SYST35144 Final Project

Luka Necajev And Owen Ross

Objective

The objective of the project is to validate the functional knowledge of building out, using, and debugging cloud infrastructure. The main part of the project should be done using the Infrastructure as Code (IaC) approach (Terraform) to create a repeatable, flexible, and maintainable configuration that can be reliably deployed anytime anywhere.

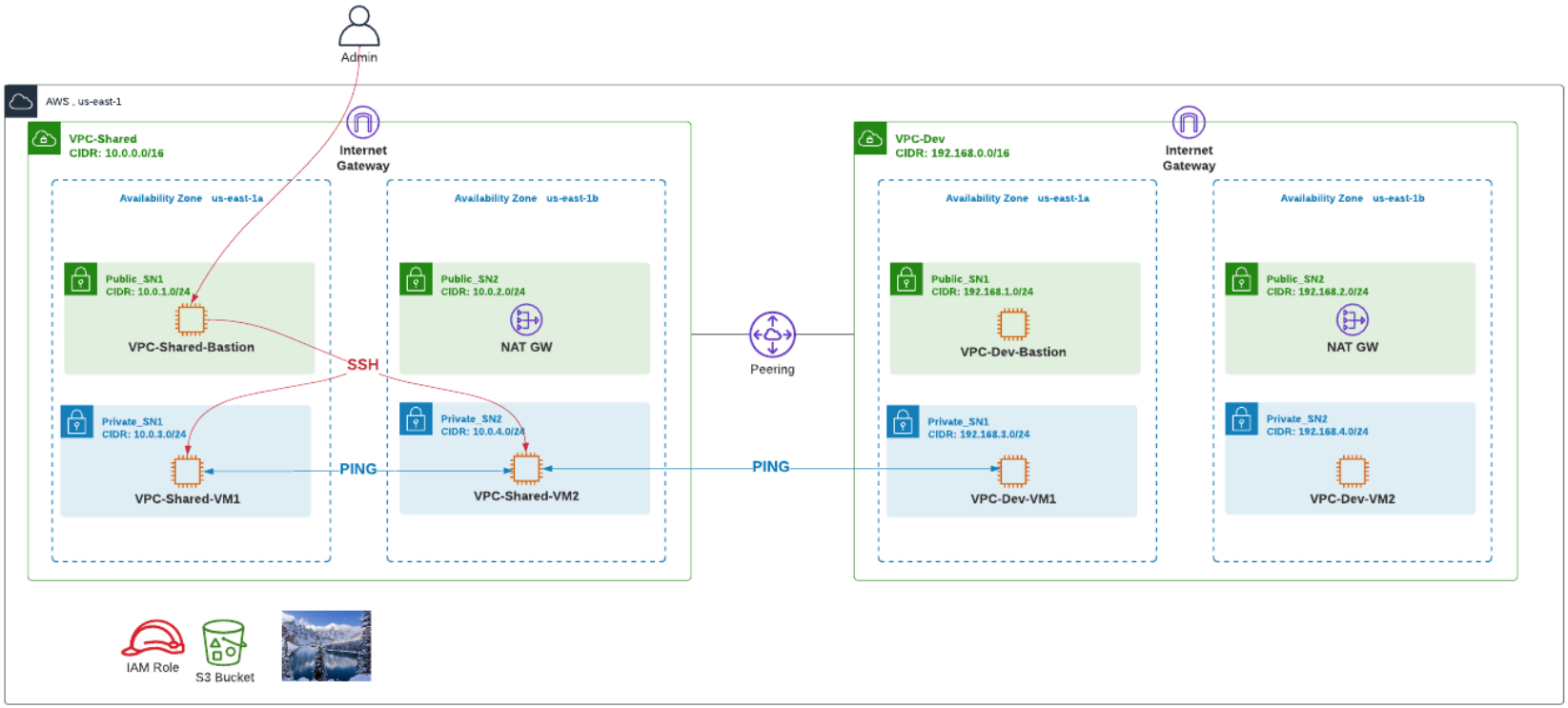
Architecture and Functional Requirements

The design diagram below depicts the real-life scenario where one VPC needs to have network connectivity to another VPC. For example, one VPC can have a Jenkins server that deploys infrastructure and application changes to all the other VPCs (dev, non-prod, and prod) and requires network connectivity to these VPCs in a hub and spoke layout.

The diagram below is **for demonstration purposes only. You are not deploying this design diagram**.

Architecture

In the scope of this assignment, you will deploy 2 VPCs (VPC-Jenkins and VPC-dev) in us-east-1 region with 2 private and 2 public subnets each interconnected via VPC peering. Detailed design diagram and deployment spec are below.



Requirements

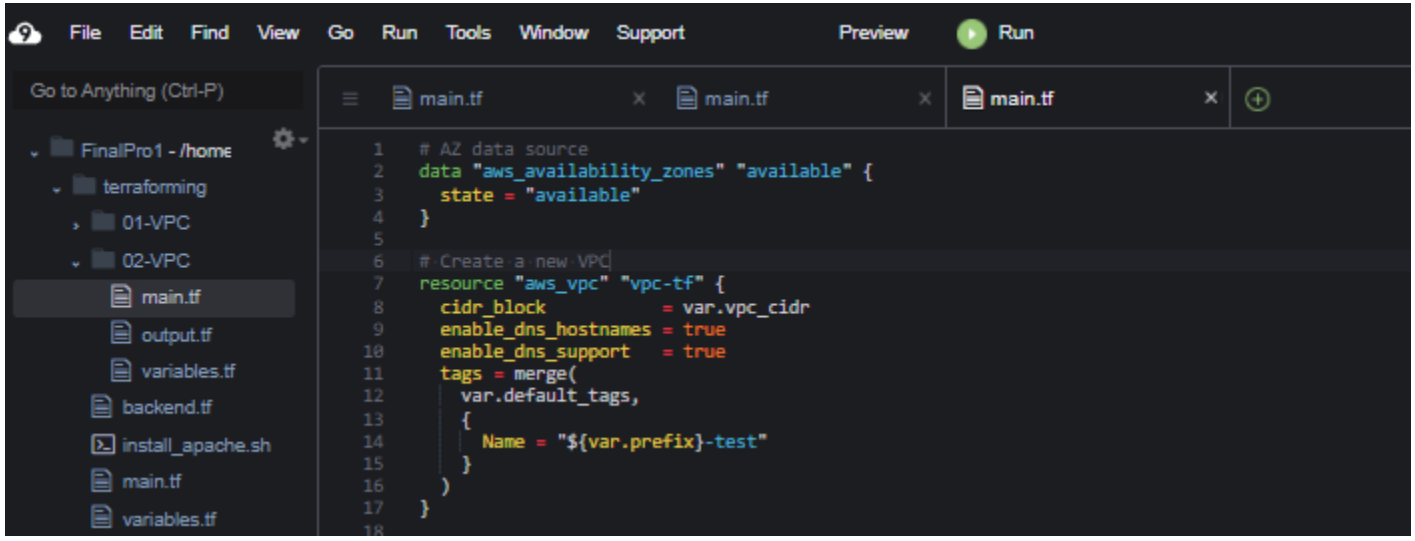
We are deploying 2 VPCs where some connectivity is permitted and the rest is blocked. Here are the requirements for your assignment.

1. Creating our VPCs

Code to Create the VPC's
VPC1

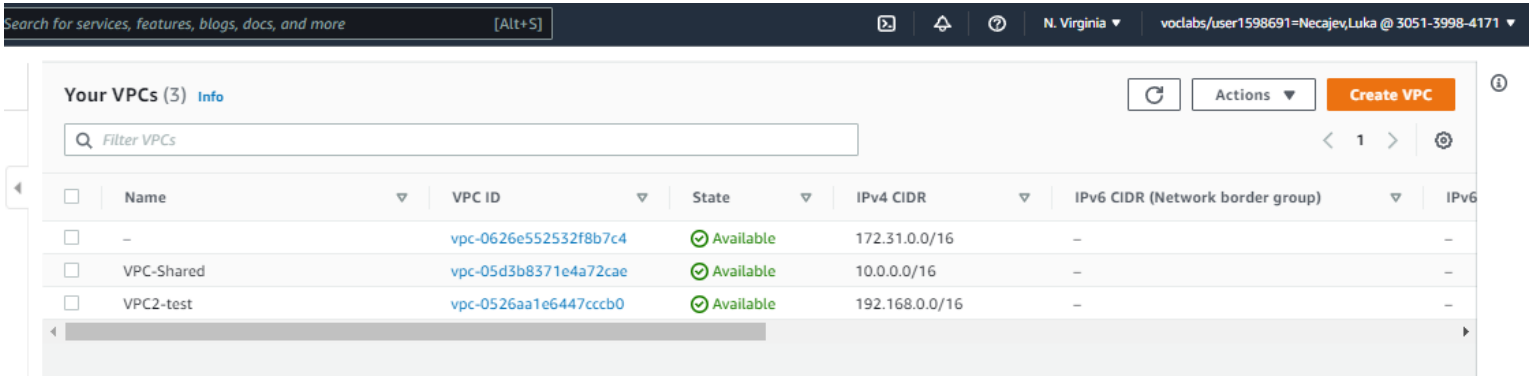
Creating the first VCPC with the name VPC-Shared

VPC2



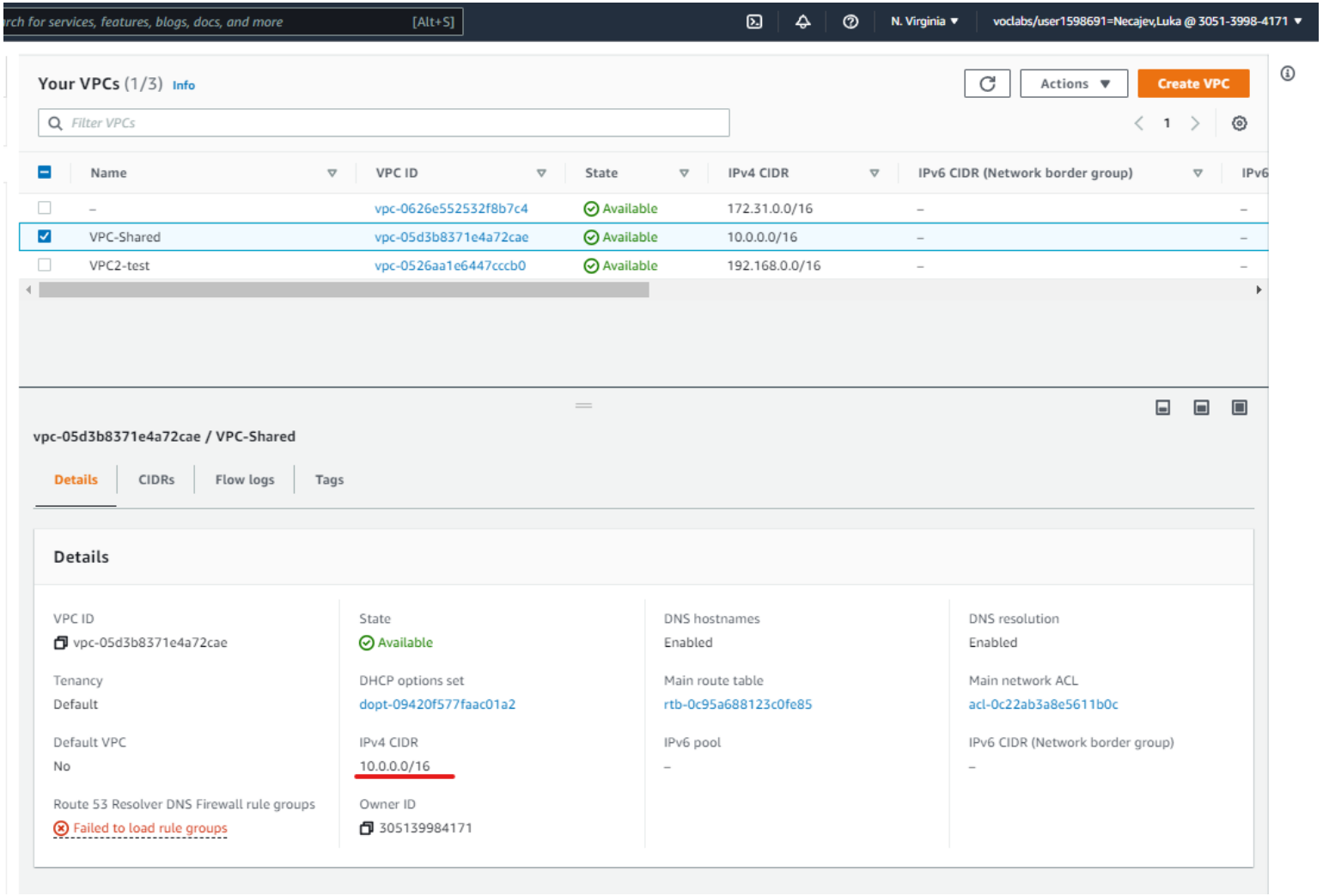
Creating the second VPC with the name VPC2-test

Showing our VPC’s



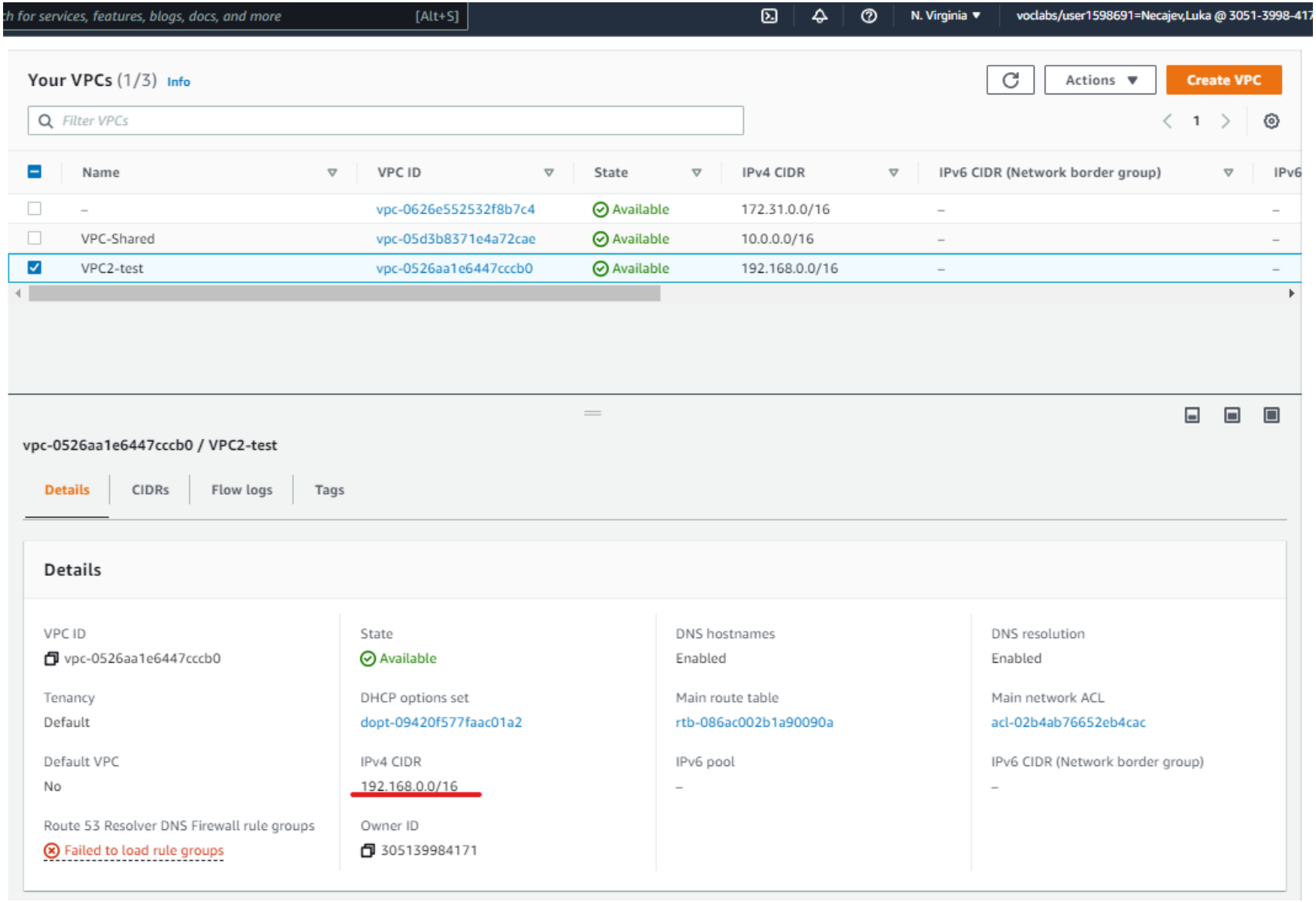
As you can see we have the default VPC, VPC1 (Shared VPC) and VPC2-(Dev VPC)

VPC1:



We set the first VPC to be in the 10.0.0.0/16 CIDR

VPC2:



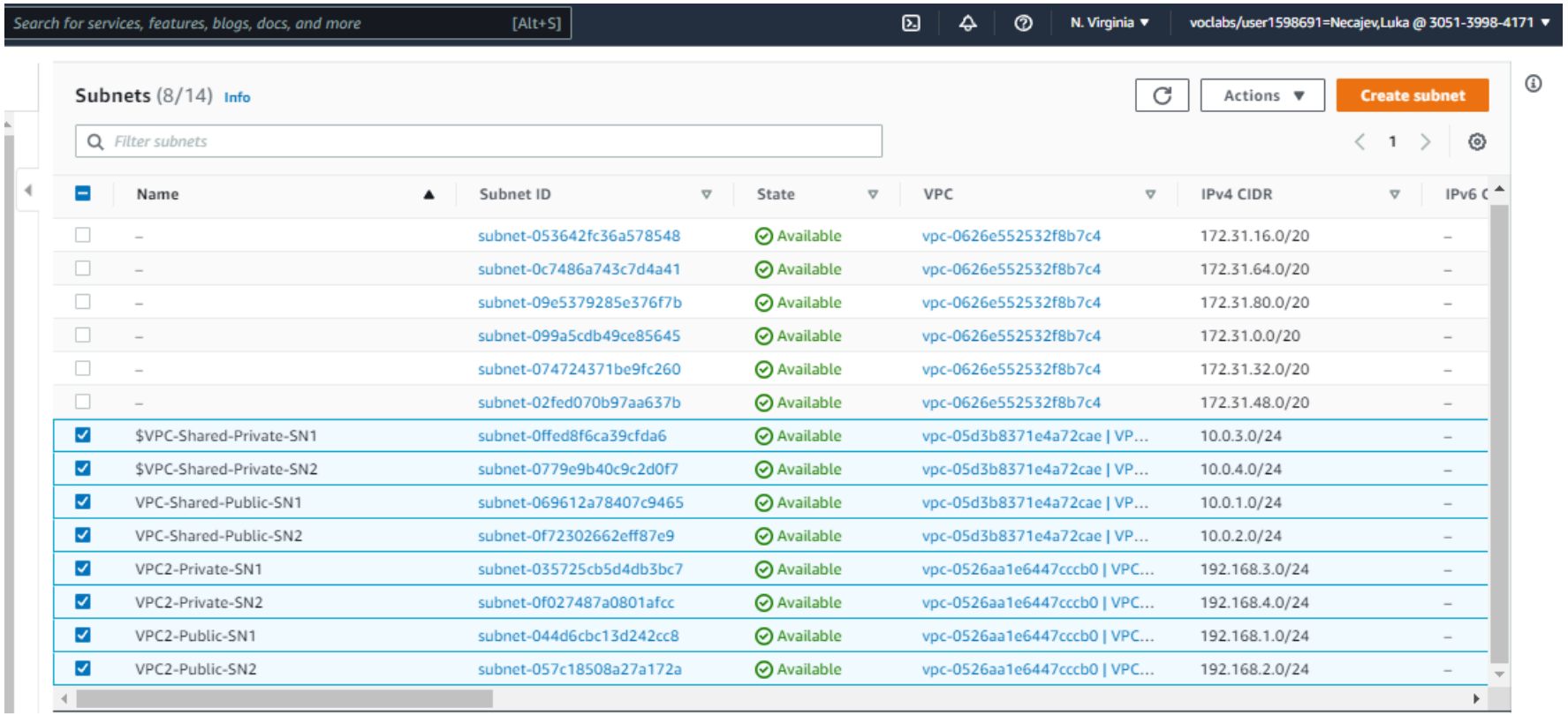
We set the second VPC to be in the 192.168.0.0/16 CIDR

2. Subnets

To Create our subnets we ran terraform code that created two private and two public subnets in EACH VPC.

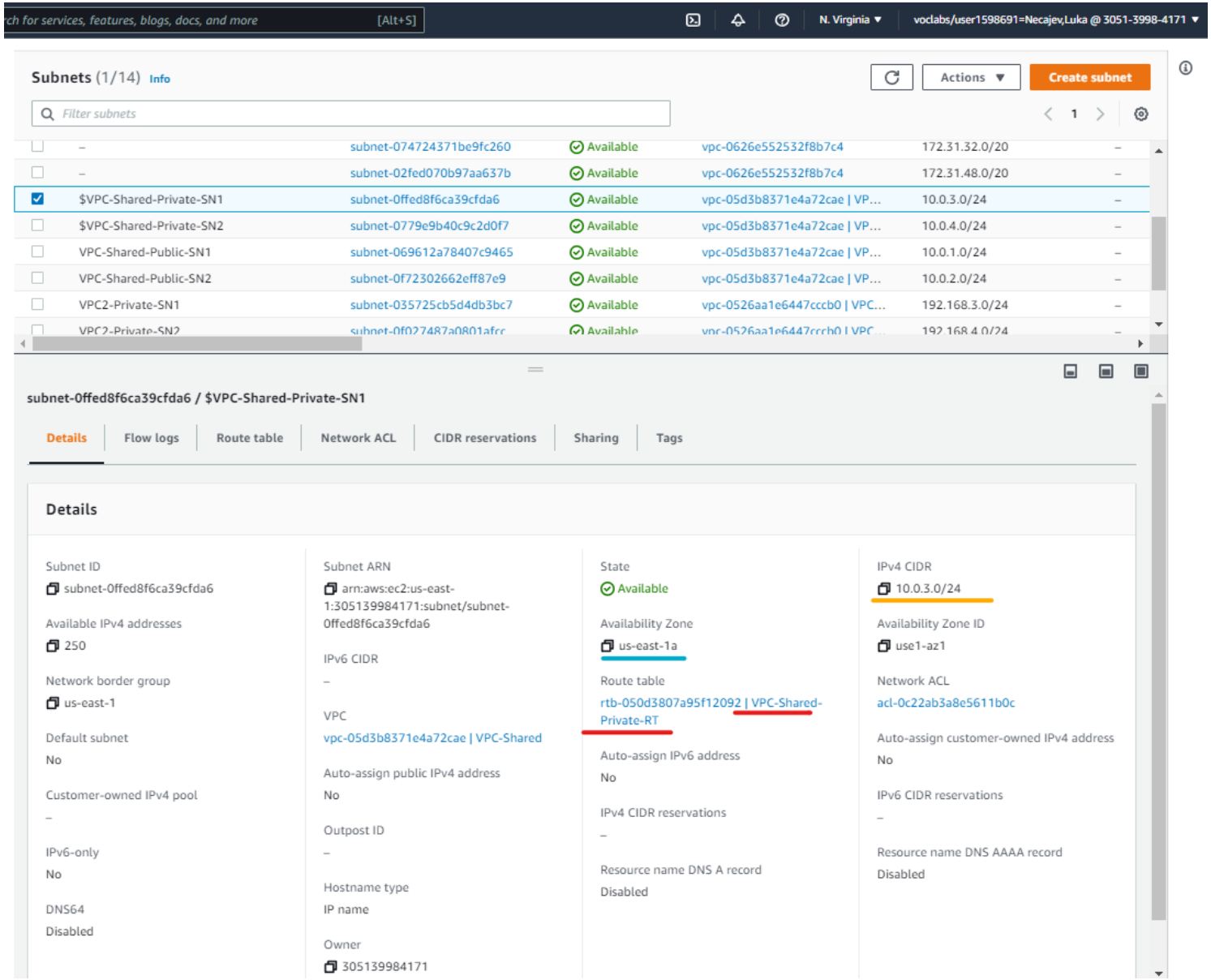
```
38  # Creating VPC-Jenkins
39  module "networking_VPC1" {
40    source        = "./01-VPC"
41    prefix        = "VPC1"
42    vpc_cidr      = "10.0.0.0/16"
43    public_cidrs  = ["10.0.1.0/24", "10.0.2.0/24"]
44    private_cidrs = ["10.0.3.0/24", "10.0.4.0/24"]
45  }
46
47  # Creating VPC-Dev
48  module "networking_VPC2" {
49    source        = "./02-VPC"
50    prefix        = "VPC2"
51    vpc_cidr      = "192.168.0.0/16"
52    public_cidrs  = ["192.168.1.0/24", "192.168.2.0/24"]
53    private_cidrs = ["192.168.3.0/24", "192.168.4.0/24"]
54  }
55
56
```

Showing the Subnets that were created in our VPC.



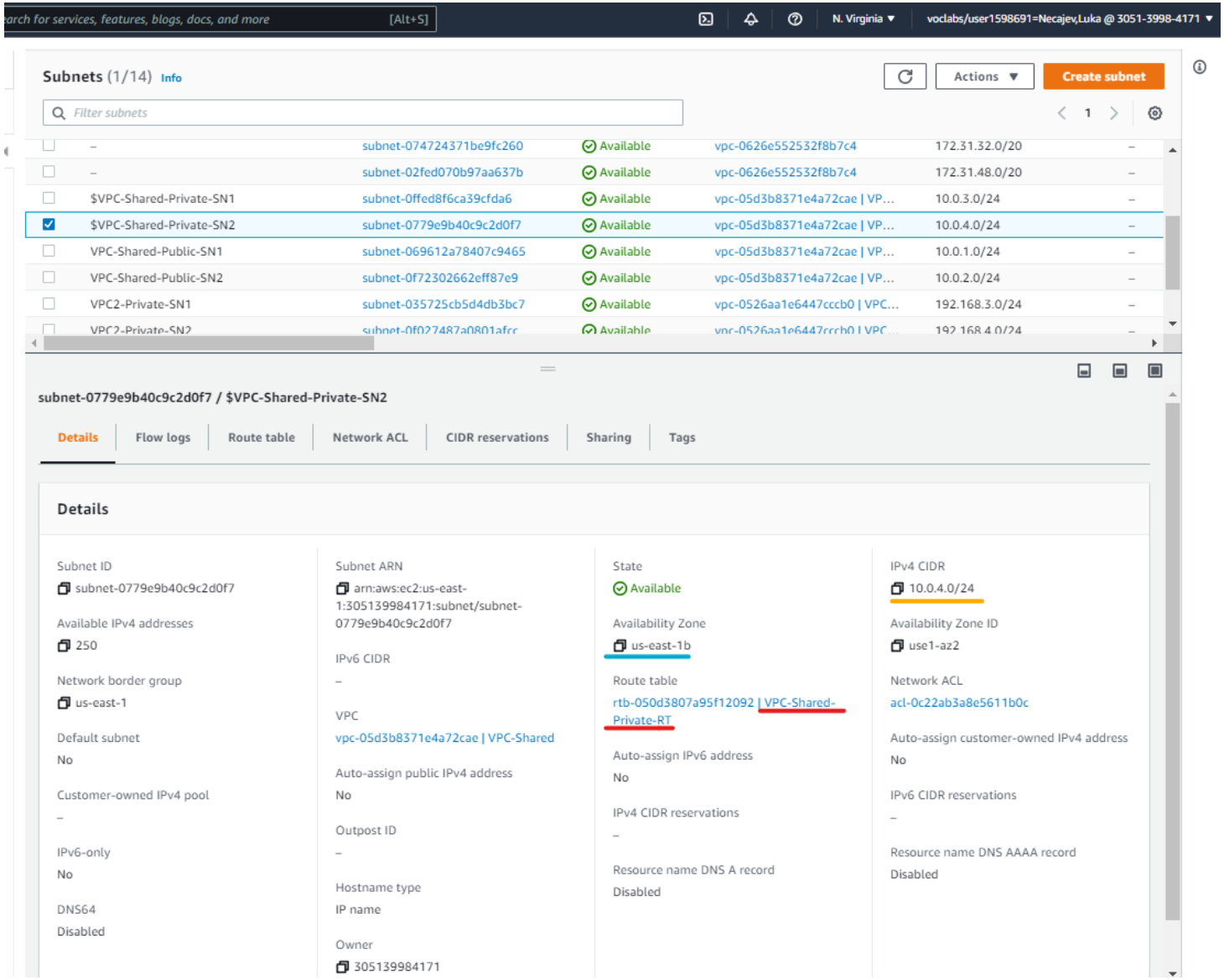
Showing The Subnet variables. Note that We are in the right CIDR, availability zone and using the correct routing table

VPC1-Private SN1



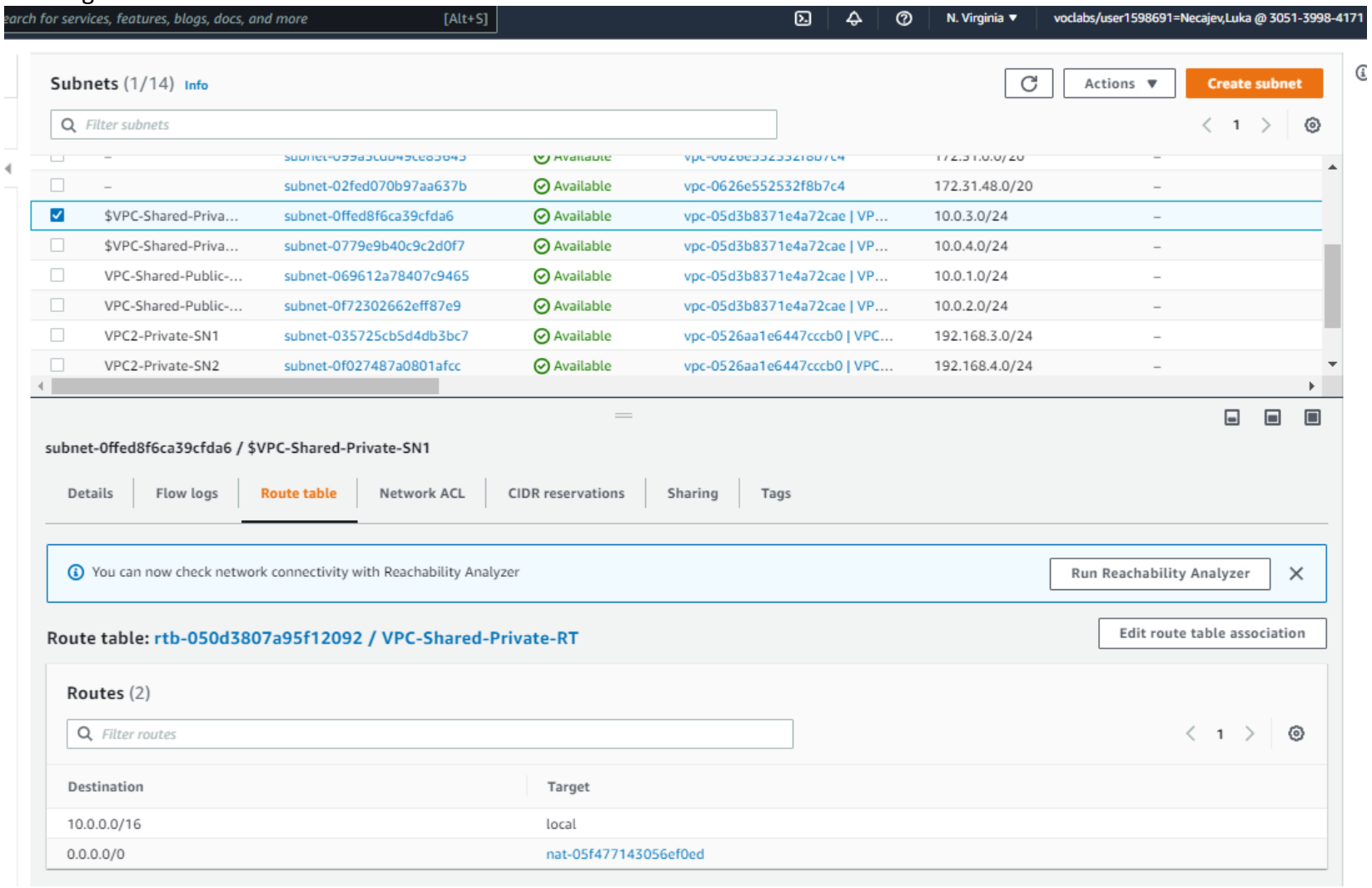
Orange is Showing the correct CIDR for private network 1. (10.0.3.0)
Blue is showing the correct Availability zone (us-east-1a)
Red is showing that we are associated with the correct routing table. (VPC-Shared-Private-RT)

VPC1-Private SN2

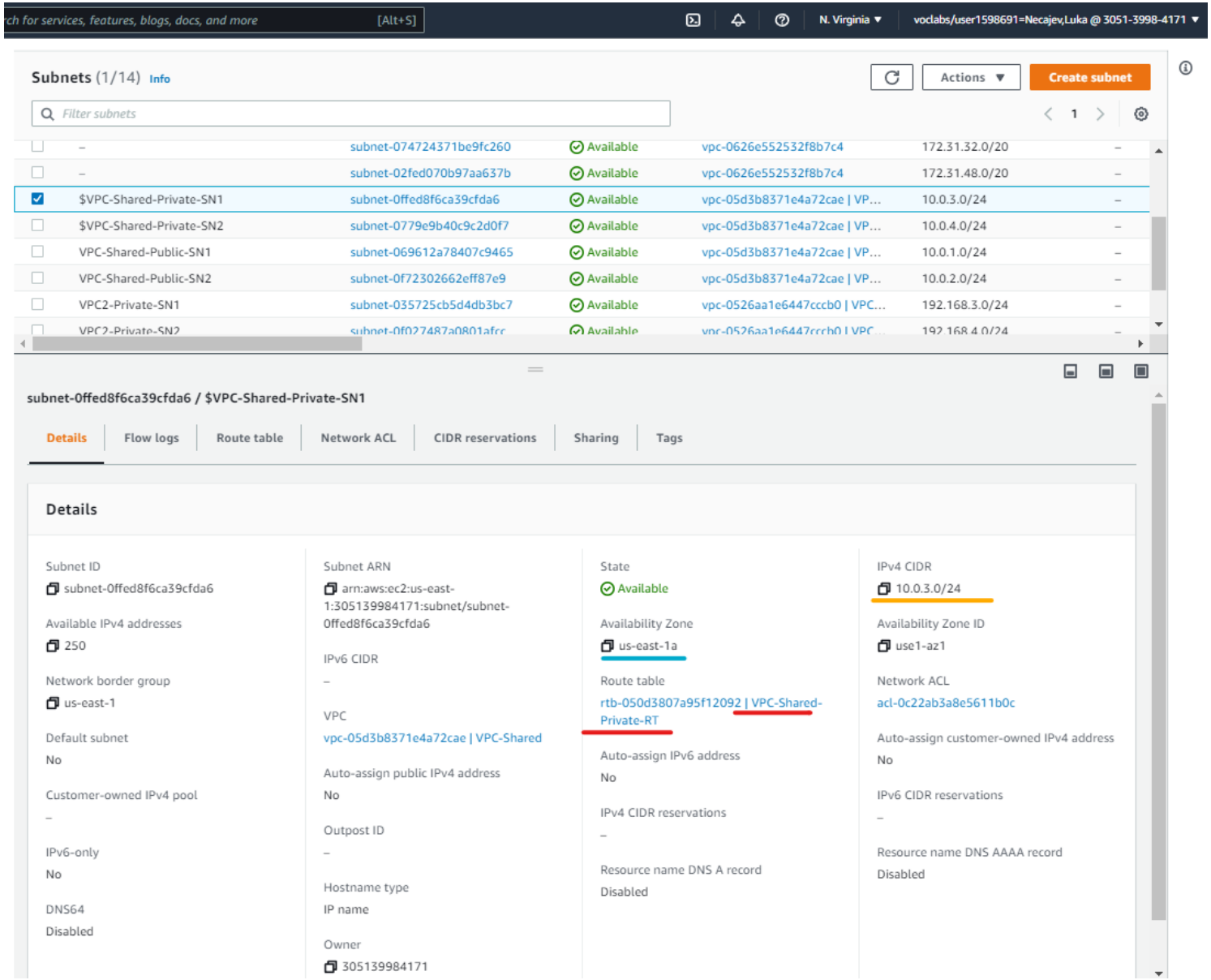


Orange is Showing the correct CIDR for private network 1. (10.0.4.0)

Blue is showing the correct Availability zone (us-east-1b)
Red is showing that we are associated with the correct routing table. (VPC-Shared-Private-RT)
Routing table

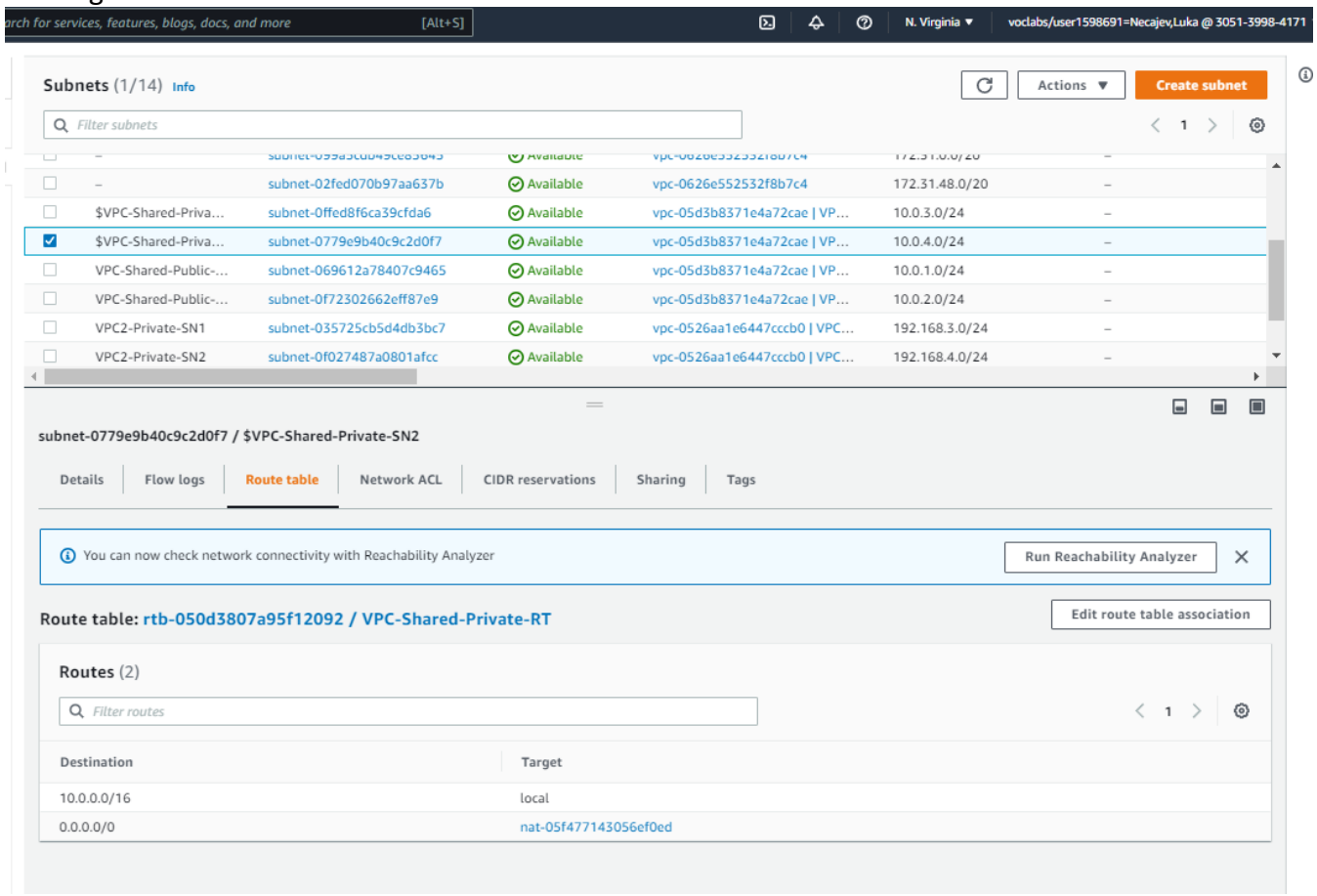


VPC1-Public SN1

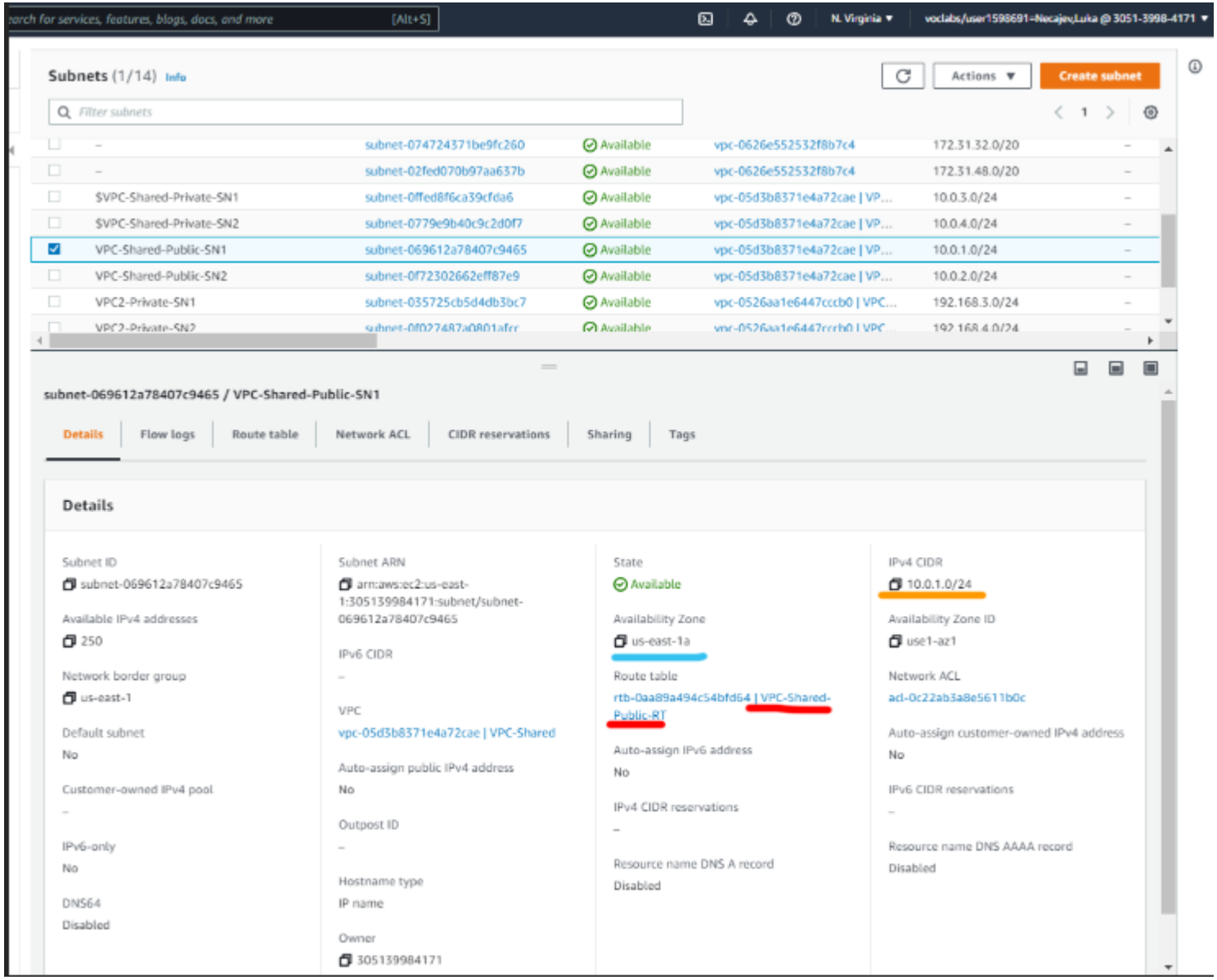


Orange is Showing the correct CIDR for private network 1. (10.0.3.0)
Blue is showing the correct Availability zone (us-east-1a)
Red is showing that we are associated with the correct routing table. (VPC-Shared-Private-RT)

Routing table

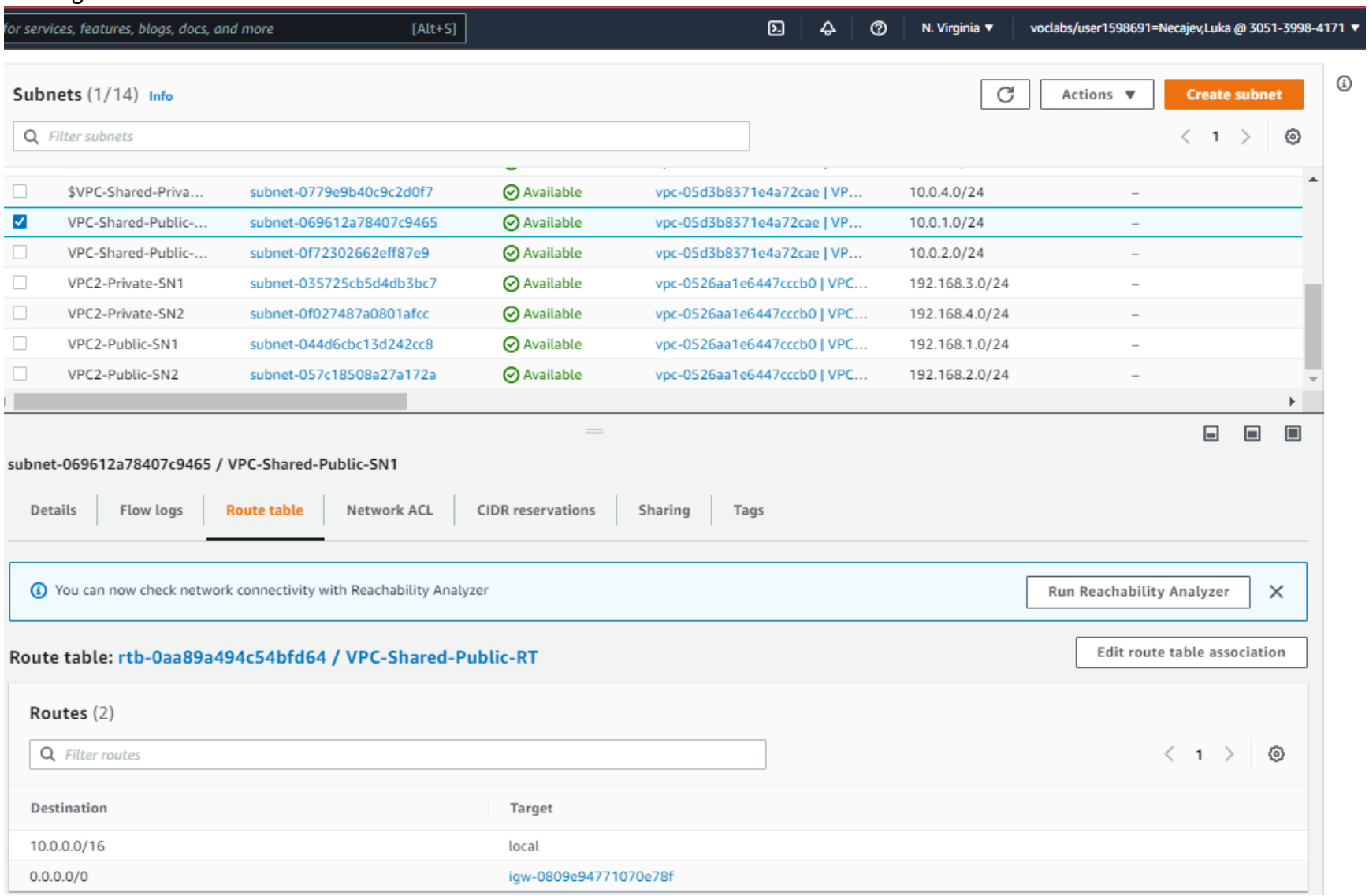


VPC1-Public SN1

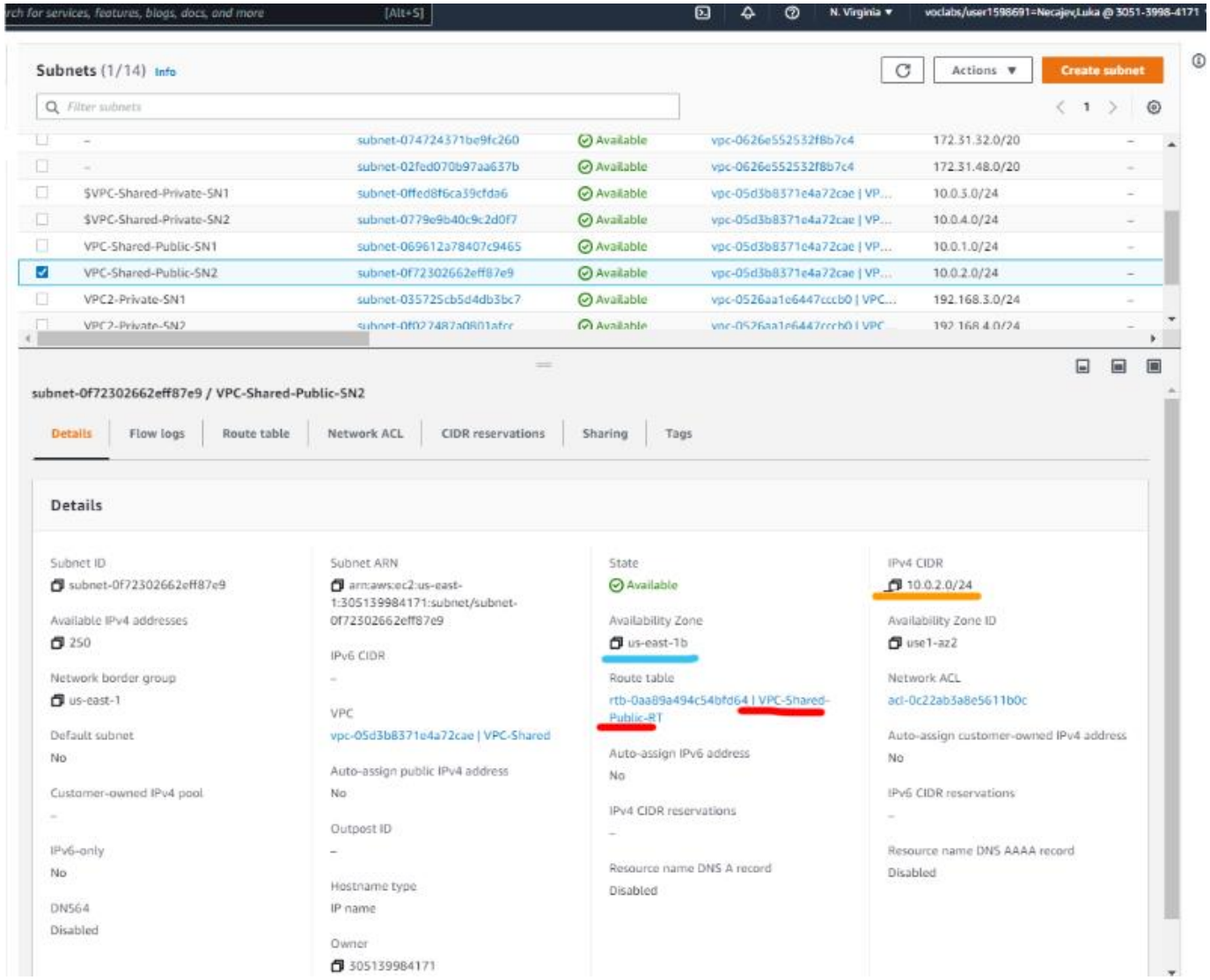


Orange is Showing the correct CIDR for private network 1. (10.0.1.0)

Blue is showing the correct Availability zone (us-east-1a)
Red is showing that we are associated with the correct routing table. (VPC-Shared-Public-RT)
Routing table

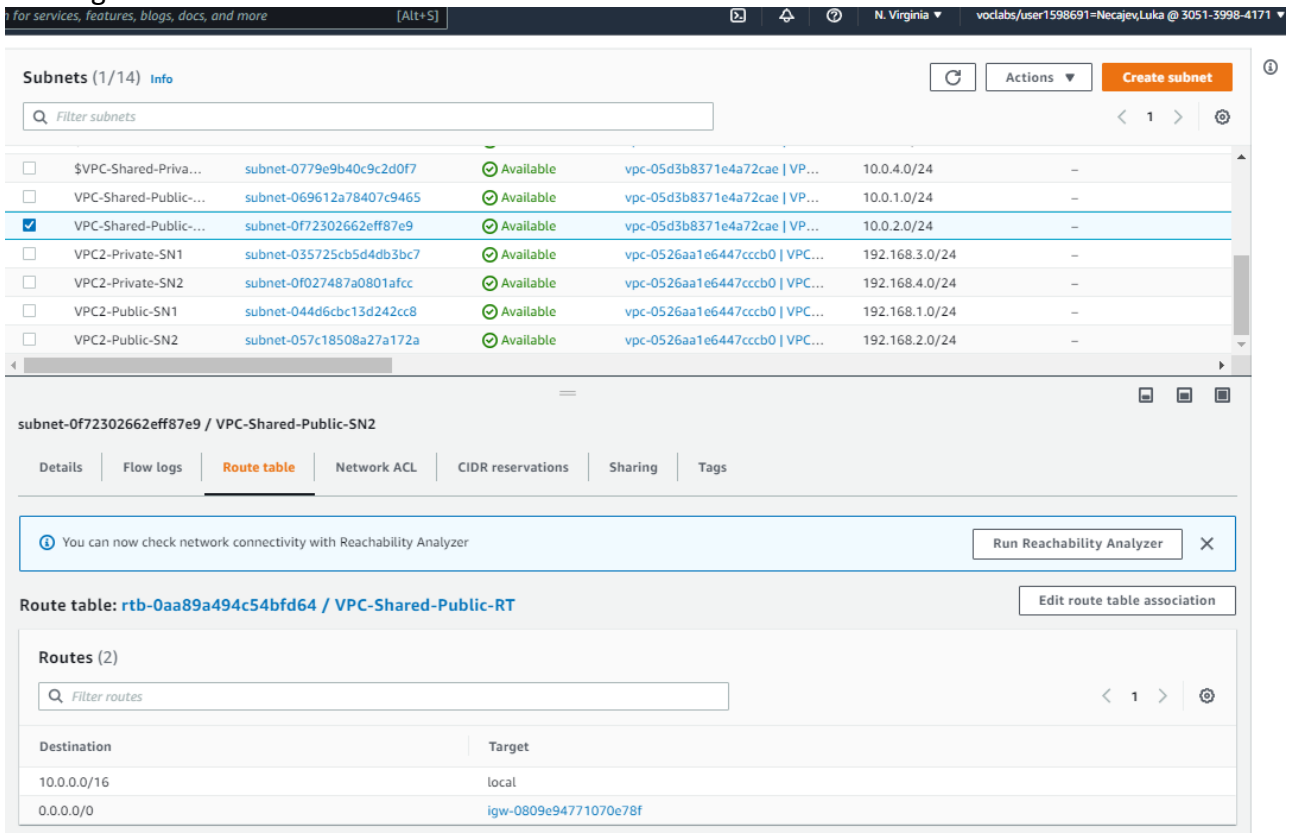


VPC1-Public SN2

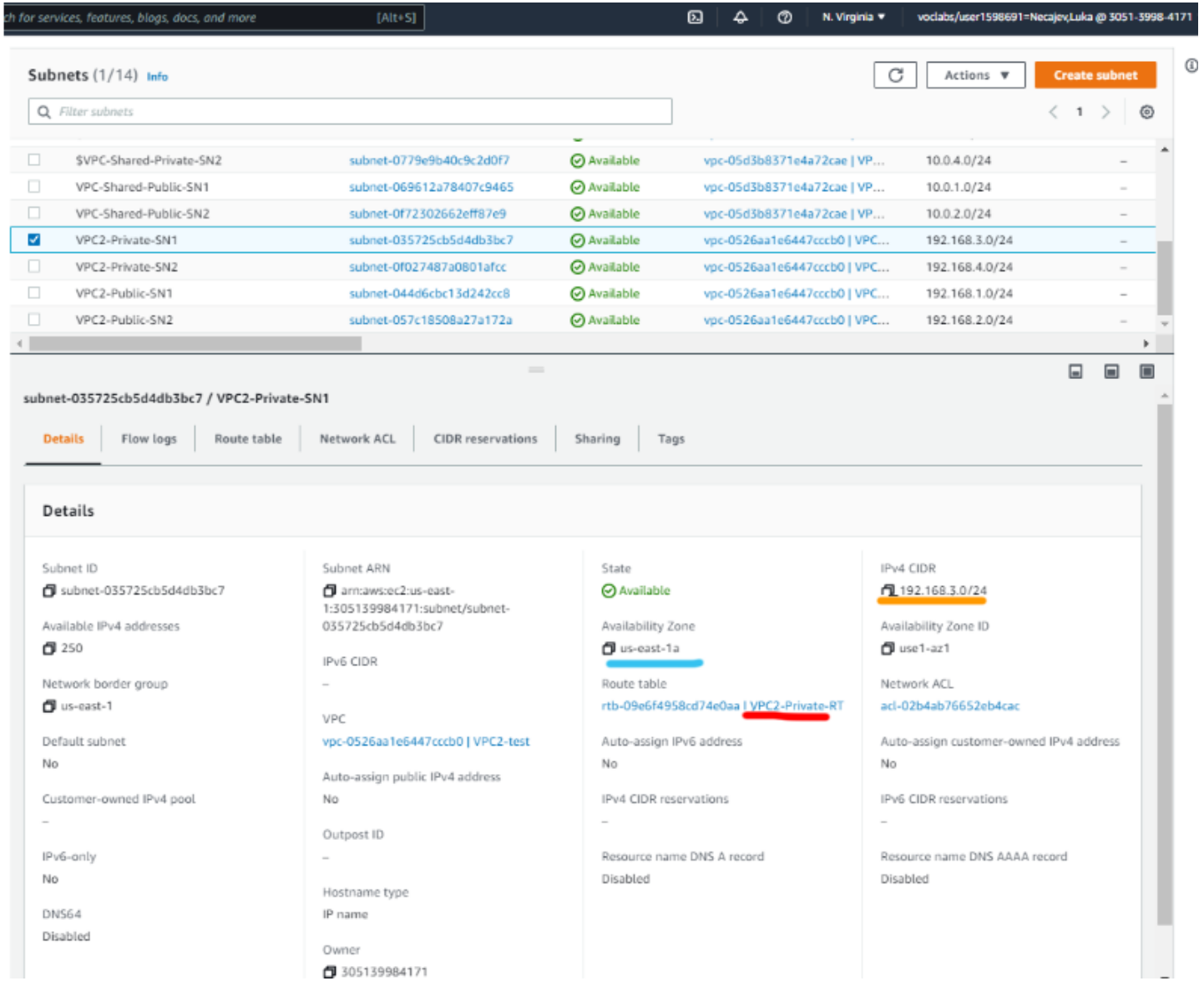


Orange is Showing the correct CIDR for private network 1. (10.0.2.0)

Blue is showing the correct Availability zone (us-east-1b)
Red is showing that we are associated with the correct routing table. (VPC-Shared-Public-RT)
Routing table

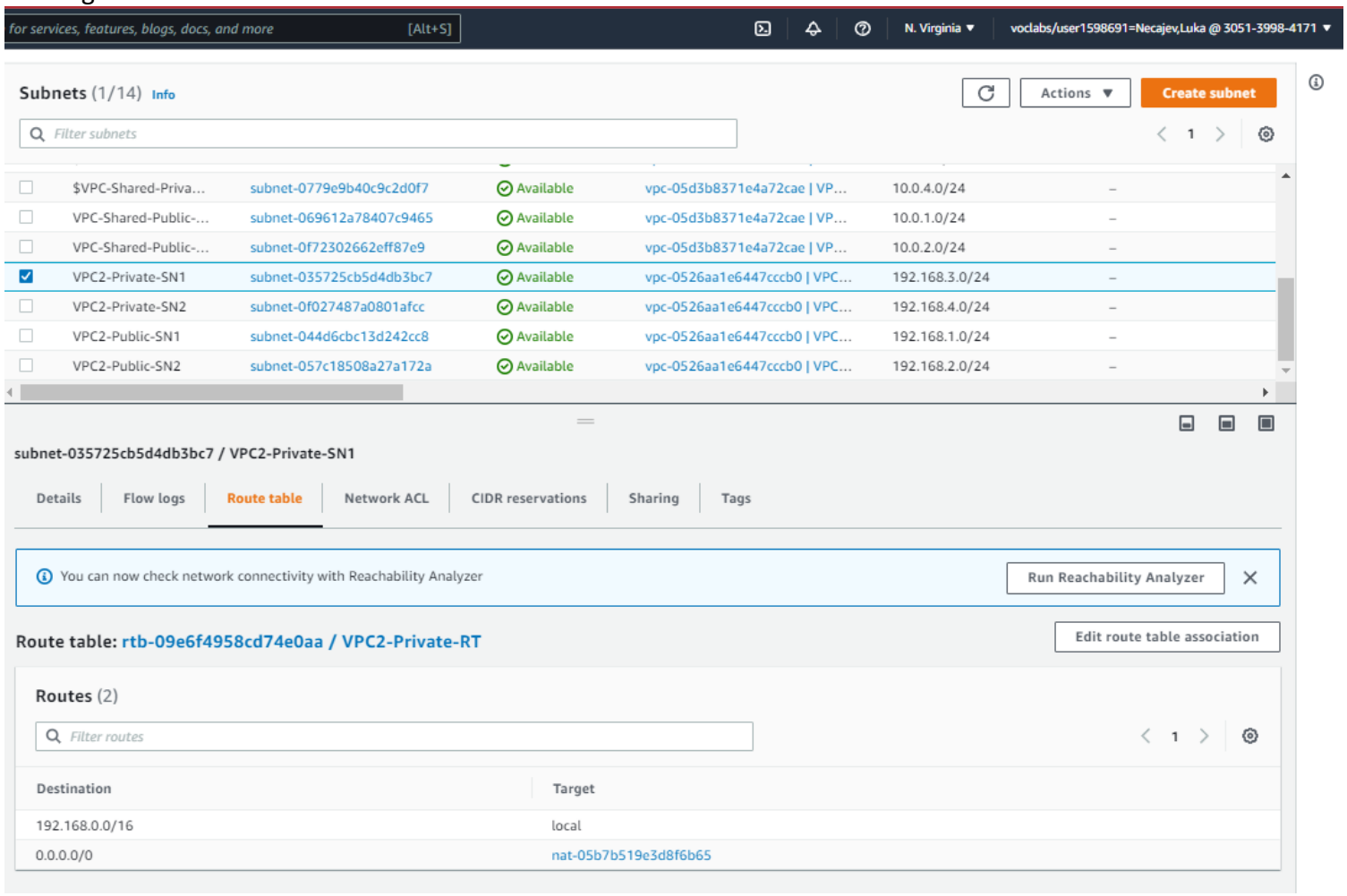


VPC2-Private SN1

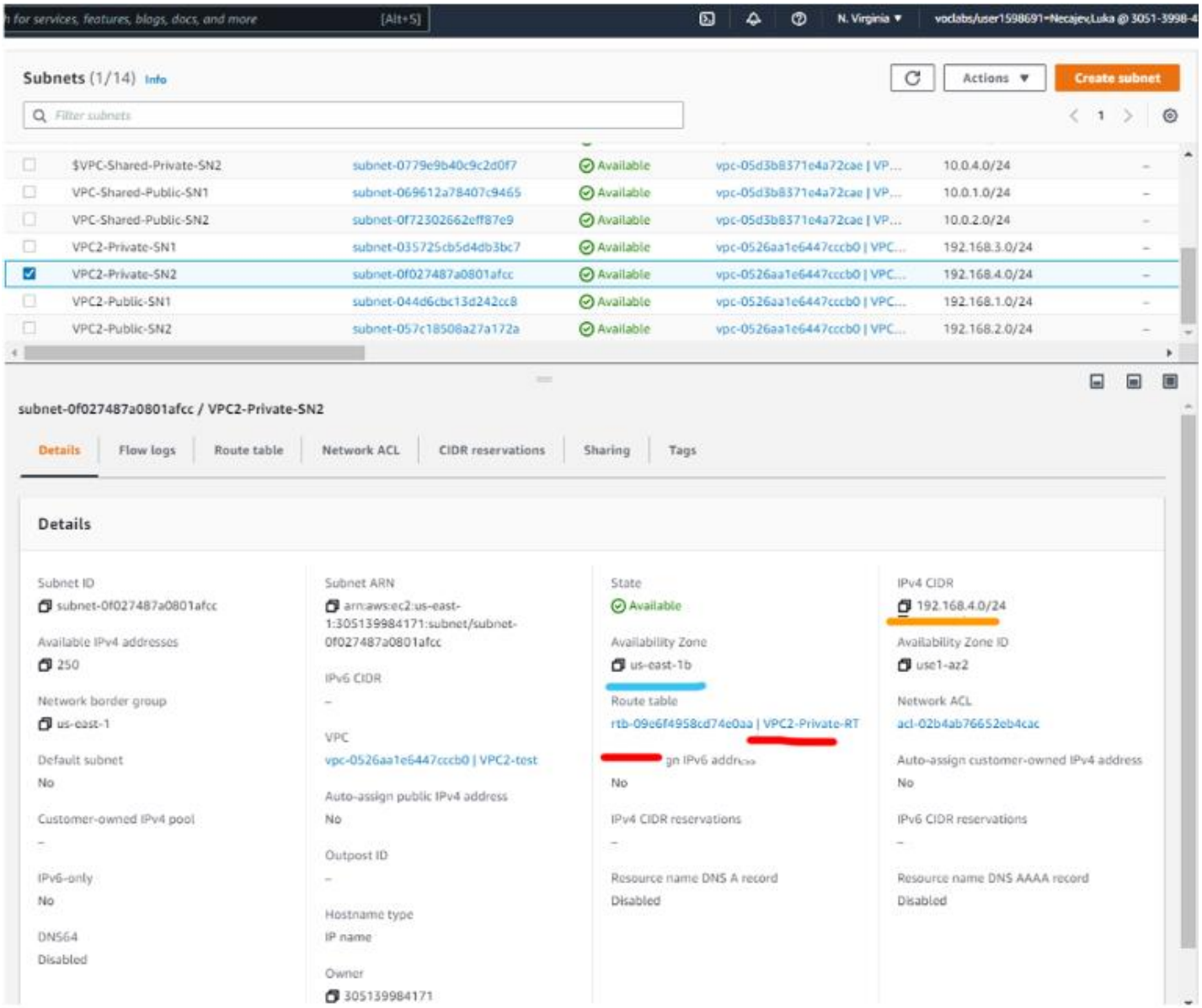


Orange is Showing the correct CIDR for private network 1. (192.168.3.0/24)
Blue is showing the correct Availability zone (us-east-1a)
Red is showing that we are associated with the correct routing table. (VPC2 -Private-RT)

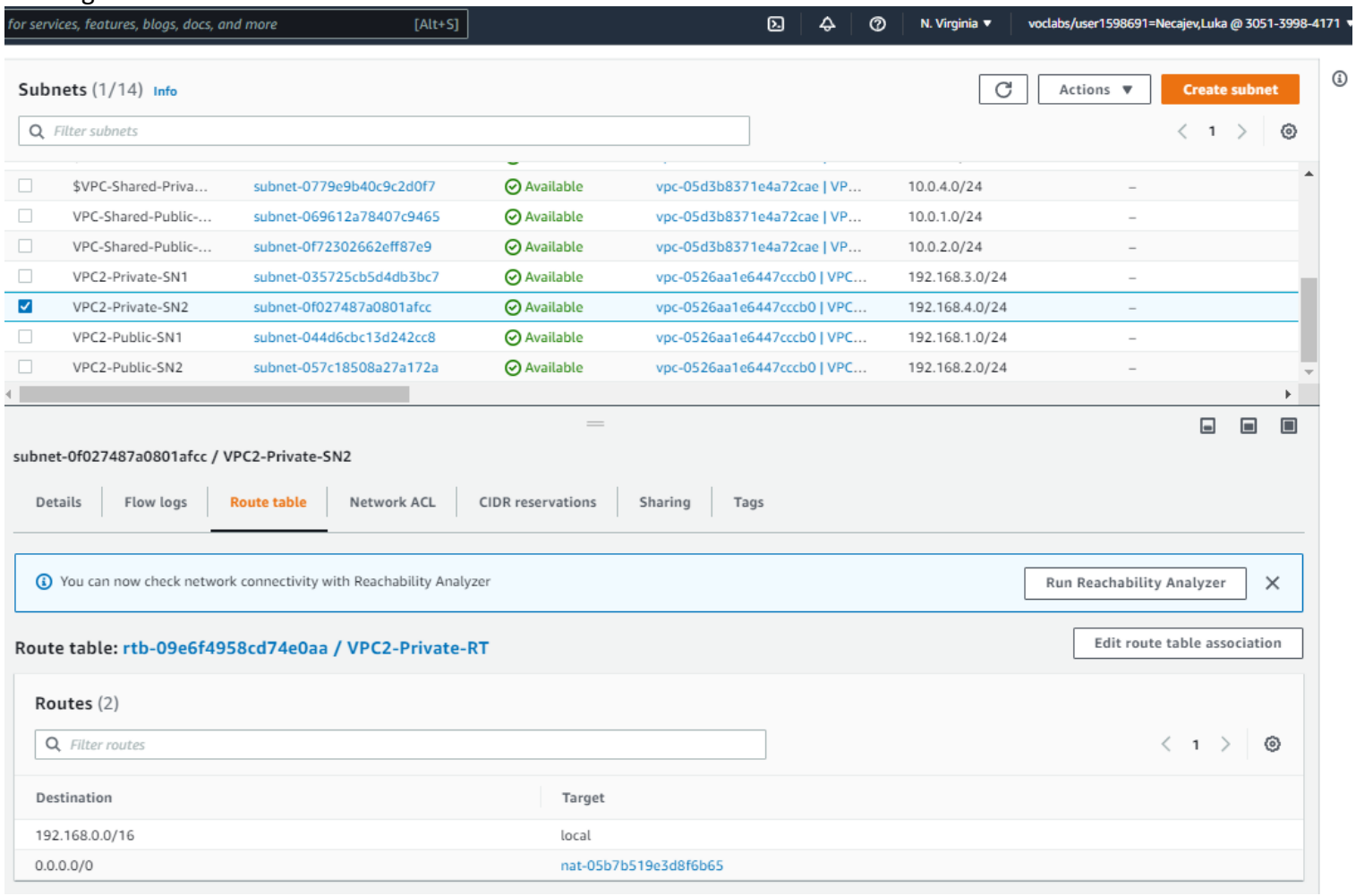
Routing table



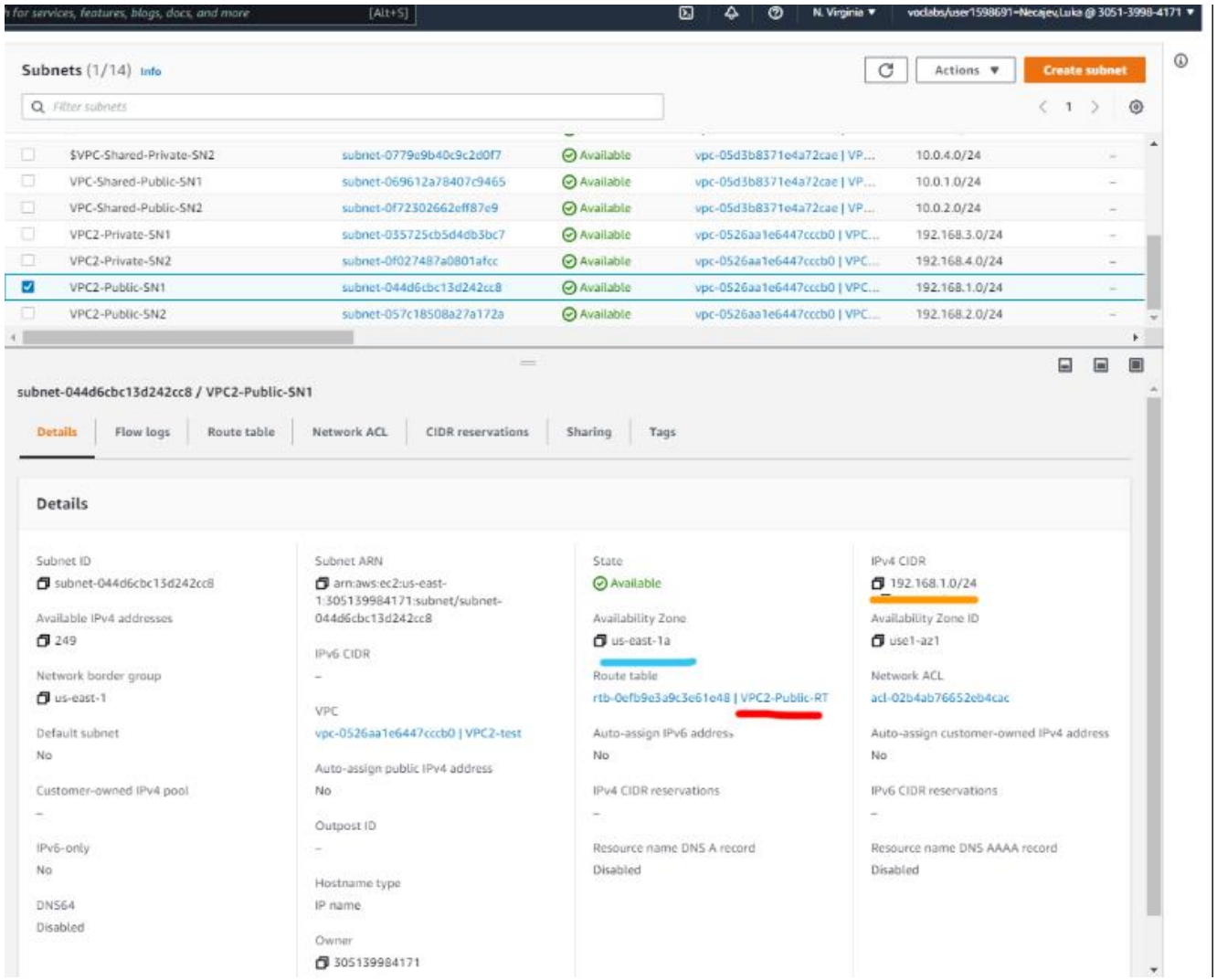
VPC2-Private SN2



Orange is Showing the correct CIDR for private network 1. (192.168.4.0/24)
Blue is showing the correct Availability zone (us-east-1b)
Red is showing that we are associated with the correct routing table. (VPC2 -Private-RT)
Routing Table

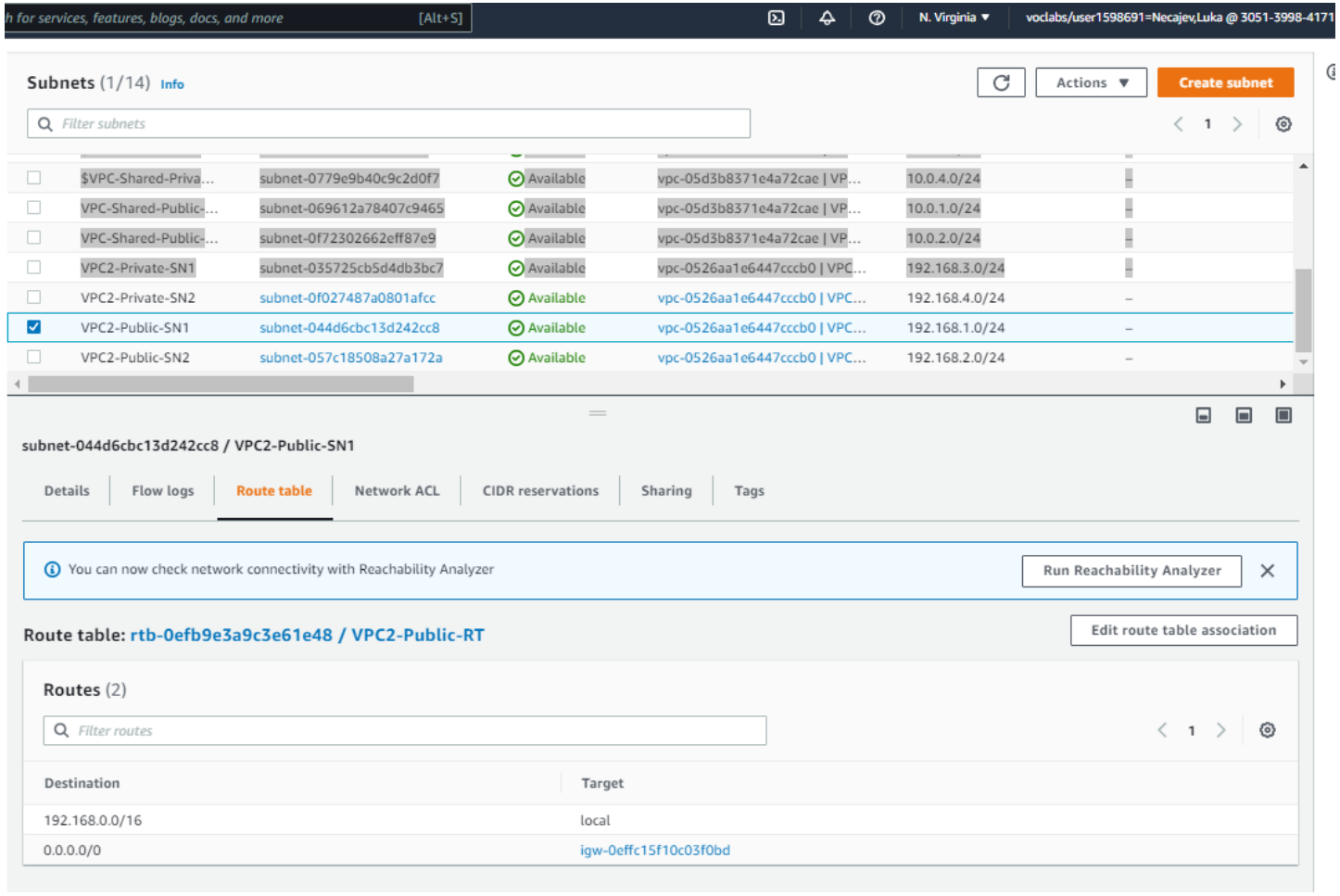


VPC2-Public SN1

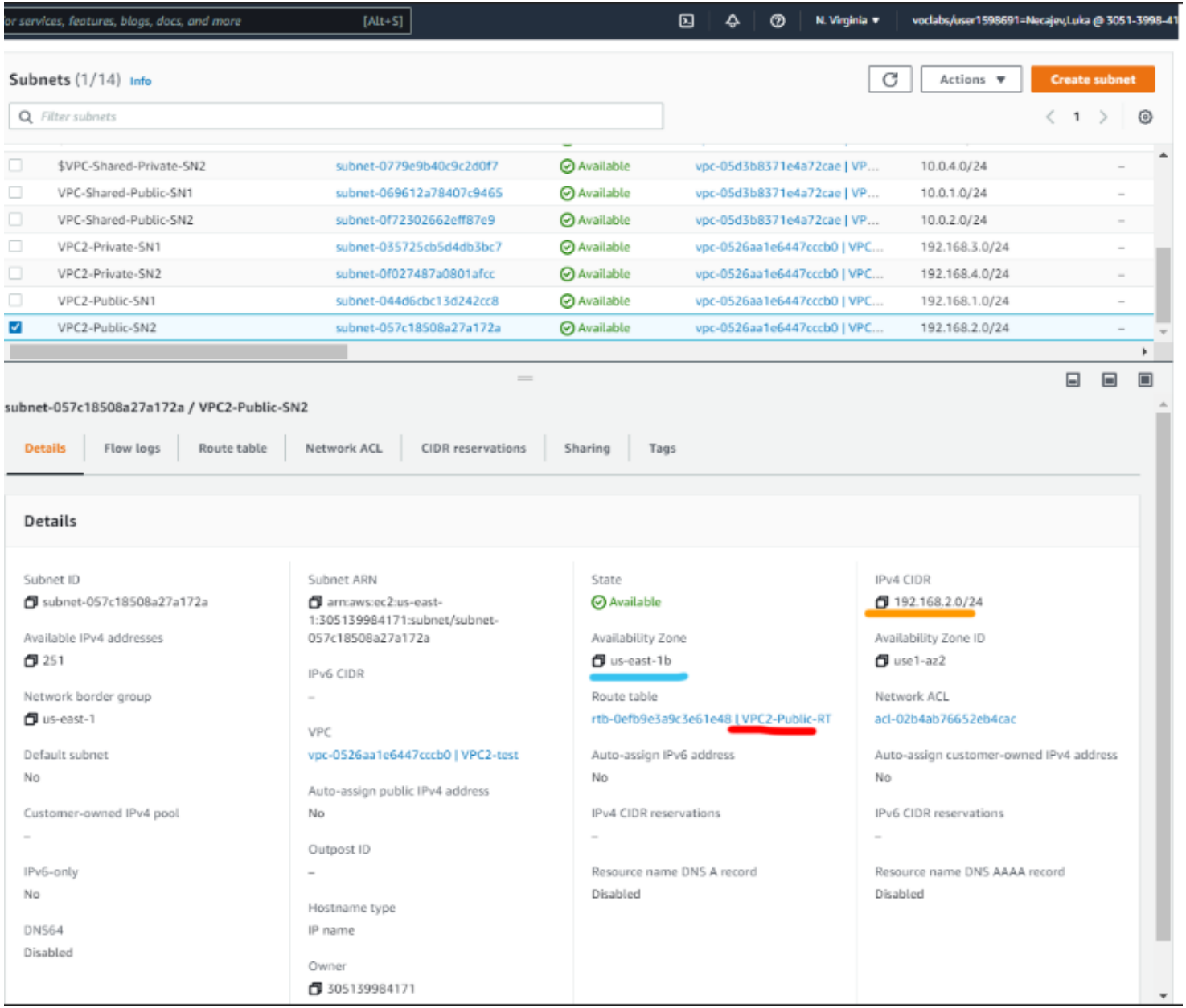


Orange is Showing the correct CIDR for private network 1. (192.168.1.0/24)
Blue is showing the correct Availability zone (us-east-1a)
Red is showing that we are associated with the correct routing table. (VPC2 -Public-RT)

Route Table

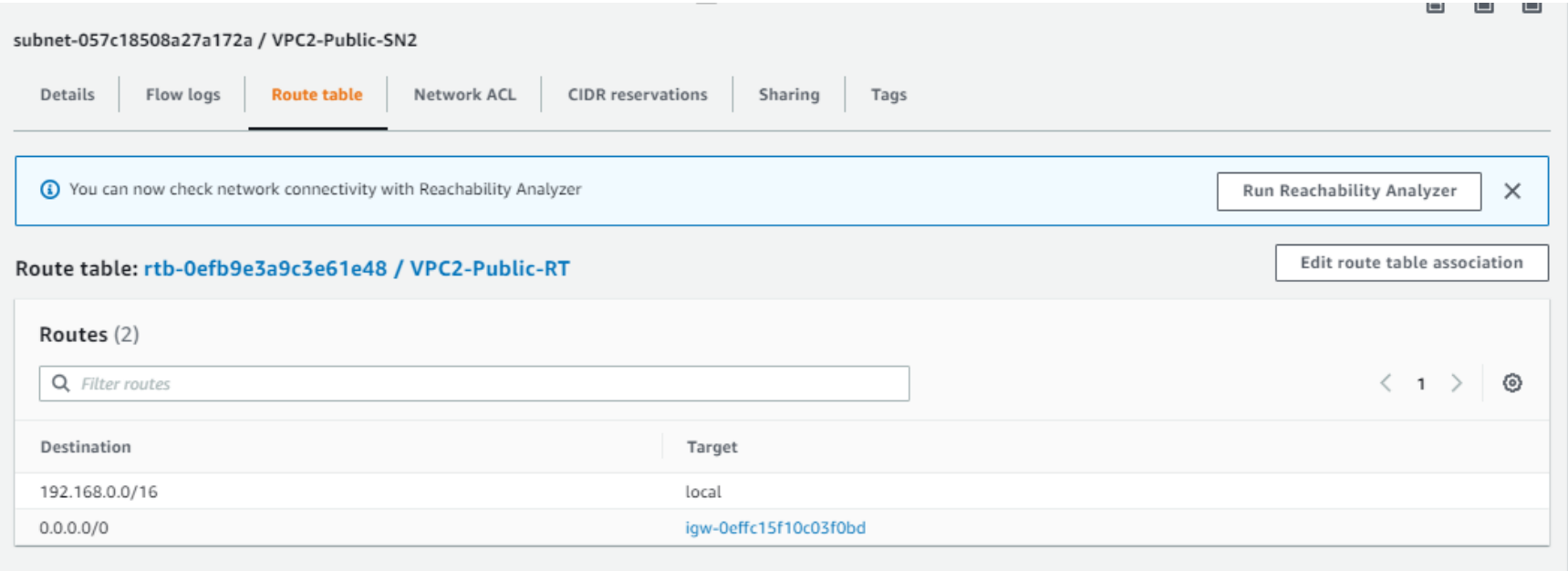


VPC2-Public SN2


or services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-41
Subnets (1/14) Info
Actions
Create subnet
Filter subnets
1
$VPC-Shared-Private-SN2 subnet-0779e9b40c9c2d0f7 Available vpc-05d3b8371e4a72cae | VP... 10.0.4.0/24
VPC-Shared-Public-SN1 subnet-069612a78407c9465 Available vpc-05d3b8371e4a72cae | VP... 10.0.1.0/24
VPC-Shared-Public-SN2 subnet-0f72302662eff87e9 Available vpc-05d3b8371e4a72cae | VP... 10.0.2.0/24
VPC2-Private-SN1 subnet-035725cb5d4db3bc7 Available vpc-0526aa1e6447cccb0 | VPC... 192.168.3.0/24
VPC2-Private-SN2 subnet-0f027487a0801afcc Available vpc-0526aa1e6447cccb0 | VPC... 192.168.4.0/24
VPC2-Public-SN1 subnet-044d6cbc13d242cc8 Available vpc-0526aa1e6447cccb0 | VPC... 192.168.1.0/24
VPC2-Public-SN2 subnet-057c18508a27a172a Available vpc-0526aa1e6447cccb0 | VPC... 192.168.2.0/24
subnet-057c18508a27a172a / VPC2-Public-SN2
Details
Flow logs
Route table
Network ACL
CIDR reservations
Sharing
Tags
Details
Subnet ID
subnet-057c18508a27a172a
Available IPv4 addresses
251
Network border group
us-east-1
Default subnet
No
Customer-owned IPv4 pool
–
IPv6-only
No
DNS64
Disabled
Subnet ARN
arn:aws:ec2:us-east-1:305139984171:subnet/subnet-057c18508a27a172a
IPv6 CIDR
–
VPC
vpc-0526aa1e6447cccb0 | VPC2-test
Auto-assign public IPv4 address
No
Outpost ID
–
Hostname type
IP name
Owner
305139984171
State
Available
Availability Zone
us-east-1b
Route table
rtb-0efb9e3a9c3e61e48 | VPC2-Public-RT
Auto-assign IPv6 address
No
IPv4 CIDR reservations
–
Resource name DNS A record
Disabled
IPv4 CIDR
192.168.2.0/24
Availability Zone ID
use1-az2
Network ACL
acl-02b4ab76652eb4cac
Auto-assign customer-owned IPv4 address
No
IPv6 CIDR reservations
–
Resource name DNS AAAA record
Disabled

Orange is Showing the correct CIDR for private network 1. (192.168.2.0/24)
Blue is showing the correct Availability zone (us-east-1b)
Red is showing that we are associated with the correct routing table. (VPC2 -Public-RT)

Route Table



As you can see the highlighted sections, We

3. Route Tables

Code to Create Routing Tables and Associations

Code to create public and private routing table for VPC1 (Shared)

```
75
76  # Create Public Route Table
77  resource "aws_route_table" "rt-public" {
78    vpc_id = aws_vpc.vpc-tf.id
79    route {
80      cidr_block = "0.0.0.0/0"
81      gateway_id = aws_internet_gateway.igw.id
82    }
83    tags = merge(
84      var.default_tags,
85      {
86        Name = "VPC-Shared-Public-RT"
87      }
88    )
89  }
90  # Create Private Route Table
91  resource "aws_route_table" "rt-private" {
92    vpc_id = aws_vpc.vpc-tf.id
93    route {
94      cidr_block = "0.0.0.0/0"
95      gateway_id = aws_nat_gateway.nat.id
96    }
97    tags = merge(
98      var.default_tags,
99      {
100       Name = "VPC-Shared-Private-RT"
101     }
102   )
103 }
104
```

Code associating the routing tables to a subnet

```
105  # Create Route Table Association
106  resource "aws_route_table_association" "association-pub" {
107    count          = var.counter
108    subnet_id      = aws_subnet.public.*.id[count.index]
109    route_table_id = aws_route_table.rt-public.id
110  }
111
112  resource "aws_route_table_association" "association-pr" {
113    count          = var.counter
114    subnet_id      = aws_subnet.private.*.id[count.index]
115    route_table_id = aws_route_table.rt-private.id
116  }
```

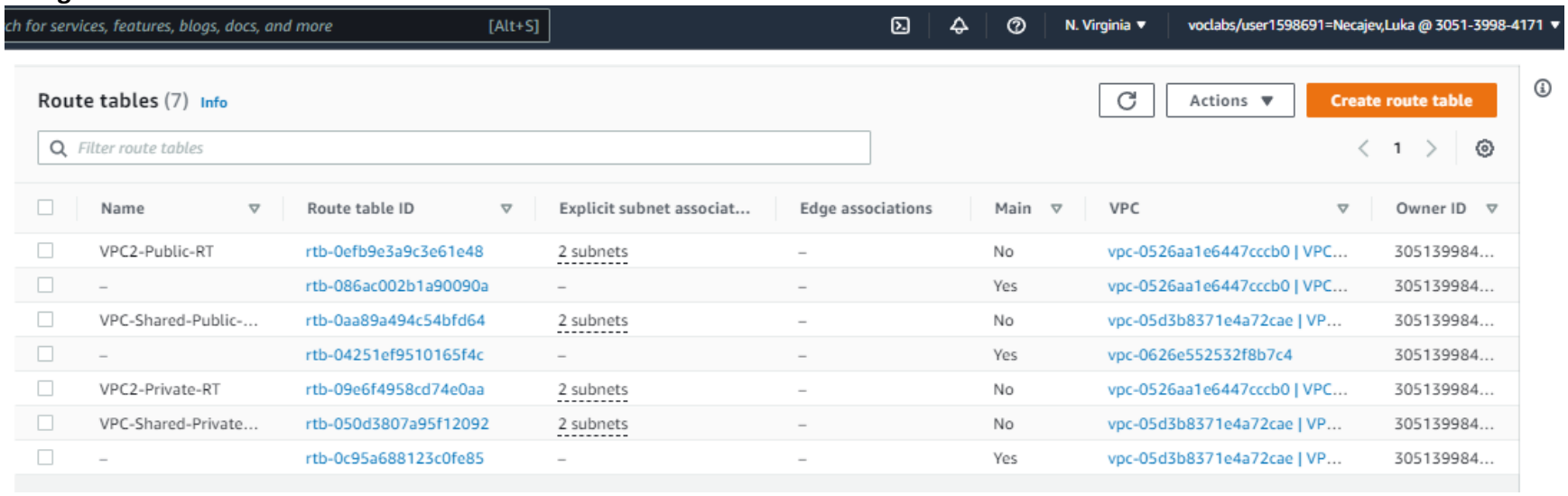
Code Creating VPC2 (DEV) Routing tables and Associations

```
# Create Public Route Table
resource "aws_route_table" "rt-public" {
  vpc_id = aws_vpc.vpc-tf.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_internet_gateway.igw.id
  }
  tags = merge(
    var.default_tags,
    {
      Name = "${var.prefix}-Public-RT"
    }
  )
}
# Create Private Route Table
resource "aws_route_table" "rt-private" {
  vpc_id = aws_vpc.vpc-tf.id
  route {
    cidr_block = "0.0.0.0/0"
    gateway_id = aws_nat_gateway.nat.id
  }
  tags = merge(
    var.default_tags,
    {
      Name = "${var.prefix}-Private-RT"
    }
  )
}

```

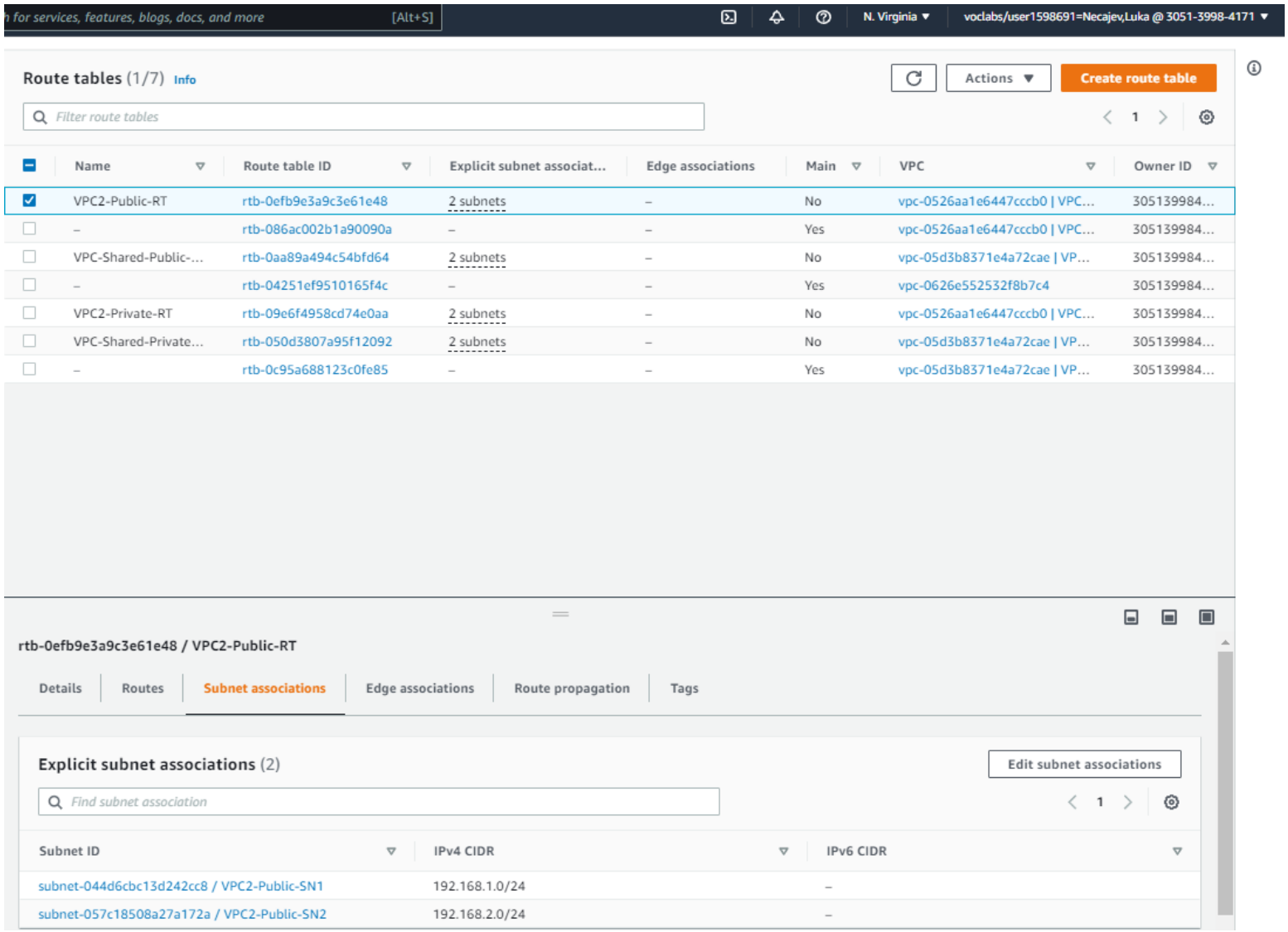
Then Create the Route Table associations for VPC2 Public and Private fields

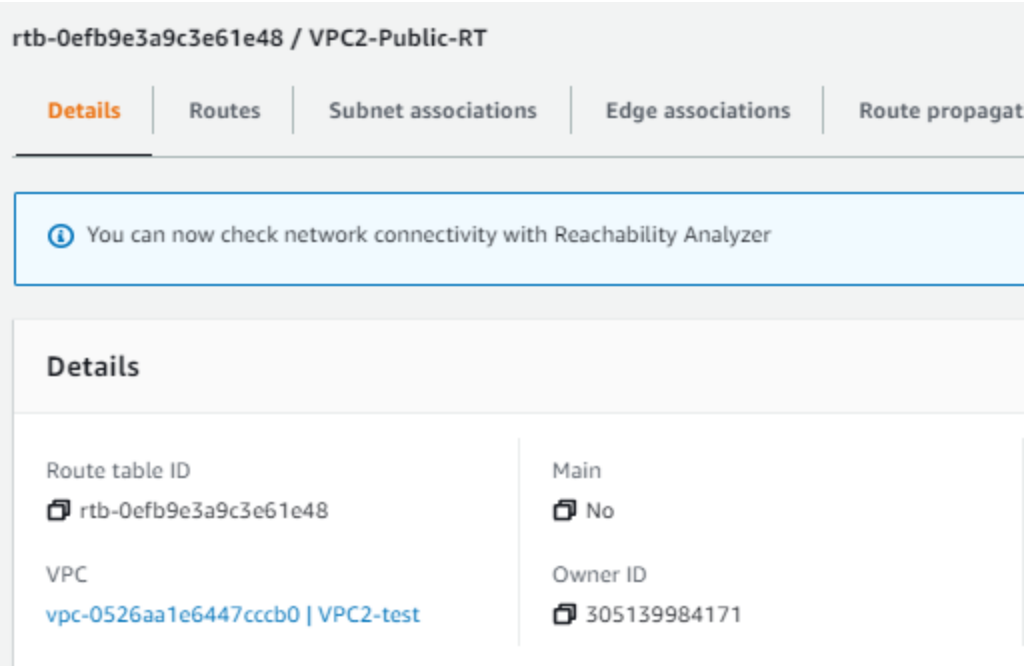
```
104  # Create Route Table Association
105  resource "aws_route_table_association" "association-pub" {
106    count          = var.counter
107    subnet_id      = aws_subnet.public.*.id[count.index]
108    route_table_id = aws_route_table.rt-public.id
109  }
110
111  resource "aws_route_table_association" "association-pr" {
112    count          = var.counter
113    subnet_id      = aws_subnet.private.*.id[count.index]
114    route_table_id = aws_route_table.rt-private.id
115  }
```

Using Console



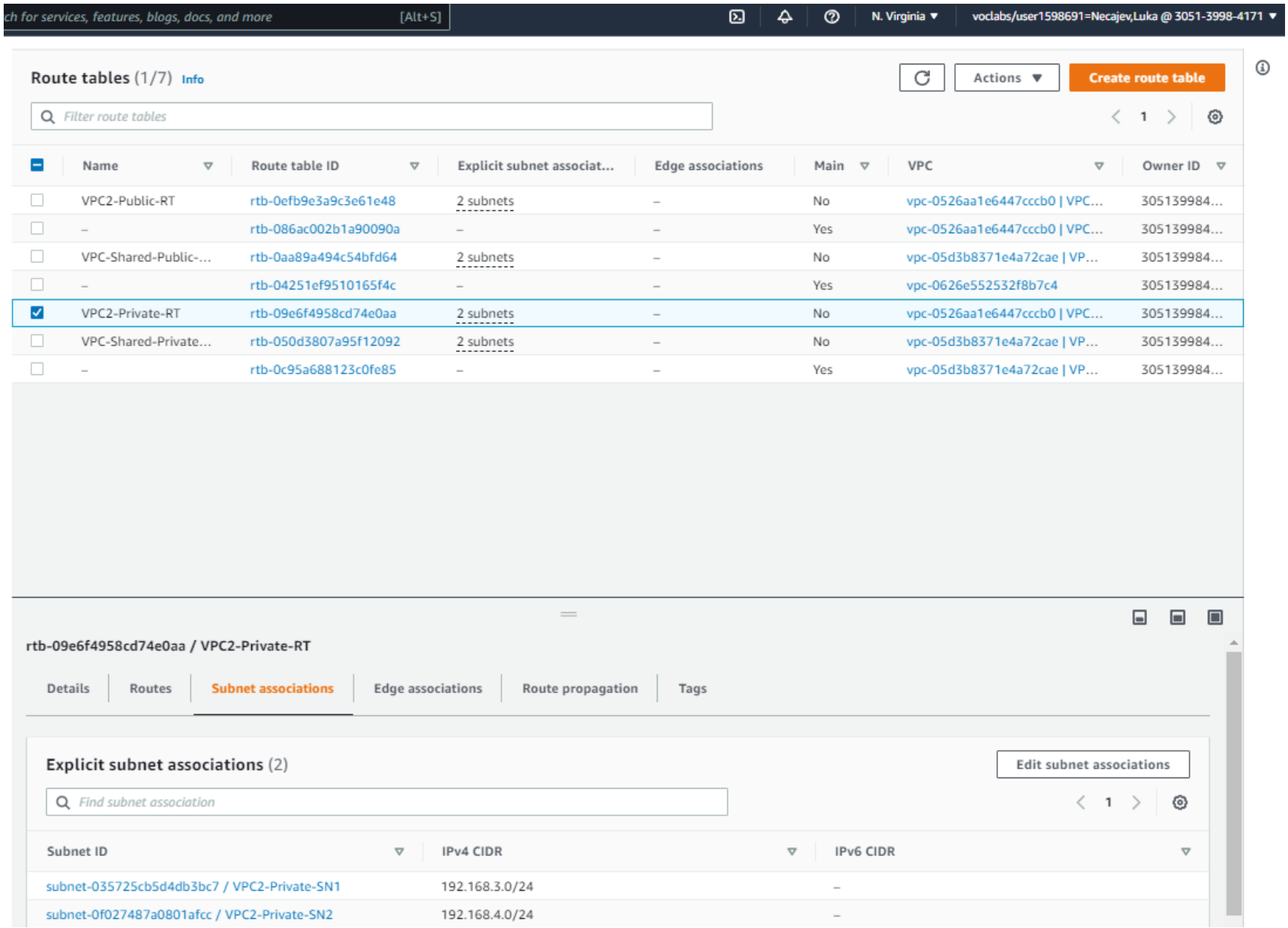
VPC2 Public Routing table

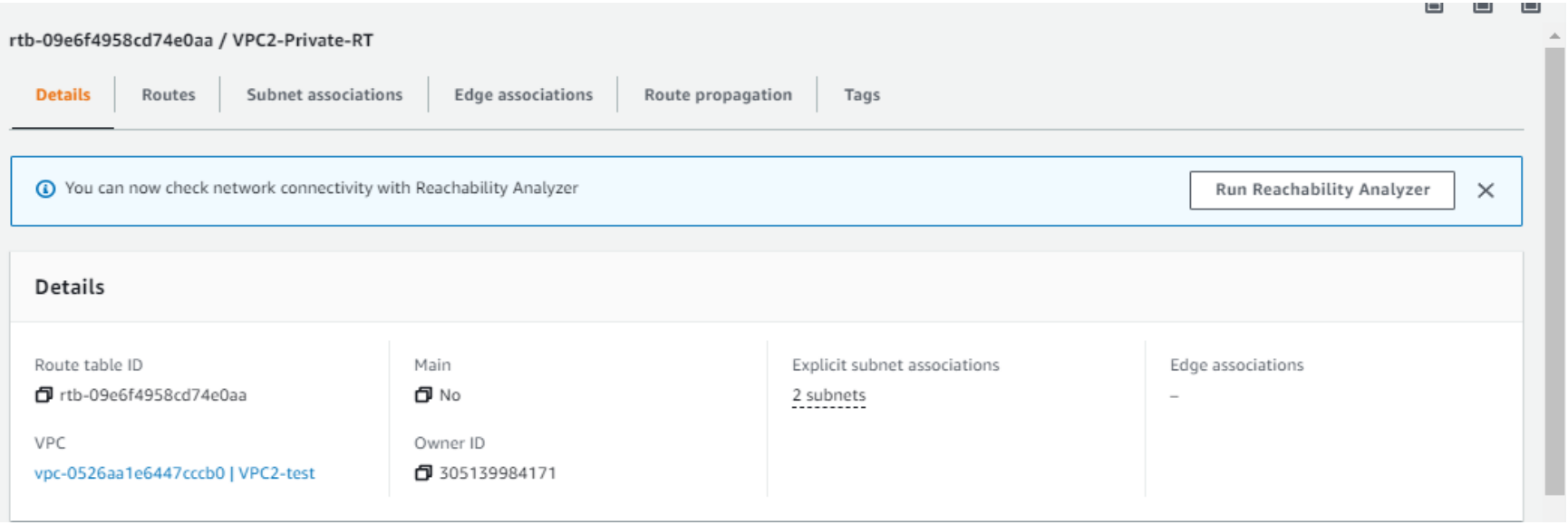

for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-4171
Route tables (1/7) Info
Actions
Create route table
Filter route tables
1
Name
Route table ID
Explicit subnet associat...
Edge associations
Main
VPC
Owner ID
VPC2-Public-RT rtb-0efb9e3a9c3e61e48 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
– rtb-086ac002b1a90090a – – Yes vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Public-... rtb-0aa89a494c54bfd64 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-04251ef9510165f4c – – Yes vpc-0626e552532f8b7c4 305139984...
VPC2-Private-RT rtb-09e6f4958cd74e0aa 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Private... rtb-050d3807a95f12092 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-0c95a688123c0fe85 – – Yes vpc-05d3b8371e4a72cae | VP... 305139984...
rtb-0efb9e3a9c3e61e48 / VPC2-Public-RT
Details
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (2)
Edit subnet associations
Find subnet association
1
Subnet ID
IPv4 CIDR
IPv6 CIDR
subnet-044d6cbc13d242cc8 / VPC2-Public-SN1 192.168.1.0/24 –
subnet-057c18508a27a172a / VPC2-Public-SN2 192.168.2.0/24 –



Showing we are connected to the right VPC (VPC2-test)

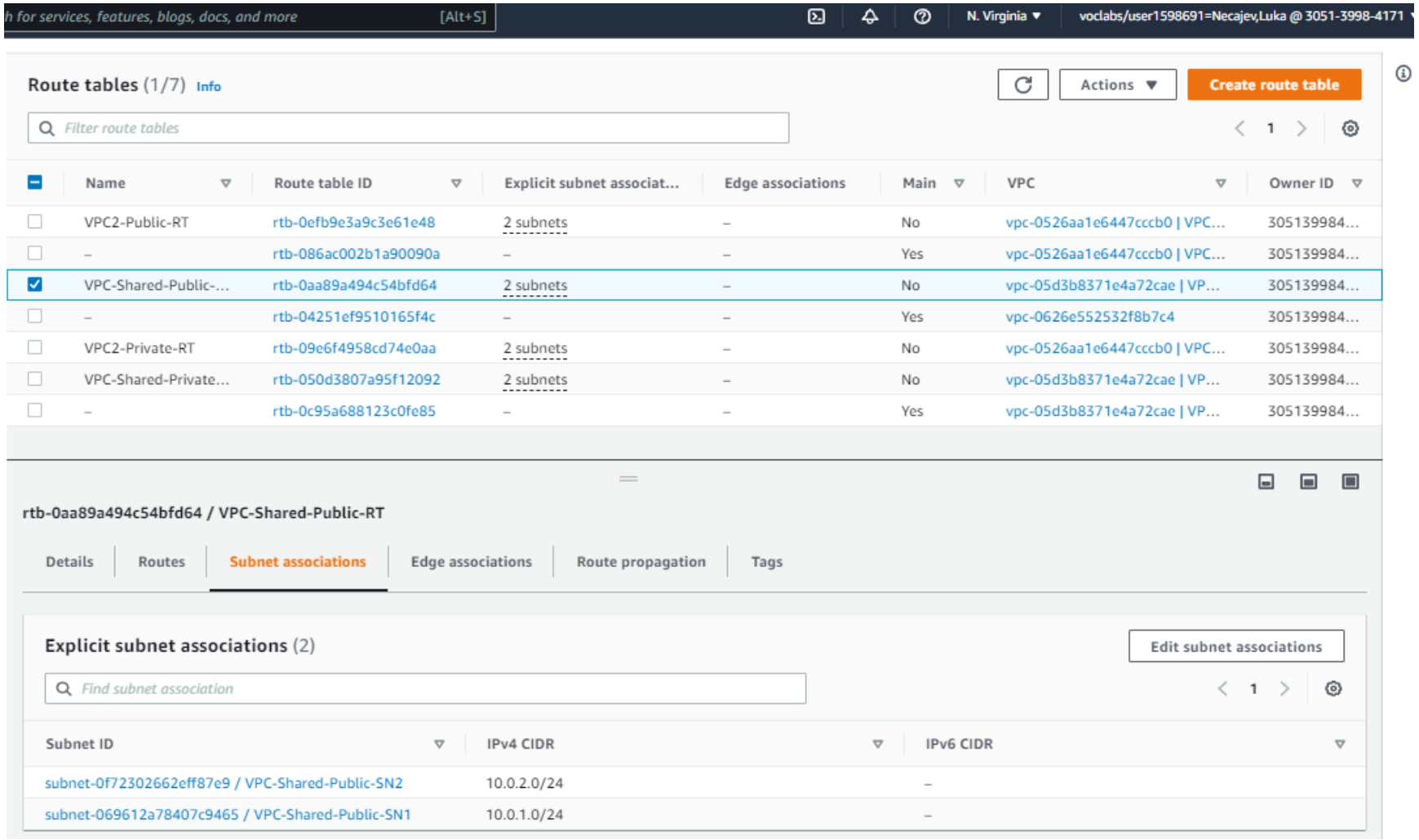
VPC2 Private Routing table associations

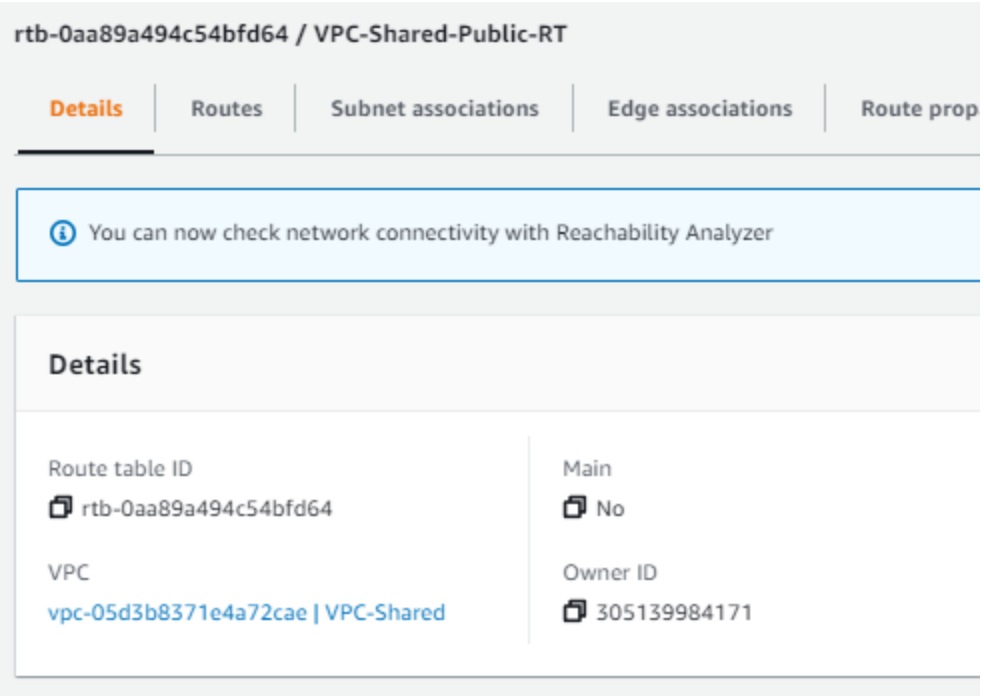

ch for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-4171
Route tables (1/7) Info
Actions
Create route table
Filter route tables
1
Name
Route table ID
Explicit subnet associat...
Edge associations
Main
VPC
Owner ID
VPC2-Public-RT rtb-0efb9e3a9c3e61e48 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
– rtb-086ac002b1a90090a – – Yes vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Public-... rtb-0aa89a494c54bfd64 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-04251ef9510165f4c – – Yes vpc-0626e552532f8b7c4 305139984...
VPC2-Private-RT rtb-09e6f4958cd74e0aa 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Private... rtb-050d3807a95f12092 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-0c95a688123c0fe85 – – Yes vpc-05d3b8371e4a72cae | VP... 305139984...
rtb-09e6f4958cd74e0aa / VPC2-Private-RT
Details
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (2)
Edit subnet associations
Find subnet association
1
Subnet ID
IPv4 CIDR
IPv6 CIDR
subnet-035725cb5d4db3bc7 / VPC2-Private-SN1 192.168.3.0/24 –
subnet-0f027487a0801afcc / VPC2-Private-SN2 192.168.4.0/24 –



Showing we are connected to the right VPC (VPC2-test)

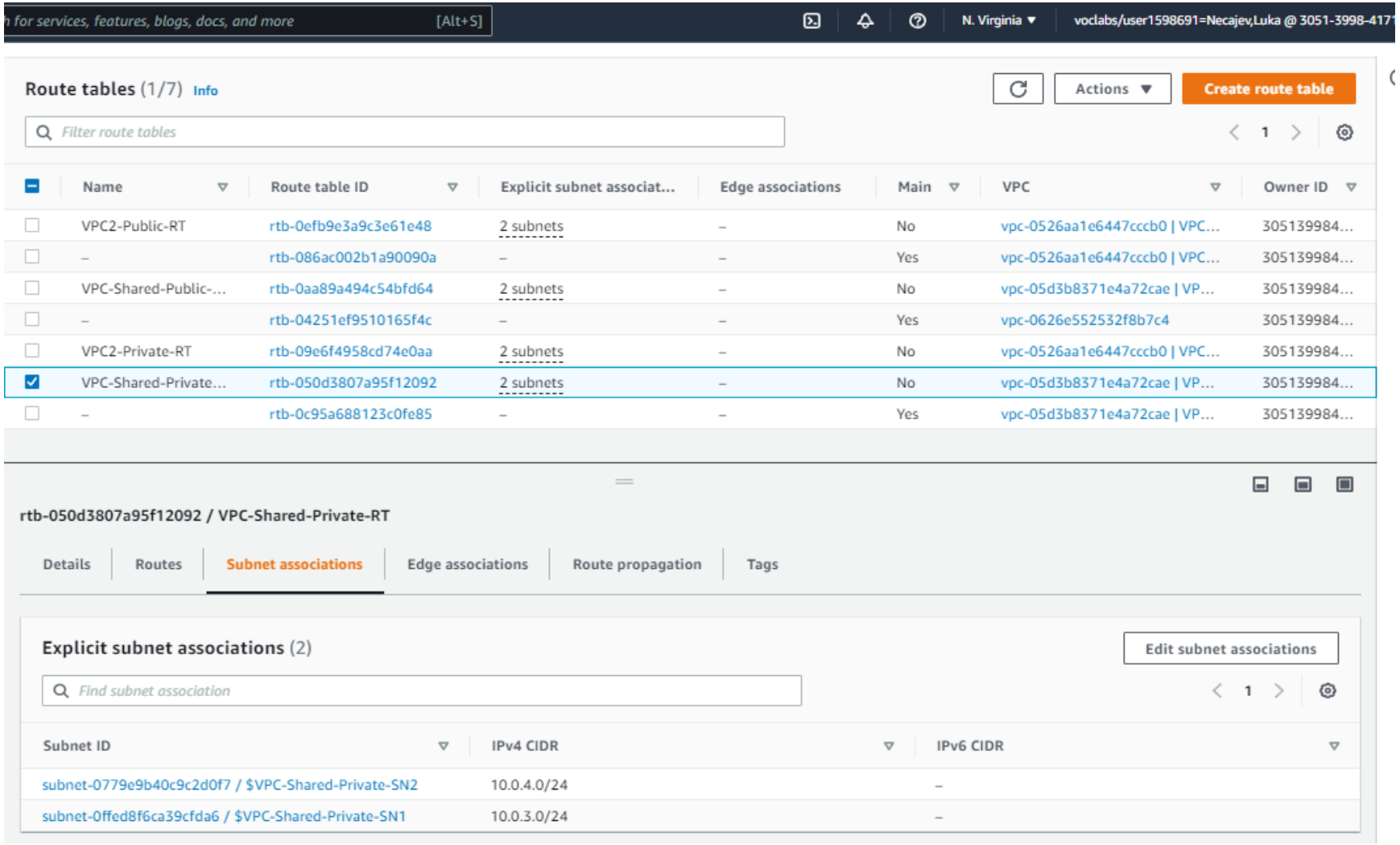
VPC1 Public Routing table associations

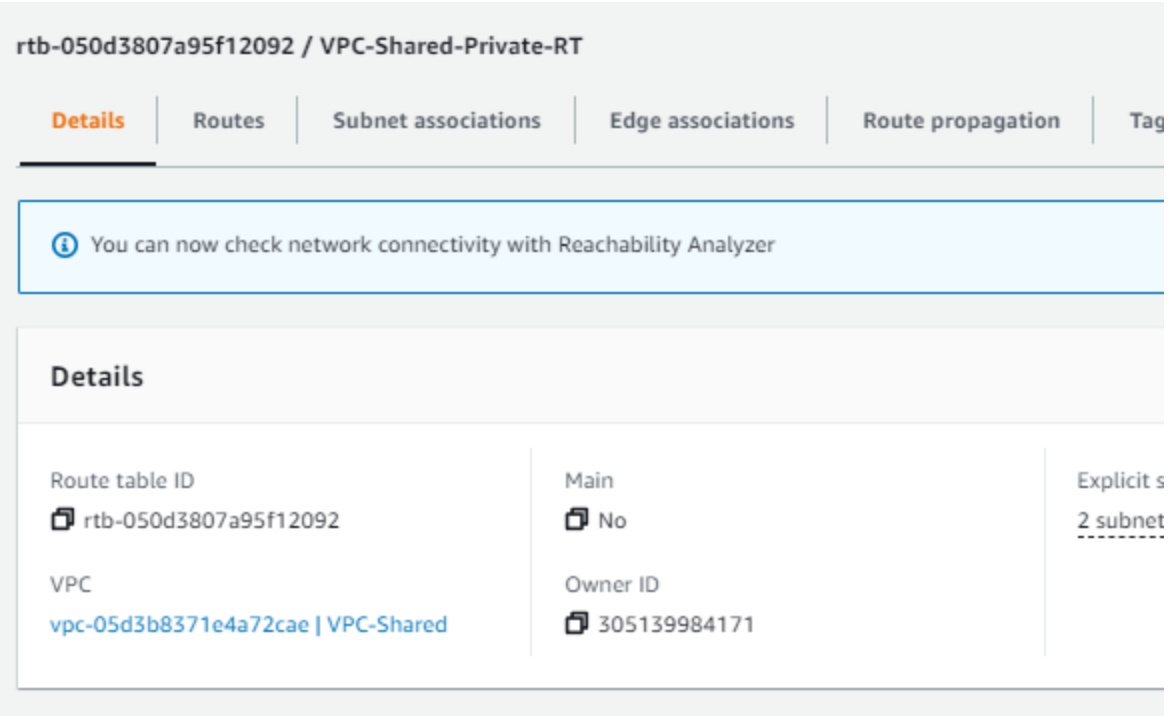

h for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-4171
Route tables (1/7) Info
Actions
Create route table
Filter route tables
1
Name
Route table ID
Explicit subnet associat...
Edge associations
Main
VPC
Owner ID
VPC2-Public-RT rtb-0efb9e3a9c3e61e48 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
– rtb-086ac002b1a90090a – – Yes vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Public-... rtb-0aa89a494c54bfd64 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-04251ef9510165f4c – – Yes vpc-0626e552532f8b7c4 305139984...
VPC2-Private-RT rtb-09e6f4958cd74e0aa 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Private... rtb-050d3807a95f12092 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-0c95a688123c0fe85 – – Yes vpc-05d3b8371e4a72cae | VP... 305139984...
rtb-0aa89a494c54bfd64 / VPC-Shared-Public-RT
Details
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (2)
Edit subnet associations
Find subnet association
1
Subnet ID
IPv4 CIDR
IPv6 CIDR
subnet-0f72302662eff87e9 / VPC-Shared-Public-SN2 10.0.2.0/24 –
subnet-069612a78407c9465 / VPC-Shared-Public-SN1 10.0.1.0/24 –



Showing we are connected to the right VPC (VPC-Shared)

VPC1 Private Routing table associations


for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-417
Route tables (1/7) Info
Actions
Create route table
Filter route tables
1
Name
Route table ID
Explicit subnet associat...
Edge associations
Main
VPC
Owner ID
VPC2-Public-RT rtb-0efb9e3a9c3e61e48 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
– rtb-086ac002b1a90090a – – Yes vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Public-... rtb-0aa89a494c54bfd64 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-04251ef9510165f4c – – Yes vpc-0626e552532f8b7c4 305139984...
VPC2-Private-RT rtb-09e6f4958cd74e0aa 2 subnets – No vpc-0526aa1e6447cccb0 | VPC... 305139984...
VPC-Shared-Private... rtb-050d3807a95f12092 2 subnets – No vpc-05d3b8371e4a72cae | VP... 305139984...
– rtb-0c95a688123c0fe85 – – Yes vpc-05d3b8371e4a72cae | VP... 305139984...
rtb-050d3807a95f12092 / VPC-Shared-Private-RT
Details
Routes
Subnet associations
Edge associations
Route propagation
Tags
Explicit subnet associations (2)
Edit subnet associations
Find subnet association
1
Subnet ID
IPv4 CIDR
IPv6 CIDR
subnet-0779e9b40c9c2d0f7 / $VPC-Shared-Private-SN2 10.0.4.0/24 –
subnet-0ffed8f6ca39cfda6 / $VPC-Shared-Private-SN1 10.0.3.0/24 –



Showing we are connected to the right VPC (VPC-Shared)

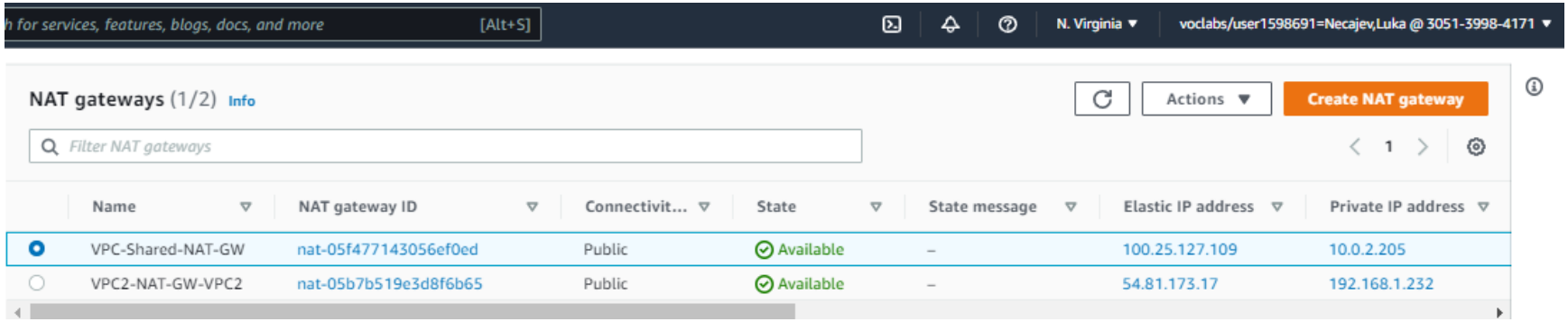
4. **NAT**

Showing Code Creating NATS

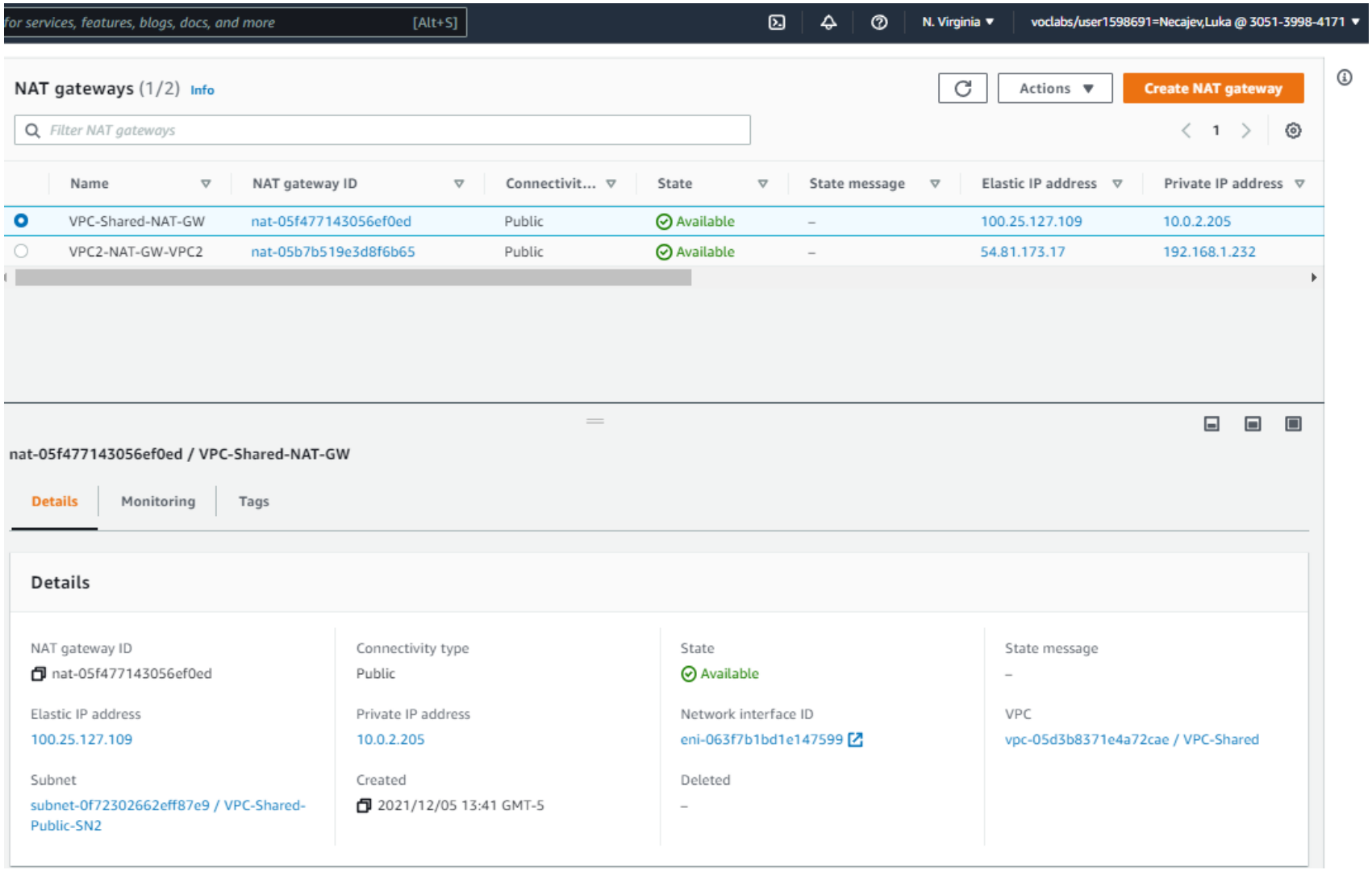
```
63  # Create NAT GW
64  resource "aws_nat_gateway" "nat" {
65    allocation_id = aws_eip.nat-eip.id
66    subnet_id     = aws_subnet.public[1].id
67
68    tags = merge(
69      var.default_tags,
70      {
71        Name = "VPC-Shared-NAT-GW"
72      }
73    )
74  }
75
```

```
58  # Create EIP
59  resource "aws_eip" "nat-eip" {
60    vpc = true
61  }
62  # Create NAT GW
63  resource "aws_nat_gateway" "nat" {
64    allocation_id = aws_eip.nat-eip.id
65    subnet_id     = aws_subnet.public[0].id
66
67    tags = merge(
68      var.default_tags,
69      {
70        Name = "${var.prefix}-NAT-GW-VPC2"
71      }
72    )
73  }
```

Showing all NATS

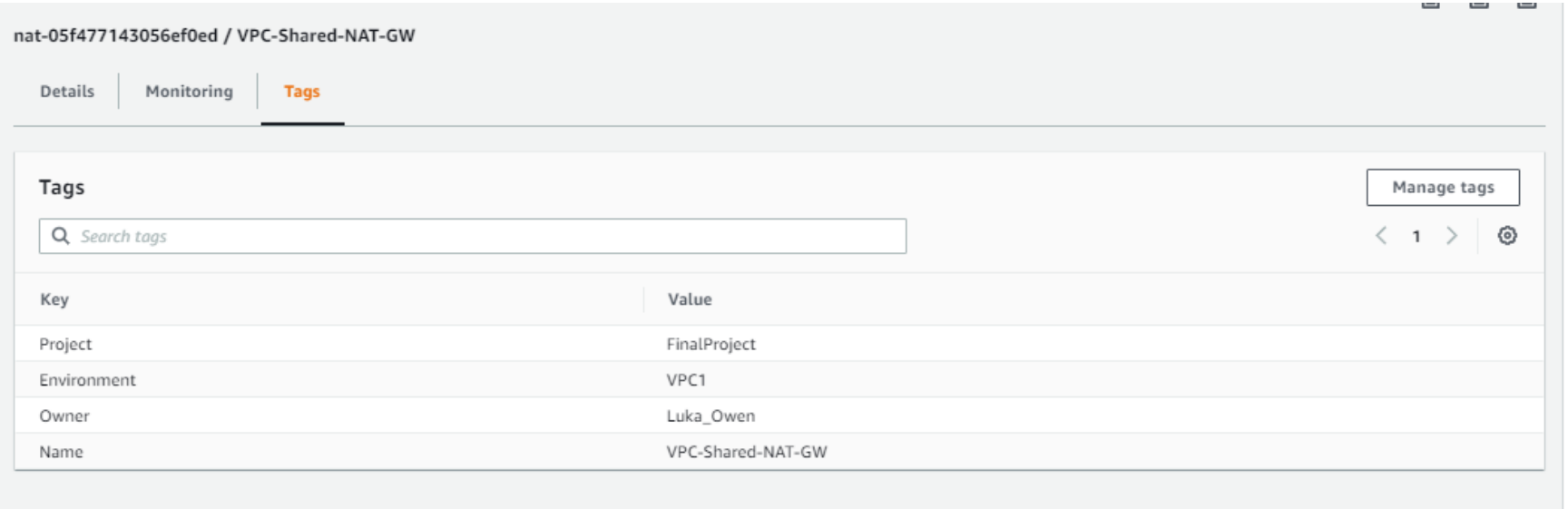


Shoiwng Nat For VPC1

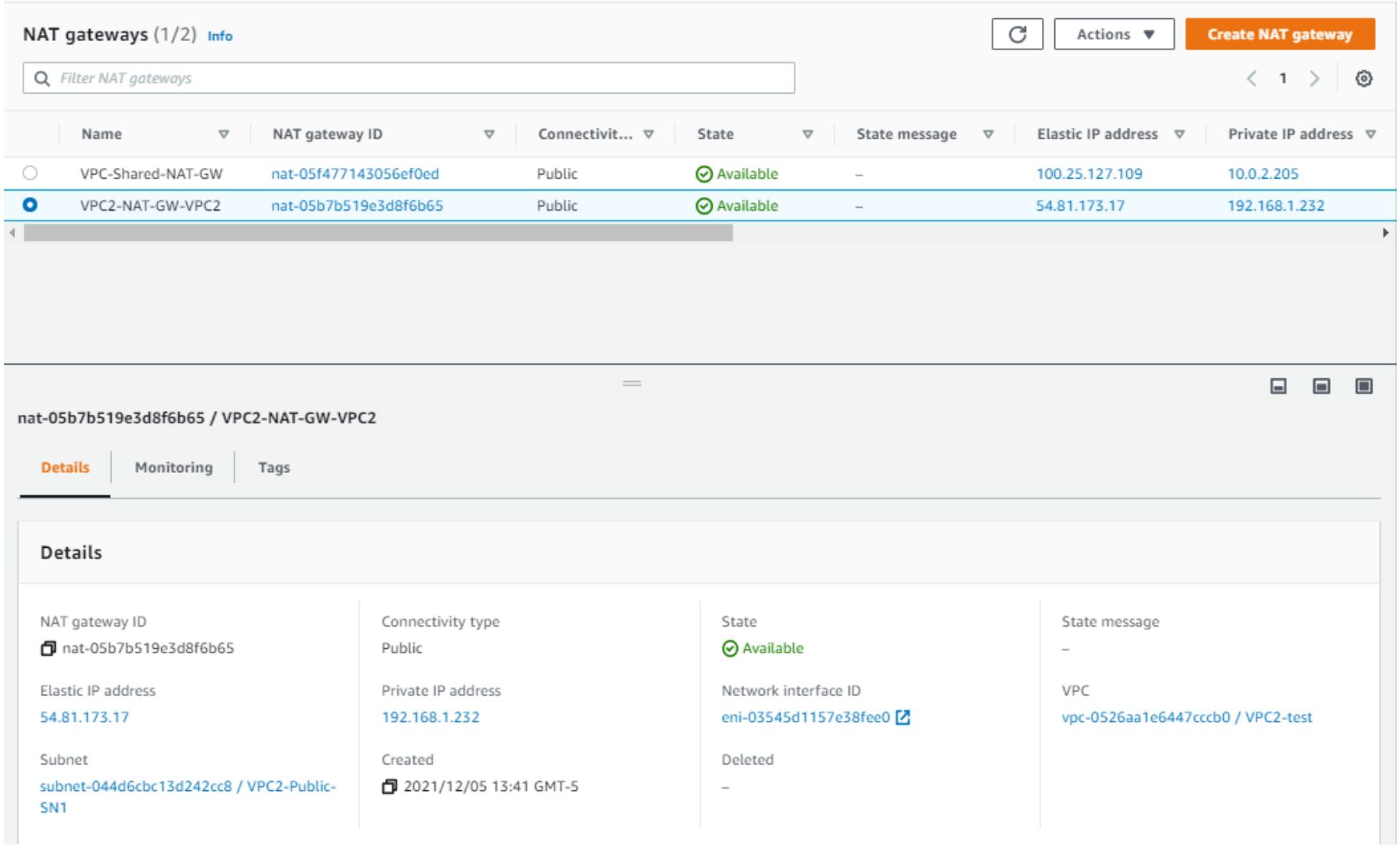


Making sure its connected to VPC-Shared

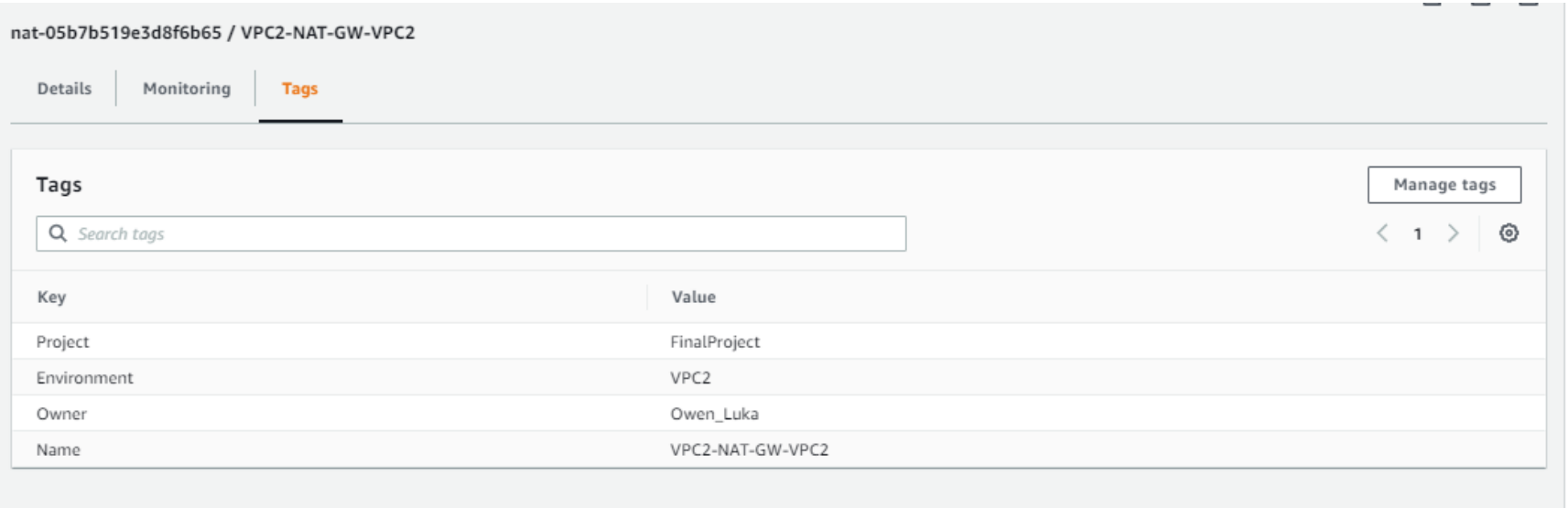
Showing the NAT tags



Showing NAT for VPC2



Making sure its connected to VPC2-Test

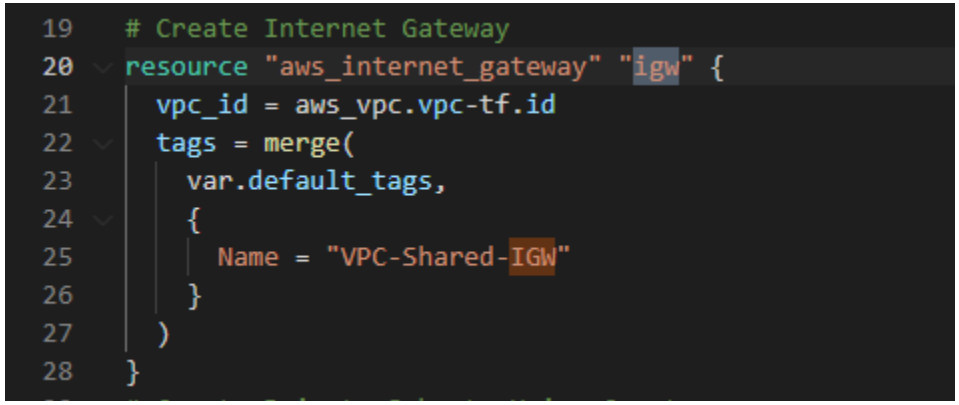


Showing the NAT tags

5. **Internet gateway**

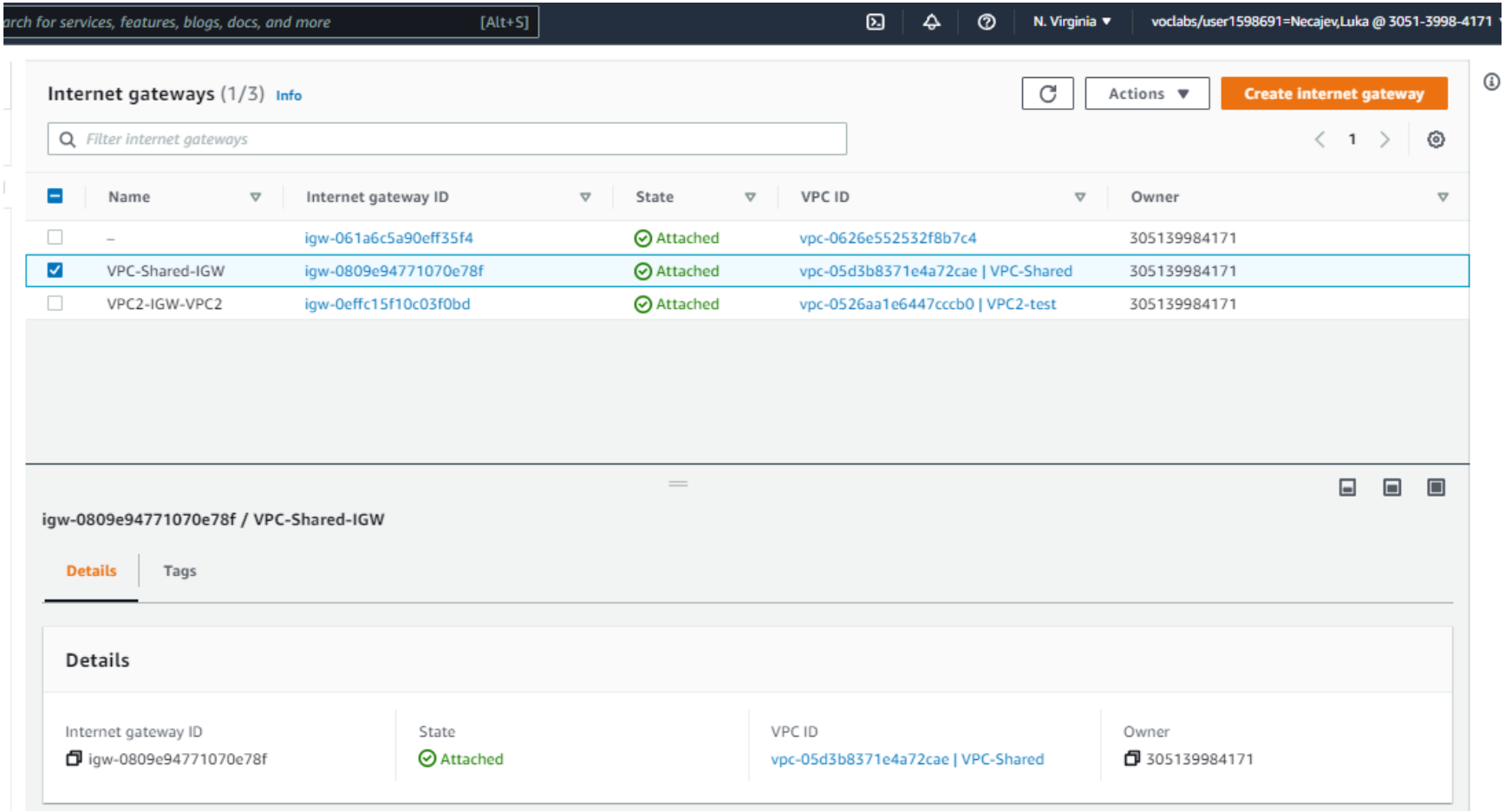
Showing Code Creating Internet Gateway

```
19  # Create Internet Gateway
20  resource "aws_internet_gateway" "igw" {
21    vpc_id = aws_vpc.vpc-tf.id
22    tags = merge(
23      var.default_tags,
24      {
25        Name = "${var.prefix}-IGW-VPC2"
26      }
27    )
28  }
```



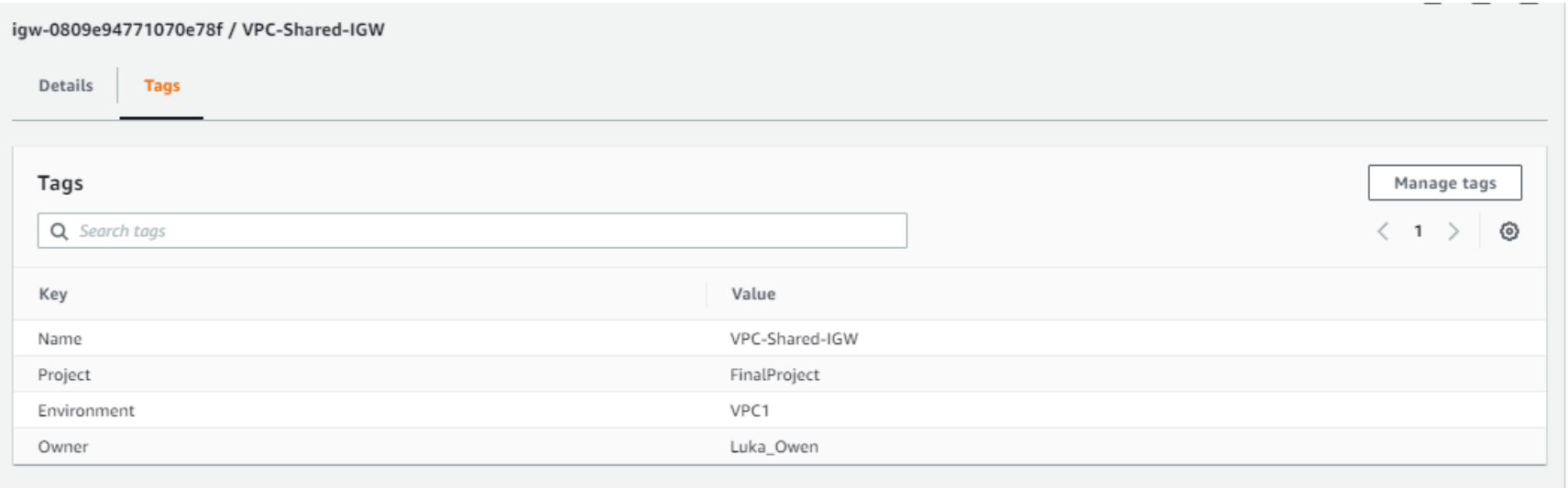
```
19  # Create Internet Gateway
20  resource "aws_internet_gateway" "igw" {
21    vpc_id = aws_vpc.vpc-tf.id
22    tags = merge(
23      var.default_tags,
24      {
25        Name = "VPC-Shared-IGW"
26      }
27    )
28  }
```

Showing Internet Gateway for VPC1

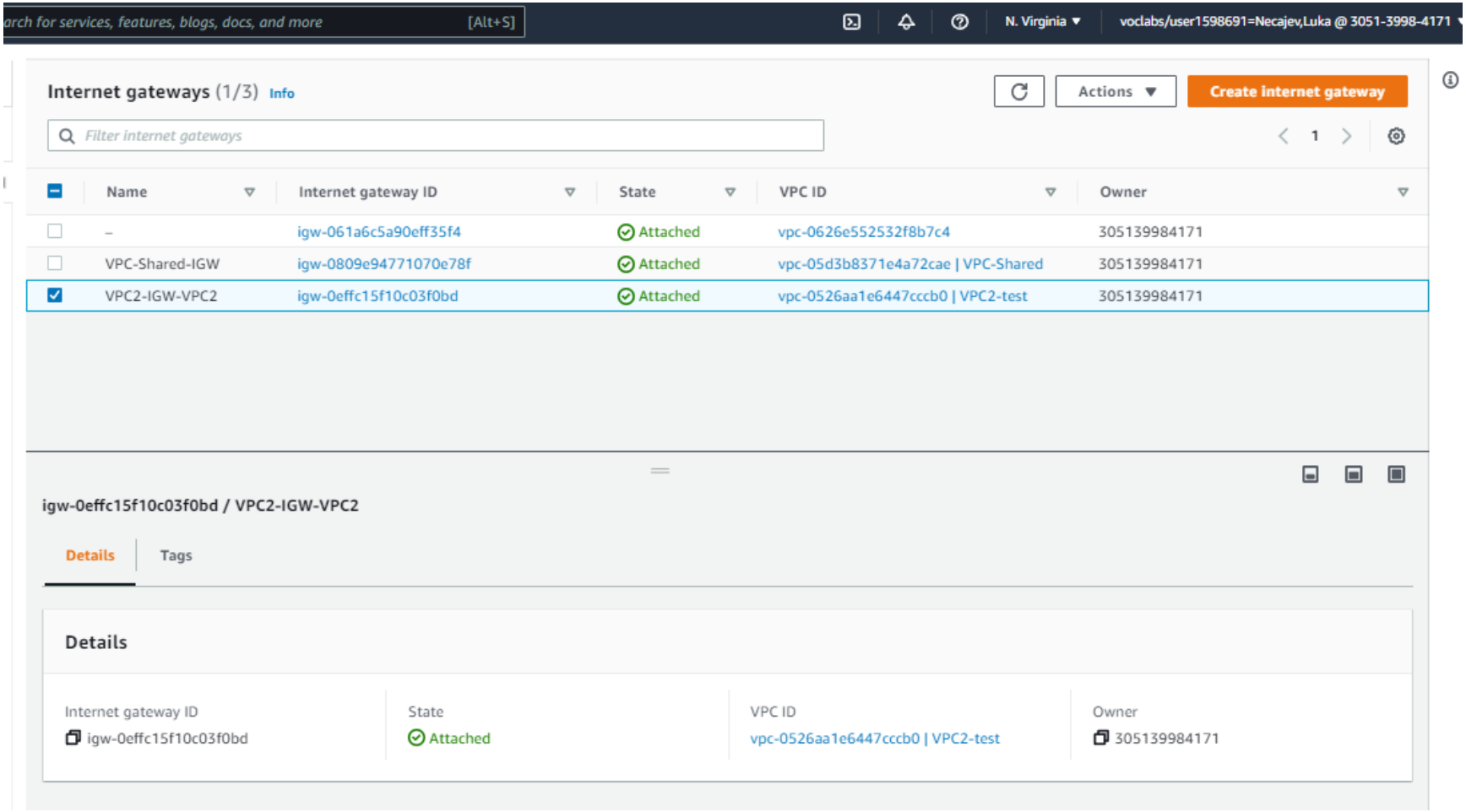


Making sure its connected to VPC-Shared

Showing related tags

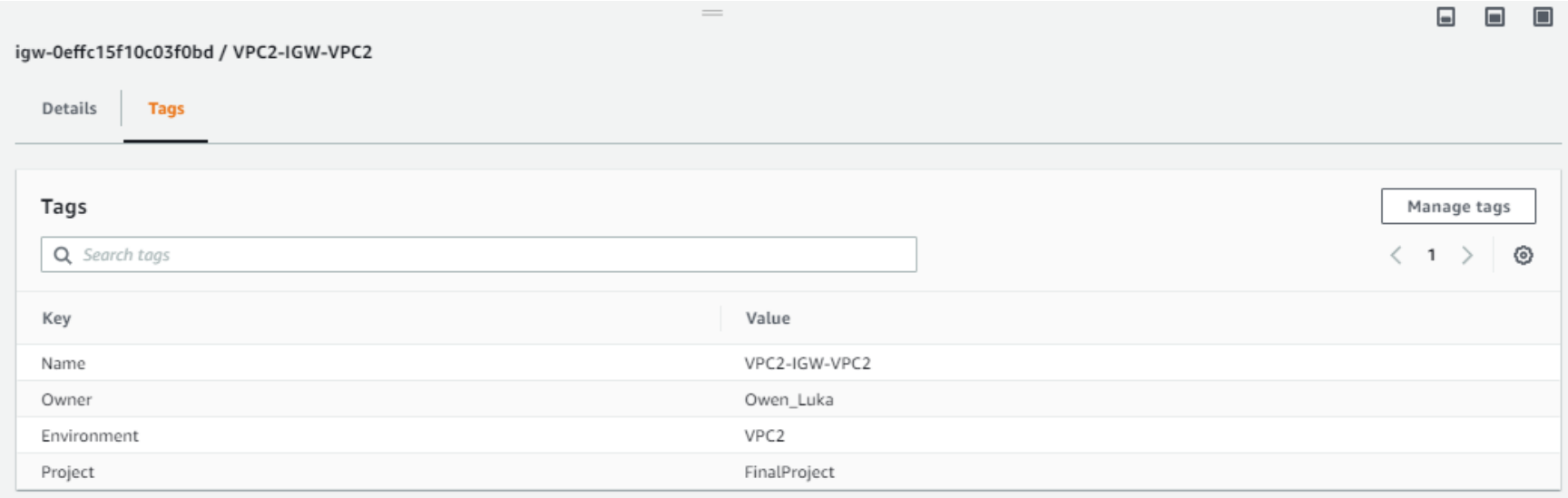


Showing Internet Gateway for VPC2

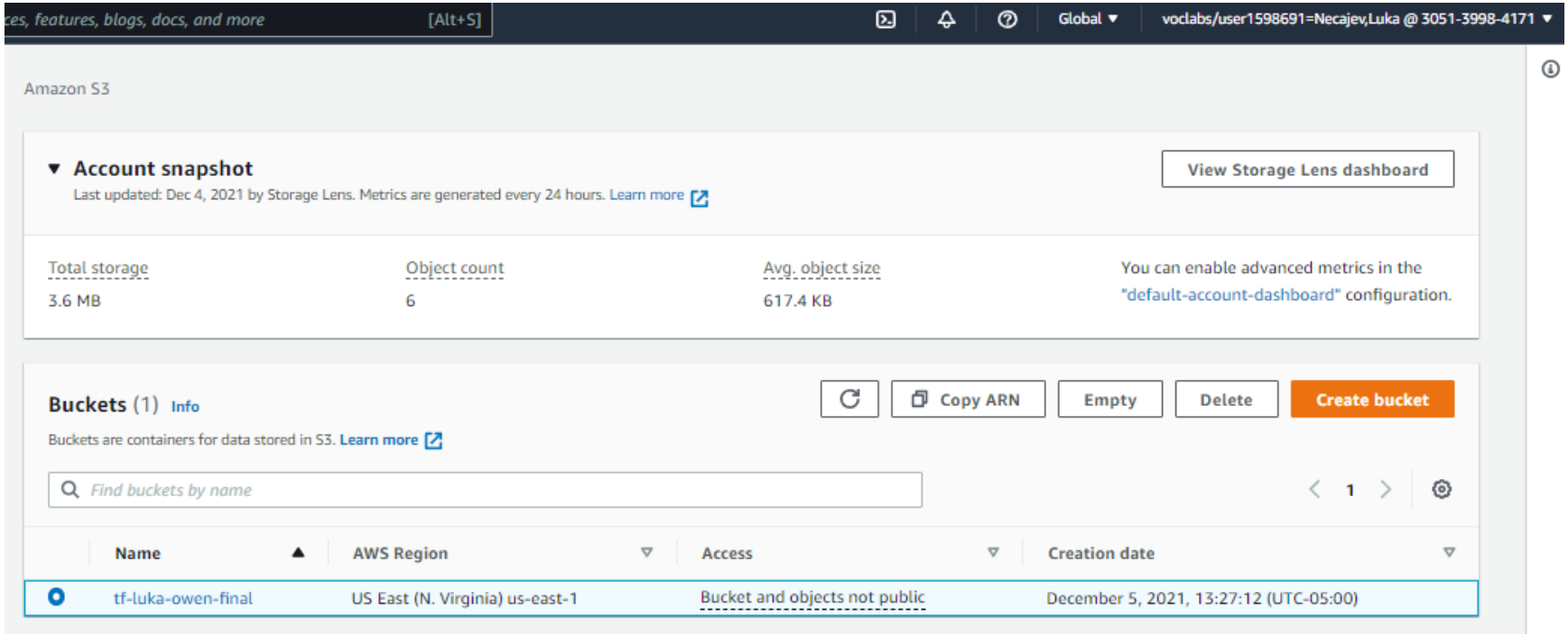


Making sure its connected to VPC2-Test

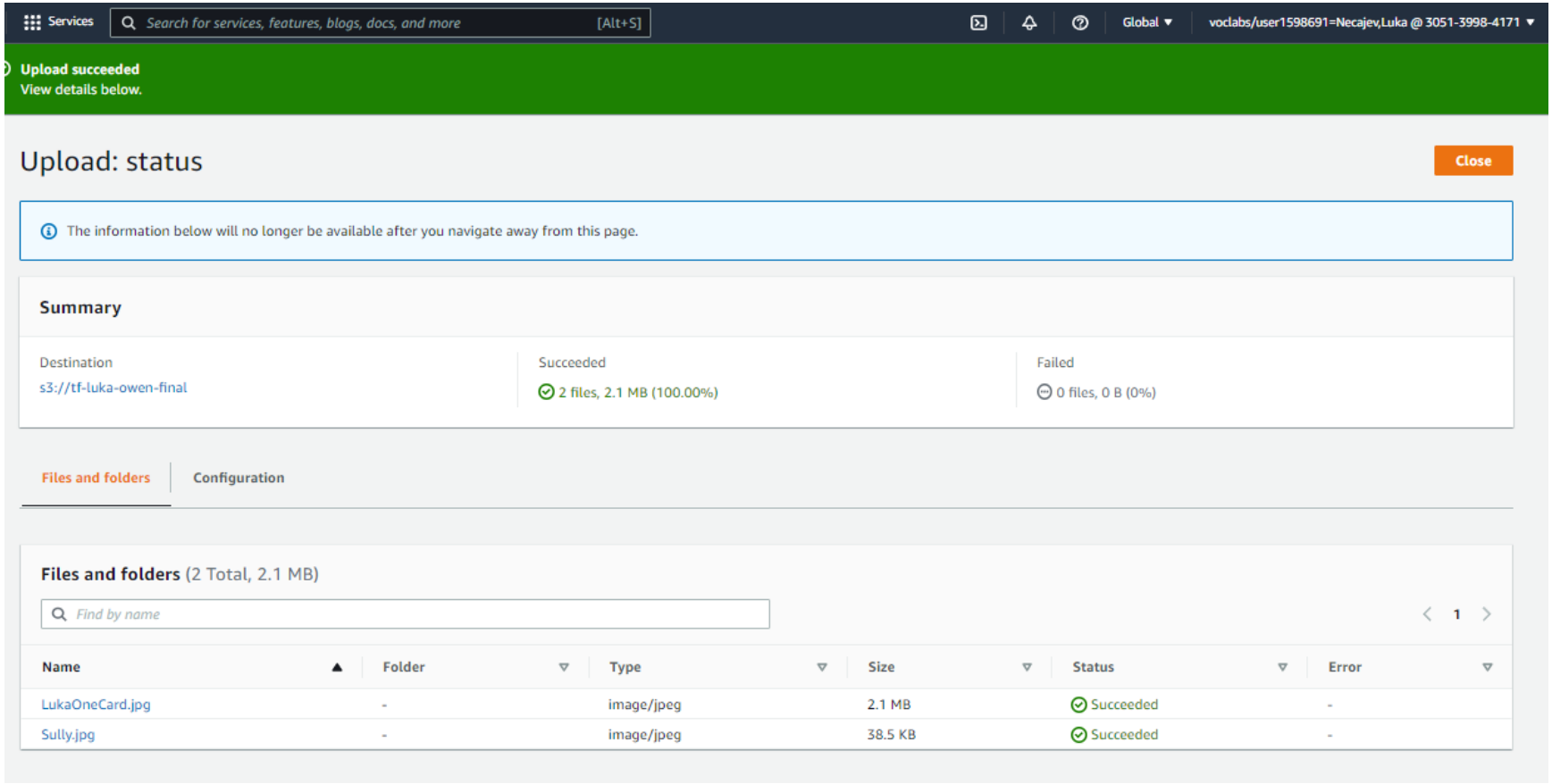
Showing related tags



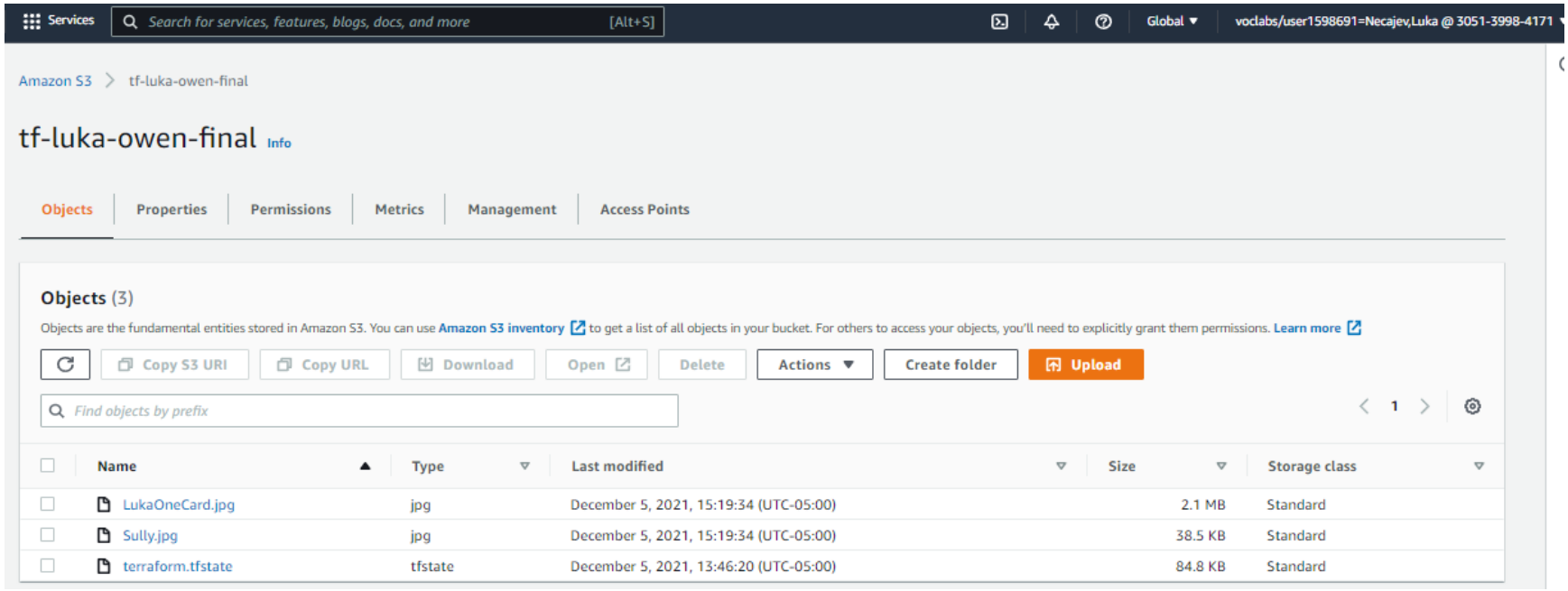
6. S3 bucket
    a. an object uploaded to the bucket,



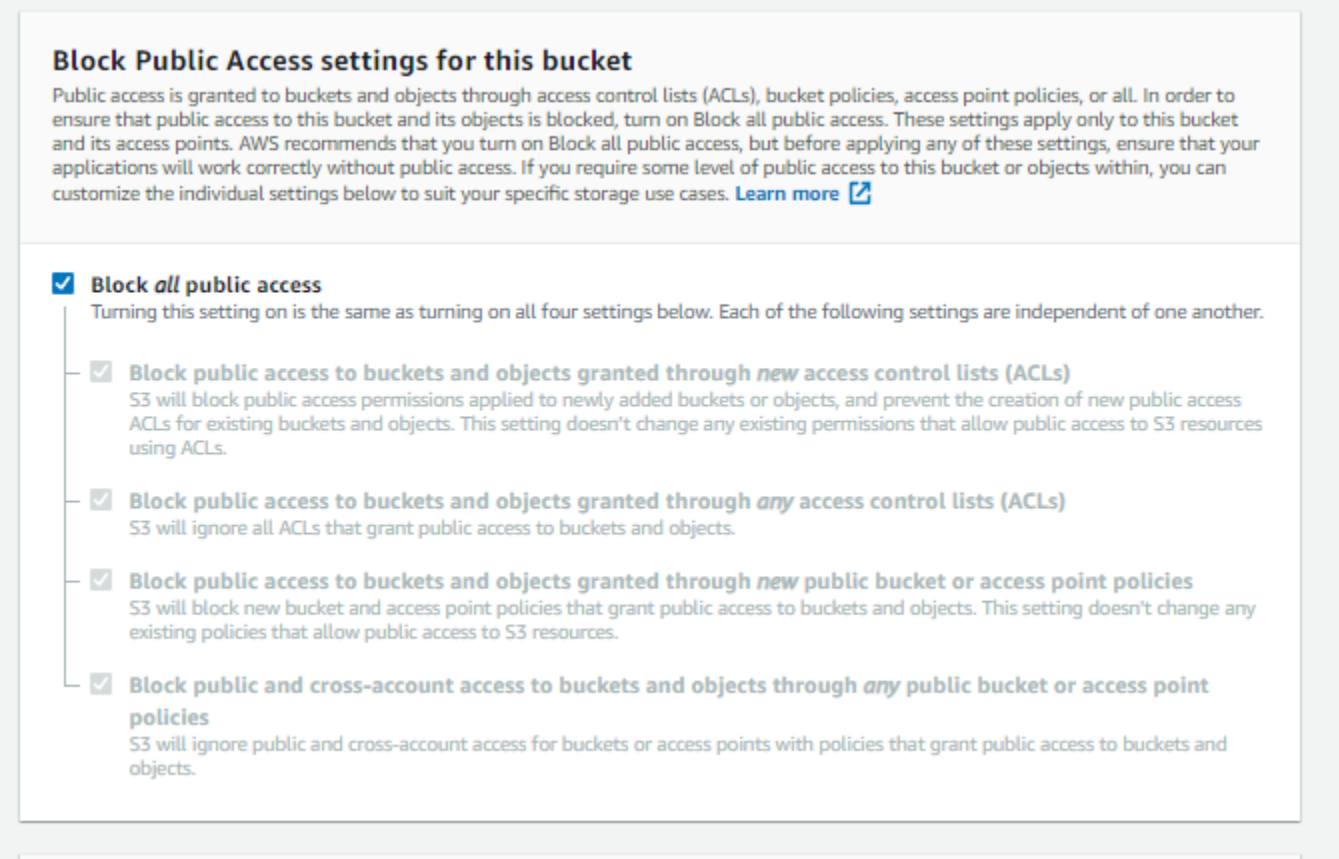
Adding the images



Showing the images and RF file


Services
Search for services, features, blogs, docs, and more [Alt+S]
Global
voclabs/user1598691=Necajev,Luka @ 3051-3998-4171
Amazon S3 > tf-luka-owen-final
tf-luka-owen-final Info
Objects
Properties
Permissions
Metrics
Management
Access Points
Objects (3)
Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more
Copy S3 URI
Copy URL
Download
Open
Delete
Actions
Create folder
Upload
Find objects by prefix
1
Name
Type
Last modified
Size
Storage class
LukaOneCard.jpg jpg December 5, 2021, 15:19:34 (UTC-05:00) 2.1 MB Standard
Sully.jpg jpg December 5, 2021, 15:19:34 (UTC-05:00) 38.5 KB Standard
terraform.tfstate tfstate December 5, 2021, 13:46:20 (UTC-05:00) 84.8 KB Standard

The bucket should be accessible by identities in your account only. The bucket should not be publicly accessible.

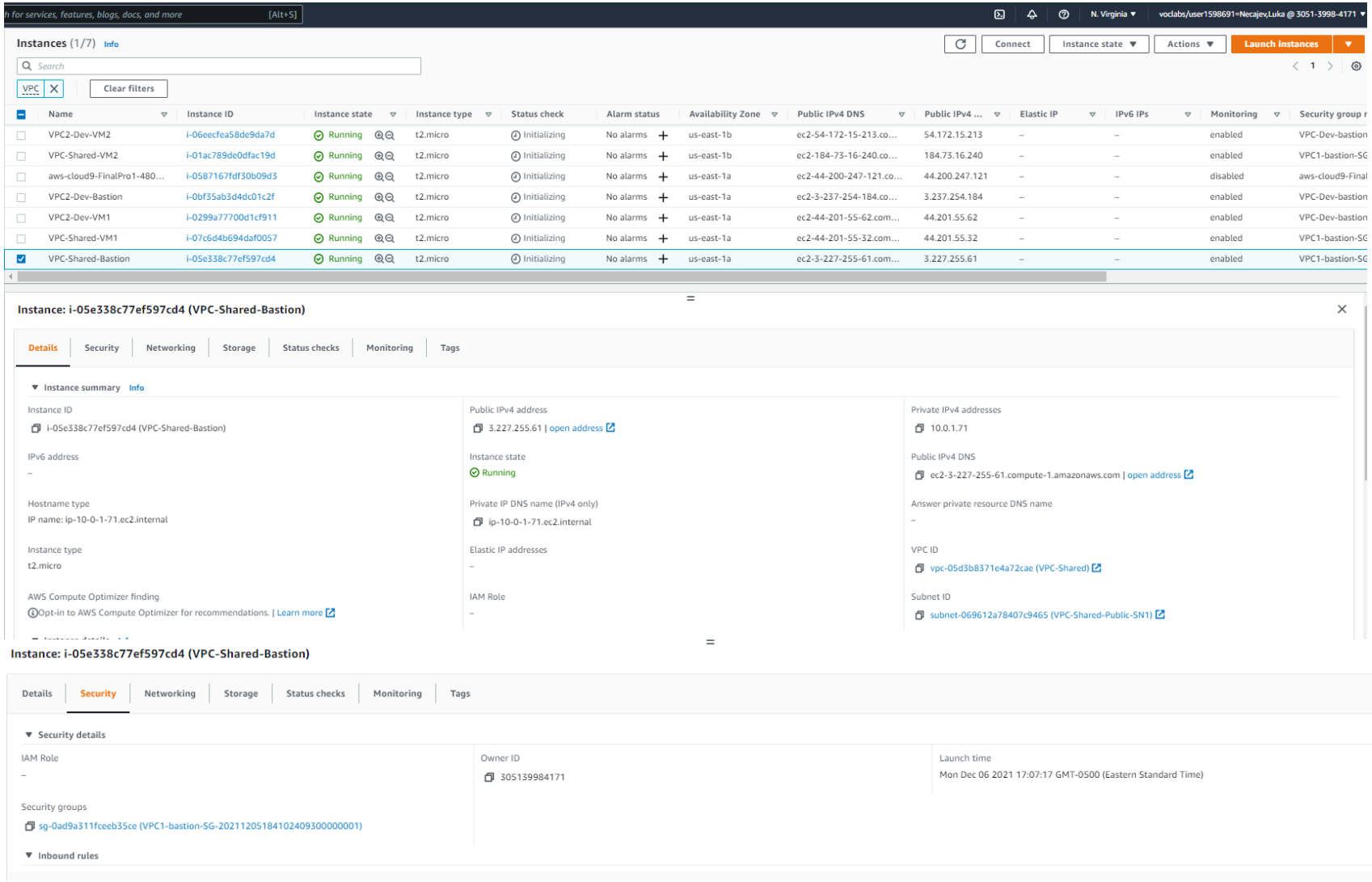


Showing the bucket has all public access blocked

7. **Bastion hosts and VM's should be deployed via Terraform**.

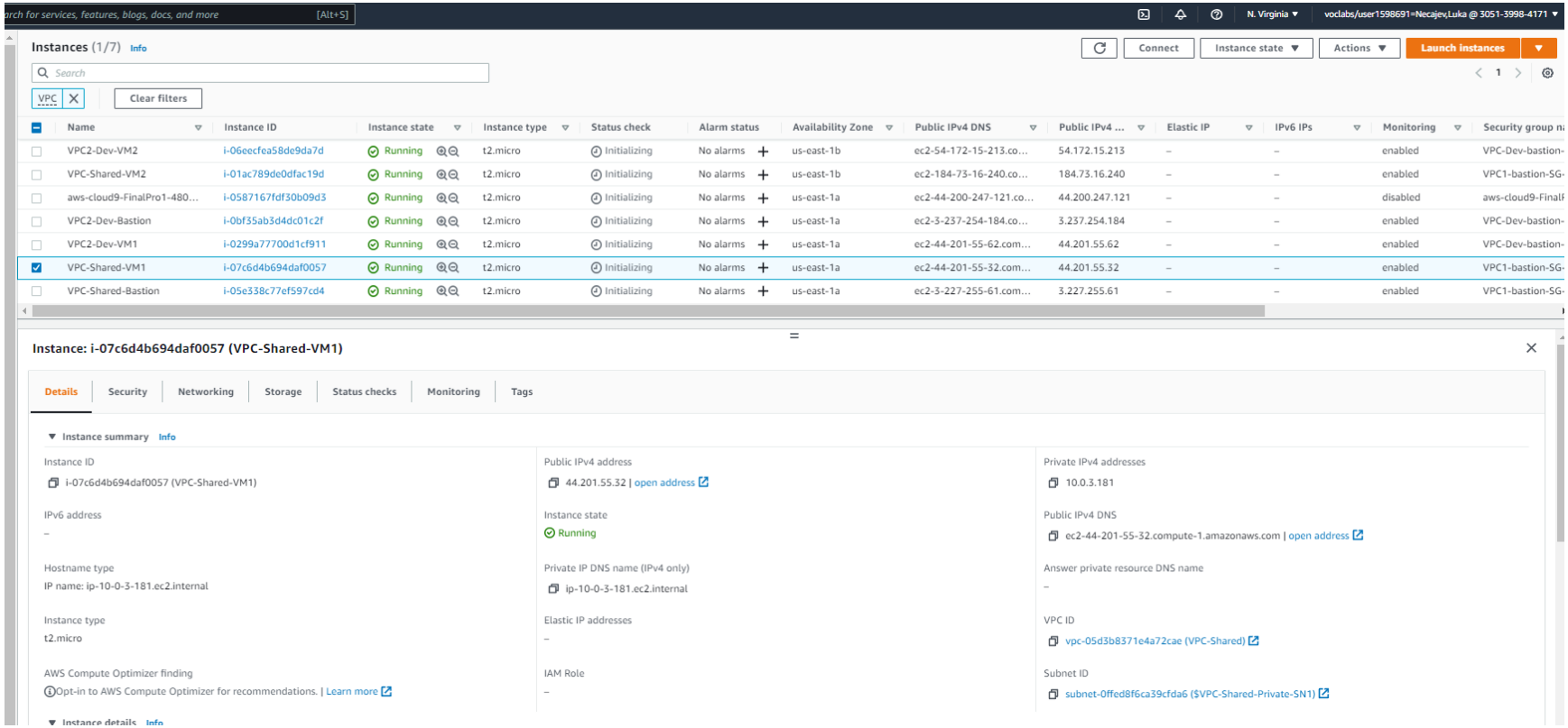
8. **All Instance photo's as proof they were created.**

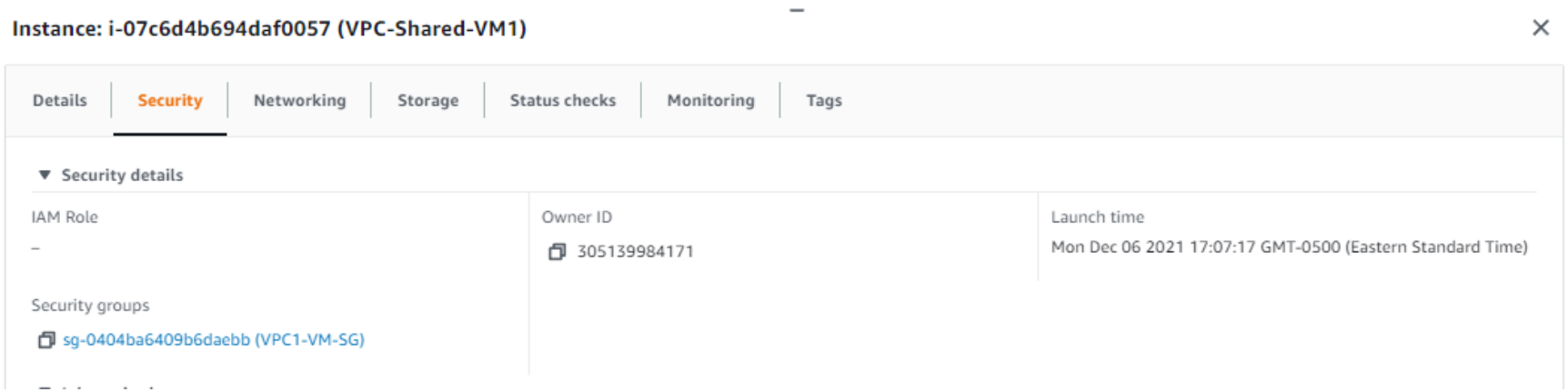
VPC1
VPC-Shared-Bastion



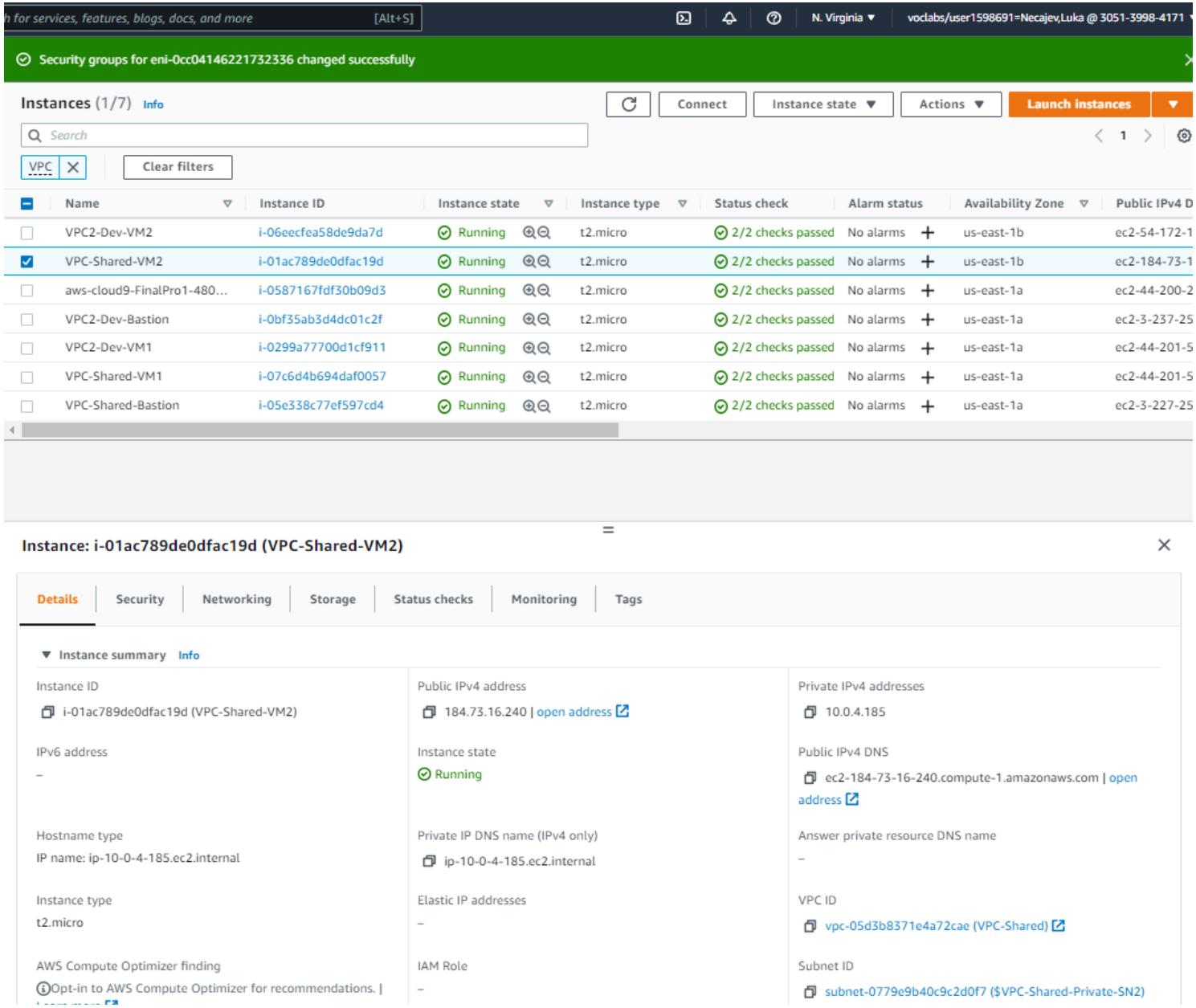
Notice that we are in the correct security group (VPC1-bastion-SG)

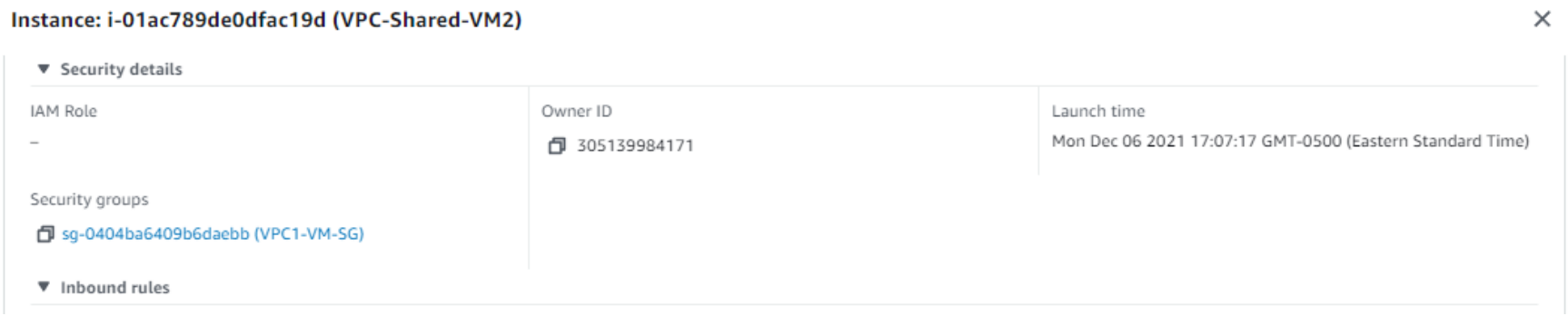
VPC-Shared-VM1





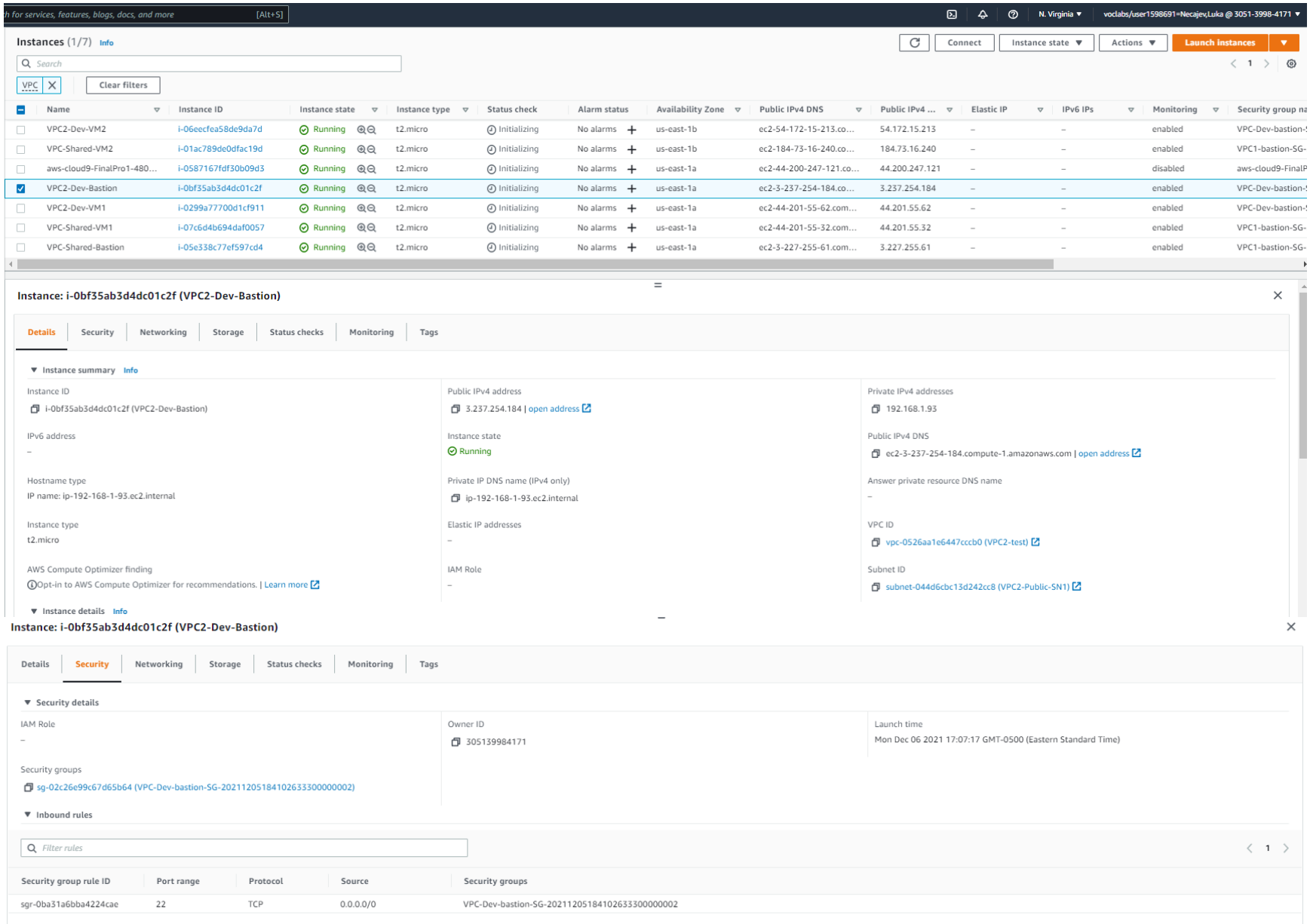
VPC-Shared-VM2


for services, features, blogs, docs, and more [Alt+S]
N. Virginia
voclabs/user1598691=Necajev,Luka @ 3051-3998-4171
Security groups for eni-0cc04146221732336 changed successfully
Instances (1/7) Info
Connect
Instance state
Actions
Launch instances
Search
1
VPC
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 D
VPC2-Dev-VM2 i-06eecfea58de9da7d Running t2.micro 2/2 checks passed No alarms us-east-1b ec2-54-172-1
VPC-Shared-VM2 i-01ac789de0dfac19d Running t2.micro 2/2 checks passed No alarms us-east-1b ec2-184-73-1
aws-cloud9-FinalPro1-480... i-0587167fdf30b09d3 Running t2.micro 2/2 checks passed No alarms us-east-1a ec2-44-200-2
VPC2-Dev-Bastion i-0bf35ab3d4dc01c2f Running t2.micro 2/2 checks passed No alarms us-east-1a ec2-3-237-25
VPC2-Dev-VM1 i-0299a77700d1cf911 Running t2.micro 2/2 checks passed No alarms us-east-1a ec2-44-201-5
VPC-Shared-VM1 i-07c6d4b694daf0057 Running t2.micro 2/2 checks passed No alarms us-east-1a ec2-44-201-5
VPC-Shared-Bastion i-05e338c77ef597cd4 Running t2.micro 2/2 checks passed No alarms us-east-1a ec2-3-227-25
Instance: i-01ac789de0dfac19d (VPC-Shared-VM2)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-01ac789de0dfac19d (VPC-Shared-VM2)
Public IPv4 address
184.73.16.240 | open address
Private IPv4 addresses
10.0.4.185
IPv6 address
–
Instance state
Running
Public IPv4 DNS
ec2-184-73-16-240.compute-1.amazonaws.com | open address
Hostname type
IP name: ip-10-0-4-185.ec2.internal
Private IP DNS name (IPv4 only)
ip-10-0-4-185.ec2.internal
Answer private resource DNS name
–
Instance type
t2.micro
Elastic IP addresses
–
VPC ID
vpc-05d3b8371e4a72cae (VPC-Shared)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. |
IAM Role
–
Subnet ID
subnet-0779e9b40c9c2d0f7 ($VPC-Shared-Private-SN2)



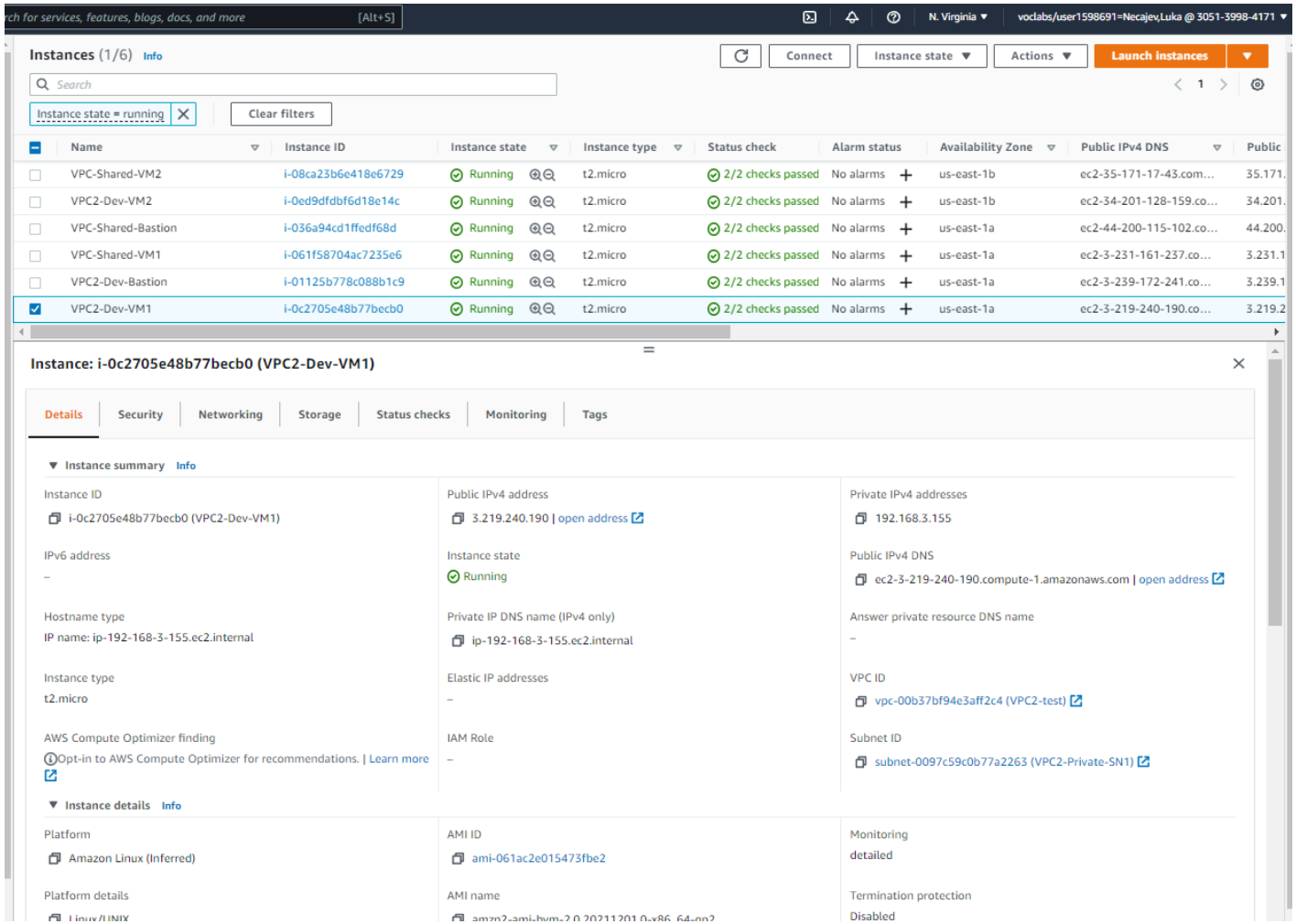
VPC2

VPC-Dev-Bastion

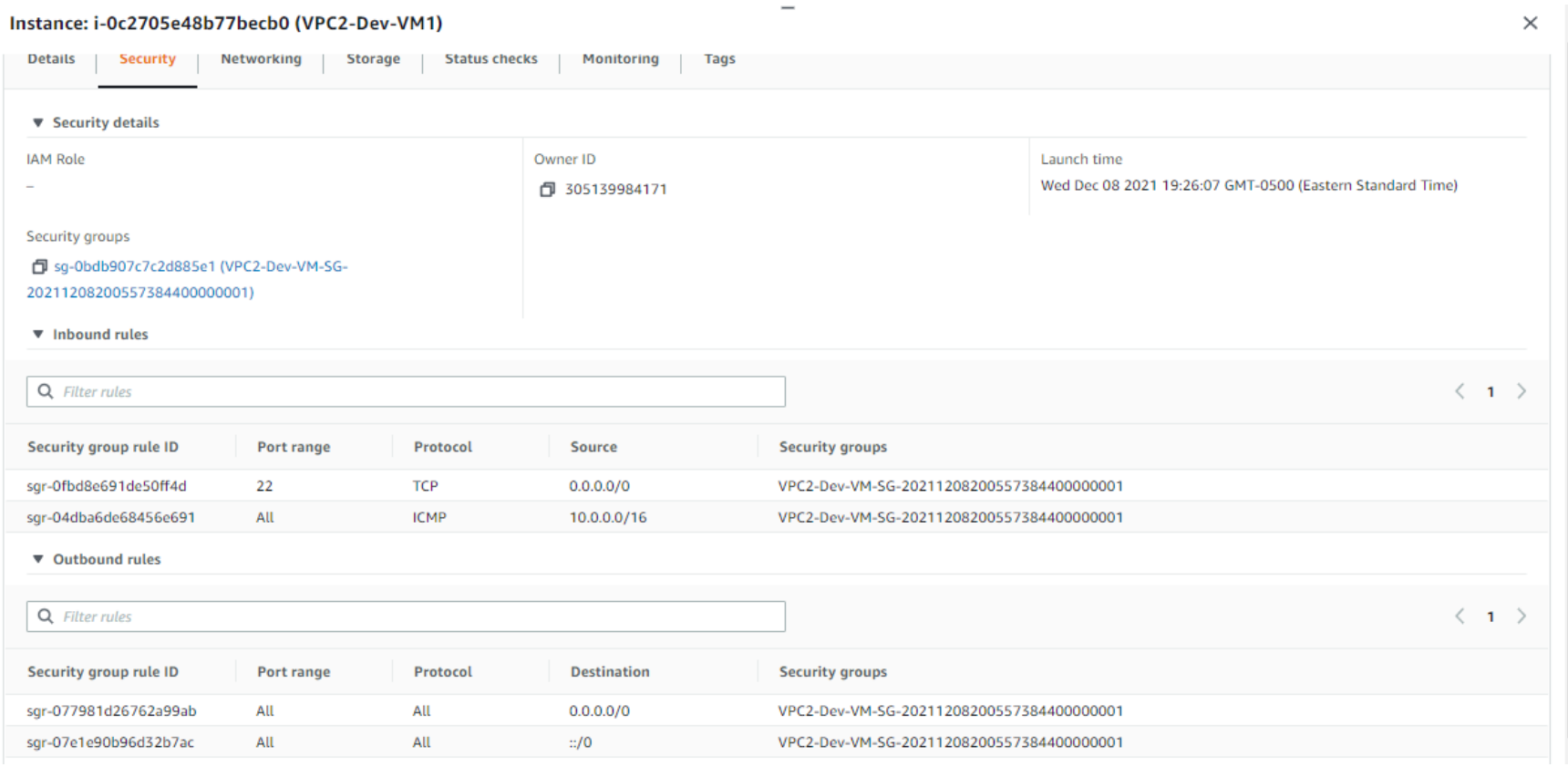

Instances (1/7) Info
Connect
Instance state
Actions
Launch instances
Search
Clear filters
Name
Instance ID
Instance state
Instance type
Status check
Alarm status
Availability Zone
Public IPv4 DNS
Public IPv4 ...
Elastic IP
IPv6 IPs
Monitoring
Security group na
VPC2-Dev-VM2 i-06eecfea58de9da7d Running t2.micro Initializing No alarms us-east-1b ec2-54-172-15-213.co... 54.172.15.213 – – enabled VPC-Dev-bastion-
VPC-Shared-VM2 i-01ac789de0dfac19d Running t2.micro Initializing No alarms us-east-1b ec2-184-73-16-240.co... 184.73.16.240 – – enabled VPC1-bastion-SG-
aws-cloud9-FinalPro1-480... i-0587167fdf30b09d3 Running t2.micro Initializing No alarms us-east-1a ec2-44-200-247-121.co... 44.200.247.121 – – disabled aws-cloud9-FinalP
VPC2-Dev-Bastion i-0bf35ab3d4dc01c2f Running t2.micro Initializing No alarms us-east-1a ec2-3-237-254-184.co... 3.237.254.184 – – enabled VPC-Dev-bastion-
VPC2-Dev-VM1 i-0299a77700d1cf911 Running t2.micro Initializing No alarms us-east-1a ec2-44-201-55-62.com... 44.201.55.62 – – enabled VPC-Dev-bastion-
VPC-Shared-VM1 i-07c6d4b694daf0057 Running t2.micro Initializing No alarms us-east-1a ec2-44-201-55-32.com... 44.201.55.32 – – enabled VPC1-bastion-SG-
VPC-Shared-Bastion i-05e338c77ef597cd4 Running t2.micro Initializing No alarms us-east-1a ec2-3-227-255-61.com... 3.227.255.61 – – enabled VPC1-bastion-SG-
Instance: i-0bf35ab3d4dc01c2f (VPC2-Dev-Bastion)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Instance summary Info
Instance ID
i-0bf35ab3d4dc01c2f (VPC2-Dev-Bastion)
Public IPv4 address
3.237.254.184 | open address
Private IPv4 addresses
192.168.1.93
IPv6 address
–
Instance state
Running
Public IPv4 DNS
ec2-3-237-254-184.compute-1.amazonaws.com | open address
Hostname type
IP name: ip-192-168-1-93.ec2.internal
Private IP DNS name (IPv4 only)
ip-192-168-1-93.ec2.internal
Answer private resource DNS name
–
Instance type
t2.micro
Elastic IP addresses
–
VPC ID
vpc-0526aa1e6447cccb0 (VPC2-test)
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role
–
Subnet ID
subnet-044d6cbc13d242cc8 (VPC2-Public-SN1)
Instance details Info
Instance: i-0bf35ab3d4dc01c2f (VPC2-Dev-Bastion)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Security details
IAM Role
–
Owner ID
305139984171
Launch time
Mon Dec 06 2021 17:07:17 GMT-0500 (Eastern Standard Time)
Security groups
sg-02c26e99c67d65b64 (VPC-Dev-bastion-SG-20211205184102633300000002)
Inbound rules
Filter rules
Security group rule ID
Port range
Protocol
Source
Security groups
sgr-0ba31a6bba4224cae
22
TCP
0.0.0.0/0
VPC-Dev-bastion-SG-20211205184102633300000002

Notice that we are in the correct security group (VPC1-bastion-SG)

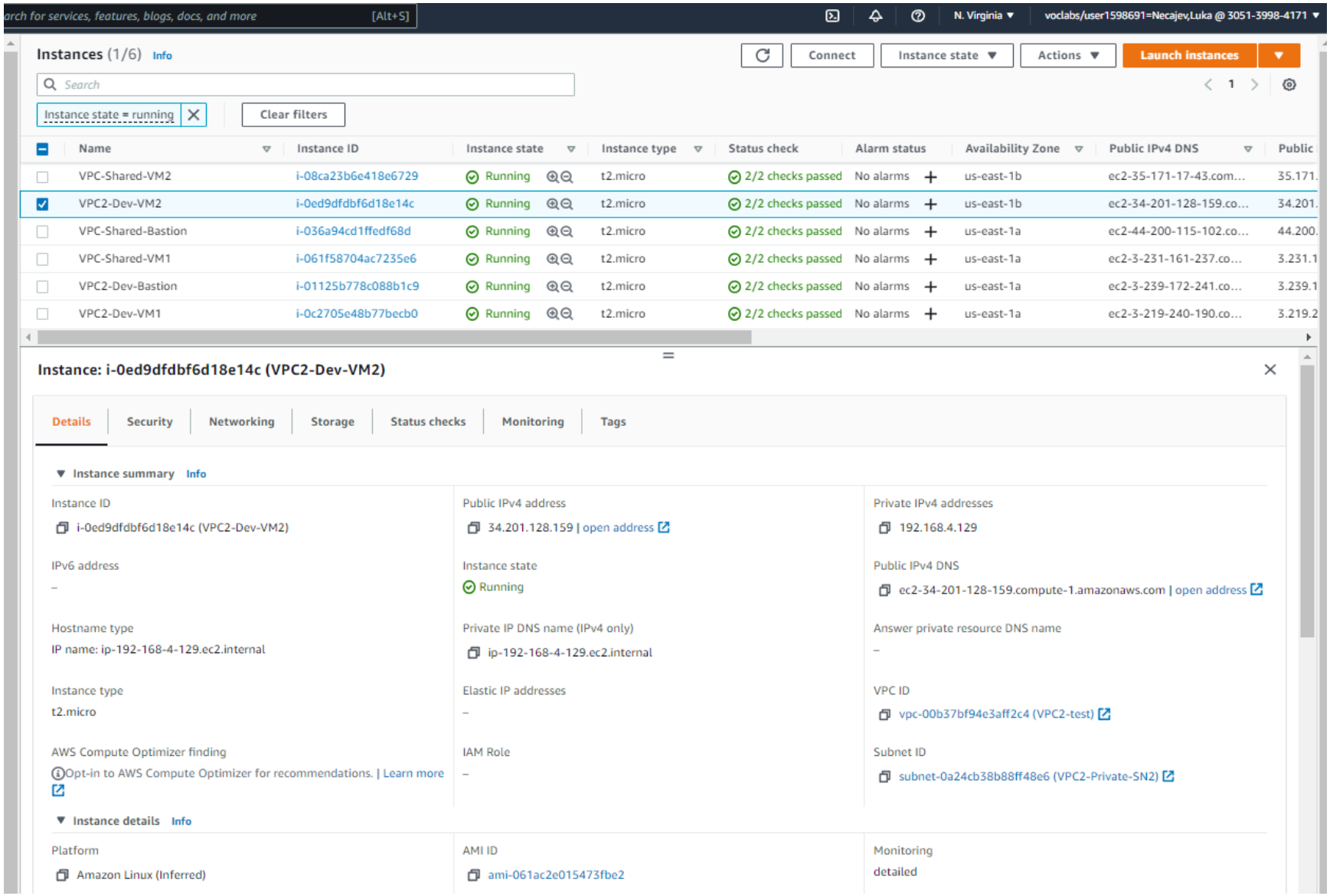
VPC2-Dev-VM1



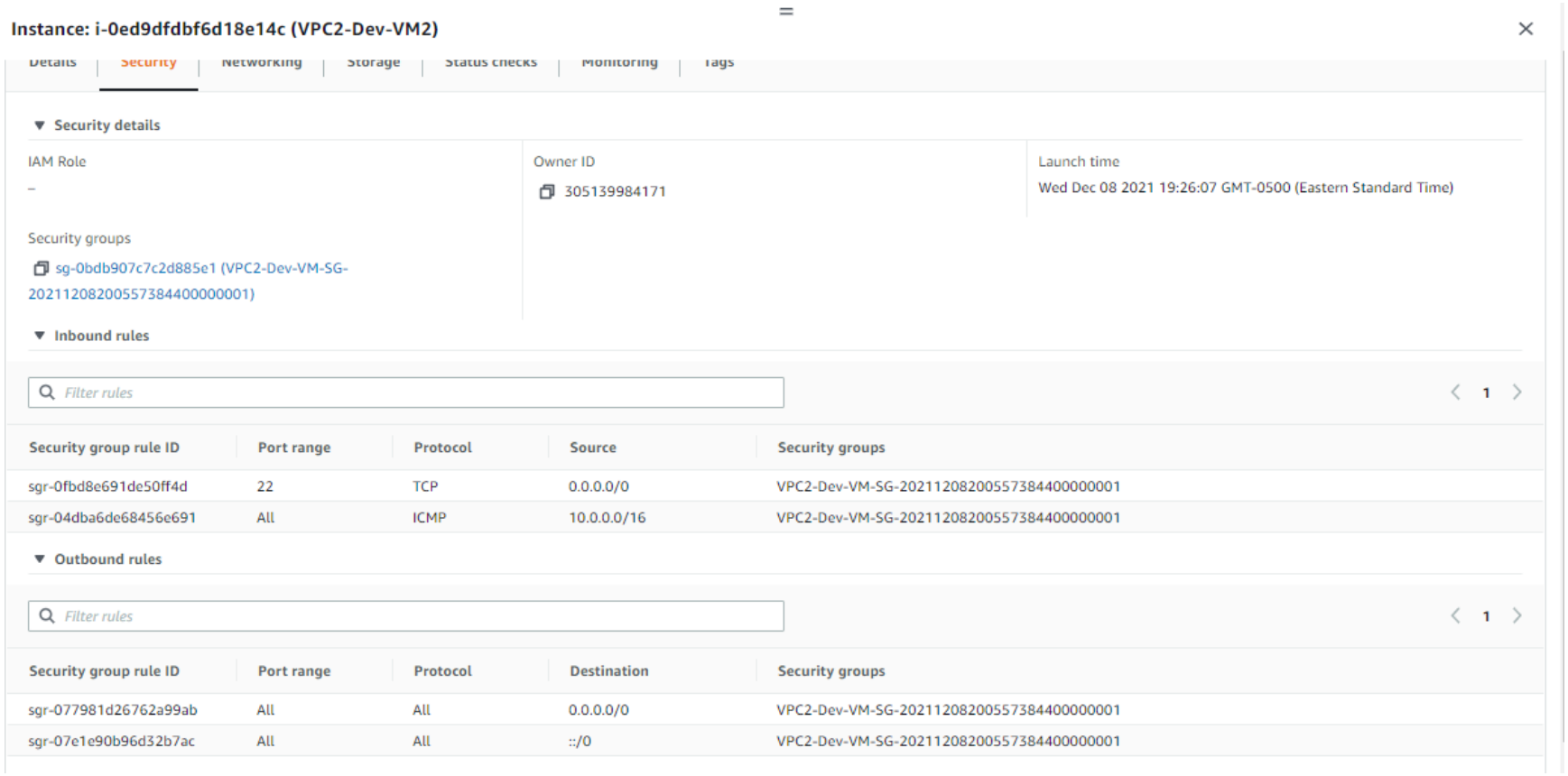
Showing the security Group


Instance: i-0c2705e48b77becb0 (VPC2-Dev-VM1)
Details
Security
Networking
Storage
Status checks
Monitoring
Tags
Security details
IAM Role
–
Owner ID
305139984171
Launch time
Wed Dec 08 2021 19:26:07 GMT-0500 (Eastern Standard Time)
Security groups
sg-0bdb907c7c2d885e1 (VPC2-Dev-VM-SG-20211208200557384400000001)
Inbound rules
Filter rules
1
Security group rule ID
Port range
Protocol
Source
Security groups
sgr-0fbd8e691de50ff4d
22
TCP
0.0.0.0/0
VPC2-Dev-VM-SG-20211208200557384400000001
sgr-04dba6de68456e691
All
ICMP
10.0.0.0/16
VPC2-Dev-VM-SG-20211208200557384400000001
Outbound rules
Filter rules
1
Security group rule ID
Port range
Protocol
Destination
Security groups
sgr-077981d26762a99ab
All
All
0.0.0.0/0
VPC2-Dev-VM-SG-20211208200557384400000001
sgr-07e1e90b96d32b7ac
All
All
::/0
VPC2-Dev-VM-SG-20211208200557384400000001

VPC2-Dev-VM2

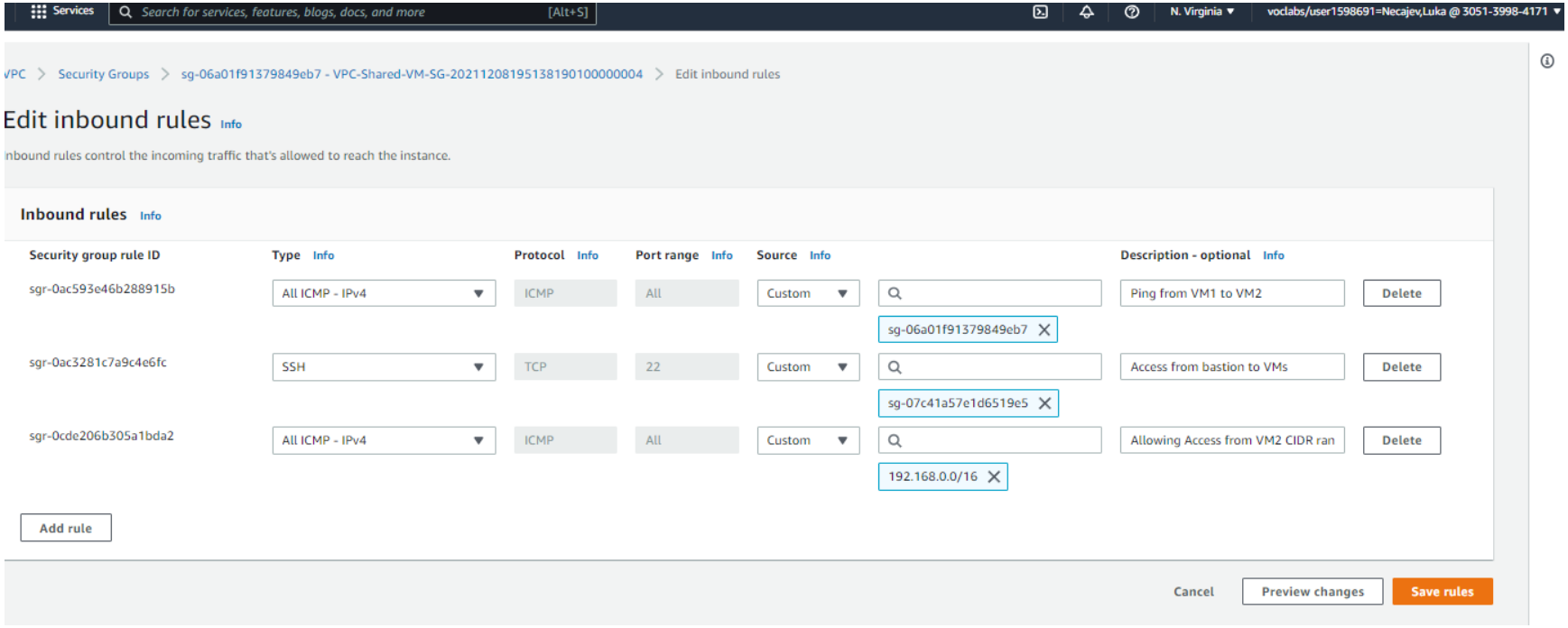


Showing the security group for VPC2 VM2



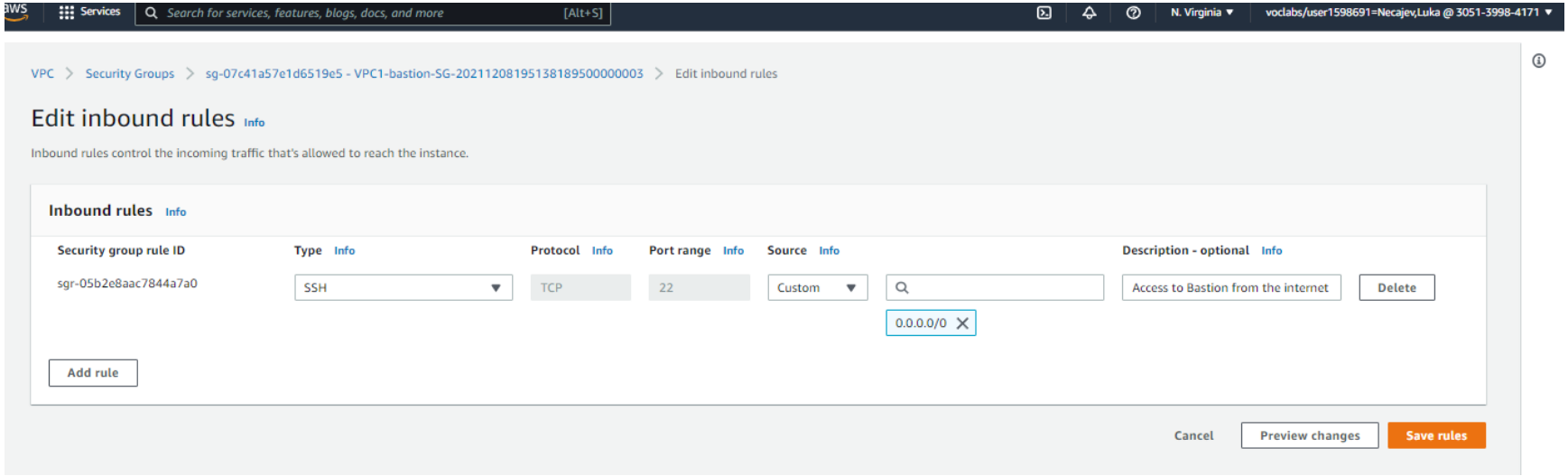
9. **Security Groups**

VPC-Shared-VM-SG



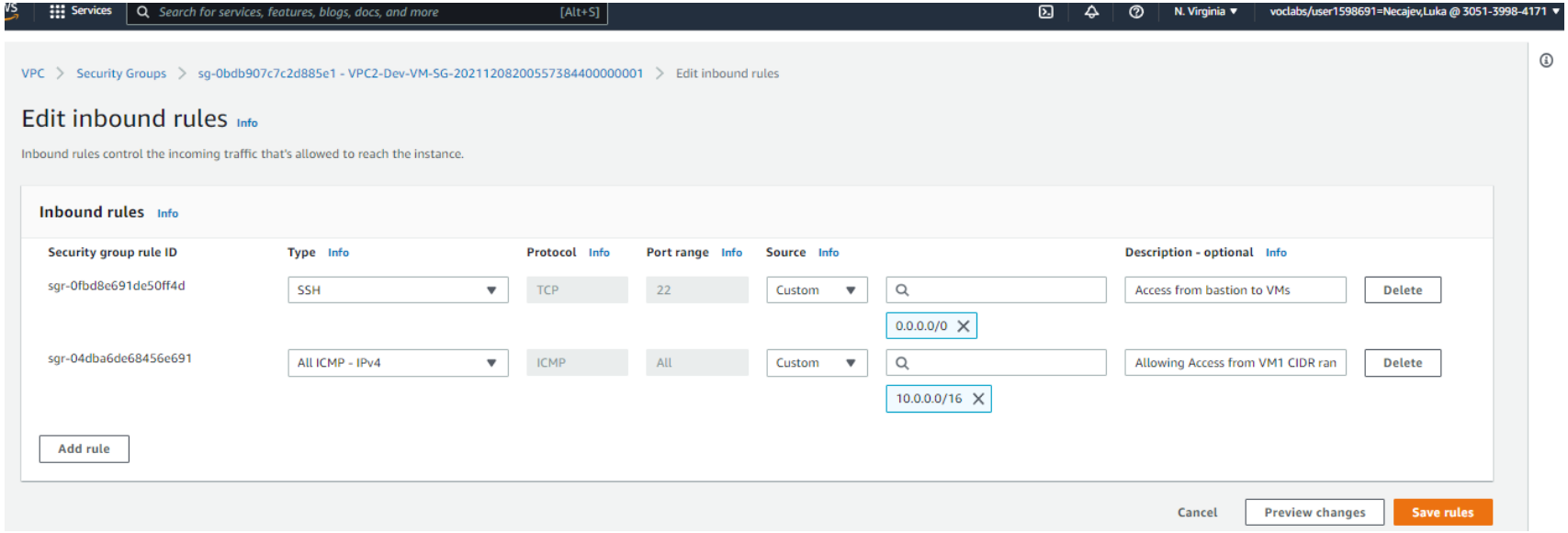
1. We are allowing ICMP only in the security groups that our VM's in VPC1 are inside. So we are specifying all ICMP to all instances inside of "VPC-Shared-VM-SG"
2. We are allowing SSH Inbound from our bastion host in VPC1. We are doing this by specifying the Bastion hosts security group. If the bastion host had a static IP we would only specify our bastion hosts IP, but since it is not static, we will reference the whole security group.
3. We are allowing all ICMP traffic to come in from the IP address range (CIDR) of our second VPC (Ip == 192.168.0.0/16)

VPC1-bastion-SG



1. We are allowing SSH Inbound to our bastion host from anywhere on the internet. So any admin with its ip address can access it.

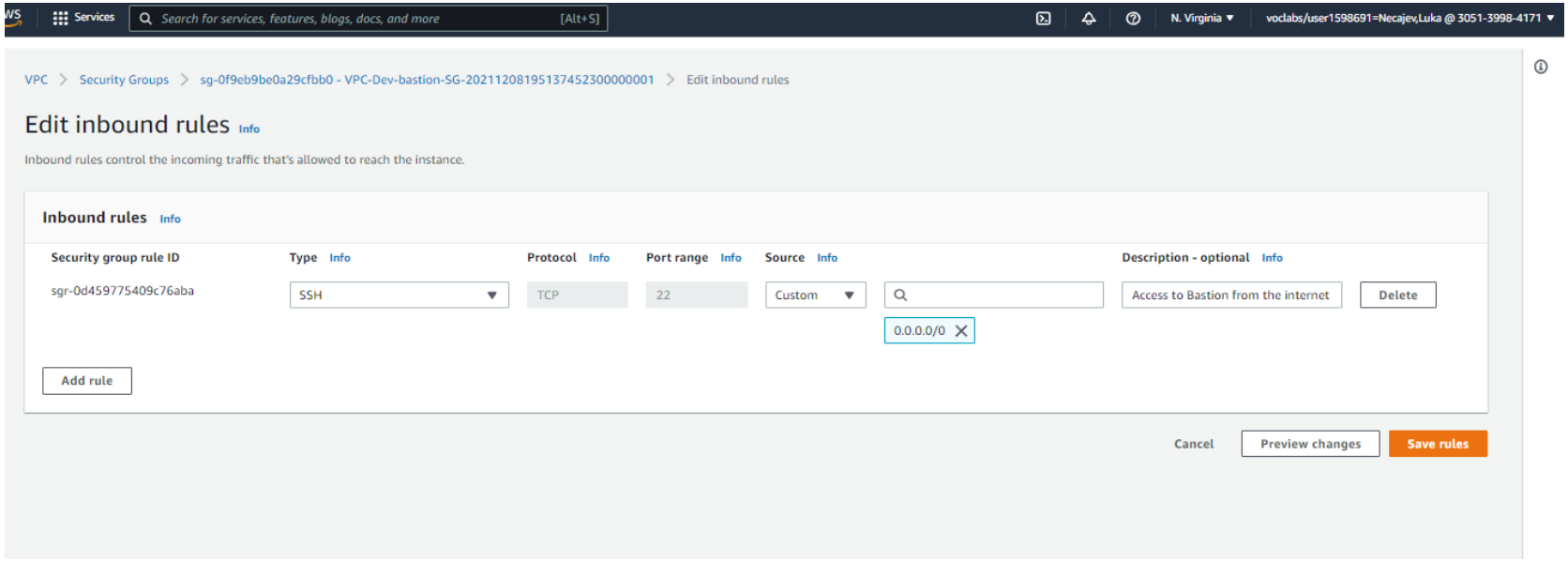
VPC2-Dev-VM-SG



1. We are allowing SSH Inbound from our bastion host in VPC2. We are doing this by specifying the Bastion hosts security group. If the bastion host had a static IP we would only specify our bastion hosts IP, but since it is not static, we will reference the whole security group.

2. We are allowing all ICMP traffic to come in from the IP address range (CIDR) of our first VPC (Ip == 10.0.0.0/16)

VPC2-Dev-bastion-SG



1. We are allowing SSH Inbound to our bastion host from anywhere on the internet. So any admin with its ip address can access it.

10. Network connectivity

a. Admin user should be able to ssh to VPC-Shared-Bastion and through it to VPC-Shared-VM1 and VPC-Shared-VM2.

SSH into VPC bastion

```
ddd_v1_w_MVbI_882554@runweb44202:~$ ssh -i ~/.ssh/labsuser.pem ec2-user@3.227.255.61
Last login: Mon Dec  6 22:29:05 2021 from ec2-34-220-150-248.us-west-2.compute.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-1-71 ~]$
[ec2-user@ip-10-0-1-71 ~]$
[ec2-user@ip-10-0-1-71 ~]$
[ec2-user@ip-10-0-1-71 ~]$
```

SSH into VPC-Shared-VM1

Create the key file and place it into our bastion host. Then change the settings so its executable

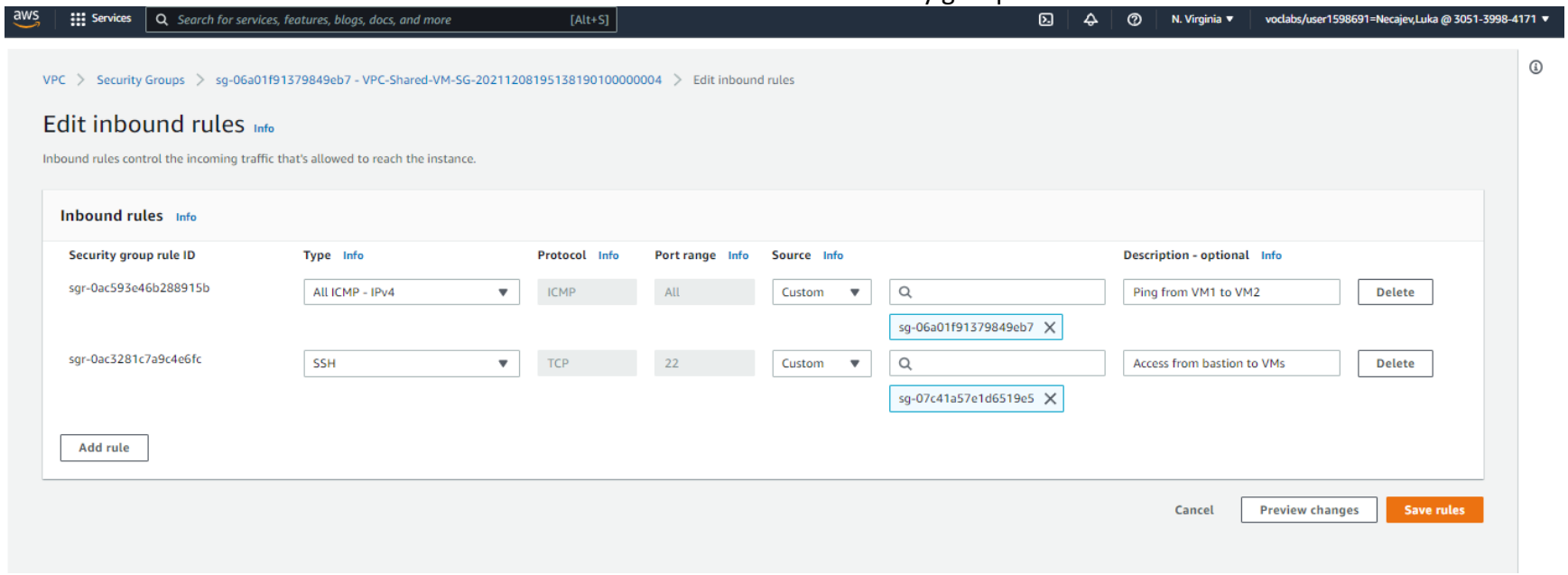
```
[ec2-user@ip-10-0-1-71 ~]$ nano key.pem
[ec2-user@ip-10-0-1-71 ~]$ sudo chmod 777 key.pem
```

```
[ec2-user@ip-10-0-1-71 ~]$ ssh -i key.pem ec2-user@3.227.255.61
Last login: Mon Dec  6 22:32:34 2021 from ec2-34-220-150-248.us-west-2.compute.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
```

This means that we will need port 22 open on the bastion sg to allow outside users to ssh to the bastion
This means that we will need port 22 open on our vm sg to allow bastion host sg in

b. VPC-Shared-VM1 should be able to ping VPC-Shared-VM2 and vice versa

We will need to allow ICMP to all instances within the VPC1-VM-SG vm security group



```
--- 10.0.4.179 ping statistics ---
48 packets transmitted, 48 received, 0% packet loss, time 47989ms
rtt min/avg/max/mdev = 0.708/0.881/1.447/0.126 ms
[ec2-user@ip-10-0-3-42 ~]$ ping 10.0.4.179
PING 10.0.4.179 (10.0.4.179) 56(84) bytes of data.
64 bytes from 10.0.4.179: icmp_seq=1 ttl=255 time=0.675 ms
64 bytes from 10.0.4.179: icmp_seq=2 ttl=255 time=0.818 ms
64 bytes from 10.0.4.179: icmp_seq=3 ttl=255 time=0.734 ms
```

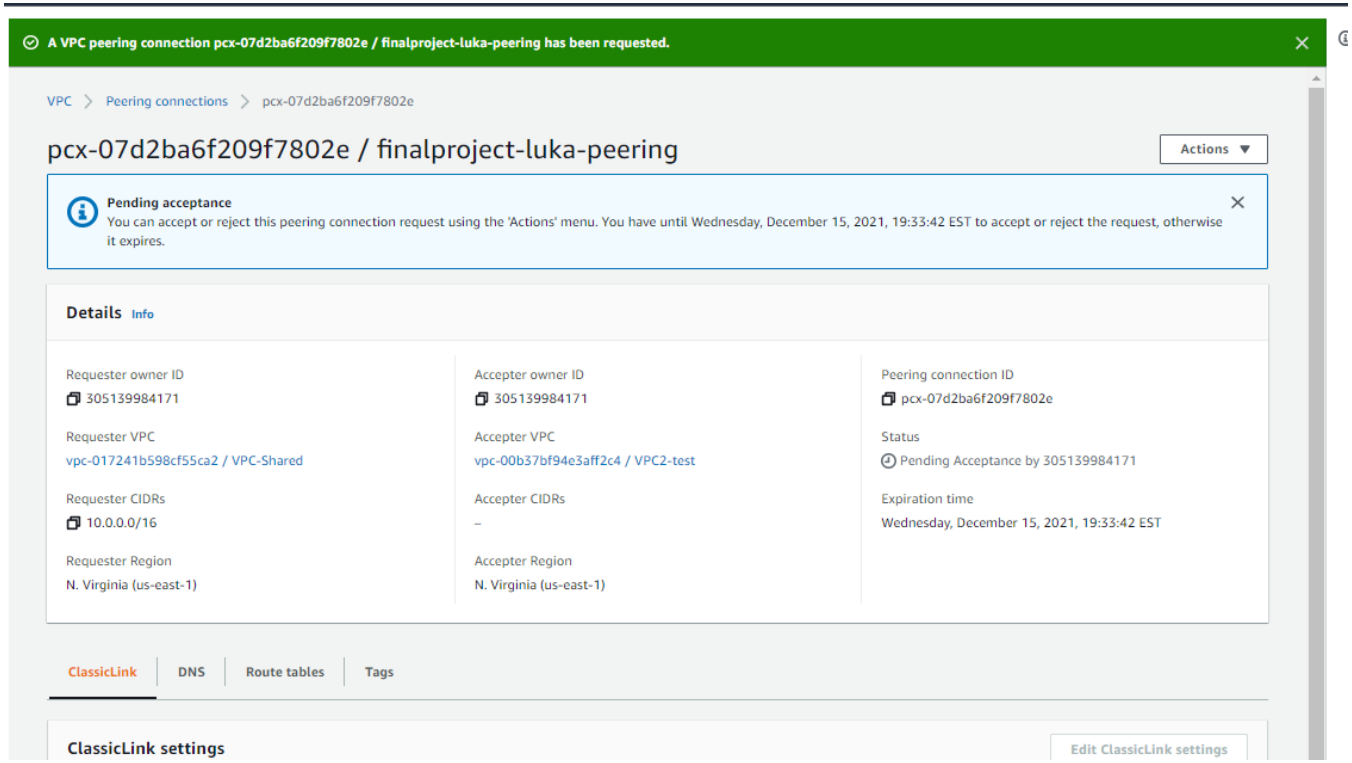
c. **Bonus!** VPC-Shared-VM1 should be able to copy an image from the S3 bucket you created.

aws s3api get-object --bucket tf-luka-owen-final --key Sully.jpg sully.jpg

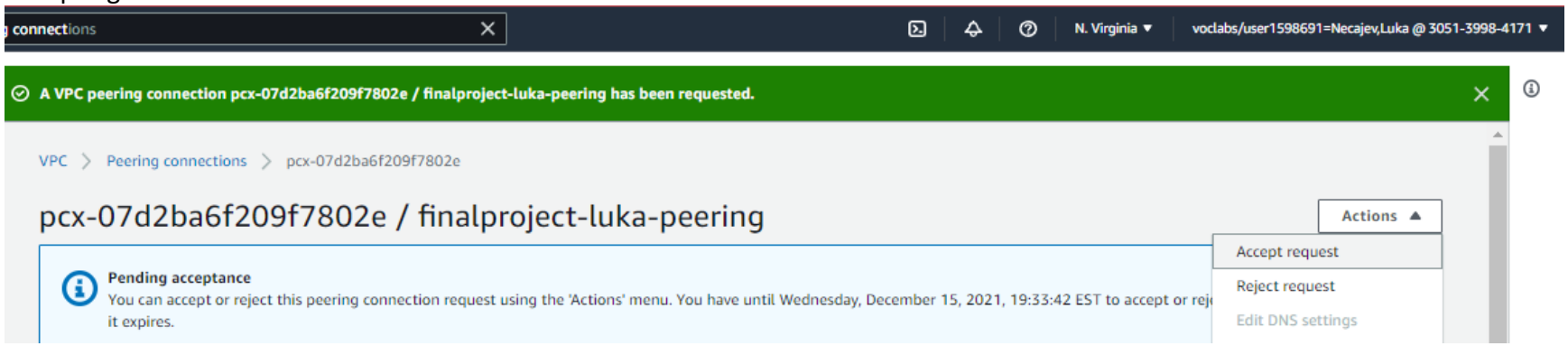
```
ddd_v1_w_MVbI_882554@runweb44484:~$ aws s3api get-object --bucket tf-luka-owen-final --key Sully.jpg sully.jpg
{
    "AcceptRanges": "bytes",
    "LastModified": "2021-12-08T20:17:08+00:00",
    "ContentLength": 39373,
    "ETag": "\"0dbc2ff353c410691977bd542524e328\"",
    "ContentType": "image/jpeg",
    "Metadata": {}
}
ddd_v1_w_MVbI_882554@runweb44484:~$ ls
sully.jpg
ddd_v1_w_MVbI_882554@runweb44484:~$
```

d. **VPC-Shared-VM2 should be able to ping VPC-Dev-VM1; VPC-Shared-VM1 should not be able to ping VPC-Dev-VM1**

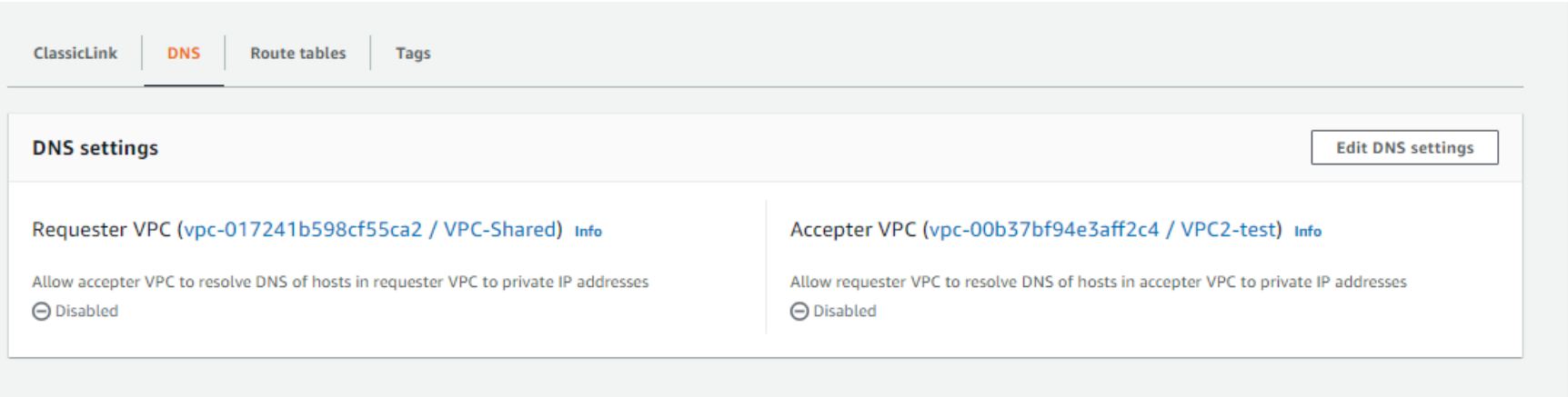
Creating the peering connection



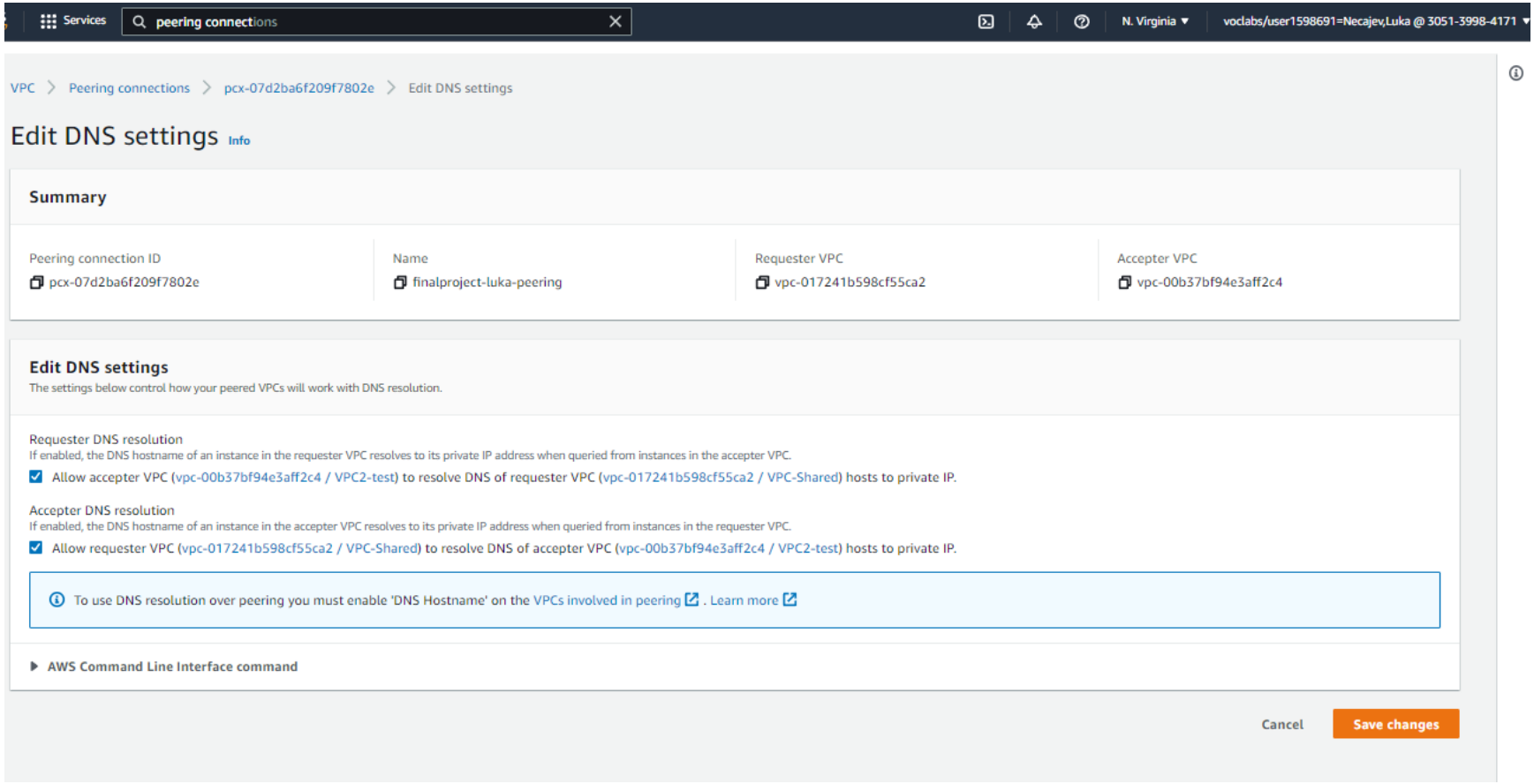
Accepting the connection.



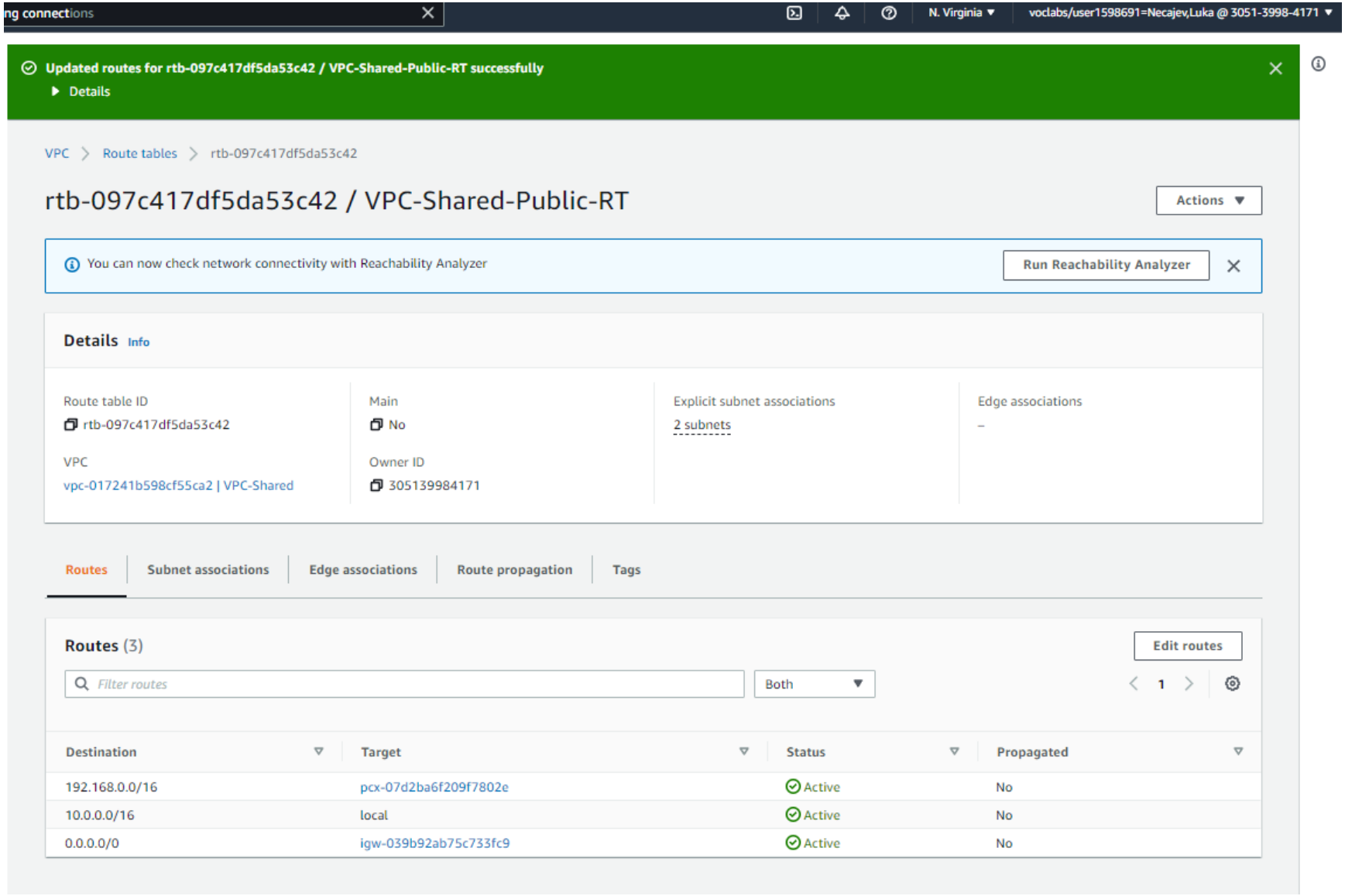
Allowing DNS to work in our private subnets over peering



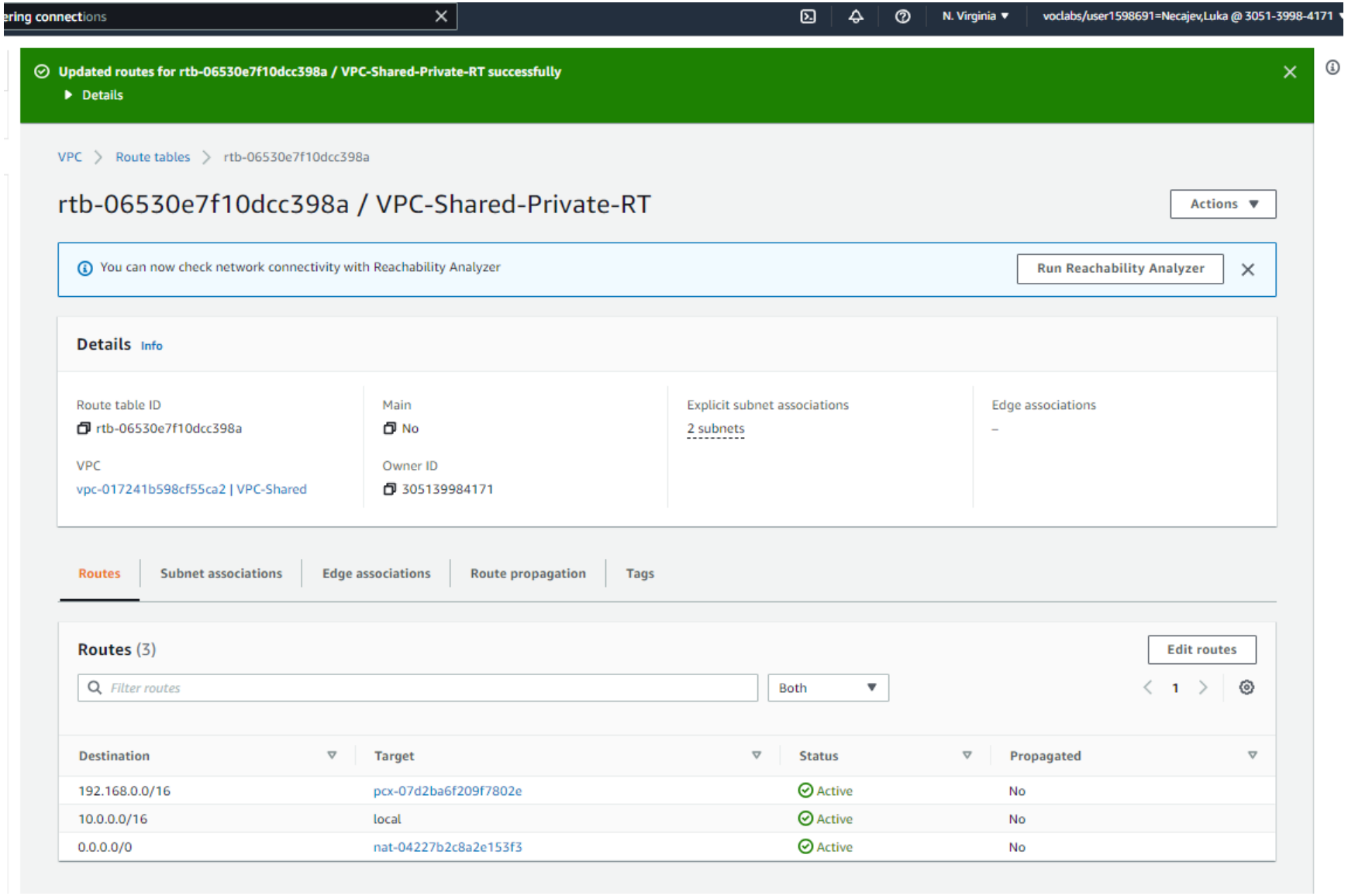
Accepting both the requester and acceptor DNS name resolution as we do not mind if both users are able to recognize eachother by an ip/hostname.



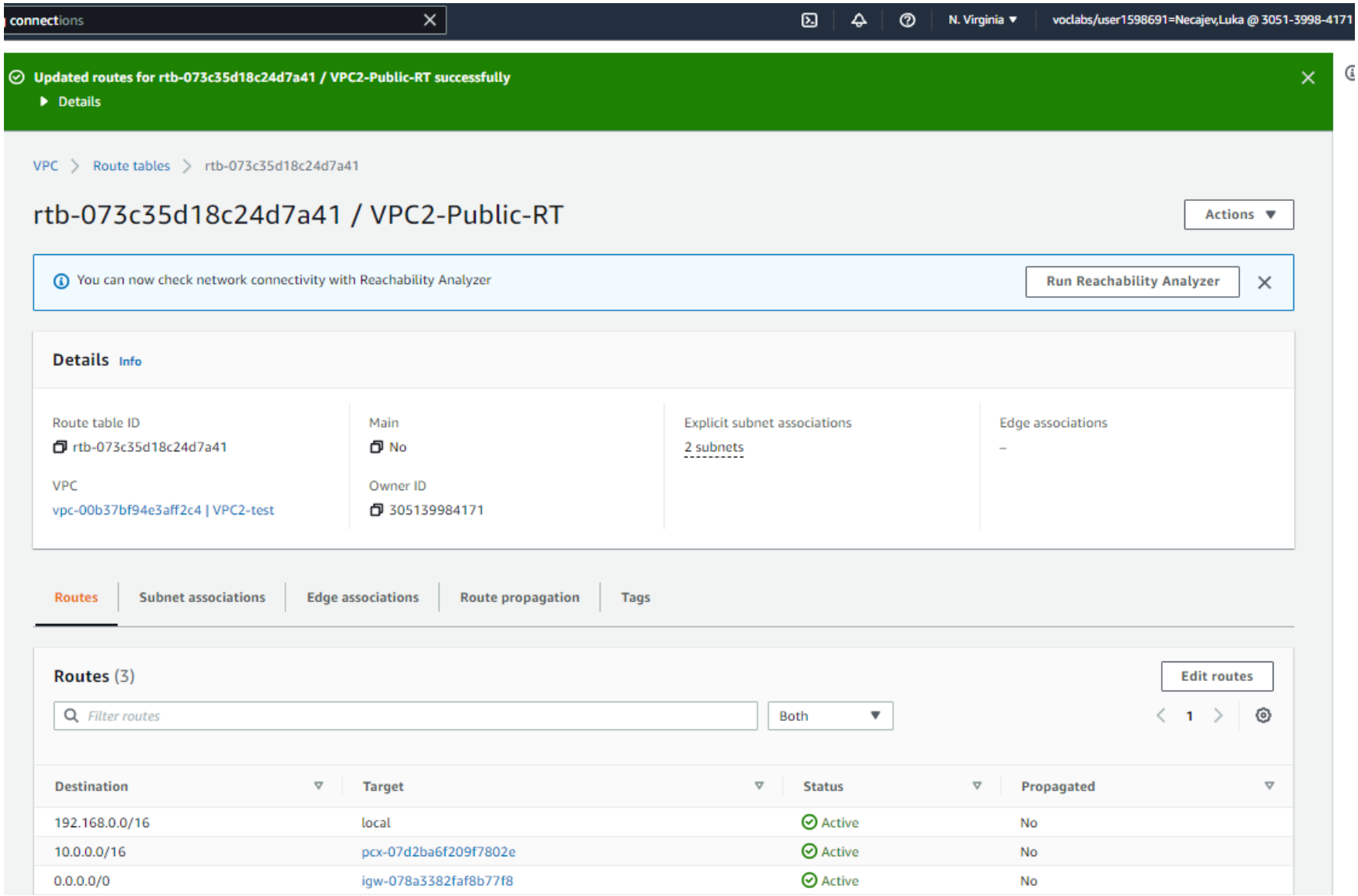
Updating the traffic routes.



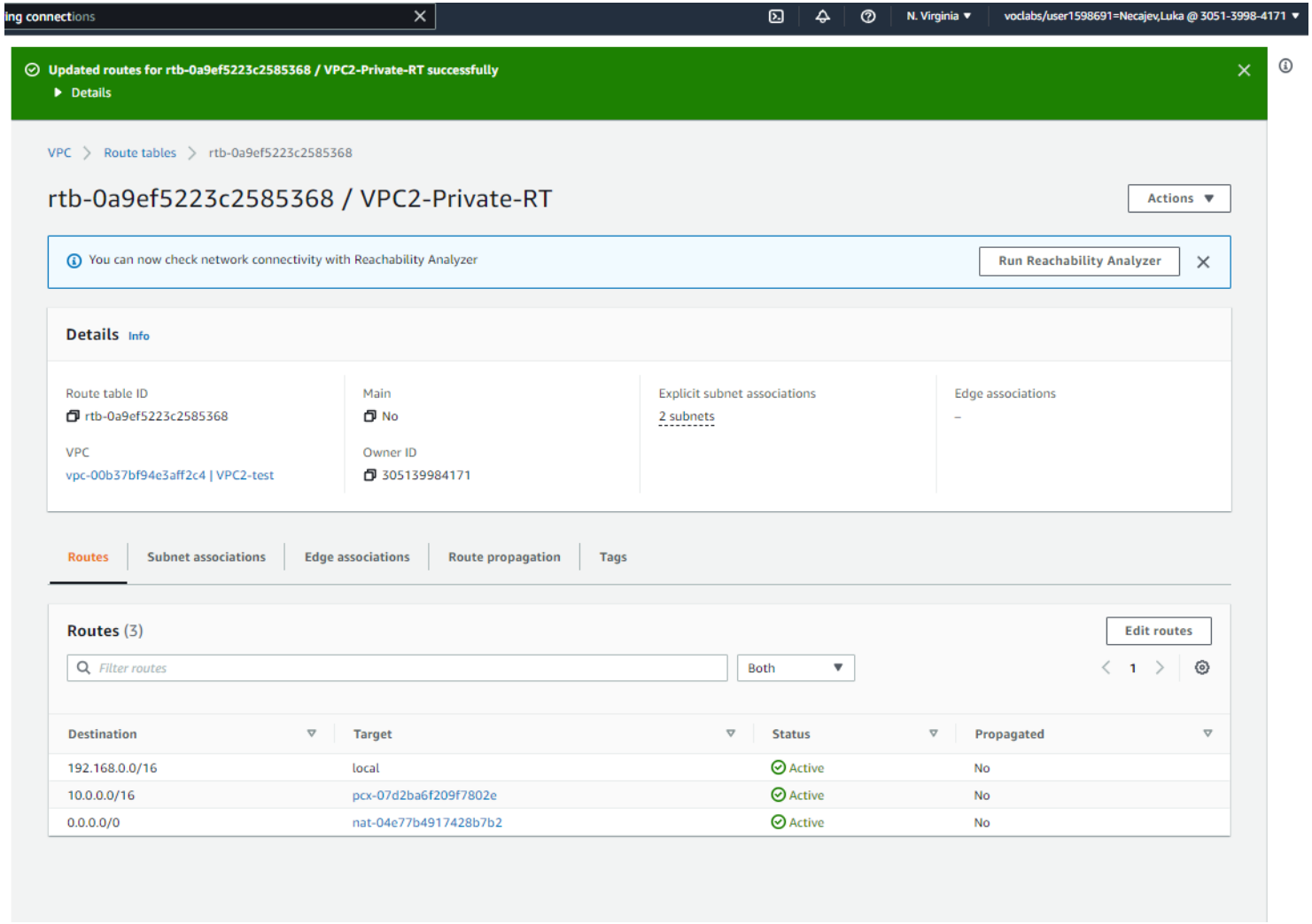
Changing Public VPC1 to allow our data from VPC2



Allowing our private routing table access to the other VPC using peering



Allowing our public routing table to send messages back to the Sharing VPC using peering

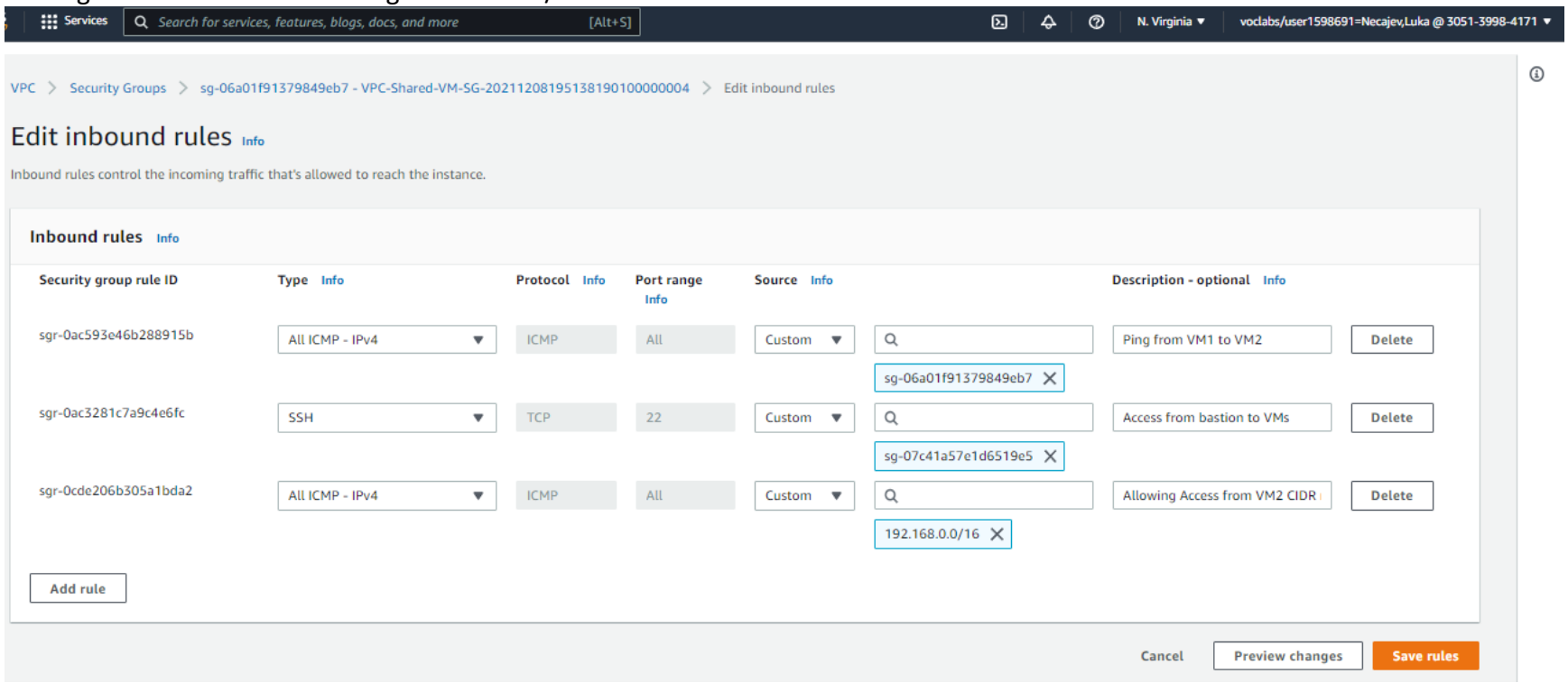


Allowing our private routing table to send messages back to the Sharing VPC using peering

Updating Security Groups to allow ICMP messages to VPC2 From VM1 in VPC1

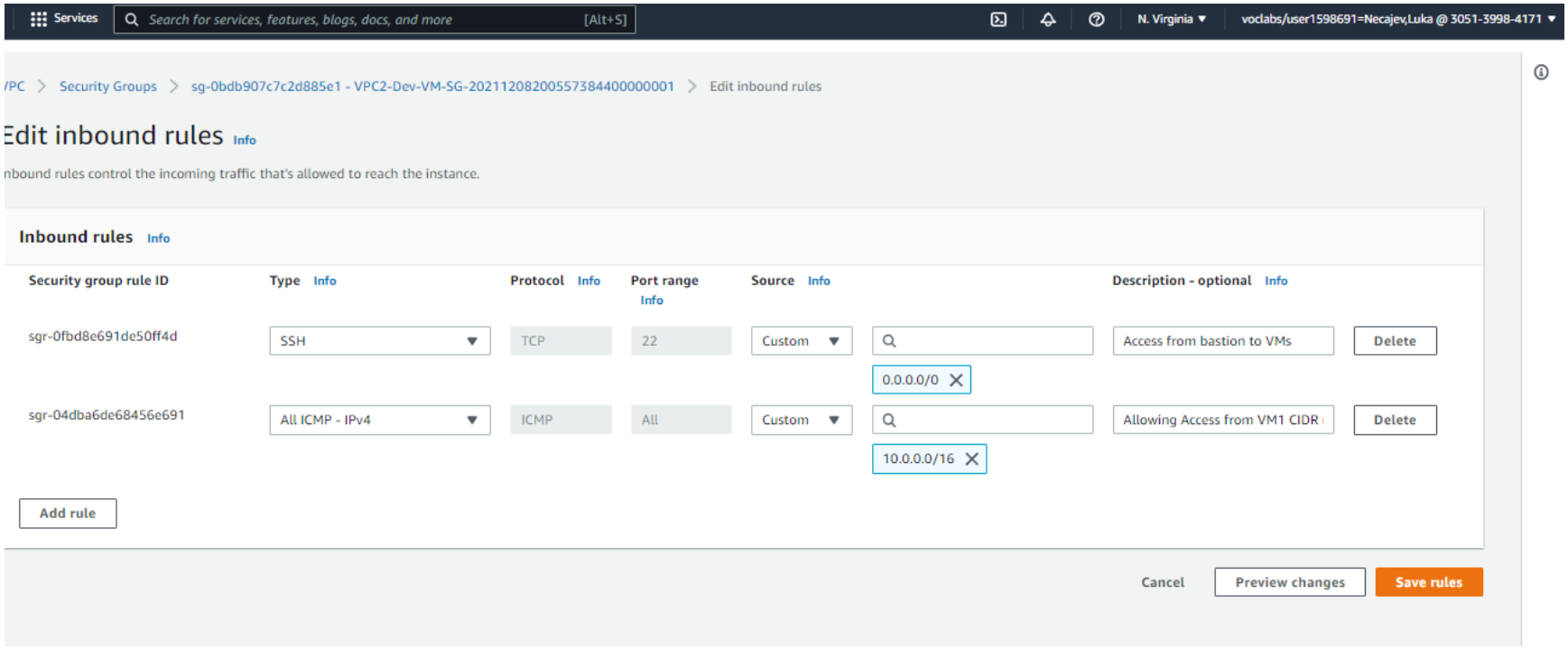
Security-Group-VPC1

What we had to do here was change the Routing table to allow all items that are coming from the second VPC's(192.168.0.0/16) subnet to our first VPC's subnet(10.0.0.0/16) into our VPC network. To do this we have to add the rule to allow all ICMP packets coming from VPC2's subnet being 192.168.0.0/16.



Security-Group-VPC2

What we had to do here was change the Routing table to allow all items that are coming from the first VPC's(10.0.0.0/16) subnet to our second VPC's subnet(192.168.0.0/16). To do this we have to add the rule to allow all ICMP packets coming from VPC1's subnet being 10.0.0.0/16.



Testing ping from VPC1 VM2 to VPC2 VM1

Getting Bastion Host IP

8d (VPC-Shared-Bastion)

orking | Storage | Status checks | Monitoring | Tags

ared-Bastion)

Public IPv4 address
44.200.115.102 | open address

Private IPv4 addresses
10.0.1.183

Instance state

Public IPv4 DNS

Getting VPC1 VM1 IP

(VPC-Shared-VM1)

ing | Storage | Status checks | Monitoring | Tags

d-VM1)

Public IPv4 address
3.231.161.237 | open address

Private IPv4 addresses
10.0.3.42

Getting VPC2 VM1 IP

lc (VPC2-Dev-VM2)

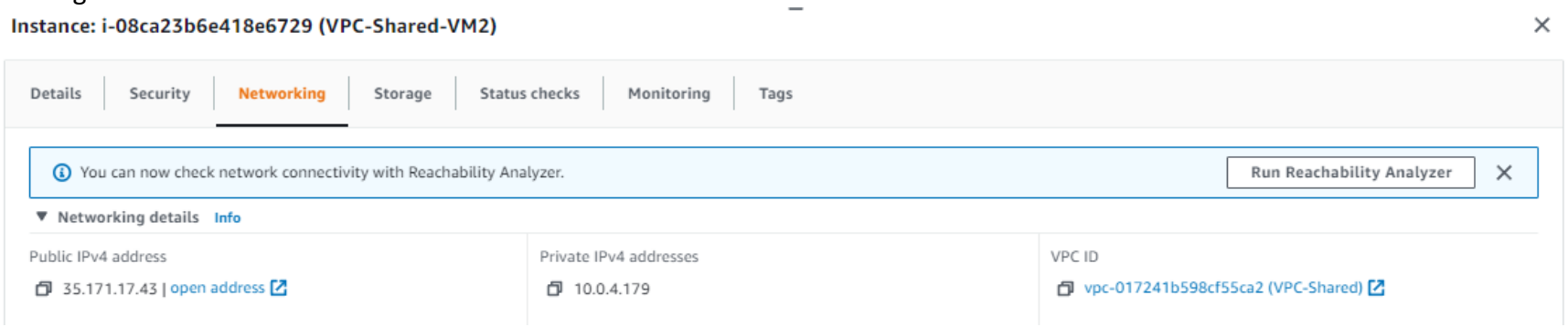
king | Storage | Status checks | Monitoring | Tags

v-VM2)

Public IPv4 address
34.201.128.159 | open address

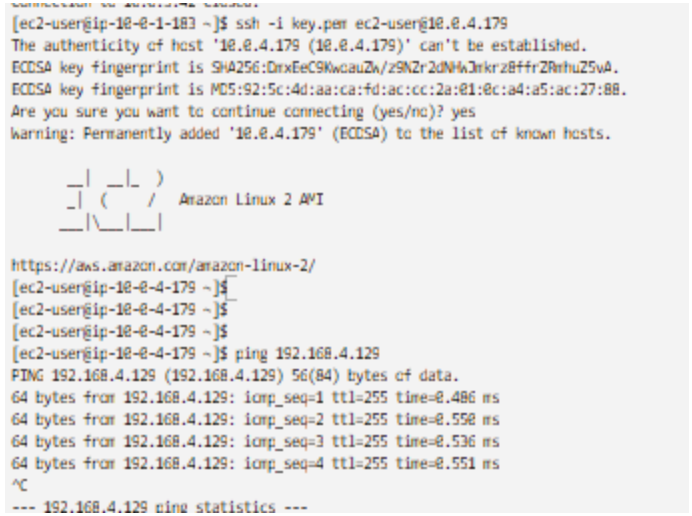
Private IPv4 addresses
192.168.4.129

Getting VPC1 VM2 IP



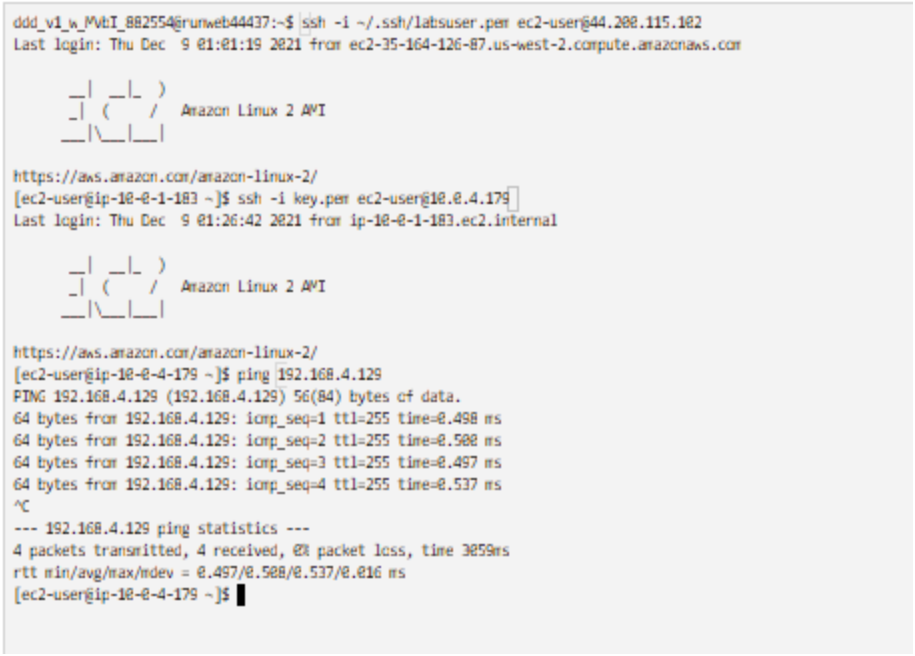
Signing into bastion VM -> Then Signing into VM1 in VPC1 Shared -> Then trying to ping EC2 instance in other VPC

```
ddd_v1_w_MVtI_882554@runweb44416:~$ ssh -i ~/.ssh/labsuser.pem ec2-user@44.208.115.102
The authenticity of host '44.208.115.102 (44.208.115.102)' can't be established.
ECDSA key fingerprint is SHA256:YwcYv/YZMmCCCfiJLMciCTXxlmCt4v58ZmCSY6Ddl+Y.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '44.208.115.102' (ECDSA) to the list of known hosts.
Last login: Wed Dec  8 20:12:08 2021 from ec2-34-220-39-163.us-west-2.compute.amazonaws.com

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-1-183 ~]$
[ec2-user@ip-10-0-1-183 ~]$
[ec2-user@ip-10-0-1-183 ~]$
[ec2-user@ip-10-0-1-183 ~]$ ls
key.pem
[ec2-user@ip-10-0-1-183 ~]$ ssh -i key.pem ec2-user@10.0.3.42
```



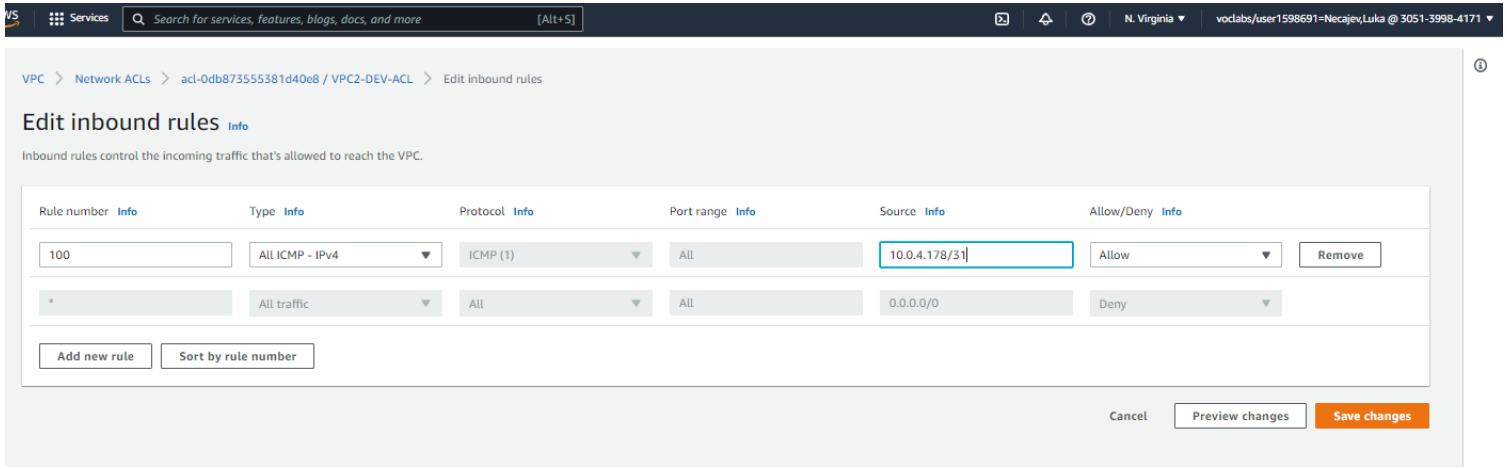
Now we can isolate the other ip's in the NACL to allow and deny specific items in and add the

Logging into the bastion host on VPC1 shared -> logging into VM2 in VPC1 Shared -> pinging VPC2-Dev-VM1



Testing ping from VPC1 VM1 to VPC2 VM1

Edit the Inbound NACL (VPC2-DEV-ACL) to not allow anything into the Subnet that is not VPC1 VM2



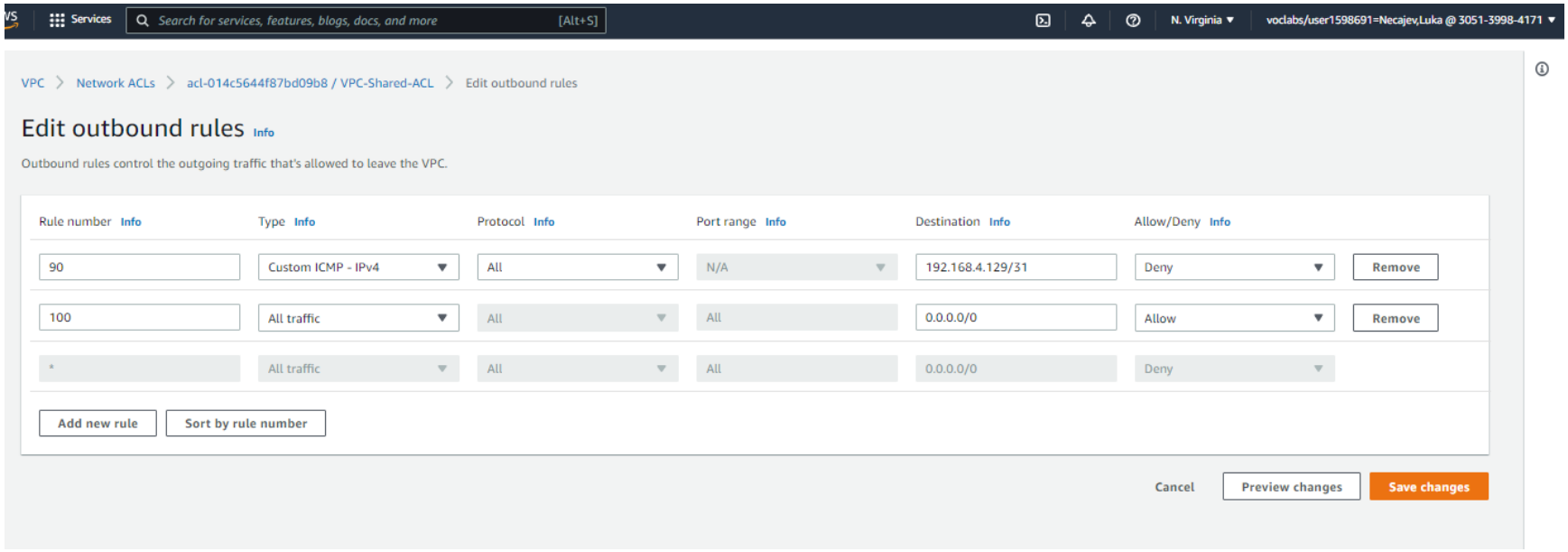
Logging out of that instance, logging into VPC VM1 to test if I can ping VPC2 VM1

```
[ec2-user@ip-10-0-1-183 ~]$ ssh -i key.pem ec2-user@10.0.3.42
Last login: Thu Dec  9 01:01:36 2021 from ip-10-0-1-183.ec2.internal

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-3-42 ~]$
[ec2-user@ip-10-0-3-42 ~]$
[ec2-user@ip-10-0-3-42 ~]$ ping 192.168.3.155
PING 192.168.3.155 (192.168.3.155) 56(84) bytes of data.
```

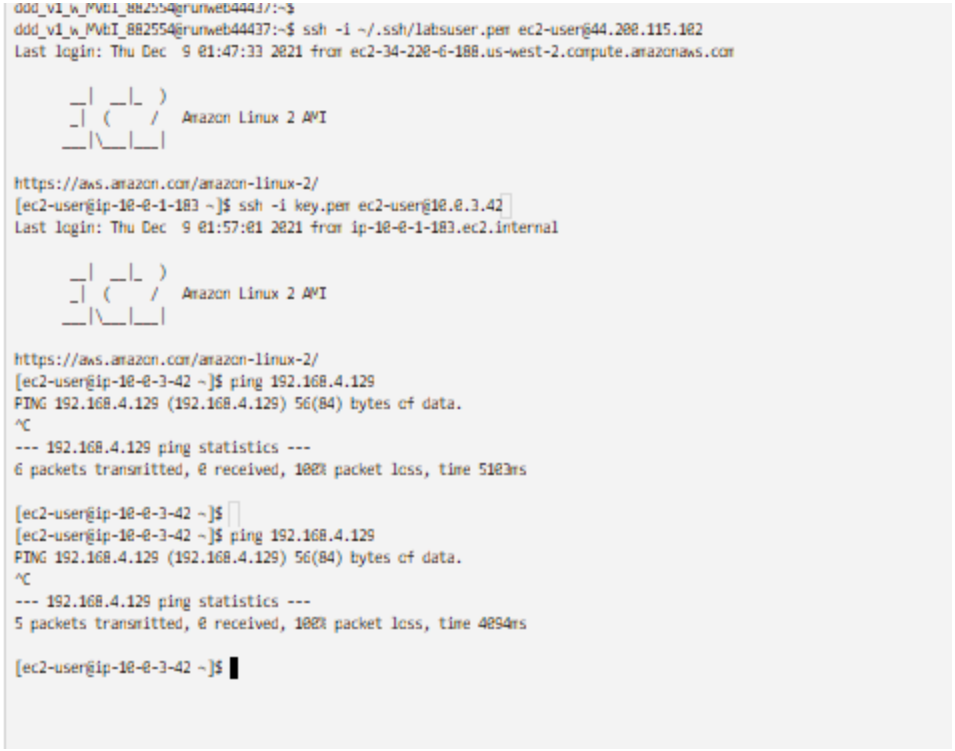
There is no response from the VM1 and there will be no response when pinging other VM's

e. **Bonus!** VPC-Shared-VM1 and VPC-Shared-VM2 should **not** be able to ping VPC-Dev-VM2

Since we do not want them to be able to ping VM2 and we want VM2 to be able to ping VM1, we can just add an outbound rule that would stop the packet from ever exiting VPC1 subnet.



I am making sure that this is denying only the IP I specify which is VM2's IP



Functionality Validation

Please find the table below with the functionality validation artifacts that should be part of your final project's delivery.

Functionality	Artifacts to provide	Comments

TF code building out VPCs, subnets, route tables, GWs, SGs and bastion hosts	Terraform code is deployable, well structured, modular and uses variables Terraform code should be submitted as a zip file The Terraform code should use a remote state stored in S3. There is a clear README.md that explains all the steps for successful deployment	You can use external TF modules You can deploy part of the required infrastructure via TF for a reduced mark. TF code that cannot be deployed will incur zero mark for this part of the assignment. README.md **must** be added to the root of the zip file.
Building out the rest of the infrastructure on the diagram: S3 bucket to store TF remote state Peering connection Updated route tables (if needed) Updated SGs (if needed)	AWS CLI command screenshots (if done with AWS CLI) and AWS Console screenshots clearly showing the infrastructure components have been deployed in **your** account.	This part can be fully or partially implemented with TF for extra marks. All the deployed infrastructure should be properly tagged where possible.
Create S3 bucket with to store remote Terraform state.	AWS CLI command screenshots (if done with AWS CLI) and AWS Console screenshots clearly showing the infrastructure components have been deployed in **your** account. Screenshot of the S3 bucket resource policy and of the permissions tab clearly showing that the bucket is not publicly accessible. Any other screenshots that prove the bucket is private.	In case you are struggling with the private bucket you can make it public. Mark will be reduced.

Admin users should be able to ssh to PC-Shared-VM1 and PC-Shared-VM2 in the private subnets	Explain the process of logging into Linux EC2 instances running in a private subnet. Support your explanation with the relevant screenshots from all the steps required to log into PC-Shared-VM1. Compliment the screenshots with a brief explanation.	You can ssh from your laptop or AWS Academy Terminal. You can use your own ssh key or AWS Academy provided ssh key.
VPC-Shared-VM1 should be able to ping VPC-Shared-VM2 and vice versa	Screenshots of relevant SGs and ping command	
Bonus! VPC-Shared-VM1 should be able to copy an image from the S3 bucket you created	Screenshots of VPC-Shared-VM1 IAM role and AWS CLI command performed on VPC-Shared-VM1 that copies the image Explanation of: - What kind of permissions were used to access S3 bucket? Why we cannot access this image via browser but we can do it from the VPC-Shared-VM1? - Detailed explanation or diagram of the route that the request made to reach S3 bucket.	Picture is better than a thousand words.
VPC-Shared-VM2 should be able to ping VPC-Dev-VM1 VPC-Shared-VM2 should **not** be able to ping VPC-Dev-VM1	Screenshots of relevant SGs, route tables and ping commands	

Bonus! VPC-Shared-VM1 and VPC-Shared-VM2 should **not** be able to ping VPC-Dev-VM2	Screenshots of relevant SGs, route tables and ping commands.	Blocking the ping with SG is only a partial solution. The route tables should not have routing rules that allow traffic flow between VPC-Shared-VM2 and VPC-Dev-VM1

Grading Guidelines

Deliverable	Points
TF code building out VPCs, subnets, route tables, GWs, SGs and bastion hosts	**50**

Building out the rest of the infrastructure on the diagram: S3 bucket to store TF remote state Linux VMs in private subnets Peering connection Updated route tables (if needed) Updated SGs (if needed)	**25**
Create S3 bucket with an image.	**5**
Admin users should be able to ssh to PC-Shared-VM1 and PC-Shared-VM2 in the private subnets	**5**
VPC-Shared-VM1 should be able to ping VPC-Shared-VM2 and vice versa	**5**
Bonus! VPC-Shared-VM1 should be able to copy an image from the S3 bucket you created	**10**
VPC-Shared-VM2 should be able to ping VPC-Dev-VM1 VPC-Shared-VM1 should **not** be able to ping VPC-Dev-VM1	**10**
Bonus! VPC-Shared-VM1 and VPC-Shared-VM2 should **not** be able to ping VPC-Dev-VM2	**10**