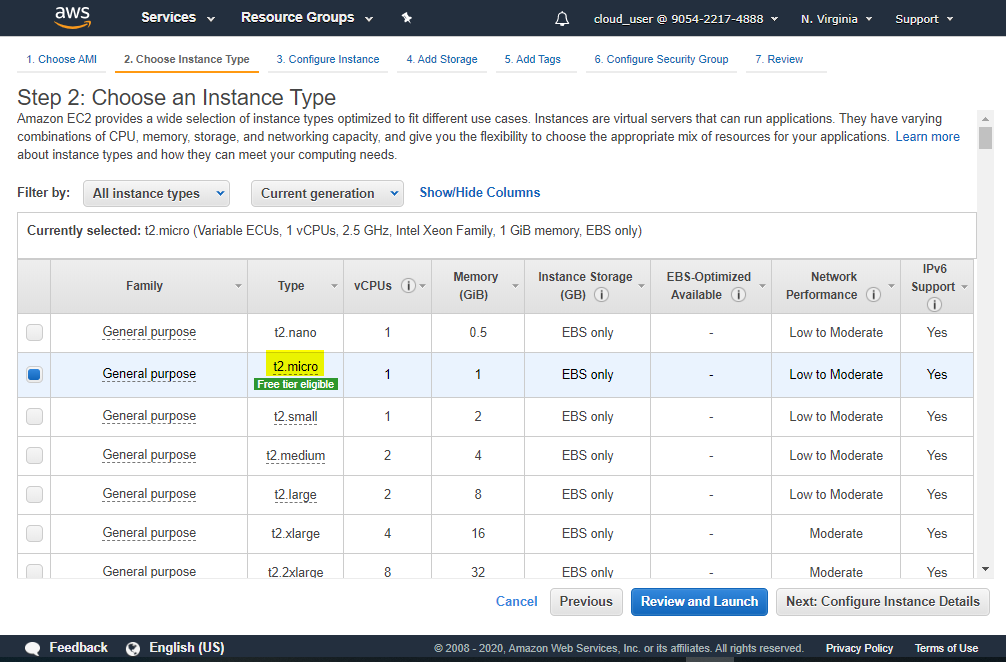


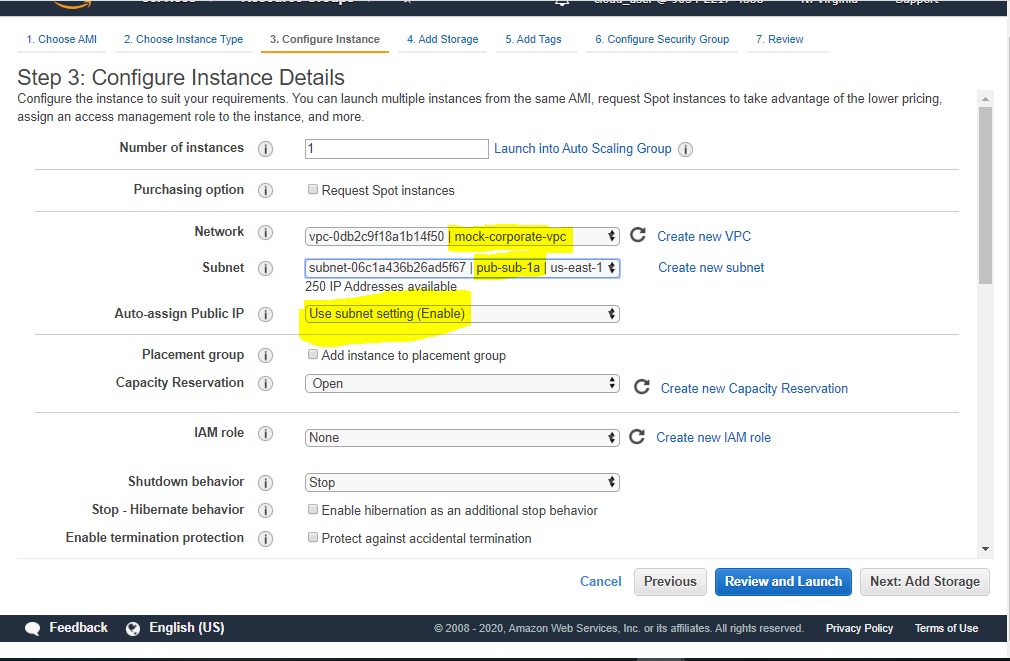


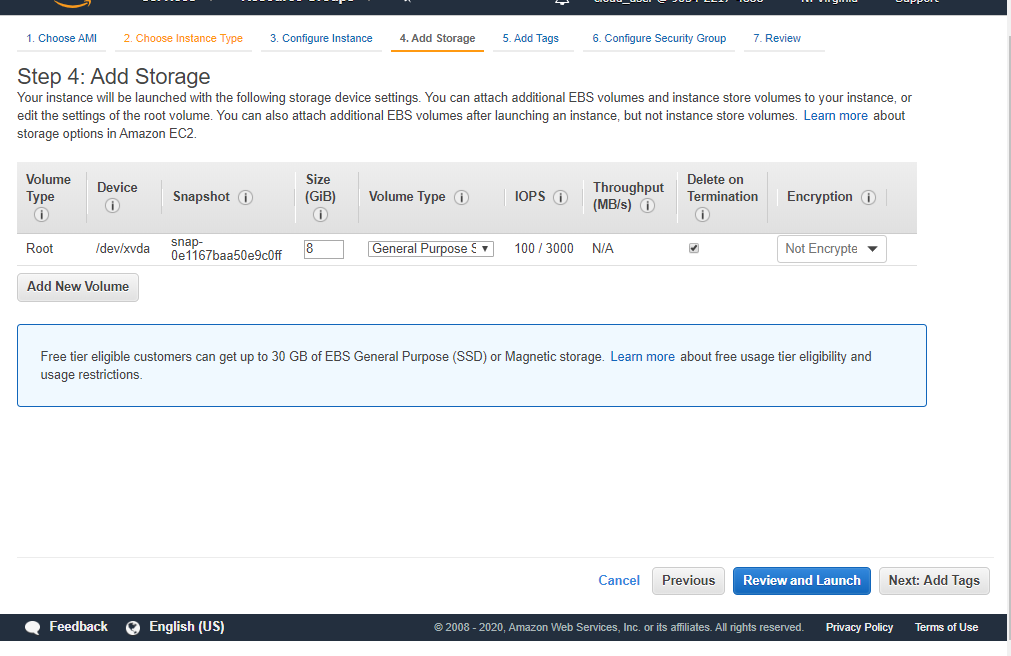


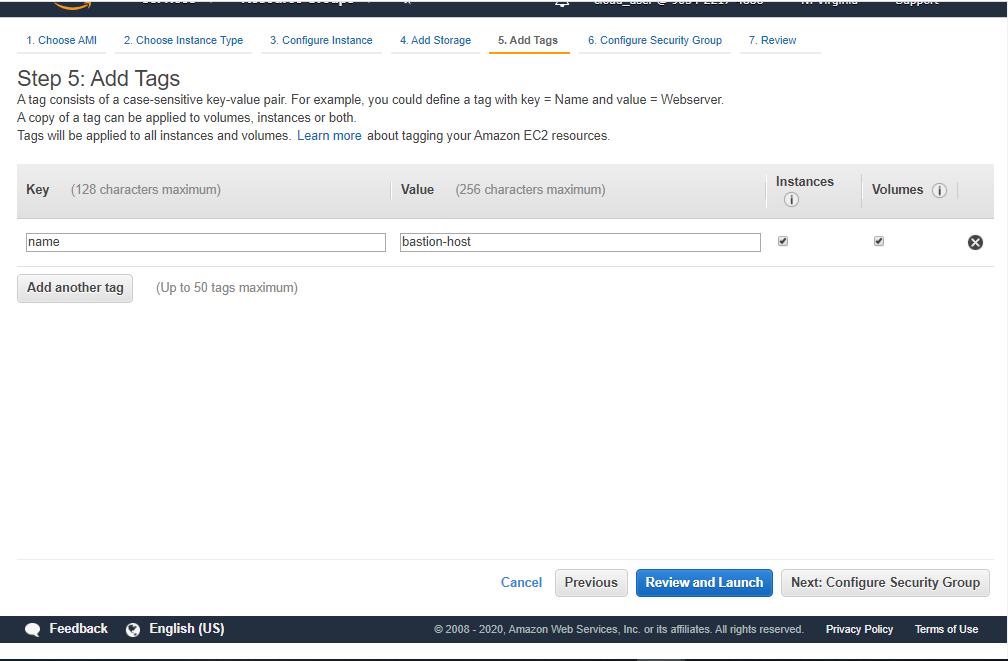
1. Now, lets create a bastion host (EC2 instance in public subnet, accessible from internet)



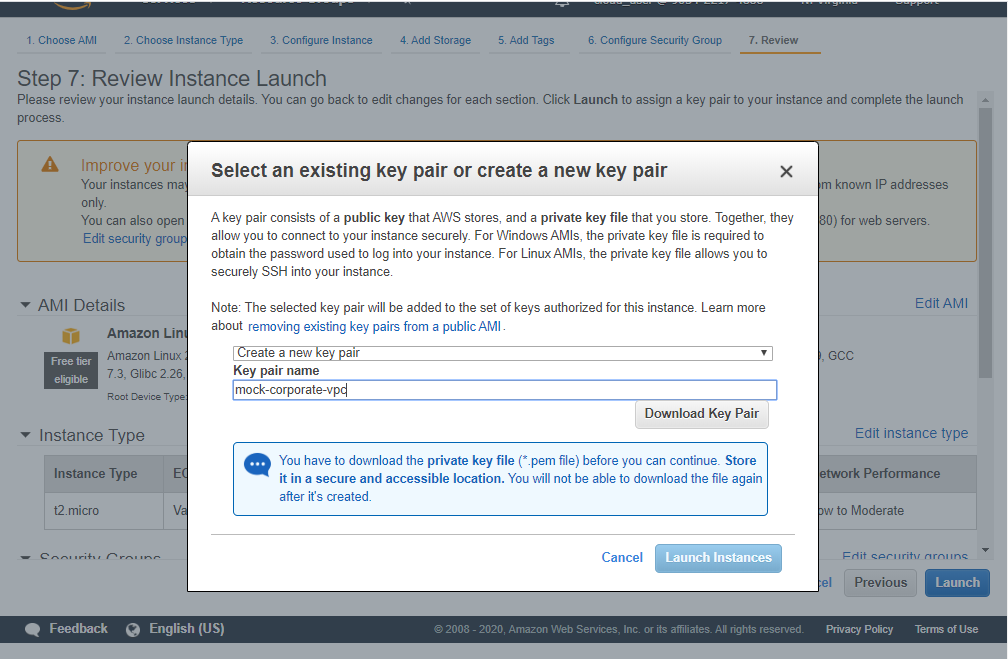




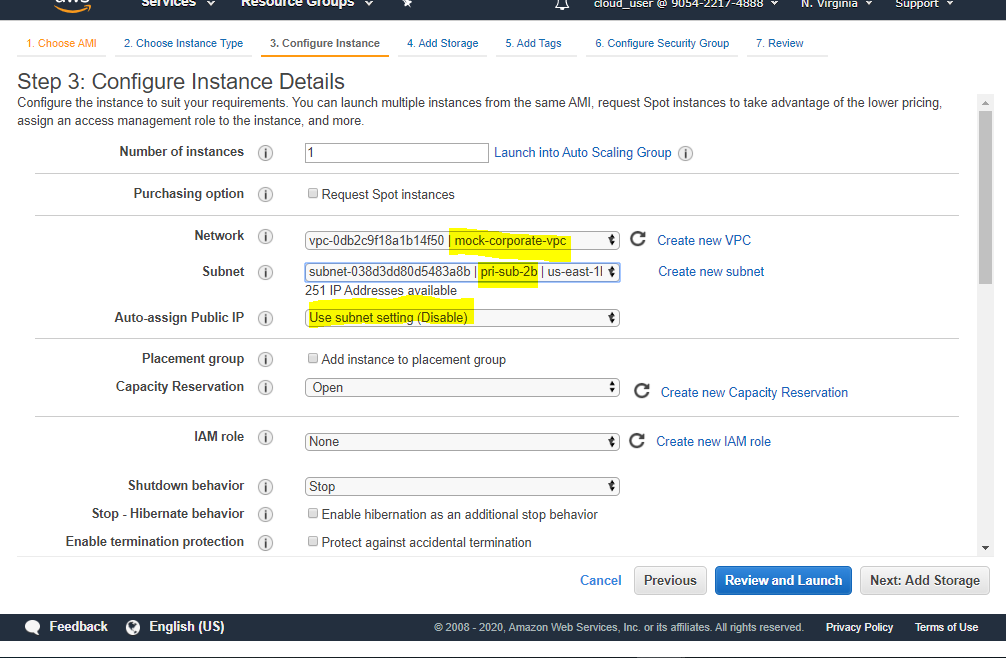


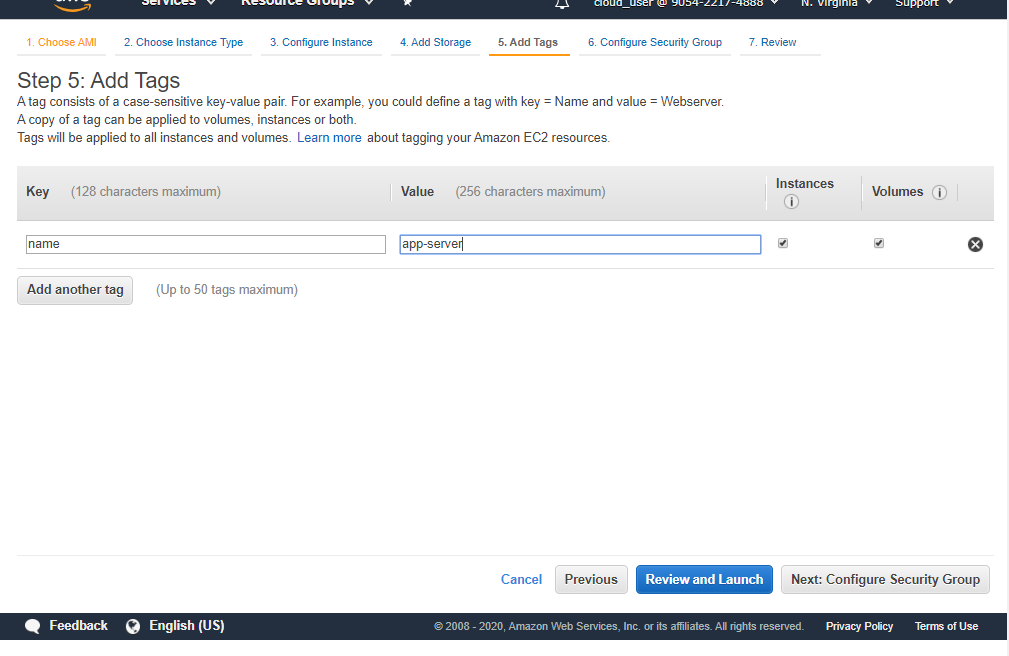


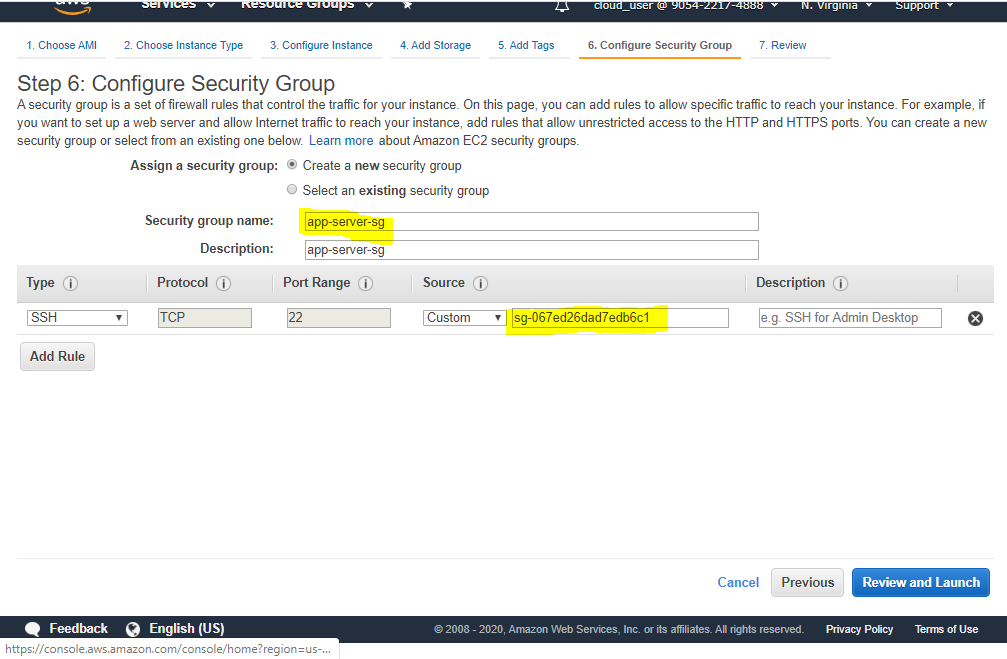


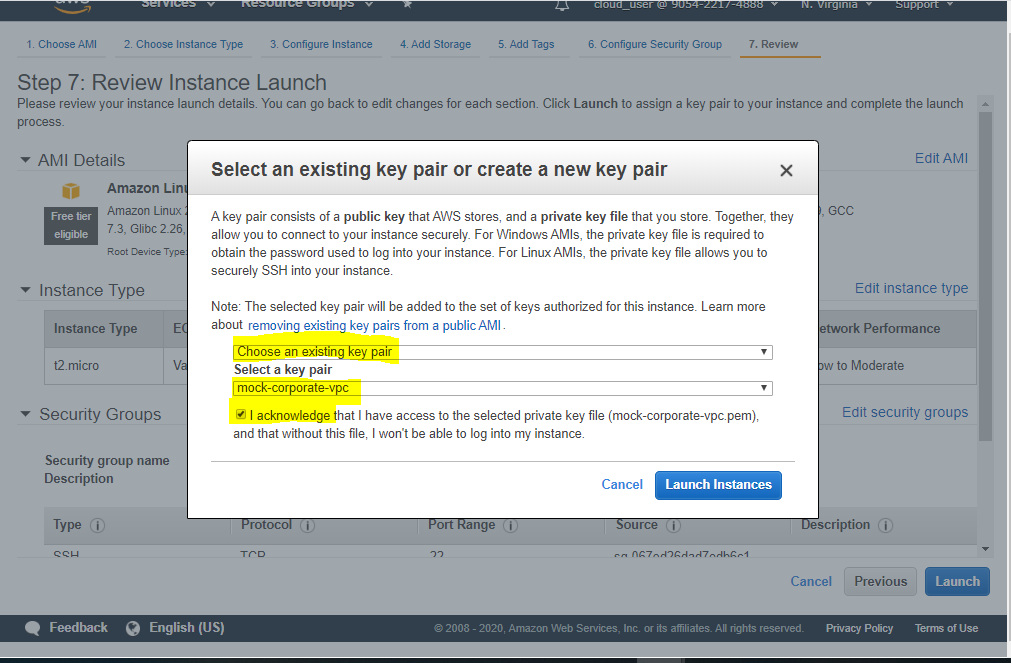


1. Create another EC2 instance, but inside a private subnet. The process should be similar except for the steps shown below. Note: It should allow traffic ONLY from the security group associated with the bastion host

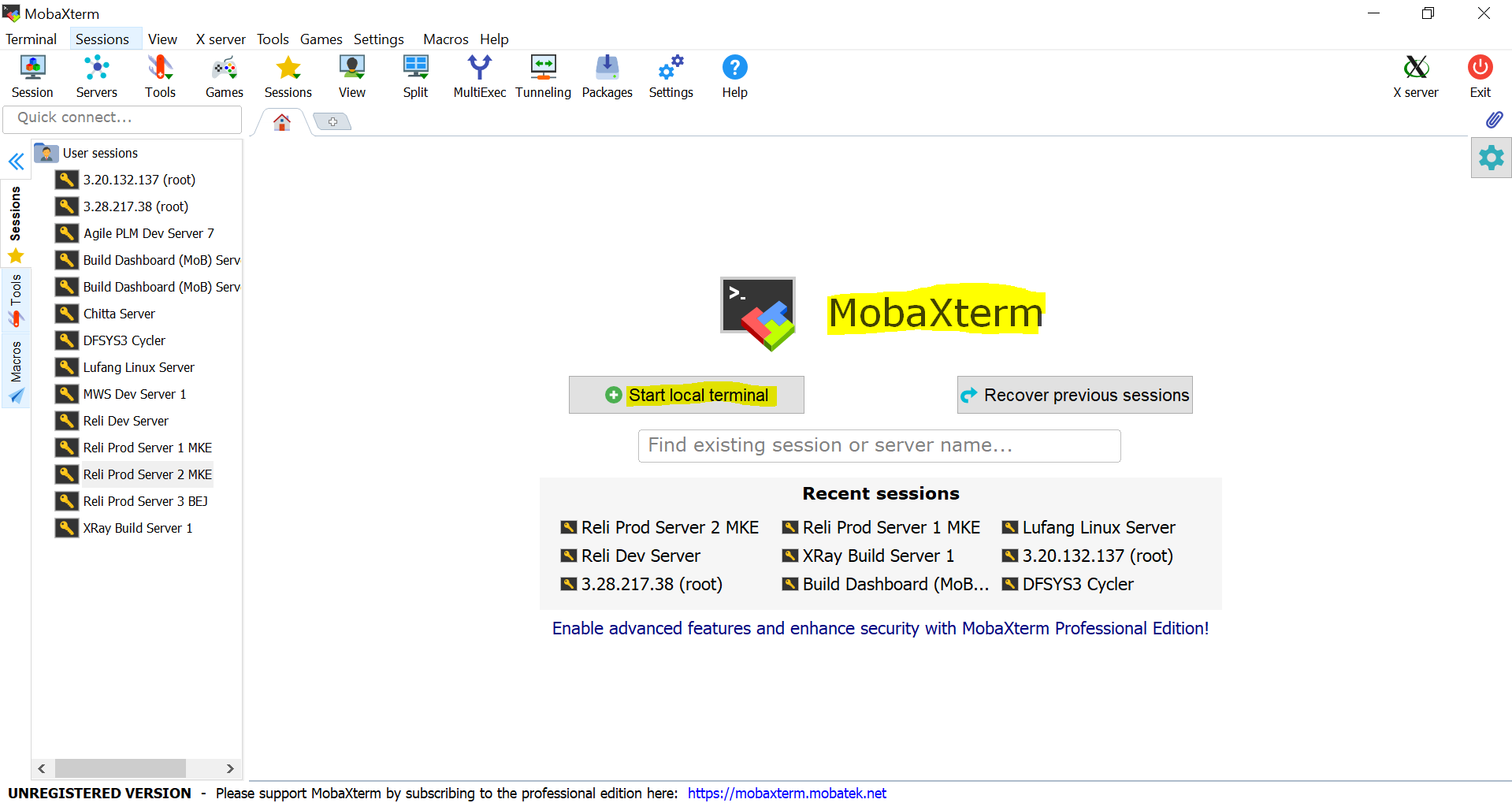


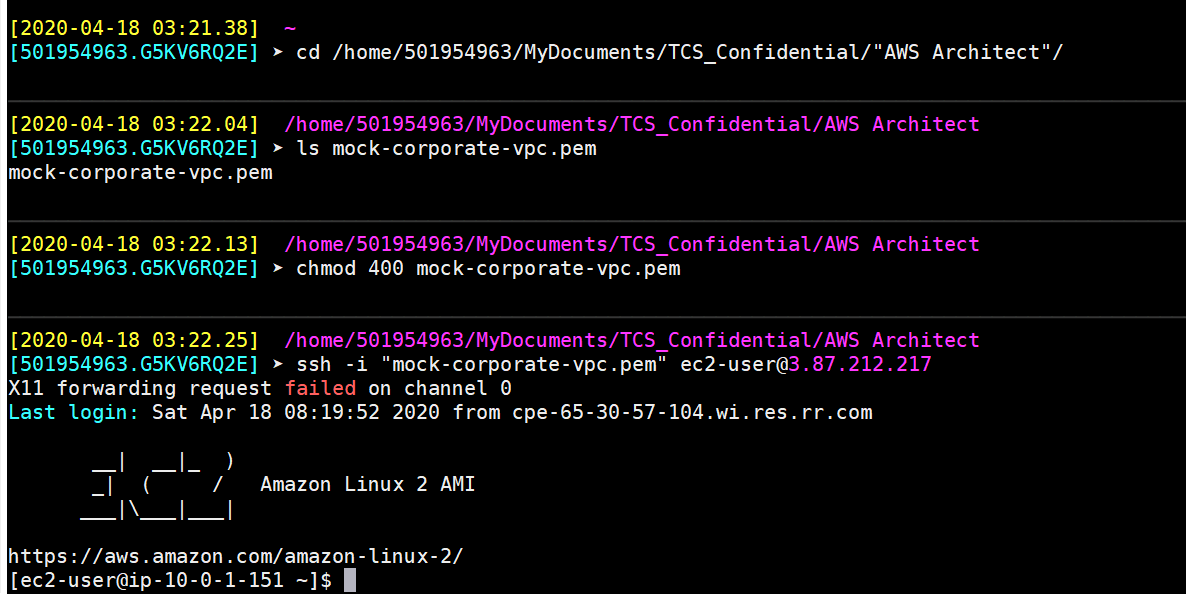


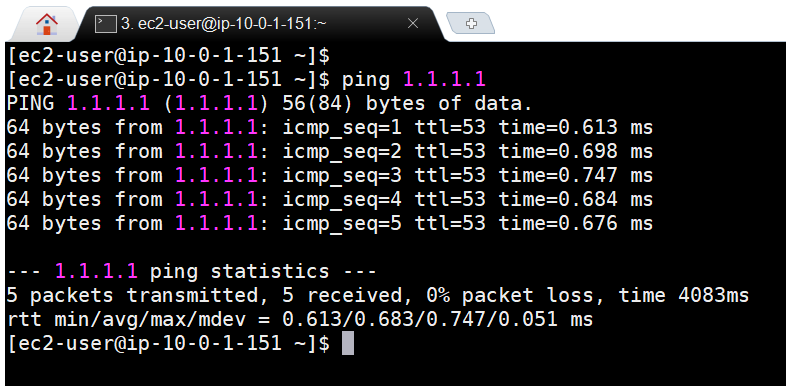




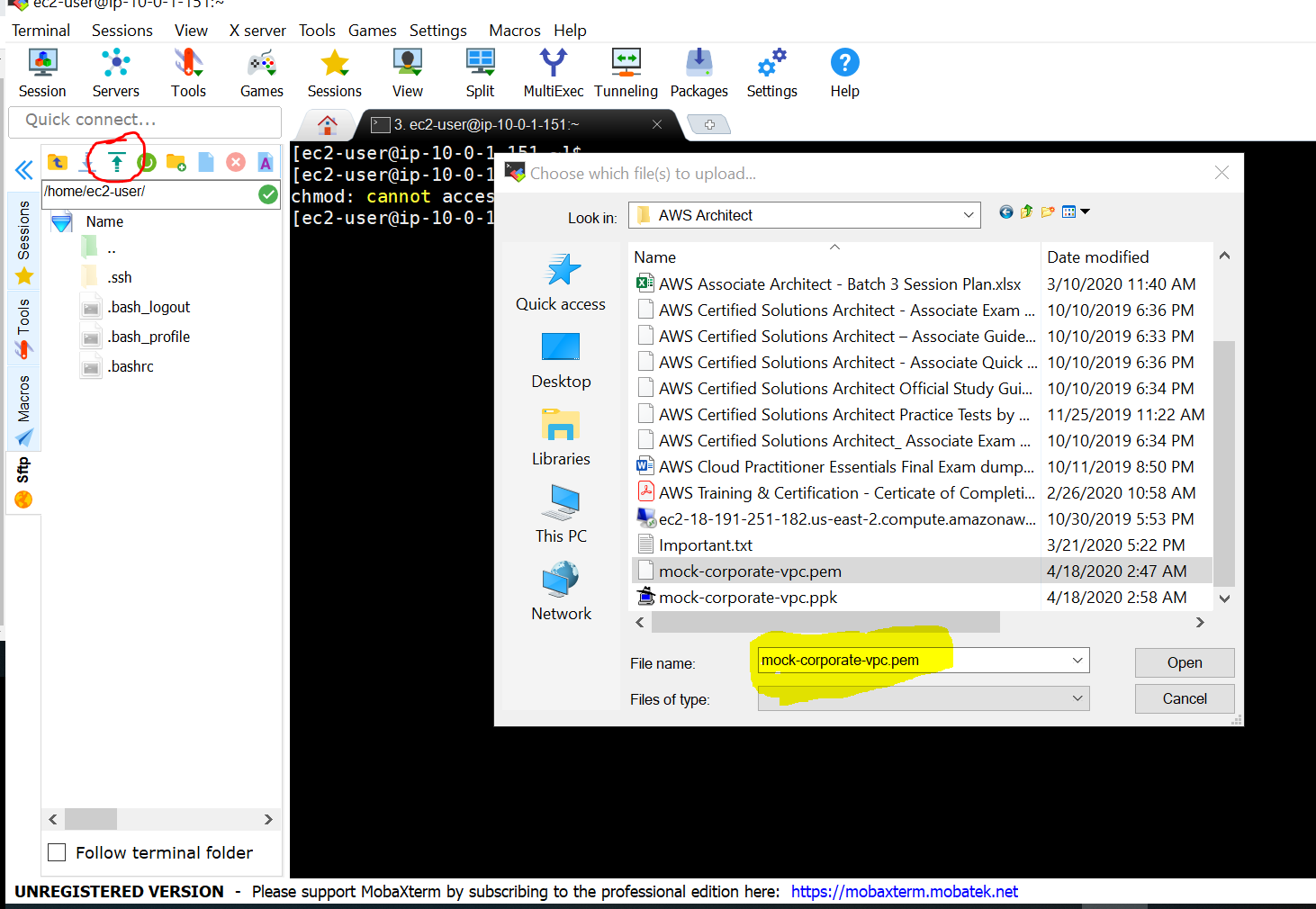
1. I would recommend installing MobaXterm to connect to EC2. Putty is not very reliable. Follow the steps to connect to the bastion host. As shown in screenshots, we can access internet from bastion host



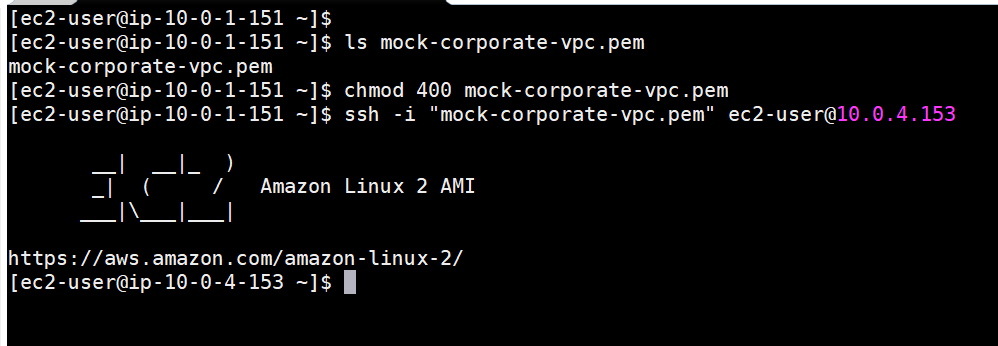




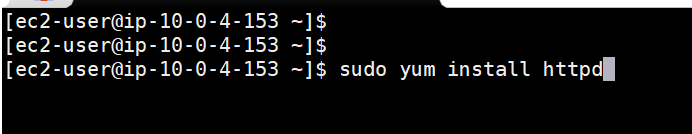
1. Next, we would upload the pem file to the bastion host, so that we can use the same to connect to the App Server (private EC2 instance)

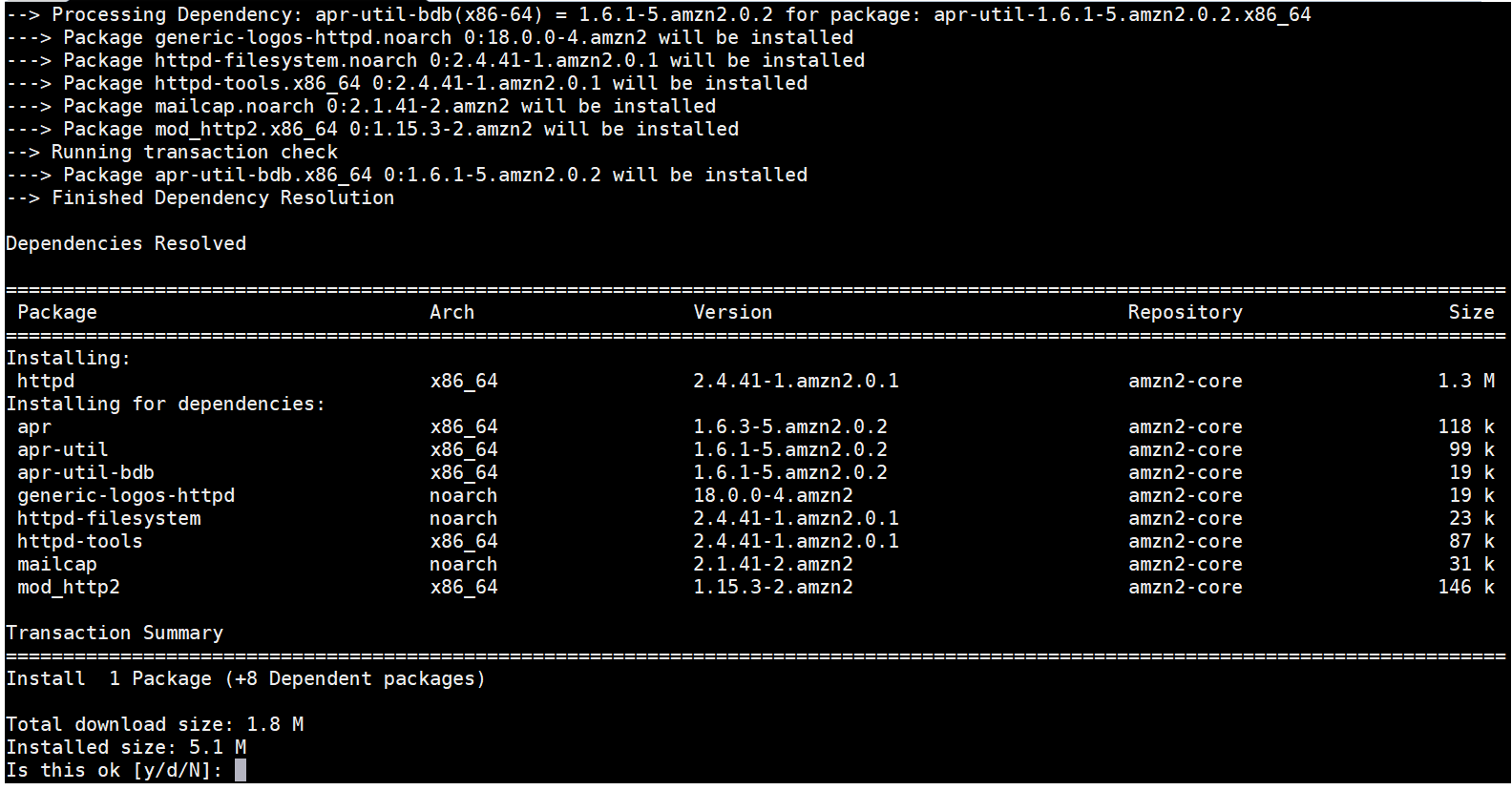


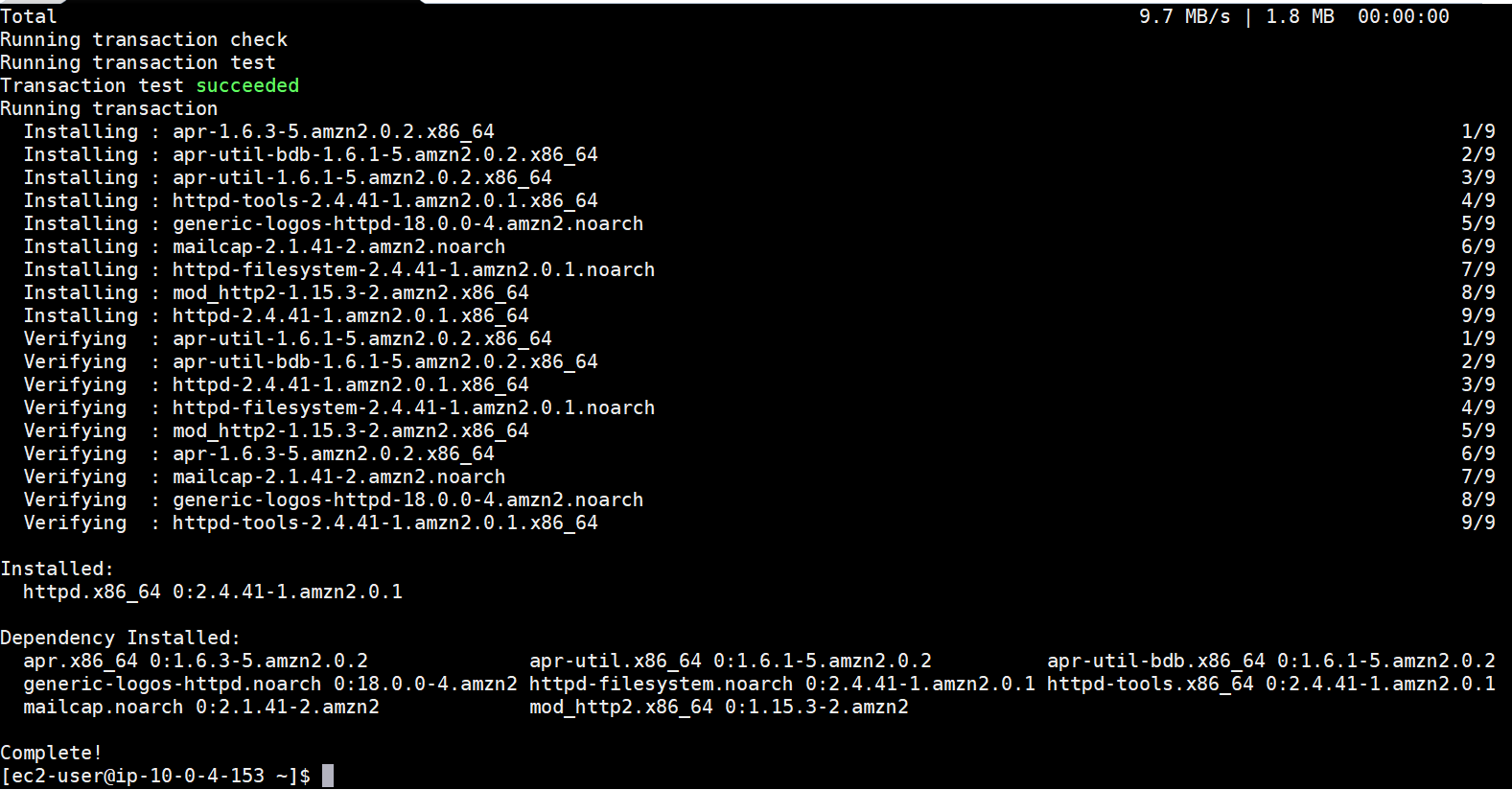
1. Once uploaded, use following steps to connect to the App server



1. Once logged inside App server, try installing httpd or Apache server from internet. A success indicates that the NAT GW is doing its job



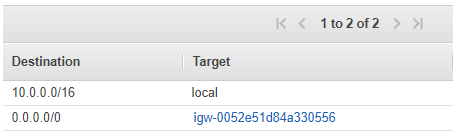




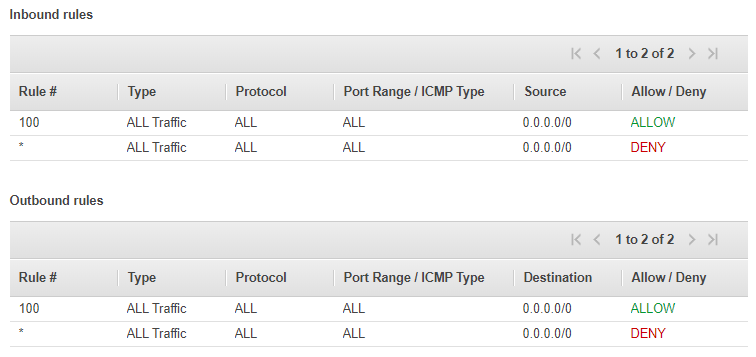
1. This concludes the experiment

# Difference between a Route Table, A NACL & A Security Group

* Route Table
  + It just says how to route a traffic inside or outside the VPC
  + Associated with a VPC & a subnet / multiple subnets



* NACL
  + It allows to specify inbound and outbound rules SEPERATELY
  + Allows to specify ALLOW/DENY rules
  + Rules are evaluated with lower Rule# first
  + If no rules match, default deny (Rule \*) applies
  + Can specify following parameters : (almost SAME AS SGs)
    - Rule#
    - Type (HTTP, HTTPS, POP3, SSH, TCP, UDP, telnet, SMTP etc)
    - Protocol (gets auto-populated based on Type selection)
    - Port Range (can specify for custom rules)
    - Source / Destination
    - Allow / Deny
  + Cannot specify an AWS resource as source / destination



* Security Group
  + It allows to specify inbound rules. All the inbound traffic gets automatically allowed to leave the instance
  + Automatically assumes all rules as ALLOW
  + Rules are combined and the combined decision decides what to do with the traffic. If multiple rules match the most specific one is chosen
  + Can specify following parameters : (almost SAME AS NACLs)
    - Type (HTTP, HTTPS, POP3, SSH, TCP, UDP, telnet, SMTP etc)
    - Protocol (gets auto-populated based on Type selection)
    - Port Range (can specify for custom rules)
    - Source / Destination
  + Can specify an AWS resource as source / destination

