AWS Networking Infrastructure Using Powershell

By: Salma Salah
Guidance & Support: Eng Saad El-Kennawy

Task Scenario

Create a Networking Architecture containing:

- A VPC with CIDR entered by user and Tags entered by user
- Creating Internet Gateway
- With the IGW attached to the VPV
- Also Tags are Entered by the User
- Subnets With Number of Subnets and CIDR of each Subnet entered by the User
- Also Tags are Entered by the User
- Creating a route table with each subnet created subnet and attach it to the Subnet
- Creating a route with each subnet created subnet

Task Scenario

Create a Networking Architecture containing:

- Creating NAT Gateway in a specified public subnet
- With Alerting the user that creation of a NAT GW will need an Elastic IP allocated first which will add additional cost
- Displaying NAT GW Details

Part 01

1. Adding Credentials and setting the session

2.Creating VPC with Tags

- With VPC CIDR is Entered By the User
- Also Tags are Entered by the User

First: Adding Credentials and setting the session

```
#-----------------------------------------------------------------------------------------
#Author: Salma Salah
#Date:   18.12.2023
#Project:Creating VPC, Subnets, Internet Gateway
#-----------------------------------------------------------------------------------------
#Importing modules
#-----------------------------------------------------------------------------------------
Get-Module -ListAvailable
Import-Module AWSPowerShell -Verbose
#-----------------------------------------------------------------------------------------
#AWS Credentials
$UserSecretKey  = "add your secret key"
$UserAccessKey = "add your access key"
$ProfileName  = "salma"
$region = "us-east-1"
#-----------------------------------------------------------------------------------------
#Setting Credentials
#-----------------------------------------------------------------------------------------
$SetCredentials = Set-AWSCredential -AccessKey $UserAccessKey -SecretKey $UserSecretKey -StoreAs $ProfileName
#-----------------------------------------------------------------------------------------
#Setting Sessions
#-----------------------------------------------------------------------------------------
$session = Initialize-AWSDefaults -ProfileName $ProfileName  -Region $region


#-----------------------------------------------------------------------------------------
#creating VPC
#-----------------------------------------------------------------------------------------
write-host "Welcome to AWS Powershell!!!!!"
write-host "Do you want to create a VPC ?"
```

```
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
```

Second: Creating VPC with Tags (1)

```
#-----------------------------------------------------------------------------------------
#creating VPC
#-----------------------------------------------------------------------------------------
write-host "Welcome to AWS Powershell!!!!!"
write-host "Do you want to create a VPC ?"
$VPCcreation = read-host
$VPCchoice = $VPCcreation
if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice-eq 'y') -or ($VPCchoice -eq 'Y') )
    {
        $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
        $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
        $vpcID= $vpcInformation.VpcId
        write-host "Your VPC has SUCCSEFULLY been create !!!!"
    }
#-----------------------------------------------------------------------------------------
#Adding tags
#-----------------------------------------------------------------------------------------
#Asking user if want to add tags to the VPC
#-----------------------------------------------------------------------------------------
write-host "Do you want to add tag to your VPC "
    $VPCtags = read-host
    $choice = $VPCtags
    if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice-eq 'y') -or ($choice -eq 'Y') )
    {
        do
        {
            write-host "please enter name of the tag to your VPC"
            $VPCtagname = read-host
            $VPCtag = $VPCtagname
            write-host "please enter value for the tag to your VPC"
```

```
PS D:\GBG\workspace>
PS D:\GBG\workspace>
```

Second: Creating VPC with Tags (2)

```
#-----------------------------------------------------------------------------------------
#Asking user if want to add tags to the VPC
#-----------------------------------------------------------------------------------------
write-host "Do you want to add tag to your VPC "
    $VPCtags = read-host
    $choice = $VPCtags
    if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice-eq 'y') -or ($choice -eq 'Y') )
    {
        do
        {
            write-host "please enter name of the tag to your VPC"
            $VPCtagname = read-host
            $VPCtag = $VPCtagname
            write-host "please enter value for the tag to your VPC"
            $VPCtagvalue = read-host
            $VPCtagCONTENT= $VPCtagvalue
            $tag = New-Object Amazon.EC2.Model.Tag
            $tag.Key = $VPCtag
            $tag.Value = $VPCtagCONTENT
            New-EC2Tag -Resource $vpcID -Tag $tag
            write-host "Do you want to add another tag to your VPC ?"
            $VPCanothertag= read-host

        }
        while
        (
         ( $VPCanothertag -eq "yes" -or ( $VPCanothertag -eq 'Yes') -or ( $VPCanothertag -eq 'y') -or ( $VPCanothertag -eq 'Y') )
        )
    }
write-host "The VPC has SUCCSEFULLY been created with Tags "
```

```
PS D:\GBG\workspace>
PS D:\GBG\workspace>
```

Third: The user Enters VPC CIDR

```
#-----------------------------------------------------------------------------------------
#creating VPC
#-----------------------------------------------------------------------------------------
write-host "Welcome to AWS Powershell!!!!!"
write-host "Do you want to create a VPC ?"
$VPCcreation = read-host
$VPCchoice = $VPCcreation
if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice-eq 'y') -or ($VPCchoice -eq 'Y') )
    {
        $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
        $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
        $vpcID= $vpcInformation.VpcId
        write-host "Your VPC has SUCCSEFULLY been create !!!!"
    }

#-----------------------------------------------------------------------------------------
```

```
PS D:\GBG\workspace> $VPCcreation = read-host
y
PS D:\GBG\workspace> $VPCchoice = $VPCcreation
PS D:\GBG\workspace> if ($VPCchoice -eq "yes" -or ($VPCchoice -eq 'Yes') -or ($VPCchoice-eq 'y') -or ($VPCchoice -eq 'Y') )
>>     {
>>         $vpccidr = read-host "Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !"
>>         $vpcInformation= New-EC2Vpc -CidrBlock $vpccidr -InstanceTenancy default -Region $region
>>         $vpcID= $vpcInformation.VpcId
>>         write-host "Your VPC has SUCCSEFULLY been create !!!!"
>>     }
Enter a valid cidr block for your VPC ex: 10.0.0.0/16 !: 200.0.0.0/16
Your VPC has SUCCSEFULLY been create !!!!
PS D:\GBG\workspace>
```

Third: The user Enters VPC Tags (1)

```
PS D:\GBG\workspace> write-host "Do you want to add tag to your VPC "
Do you want to add tag to your VPC
PS D:\GBG\workspace>     $VPCtags = read-host
y
PS D:\GBG\workspace>     $choice = $VPCtags
PS D:\GBG\workspace> if ($choice -eq "yes" -or ($choice -eq 'Yes') -or ($choice-eq 'y') -or ($choice -eq 'Y') )
>>     {
>>         do
>>         {
>>             write-host "please enter name of the tag to your VPC"
>>             $VPCtagname = read-host
>>             $VPCtag = $VPCtagname
>>             write-host "please enter value for the tag to your VPC"
>>             $VPCtagvalue = read-host
>>             $VPCtagCONTENT= $VPCtagvalue
>>             $tag = New-Object Amazon.EC2.Model.Tag
>>             $tag.Key = $VPCtag
>>             $tag.Value = $VPCtagCONTENT
>>             New-EC2Tag -Resource $vpcID -Tag $tag
>>             write-host "Do you want to add another tag to your VPC ?"
>>             $VPCanothertag= read-host
>>
>>         }
>>         while
>>         (
>>           ( $VPCanothertag -eq "yes" -or ( $VPCanothertag -eq 'Yes') -or ( $VPCanothertag -eq 'y') -or ( $VPCanothertag -eq 'Y') )
>>         )
>>
>>     }
please enter name of the tag to your VPC
vpc powershell
please enter value for the tag to your VPC
first ps vpc
Do you want to add another tag to your VPC ?
```

Third: The user Enters VPC Tags (2)

```
60          write-host "Do you want to add another tag to your VPC ?"
```

```
>>          ( $VPCanothertag -eq "yes" -or ( $VPCanothertag -eq 'Yes') -or ( $VPCanothertag -eq 'y') -or ( $VPCanothertag -eq 'Y') )
>>        )
>>
>>    }
please enter name of the tag to your VPC
vpc powershell
please enter value for the tag to your VPC
first ps vpc
Do you want to add another tag to your VPC ?
yes
please enter name of the tag to your VPC
cidr
please enter value for the tag to your VPC
200.0.0.0/16
Do you want to add another tag to your VPC ?
y
please enter name of the tag to your VPC
date
please enter value for the tag to your VPC
20.12.3023
Do you want to add another tag to your VPC ?
y
please enter name of the tag to your VPC
created by
please enter value for the tag to your VPC
Salma Salah
Do you want to add another tag to your VPC ?
n
PS D:\GBG\workspace>
PS D:\GBG\workspace> write-host "The VPC has SUCCSEFULLY been created with Tags "
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace>
```

Fourth: VPC After Creation with Tags

```
238
239   #GET A SPACIFIC ATTRIBUTE-------------------------------
240   Get-EC2VpcAttribute -VpcId  vpc-05f1a38a6eeba4378 -Attribute enableDnsHostnames
241   Get-EC2VpcAttribute -VpcId  vpc-05f1a38a6eeba4378 -Attribute VpcId
```

```
CidrBlock                   : 200.0.0.0/16
CidrBlockAssociationSet     : {vpc-cidr-assoc-092eb563779fab5c1}
DhcpOptionsId               : dopt-07a57c47179ae8793
InstanceTenancy             : default
Ipv6CidrBlockAssociationSet : {}
IsDefault                   : False
OwnerId                     : 821594020462
State                       : available
Tags                        : {created by, vpc powershell, date, cidr}
VpcId                       : vpc-0e2bc2e5594d2a579

PS D:\GBG\workspace>
```

Part 02

Creating Internet Gateway

- With the IGW attached to the VPV
- Also Tags are Entered by the User

First: Crating Internet Gateway With Tags

```
#-----------------------------------------------------------------------------------------
#creating InternetGateway
#-----------------------------------------------------------------------------------------
$Internetgatewayinfo= New-EC2InternetGateway
$Internetgatewayinfo
$IGWID= $Internetgatewayinfo.InternetGatewayId
#-----------------------------------------------------------------------------------------
#Adding TAGS to the Internet Gateway
#-----------------------------------------------------------------------------------------
write-host "Do you want to add tag to the Internet Gateway ? "
$IGWtags = read-host
$IGWchoice = $IGWtags
if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
{
    do
    {
        write-host "please enter name of the tage"
        $IGWtagname = read-host
        $IGWtag = $IGWtagname
        write-host "please enter value for the tage"
        $IGWtagvalue = read-host
        $IGWcontent= $IGWtagvalue
        $tag = New-Object Amazon.EC2.Model.Tag
        $tag.Key = $IGWtag
        $tag.Value = $IGWcontent
        New-EC2Tag -Resource $IGWID -Tag $tag
        write-host "Do you want to add another tag to the Internet Gateway ?"
        $IGWanothertag= read-host

    }
```

```
>>          $VPCtag = $VPCtagname
>>          write-host "please enter value for the tag to your VPC"
```

Second: Attaching Internet Gateway to the VPC

```
    }
    while
    (
      ( $IGWanothertag -eq "yes" -or ( $IGWanothertag -eq 'Yes') -or ( $IGWanothertag -eq 'y') -or ( $IGWanothertag -eq 'Y') )
    )
}
#-----------------------------------------------------------------------------------
#Attaching Internet Gateway to the VPC
#-----------------------------------------------------------------------------------
Add-EC2InternetGateway -InternetGatewayId $IGWID  -VpcId $vpcID
Get-EC2InternetGateway
#-----------------------------------------------------------------------------------
```

Third: Crating Internet Gateway With Tags (During Run(1))

```
{
    $privatesubnetcidr = read-host "Enter your subnet cidr !"
    $privatesubnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId  -CidrBlock $subnetcidr
    $privatesubnetid = $privatesubnetInformation.SubnetId
}

#-----------------------------------------------------------------------------------------
#creating InternetGateway
#-----------------------------------------------------------------------------------------
$Internetgatewayinfo= New-EC2InternetGateway
$Internetgatewayinfo
$IGWID= $Internetgatewayinfo.InternetGatewayId
#-----------------------------------------------------------------------------------------
```

```
PS D:\GBG\workspace> $Internetgatewayinfo= New-EC2InternetGateway
PS D:\GBG\workspace> $Internetgatewayinfo

Attachments InternetGatewayId     OwnerId      Tags
----------- ----------------     -------      ----
{}          igw-0d1b681dff91fe07a 821594020462 {}

PS D:\GBG\workspace> $IGWID= $Internetgatewayinfo.InternetGatewayId
PS D:\GBG\workspace> write-host "Do you want to add tag to the Internet Gateway ? "
Do you want to add tag to the Internet Gateway ?
PS D:\GBG\workspace> $IGWtags = read-host
GWtagname
PS D:\GBG\workspace> $IGWchoice = $IGWtags
PS D:\GBG\workspace> if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
>> {
>>     do
>>     {
>>         write-host "please enter name of the tage"
>>         $IGWtagname = read-host
>>         $IGWtag = $I        write-host "please enter value for the tage"
```

Third: Crating Internet Gateway With Tags (During Run (2))

```
PS D:\GBG\workspace> write-host "Do you want to add tag to the Internet Gateway ? "
Do you want to add tag to the Internet Gateway ?
PS D:\GBG\workspace> $IGWtags = read-host
y
PS D:\GBG\workspace> $IGWchoice = $IGWtags
PS D:\GBG\workspace> $IGWchoice
y
PS D:\GBG\workspace> if ($IGWchoice -eq "yes" -or ($IGWchoice -eq 'Yes') -or ($IGWchoice -eq 'y') -or ($IGWchoice -eq 'Y') )
>> {
>>     do
>>     {
>>         write-host "please enter name of the tage"
>>         $IGWtagname = read-host
>>         $IGWtag = $IGWtagname
>>         write-host "please enter value for the tage"
>>         $IGWtagvalue = read-host
>>         $IGWcontent= $IGWtagvalue
>>         $tag = New-Object Amazon.EC2.Model.Tag
>>         $tag.Key = $IGWtag
>>         $tag.Value = $IGWcontent
>>         New-EC2Tag -Resource $IGWID -Tag $tag
>>         write-host "Do you want to add another tag to the Internet Gateway ?"
>>         $IGWanothertag= read-host
>>
>>     }
>>     while
>>     (
>>       ( $IGWanothertag -eq "yes" -or ( $IGWanothertag -eq 'Yes') -or ( $IGWanothertag -eq 'y') -or ( $IGWanothertag -eq 'Y') )
>>     )
>>
>> }
please enter name of the tage
```

Fourth: User Enters the Tags

```
154 {
155     do
156     {
```

```
>>         $IGWtag = $IGWtagname
>>         write-host "please enter value for the tage"
>>         $IGWtagvalue = read-host
>>         $IGWcontent= $IGWtagvalue
>>         $tag = New-Object Amazon.EC2.Model.Tag
>>         $tag.Key = $IGWtag
>>         $tag.Value = $IGWcontent
>>         New-EC2Tag -Resource $IGWID -Tag $tag
>>         write-host "Do you want to add another tag to the Internet Gateway ?"
>>         $IGWanothertag= read-host
>>
>>     }
>>     while
>>     (
>>       ( $IGWanothertag -eq "yes" -or ( $IGWanothertag -eq 'Yes') -or ( $IGWanothertag -eq 'y') -or ( $IGWanothertag -eq 'Y') )
>>     )
>>
>> }
please enter name of the tage
vpc igw
please enter value for the tage
this is by powershell
Do you want to add another tag to the Internet Gateway ?
yes
please enter name of the tage
date
please enter value for the tage
20.12.2023
Do you want to add another tag to the Internet Gateway ?
n
PS D:\GBG\workspace>
```

Fifth: Internet Gateway With Tags After Creation

```
}


#-----------------------------------------------------------------------------------------
#Attaching Internet Gateway to the VPC
#-----------------------------------------------------------------------------------------
Add-EC2InternetGateway -InternetGatewayId $IGWID  -VpcId $vpcID
Get-EC2InternetGateway
#-----------------------------------------------------------------------------------------
```

```
n
PS D:\GBG\workspace> $vpcID
vpc-0e2bc2e5594d2a579
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace> #Attaching Internet Gateway to the VPC
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace>  Add-EC2InternetGateway -InternetGatewayId $IGWID  -VpcId $vpcID
PS D:\GBG\workspace>  Get-EC2InternetGateway

Attachments            InternetGatewayId     OwnerId      Tags
-----------            -----------------     -------      ----
{vpc-07e368fb8901161b4} igw-0c347142903731a40 821594020462 {}
{vpc-0e2bc2e5594d2a579} igw-0d1b681dff91fe07a 821594020462 {vpc igw, date}

PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
```

Part 03

Creating Subnets with Tags

- With Number of Subnets and CIDR of each Subnet entered by the User
- Also Tags are Entered by the User

First: Get Number of subnets from the user and Two validation

1. That user Enters a Number
2. That this number is not 0

```
write-host "Now let's Create subnets"
#-----------------------------------------------------------------------------------
#creating public subnets with TAGS
#-----------------------------------------------------------------------------------
#To ensure that user enters a number not a charecter
#-----------------------------------------------------------------------------------
[int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
#-----------------------------------------------------------------------------------
#To ensure that user enters a number not 0
#-----------------------------------------------------------------------------------
if ($NOofpulicsubnets -lt 1)
  {
    do
    {
        [int]$NOofpulicsubnets= read-host "Please write a valied number "
    }
    while
      (
        $NOofpulicsubnets -lt 1
      )
  }
```

Second: During Getting Number of Subnets From User and Testing the Validation

```
121    write-host "Now let's Create subnets"
```

```
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
How many public subnets do you want to create ?: 0
PS D:\GBG\workspace> if ($NOofpulicsubnets -lt 1)
>>   {
>>     do
>>     {
>>         [int]$NOofpulicsubnets= read-host "Please write a valied number "
>>     }
>>      while
>>       (
>>         $NOofpulicsubnets -lt 1
>>       )
>>   }
Please write a valied number : 0
Please write a valied number : 0
Please write a valied number : 00
Please write a valied number : 3
PS D:\GBG\workspace>
```

Third: Creating Subnets according to the number Entered by user and creating tags Entered by user and asking user if want to add more tags

```
#-------------------------------------------------------------------------------
#For loop to create subnets according to number enterd by user get CIDR from user
#Ask if user want to add tags
#Do-while loop to create another tag if user wants
#-------------------------------------------------------------------------------
for($counter=1; $counter -le $NOofpulicsubnets; $counter ++)
{
    $subnetcidr = read-host "Enter your subnet cidr !"
    $subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId  -CidrBlock $subnetcidr
    #to enable auto-assign public IPv4
    $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
    $subnetid = $subnetInformation.SubnetId
    write-host "Do you want to add tag to the subnet "
    $subnettags = read-host
    $answer = $subnettags
    if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer-eq 'y') -or ($answer -eq 'Y') )
    {
        do
        {
            write-host "please enter name of the tage"
            $subnettagname = read-host
            $tagname = $subnettagname
            write-host "please enter value for the tage"
            $subnettagvalue = read-host
            $tagheder= $subnettagvalue
            $tag = New-Object Amazon.EC2.Model.Tag
            $tag.Key = $tagname
            $tag.Value = $tagheder
            New-EC2Tag -Resource $subnetid -Tag $tag
            write-host "Do you want to add another tag ?"
```

```
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
```

Third: Creating Subnets according to the number Entered by user and creating tags Entered by user and asking user if want to add more tags

```
$subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId  -CidrBlock $subnetCidr
#to enable auto-assign public IPv4
$publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
$subnetid = $subnetInformation.SubnetId
write-host "Do you want to add tag to the subnet "
$subnettags = read-host
$answer = $subnettags
if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer-eq 'y') -or ($answer -eq 'Y') )
{
    do
    {
        write-host "please enter name of the tage"
        $subnettagname = read-host
        $tagname = $subnettagname
        write-host "please enter value for the tage"
        $subnettagvalue = read-host
        $tagheder= $subnettagvalue
        $tag = New-Object Amazon.EC2.Model.Tag
        $tag.Key = $tagname
        $tag.Value = $tagheder
        New-EC2Tag -Resource $subnetid -Tag $tag
        write-host "Do you want to add another tag ?"
        $anothertag= read-host

    }
    while
    (
      ( $anothertag -eq "yes" -or ( $anothertag -eq 'Yes') -or ( $anothertag -eq 'y') -or ( $anothertag -eq 'Y') )
    )
}
```

```
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
```

Part 04

Creating a route table and a route with each subnet created subnet

- In the same For loop that has loops equal number of subnets entered by user after the subnet is created a route table will be created and associated to this subnet
- Also a route will be created that has internet gateway to make this subnet internet accessible

First: Creating route table associated for each subnet and a route

```
    }
    #-----------------------------------------------------------------------------------------
    #creating Routetables
    #Note when a new subnet is created a new route table should be created and assosiated to this subnet
    #-----------------------------------------------------------------------------------------------
    $routetableinfo= New-EC2RouteTable -VpcId $vpcInformation.VpcId
    $routetableID= $routetableinfo.RouteTableId

    #-----------------------------------------------------------------------------------------------
    #Assosiating Routetables each routetable to a specific subnet
    #-----------------------------------------------------------------------------------------------
    Register-EC2RouteTable -RouteTableId $routetableID -SubnetId  $subnetid
    write-host "The Route table has succesfully been created and assosiated to the subnet"
    write-host "Now a route will be created that has IGW so your subnet will be internet accessable"


    #-----------------------------------------------------------------------------------------------
    #creating a Route in each route table for each public subnet to be able to acess the Internet
    #-----------------------------------------------------------------------------------------------
    $routeinformation= New-EC2Route -RouteTableId $routetableID  -DestinationCidrBlock 0.0.0.0/0 -GatewayId $IGWID
    write-host "Congrats!!!! You have successfuly created subnet with route table assosiated and a route"
}
```

```
The VPC has SUCCSEFULLY been created with Tags
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
```

Second: Creating Subnets route tables and routes (During Run)

```
#-----------------------------------------------------------------------------------------
write-host "Now let's Create subnets"
#-----------------------------------------------------------------------------------------
#creating public subnets with TAGS
#-----------------------------------------------------------------------------------------
#To ensure that user enters a number not a charecter
#-----------------------------------------------------------------------------------------
[int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
#-----------------------------------------------------------------------------------------
```

```
Enter your subnet cidr !:
PS D:\GBG\workspace> write-host "Now let's Create subnets"
Now let's Create subnets
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace> #creating public subnets with TAGS
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace> #To ensure that user enters a number not a charecter
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------------
PS D:\GBG\workspace> [int]$NOofpulicsubnets= read-host "How many public subnets do you want to create ?"
How many public subnets do you want to create ?: 3
PS D:\GBG\workspace> for($counter=1; $counter -le $NOofpulicsubnets; $counter ++)
>> {
>>     $subnetcidr = read-host "Enter your subnet cidr !"
>>     $subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId  -CidrBlock $subnetcidr
>>     #to enable auto-assign public IPv4
>>     $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
>>     $subnetid = $subnetInformation.SubnetId
>>     write-host "Do you want to add tag to the subnet "
>>     $subnettags = read-host
>>     $answer = $subnettags
>>     if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer-eq 'y') -or ($answer -eq 'Y') )
>>     {
>>         do
>>         {
>>             write-host "please enter name of the tage"
```

Second: Creating Subnets route tables and routes (During Run)

```
PS D:\GBG\workspace> for($counter=1; $counter -le $NOofpulicsubnets; $counter ++)
>> {
>>     $subnetcidr = read-host "Enter your subnet cidr !"
>>     $subnetInformation= New-EC2Subnet -VpcId $vpcInformation.VpcId  -CidrBlock $subnetcidr
>>     #to enable auto-assign public IPv4
>>     $publicipstate= $subnetInformation.MapPublicIpOnLaunch= "True"
>>     $subnetid = $subnetInformation.SubnetId
>>     write-host "Do you want to add tag to the subnet "
>>     $subnettags = read-host
>>     $answer = $subnettags
>>     if ($answer -eq "yes" -or ($answer -eq 'Yes') -or ($answer-eq 'y') -or ($answer -eq 'Y') )
>>     {
>>         do
>>         {
>>             write-host "please enter name of the tage"
>>             $subnettagname = read-host
>>             $tagname = $subnettagname
>>             write-host "please enter value for the tage"
>>             $subnettagvalue = read-host
>>             $tagheder= $subnettagvalue
>>             $tag = New-Object Amazon.EC2.Model.Tag
>>             $tag.Key = $tagname
>>             $tag.Value = $tagheder
>>             New-EC2Tag -Resource $subnetid -Tag $tag
>>             write-host "Do you want to add another tag ?"
>>             $anothertag= read-host
>>
>>         }
>>         while
>>         (
>>           ( $anothertag -eq "yes" -or ( $anothertag -eq 'Yes') -or ( $anothertag -eq 'y') -or ( $anothertag -eq 'Y') )
>>         )
>>     }
>>     #-----------------------------------------------------------------------------------
>>     #creating Routetables
>>     #Note when a new subnet is created a new route table should be created and assosiated to this subnet
>>     #-------------------------------------------------------------------------------------------
>>     $routetableinfo= New-EC2RouteTable -VpcId $vpcInformation.VpcId
>>     $routetableID= $routetableinfo.RouteTableId
```

Third: User Enters CIDR and Tags of each subnet and Route table and route created using the created subnet ID

```
>>
>>      #-----------------------------------------------------------------------------------------
>>      #Assosiating Routetables each routetable to a specific subnet
>>      #-----------------------------------------------------------------------------------------
>>      Register-EC2RouteTable -RouteTableId $routetableID -SubnetId  $subnetid
>>      write-host "The Route table has succesfully been created and assosiated to the subnet"
>>      write-host "Now a route will be created that has IGW so your subnet will be internet accessable"
>>
>>
>>      #-----------------------------------------------------------------------------------------
>>      #creating a Route in each route table for each public subnet to be able to acess the Internet
>>      #-----------------------------------------------------------------------------------------
>>      $routeinformation= New-EC2Route -RouteTableId $routetableID  -DestinationCidrBlock 0.0.0.0/0 -GatewayId $IGWID
>>      write-host "Congrats!!!! You have successfuly created subnet with route table assosiated and a route"
>> }
Enter your subnet cidr !: 200.0.20.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
please enter value for the tage
first subnet
Do you want to add another tag ?
y
please enter name of the tage
cidr
please enter value for the tage
200.0.20.0/24
Do you want to add another tag ?
n
rtbassoc-01fd1c319439df2bd
The Route table has succesfully been created and assosiated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfuly created subnet with route table assosiated and a route
Enter your subnet cidr !: 200.0.30.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
```

Third: User Enters CIDR and Tags of each subnet and Route table and route created using the created subnet ID

```
Enter your subnet cidr !: 200.0.30.0/24
Do you want to add tag to the subnet
y
please enter name of the tage
name
please enter value for the tage
second subnet
Do you want to add another tag ?
yes
please enter name of the tage
cidr
please enter value for the tage
200.0.30.0/24
Do you want to add another tag ?
n
rtbassoc-029511b38c14ccddf
The Route table has succesfully been created and assosiated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfuly created subnet with route table assosiated and a route
Enter your subnet cidr !: 200.0.40.0/24
Do you want to add tag to the subnet
Yes
please enter name of the tage
name
please enter value for the tage
third subnet
Do you want to add another tag ?
y
please enter name of the tage
cidr
please enter value for the tage
200.0.30.0/24
Do you want to add another tag ?
n
rtbassoc-0a1eefade1b073638
The Route table has succesfully been created and assosiated to the subnet
Now a route will be created that has IGW so your subnet will be internet accessable
Congrats!!!! You have successfuly created subnet with route table assosiated and a route
PS D:\GBG\workspace>
```

Part 05

Creating NAT Gateway in a specified public subnet

- With Alerting the user that creation of a NAT GW will need an Elastic IP allocated first which will add additional cost
- Displaying NAT GW Details

First: alerting user then allocation of elastic IP & NAT GW creation

File Edit Selection View Go Run ··· workspace

test.ps1

salma > test.ps1

```
223
224 #-----------------------------------------------------------------------------
225 #Asking user if want to create a NatGateway
226 #---------------------------------------------------------------------------
227 write-host "Do you want to create a Nat Gateway? "
228 $NatGW= read-host
229 $Natchoice= $NatGW
230 if ($Natchoice -eq "yes" -or ($Natchoice-eq 'Yes') -or ($Natchoice-eq 'y') -or ($Natchoice-eq 'Y') )
231     {
232
233     }
234     write-host "Take into consedtartion that creating NATGW will need an elastic IP to be allocated which have an extra cost!!! "
235 #-------------------------------------------------------------------------------
236 #Alloctaing an Elastic Ip to be used by the Nat Gateway
237 #-------------------------------------------------------------------------------
238 $ElasticIpInfo= New-EC2Address -Domain Vpc
239 $AllocationID= $ElasticIpInfo.AllocationId
240 #-------------------------------------------------------------------------------
241 #creating NATGateway
242 #-------------------------------------------------------------------------------
243 $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
244 $Natgatewayinfo
245 Get-EC2NatGateway
246
247
248
249
250
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS pwsh

```
CreateTime          : 12/21/2023 6:05:49 AM
DeleteTime          : 1/1/0001 12:00:00 AM
FailureCode         :
FailureMessage      :
```

⊗ 0 ⚠ 0 0 Ln 239, Col 1 Spaces: 4 UTF-8 CRLF PowerShell

Second: Asking user if want to create a NAT GW

```
223
224 #-----------------------------------------------------------------------------
225 #Asking user if want to create a NatGateway
226 #---------------------------------------------------------------------------
227 write-host "Do you want to create a Nat Gateway? "
228 $NatGW= read-host
229 $Natchoice= $NatGW
230 if ($Natchoice -eq "yes" -or ($Natchoice-eq 'Yes') -or ($Natchoice-eq 'y') -or ($Natchoice-eq 'Y') )
231     {
232         write-host "Take into consedtartion that creating NATGW will need an elastic IP to be allocated which have an extra cost!!! " }
233         #-------------------------------------------------------------------------------------
234         #Alloctaing an Elastic Ip to be used by the Nat Gateway
```

```
PS D:\GBG\workspace> write-host "Do you want to create a Nat Gateway? "
Do you want to create a Nat Gateway?
PS D:\GBG\workspace> $NatGW= read-host
y
PS D:\GBG\workspace> $Natchoice= $NatGW
PS D:\GBG\workspace> if ($Natchoice -eq "yes" -or ($Natchoice-eq 'Yes') -or ($Natchoice-eq 'y') -or ($Natchoice-eq 'Y') )
>>     {
>>         write-host "Take into consedtartion that creating NATGW will need an elastic IP to be allocated which have an extra cost!!! " }
Take into consedtartion that creating NATGW will need an elastic IP to be allocated which have an extra cost!!!
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace>
```

Third: Elastic IP after allocation

```
234     }
235 #-----------------------------------------------------------------------------------
236 #Alloctaing an Elastic Ip to be used by the Nat Gateway
237 #-----------------------------------------------------------------------------------
238 $ElasticIpInfo= New-EC2Address -Domain Vpc
239 $AllocationID= $ElasticIpInfo.AllocationId
240
241
242
243
244
245
```

```
PS D:\GBG\workspace> $ElasticIpInfo= New-EC2Address -Domain Vpc
PS D:\GBG\workspace> $ElasticIpInfo

AllocationId          : eipalloc-0c77f616f994966c4
CarrierIp             :
CustomerOwnedIp       :
CustomerOwnedIpv4Pool :
Domain                : vpc
NetworkBorderGroup    : us-east-1
PublicIp              : 44.194.149.42
PublicIpv4Pool        : amazon

PS D:\GBG\workspace> $AllocationID= $ElasticIpInfo.AllocationId
PS D:\GBG\workspace> $AllocationID
eipalloc-0c77f616f994966c4
PS D:\GBG\workspace>
```

Fourth: NAT GW after creation

File Edit Selection View Go Run ··· workspace

test.ps1

salma > test.ps1

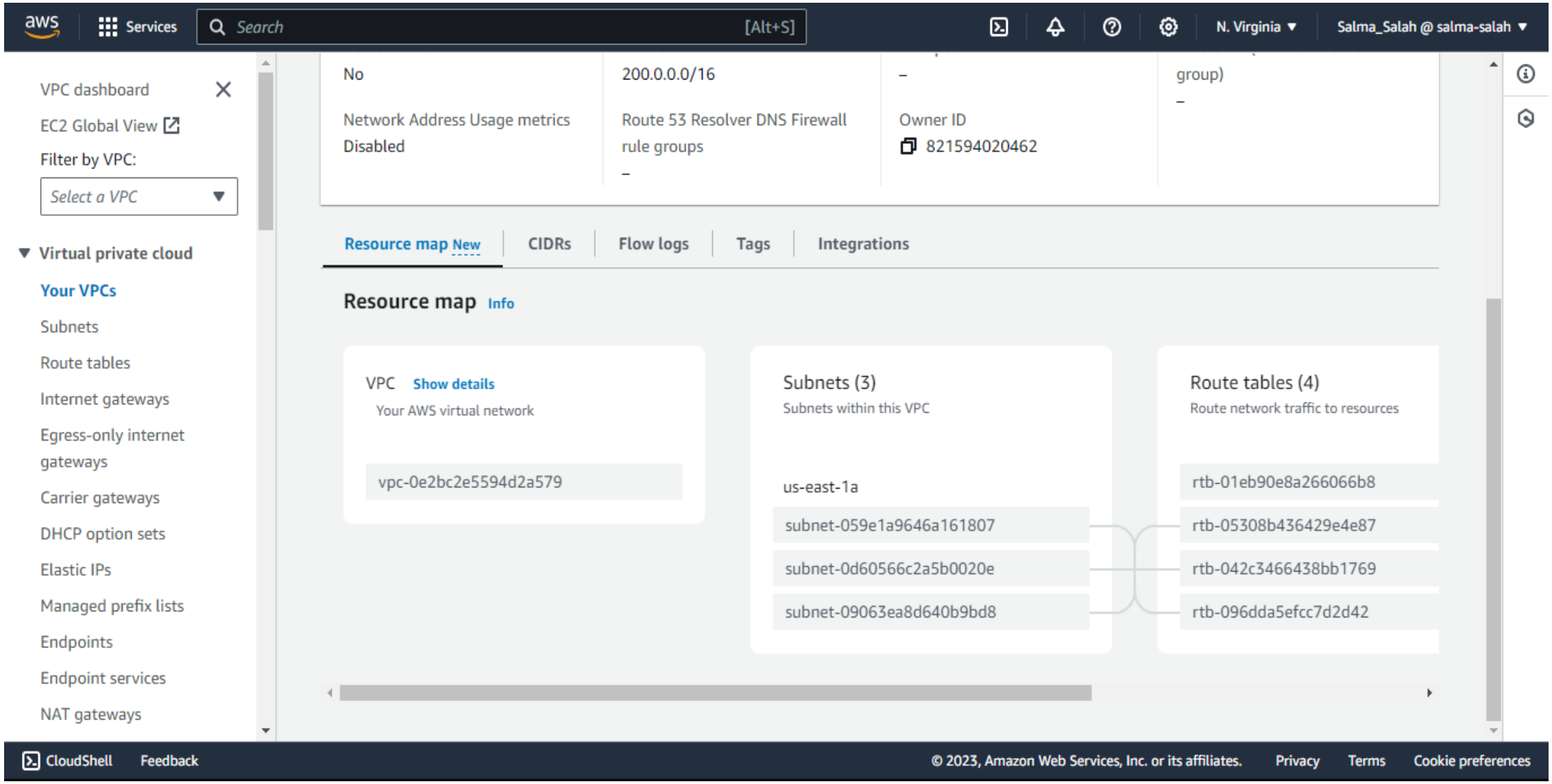
```
238     }
239
240 #-----------------------------------------------------------------------------------
241 #creating NATGateway
242 #-----------------------------------------------------------------------------------
243 $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
244 $Natgatewayinfo
245 Get-EC2NatGateway
246
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS pwsh

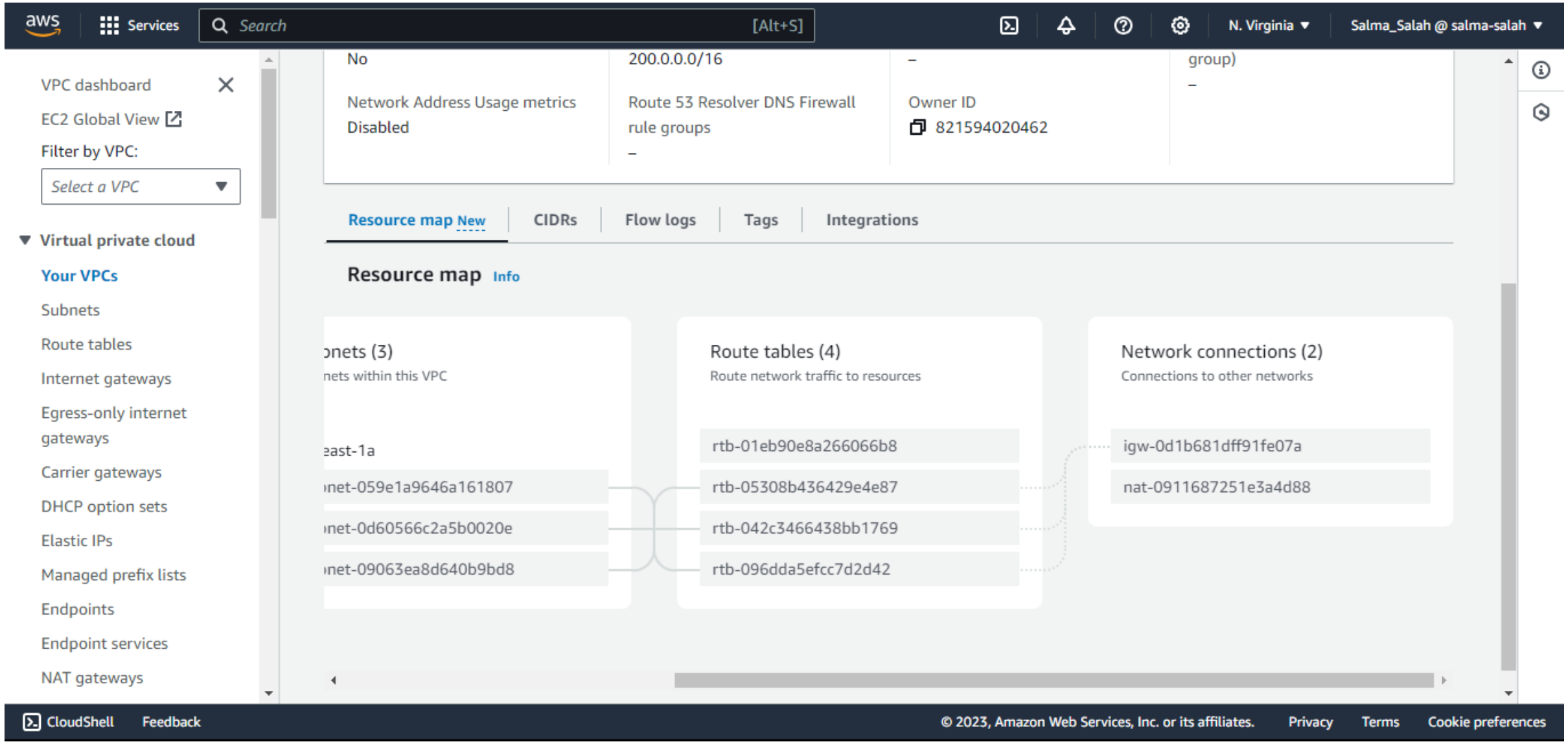
```
PS D:\GBG\workspace>
PS D:\GBG\workspace>
PS D:\GBG\workspace> #-----------------------------------------------------------------------------------
PS D:\GBG\workspace> $Natgatewayinfo= New-EC2NatGateway -SubnetId $subnetid -AllocationId $AllocationID
PS D:\GBG\workspace> $Natgatewayinfo

ClientToken                          NatGateway
-----------                          ----------
8735236a-a274-4b32-b36b-3c1e02404db2 Amazon.EC2.Model.NatGateway

PS D:\GBG\workspace> Get-EC2NatGateway

ConnectivityType     : public
CreateTime           : 12/21/2023 6:05:49 AM
DeleteTime           : 1/1/0001 12:00:00 AM
FailureCode          :
FailureMessage       :
NatGatewayAddresses  : {eipalloc-0c77f616f994966c4}
NatGatewayId         : nat-0911687251e3a4d88
ProvisionedBandwidth :
State                : available
SubnetId             : subnet-059e1a9646a161807
Tags                 : {}
VpcId                : vpc-0e2bc2e5594d2a579
```

0 0 0 Ln 247, Col 1 Spaces: 4 UTF-8 CRLF PowerShell

Finally: The Completed architecture



Finally: The Completed architecture



VPC with Tags

aws | Services | Search [Alt+S] | N. Virginia | Salma_Salah @ salma-salah

VPC dashboard
EC2 Global View
Filter by VPC:
Select a VPC

- Virtual private cloud
  - Your VPCs
  - Subnets
  - Route tables
  - Internet gateways
  - Egress-only internet gateways
  - Carrier gateways
  - DHCP option sets
  - Elastic IPs
  - Managed prefix lists
  - Endpoints
  - Endpoint services
  - NAT gateways

vpc-0e2bc2e5594d2a579 | Available | Disabled | Enabled

Tenancy Default	DHCP option set dopt-07a57c47179ae8793	Main route table rtb-01eb90e8a266066b8	Main network ACL acl-0b88e57abb177a285
Default VPC No	IPv4 CIDR 200.0.0.0/16	IPv6 pool –	IPv6 CIDR (Network border group) –
Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups –	Owner ID 821594020462	

Resource map New | CIDRs | Flow logs | Tags | Integrations

Tags

Manage tags

Search tags

1

Key	Value
created by	Salma Salah
vpc power...	first ps vpc
date	20.12.3023
cidr	200.0.0.0/16

CloudShell | Feedback | © 2023, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

nternet Gateway With Tags After Creation

aws | Services | Search [Alt+S] | N. Virginia | Salma_Salah @ salma-salah

VPC dashboard
EC2 Global View
Filter by VPC:
Select a VPC

Virtual private cloud
- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways

VPC > Internet gateways > igw-0d1b681dff91fe07a

igw-0d1b681dff91fe07a

Actions

Details Info

Internet gateway ID	State	VPC ID	Owner
igw-0d1b681dff91fe07a	Attached	vpc-0e2bc2e5594d2a579	821594020462

Tags

Manage tags

Search tags

1

Key	Value
vpc igw	this is by powershell
date	20.12.2023

CloudShell | Feedback | | Privacy | Terms | Cookie preferences

First subnet with Tags

aws | Services | Search [Alt+S] | N. Virginia | Salma_Salah @ salma-salah

VPC dashboard
EC2 Global View
Filter by VPC:
Select a VPC

- Virtual private cloud
  - Your VPCs
  - Subnets
  - Route tables
  - Internet gateways
  - Egress-only internet gateways
  - Carrier gateways
  - DHCP option sets
  - Elastic IPs
  - Managed prefix lists
  - Endpoints
  - Endpoint services
  - NAT gateways

Default subnet
No

Customer-owned IPv4 pool
–

IPv6-only
No

DNS64
Disabled

vpc-0e2bc2e5594d2a579

Auto-assign public IPv4 address
No

Outpost ID
–

Hostname type
IP name

Owner
821594020462

Auto-assign IPv6 address
No

IPv4 CIDR reservations
–

Resource name DNS A record
Disabled

Auto-assign customer-owned IPv4 address
No

IPv6 CIDR reservations
–

Resource name DNS AAAA record
Disabled

Flow logs | Route table | Network ACL | CIDR reservations | Sharing | Tags

Tags

Manage tags

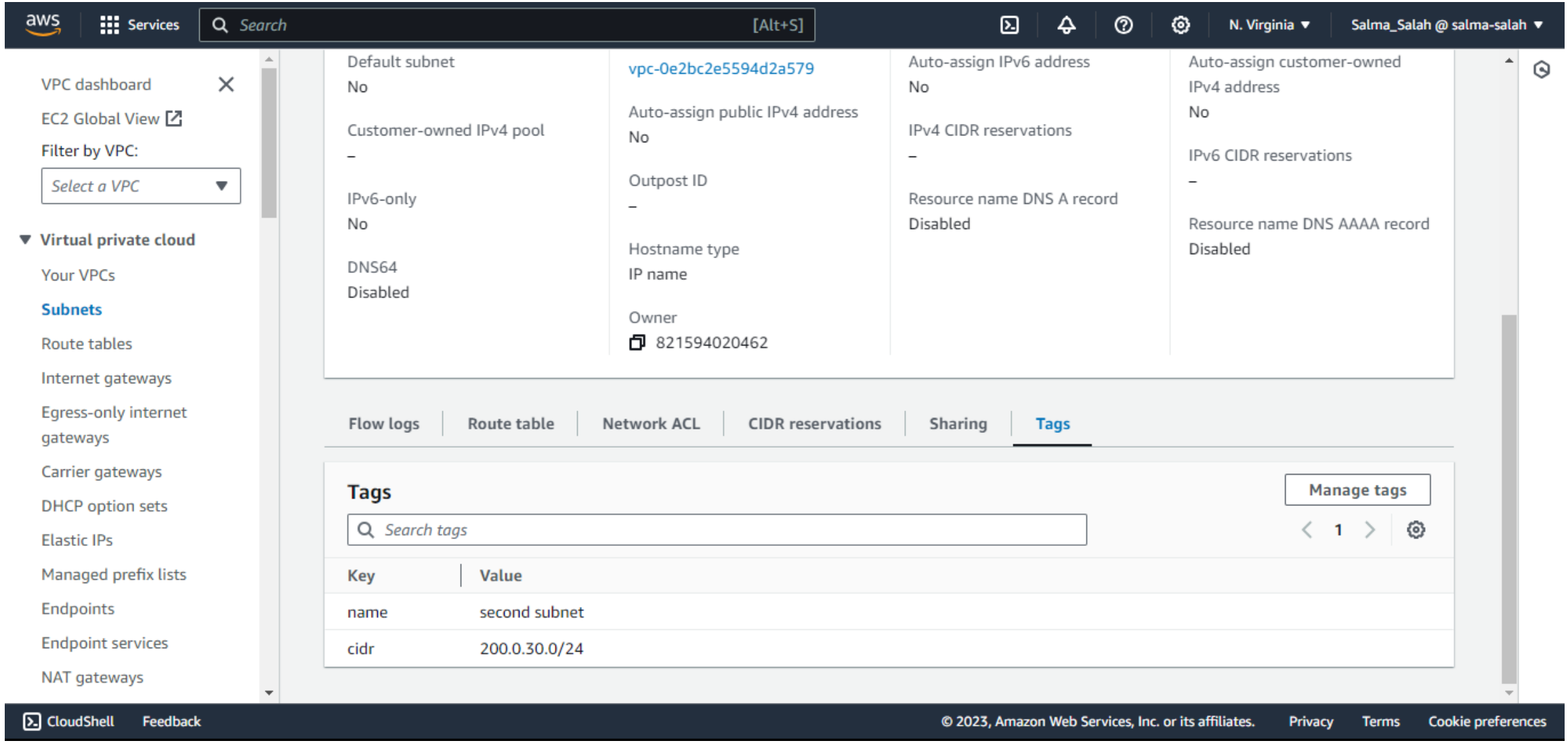
Search tags

1

Key	Value
cidr	200.0.20.0/24
name	first subnet

CloudShell | Feedback | © 2023, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

Second subnet with Tags



Third subnet with Tags

aws | Services | Search [Alt+S] | N. Virginia | Salma_Salah @ salma-salah

VPC dashboard
EC2 Global View
Filter by VPC:
Select a VPC

Virtual private cloud
- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways

Default subnet
No

Customer-owned IPv4 pool
–

IPv6-only
No

DNS64
Disabled

vpc-0e2bc2e5594d2a579

Auto-assign public IPv4 address
No

Outpost ID
–

Hostname type
IP name

Owner
821594020462

Auto-assign IPv6 address
No

IPv4 CIDR reservations
–

Resource name DNS A record
Disabled

Auto-assign customer-owned IPv4 address
No

IPv6 CIDR reservations
–

Resource name DNS AAAA record
Disabled

Flow logs | Route table | Network ACL | CIDR reservations | Sharing | Tags

Tags

Manage tags

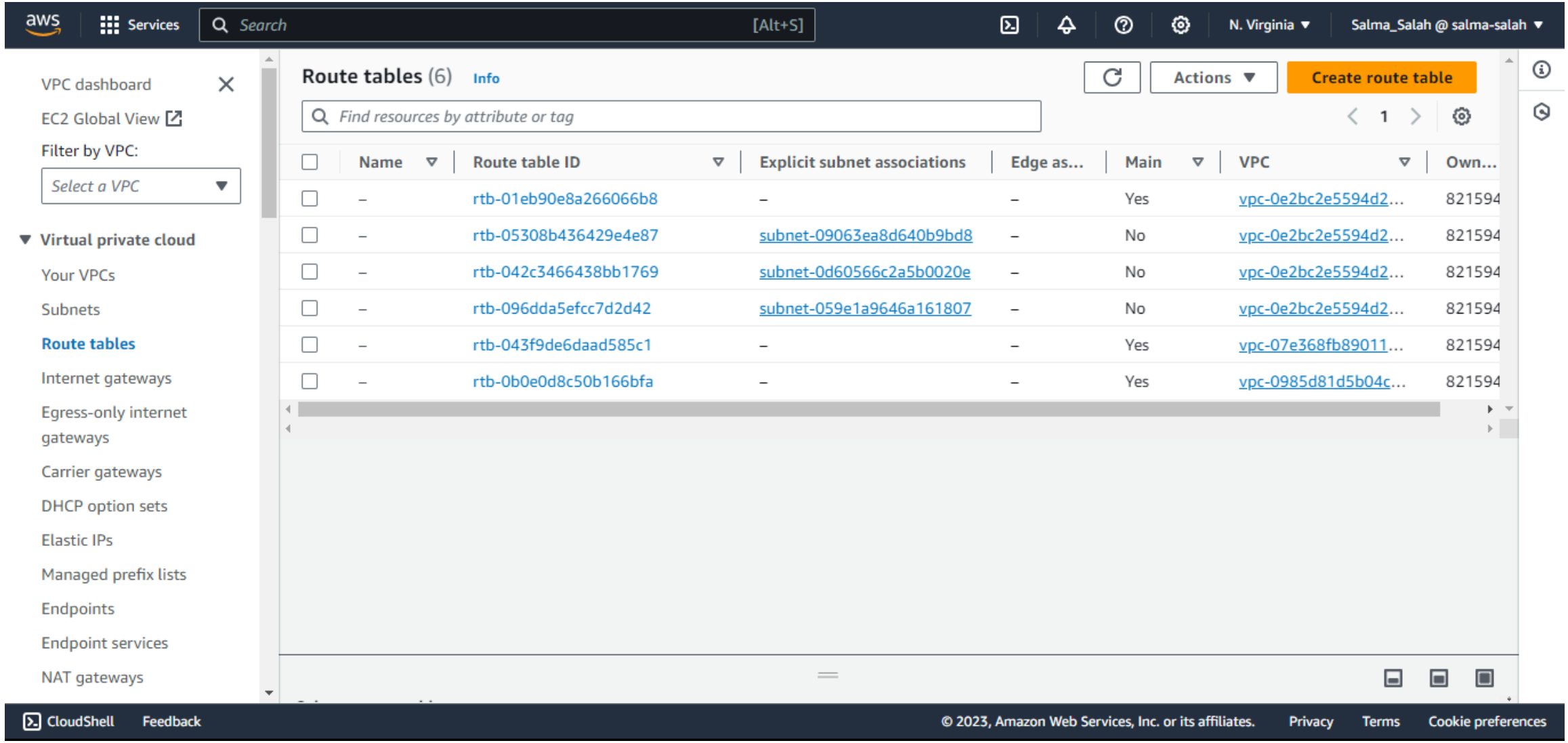
Search tags

1

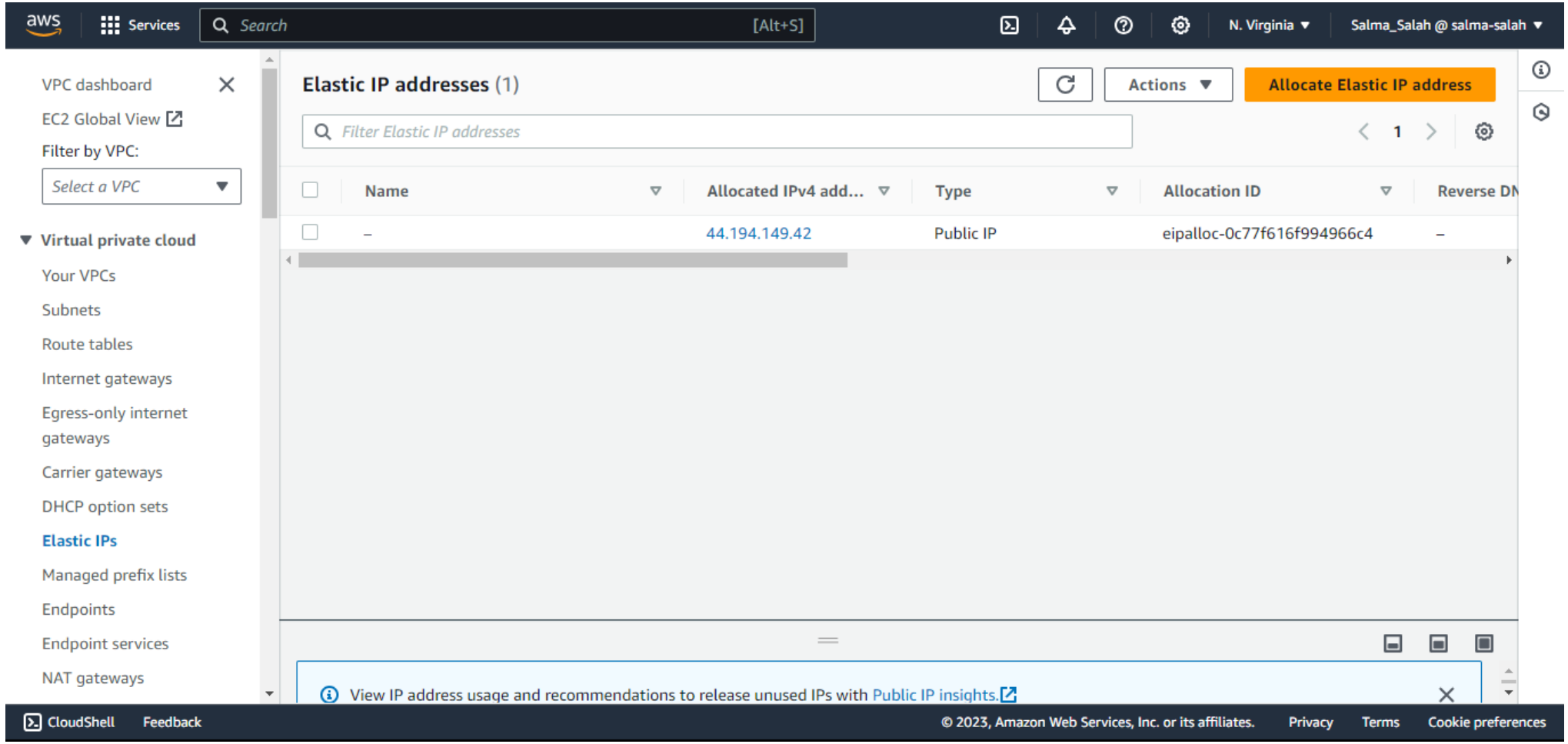
Key	Value
cidr	200.0.30.0/24
name	third subnet

CloudShell | Feedback | © 2023, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

The Three Route Tables Associated to three subnets



Elastic IP after allocation



NAT GW after creation

aws | Services | Search [Alt+S] | N. Virginia | Salma_Salah @ salma-salah

VPC dashboard
EC2 Global View
Filter by VPC:
Select a VPC

Virtual private cloud
- Your VPCs
- Subnets
- Route tables
- Internet gateways
- Egress-only internet gateways
- Carrier gateways
- DHCP option sets
- Elastic IPs
- Managed prefix lists
- Endpoints
- Endpoint services
- NAT gateways

VPC > NAT gateways > nat-0911687251e3a4d88

nat-0911687251e3a4d88

Actions

Details Info

NAT gateway ID nat-0911687251e3a4d88	Connectivity type Public	State Available	State message Info –
NAT gateway ARN arn:aws:ec2:us-east-1:821594020462:natgateway/nat-0911687251e3a4d88	Primary public IPv4 address 44.194.149.42	Primary private IPv4 address 200.0.40.194	Primary network interface ID eni-084972095396ebba7
VPC vpc-0e2bc2e5594d2a579	Subnet subnet-059e1a9646a161807	Created Thursday, December 21, 2023 at 06:05:49 GMT+2	Deleted –

Secondary IPv4 addresses | Monitoring | Tags

Secondary IPv4 addresses

Edit secondary IPv4 address associations

CloudShell | Feedback | © 2023, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

Attached below my github link with the source code

https://github.com/salmasalam024/Powershell/tree/code