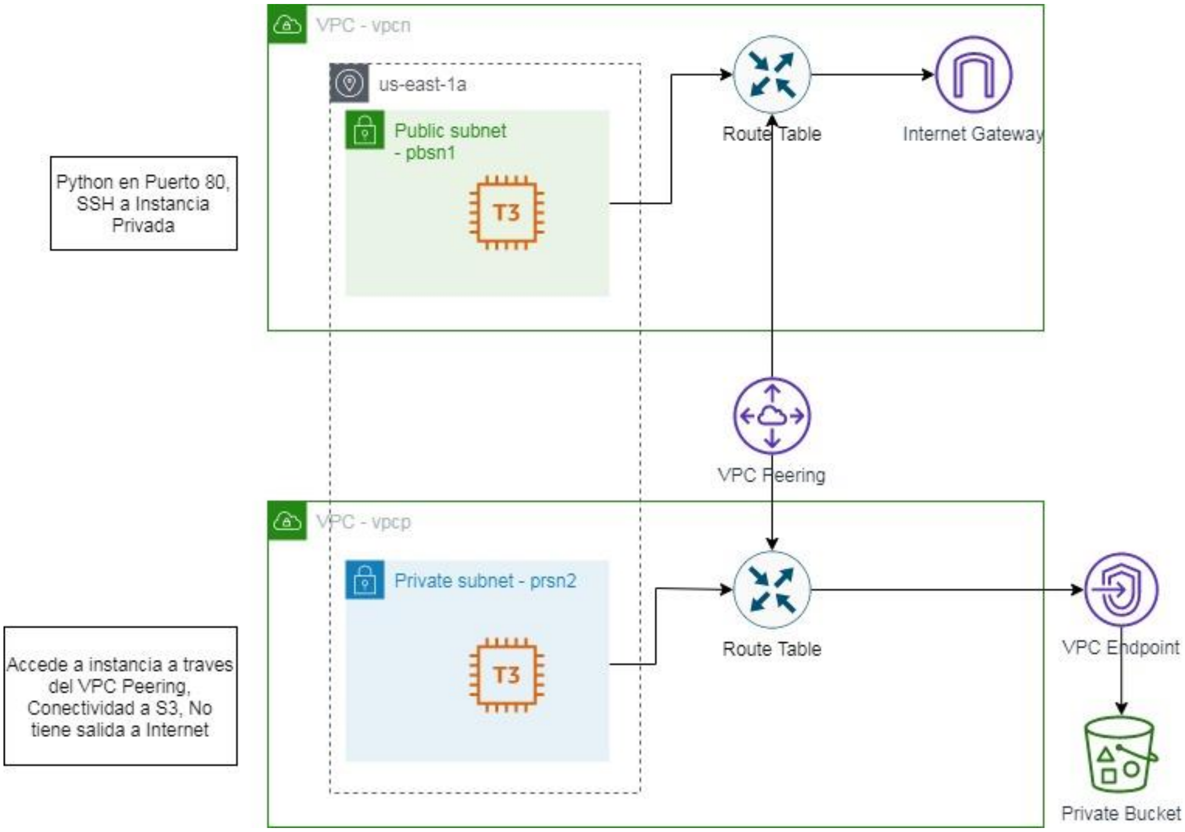
Contents

Purpose

Make a VPC Peering connection with controlled environment for an intranet subnet. In addition to have a S3 VPC Endpoint on that subnet

General Diagram

Have a public and intranet layer with controlled access.



Prerequisites

Labs1c1 have to be done and the context for Administrative user have to activated on Command Line Session.

Labs4c1 have to be done, because you learn how to: Create subnets, VPCs, IGW and Routing Tables. For this case specifically, you have to create VPC, Public Subnet, IGW, Routing Table with the same names as that laboratory, therefore we only focus on the new things.

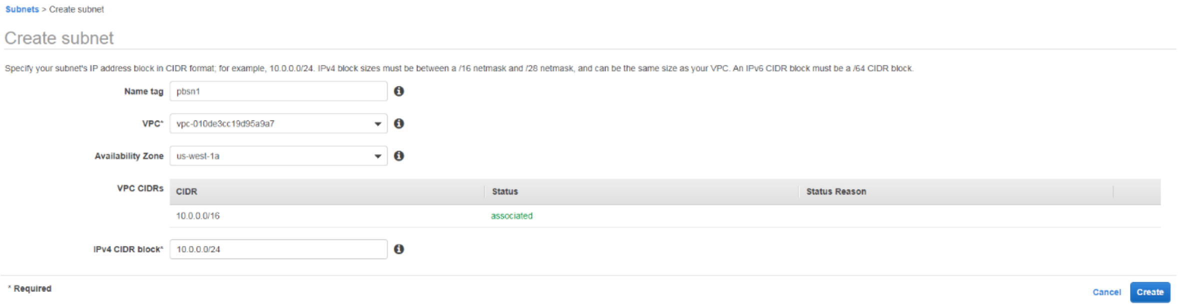
Lab 4B: VPC with VPC Peering and VPC Endpoint

Lab 4B using Web Management Console

Create VPC, subnets, IGW

Prerequisite from previous Lab: Labs4c1. Some screenshoots.

Remember to apply Public IP Mapping to Public Subnet.



Subnets > Create subnet

Create subnet

Specify your subnet's IP address block in CIDR format; for example, 10.0.0.0/24. IPv4 block sizes must be between a /16 netmask and /28 netmask, and can be the same size as your VPC. An IPv6 CIDR block must be a /64 CIDR block.

Name tag: pbsn1

VPC*: vpc-010de3cc19d95a9a7

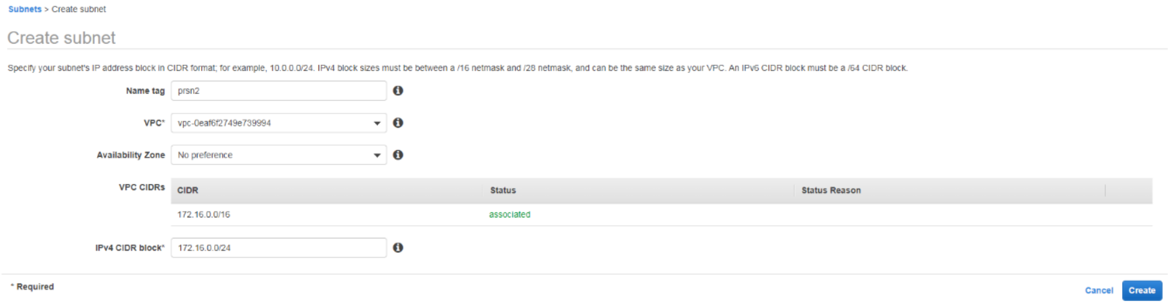
Availability Zone: us-west-1a

VPC CIDRs

CIDR	Status	Status Reason
10.0.0.0/16	associated	

IPv4 CIDR block*: 10.0.0.0/24

* Required



Subnets > Create subnet

Create subnet

Specify your subnet's IP address block in CIDR format; for example, 10.0.0.0/24. IPv4 block sizes must be between a /16 netmask and /28 netmask, and can be the same size as your VPC. An IPv6 CIDR block must be a /64 CIDR block.

Name tag: prsn2

VPC*: vpc-0eaf6f2749e739994

Availability Zone: No preference

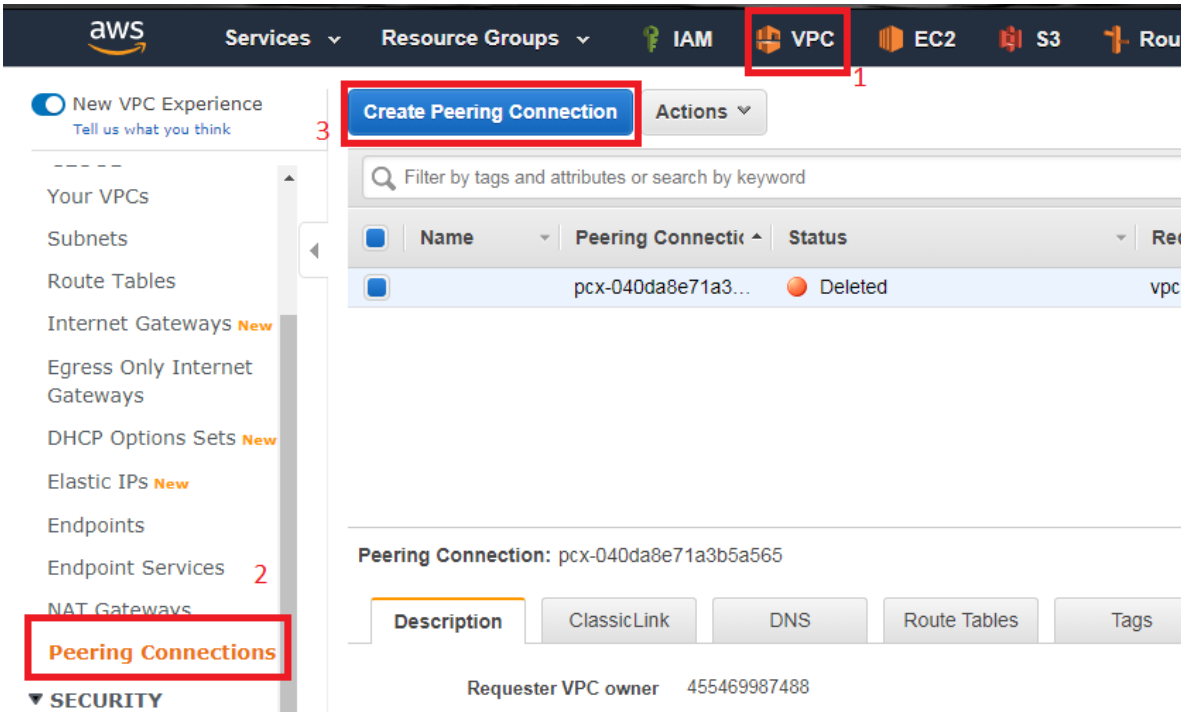
VPC CIDRs

CIDR	Status	Status Reason
172.16.0.0/16	associated	

IPv4 CIDR block*: 172.16.0.0/24

* Required

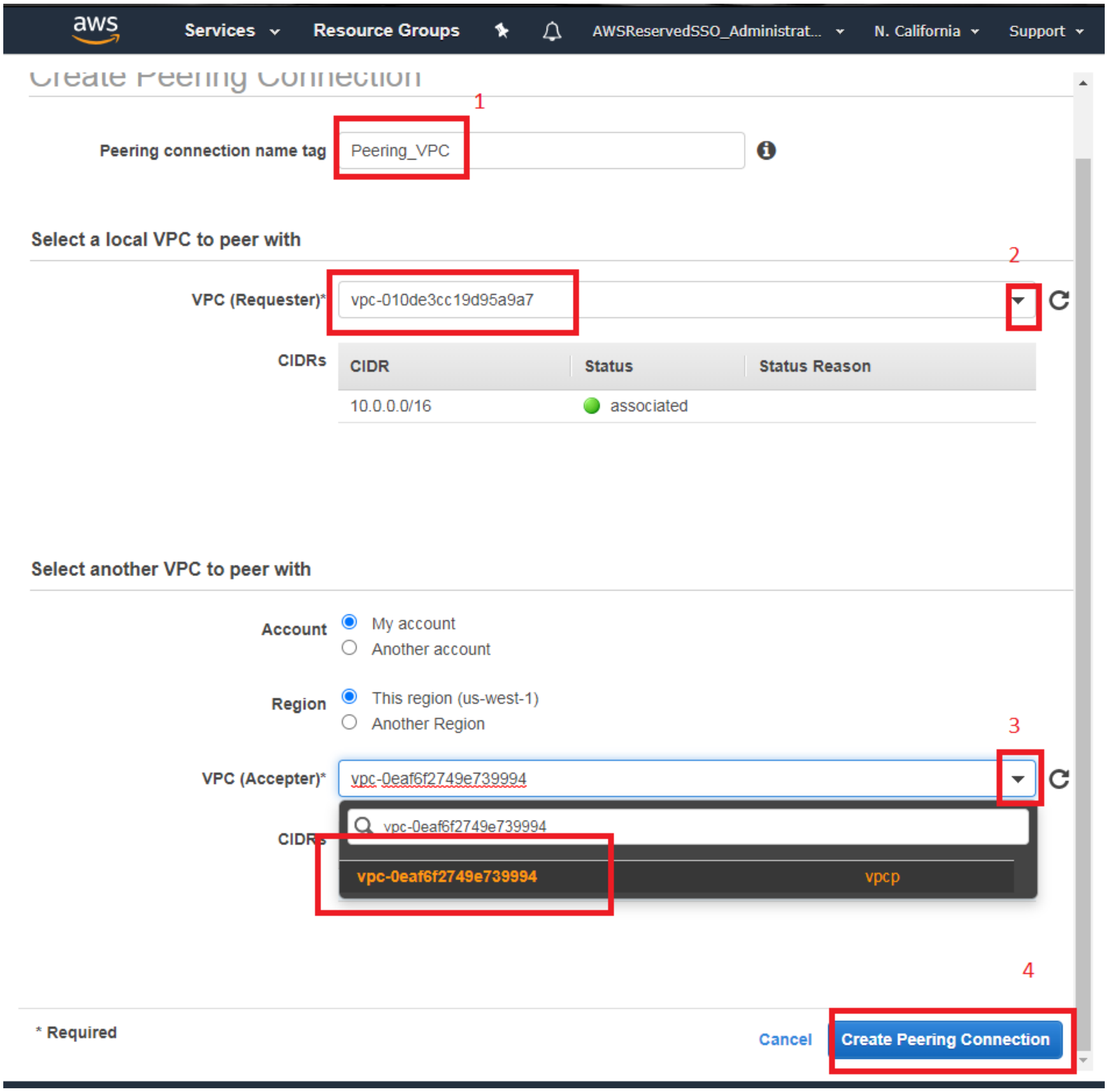
Create and accept VPC Peering



Name	Peering Connectic	Status	Re
	pcx-040da8e71a3…	Deleted	vpc

Peering Connection: pcx-040da8e71a3b5a565

Requester VPC owner 455469987488



Create Peering Connection

Peering connection name tag: Peering_VPC

Select a local VPC to peer with

VPC (Requester)*: vpc-010de3cc19d95a9a7

CIDRs

CIDR	Status	Status Reason
10.0.0.0/16	● associated	

Select another VPC to peer with

Account: ● My account ○ Another account

Region: ● This region (us-west-1) ○ Another Region

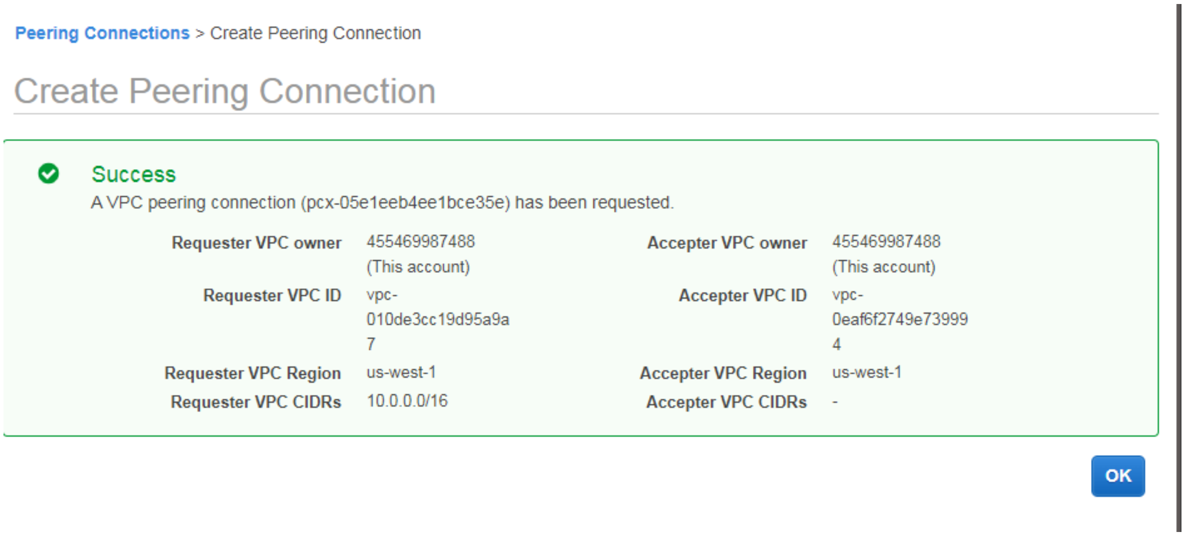
VPC (Accepter)*: vpc-0eaf6f2749e739994

CIDRs

vpc-0eaf6f2749e739994

vpc-0eaf6f2749e739994 vpcp

* Required

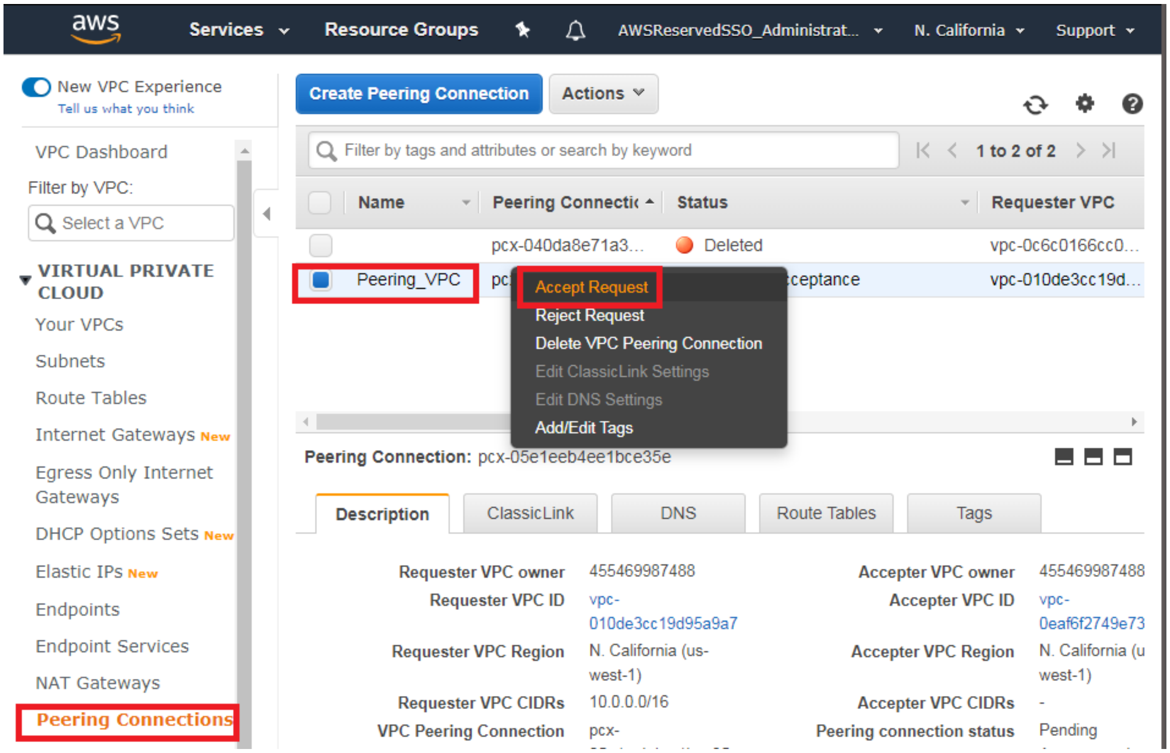


Create Peering Connection

Success

A VPC peering connection (pcx-05e1eeb4ee1bce35e) has been requested.

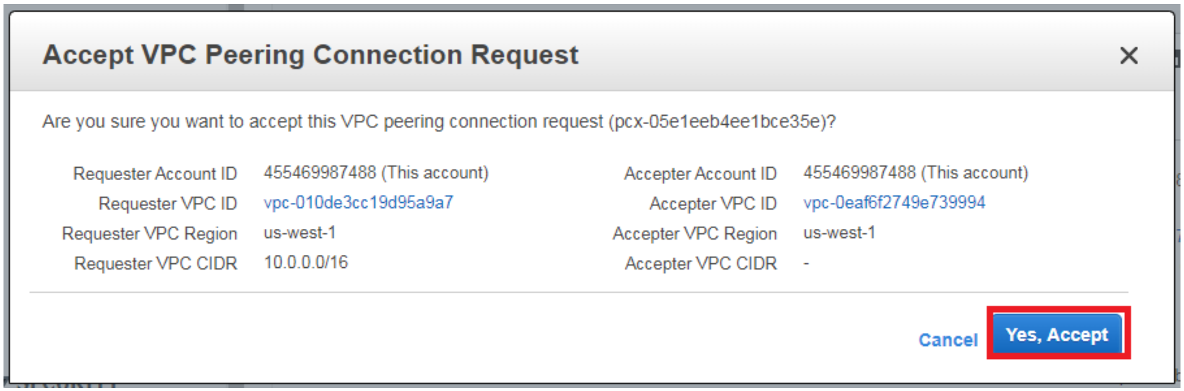
Requester VPC owner	455469987488 (This account)	Accepter VPC owner	455469987488 (This account)
Requester VPC ID	vpc-010de3cc19d95a9a7	Accepter VPC ID	vpc-0eaf6f2749e739994
Requester VPC Region	us-west-1	Accepter VPC Region	us-west-1
Requester VPC CIDRs	10.0.0.0/16	Accepter VPC CIDRs	-



	Name	Peering Connecti	Status	Requester VPC
□		pcx-040da8e71a3...	● Deleted	vpc-0c6c0166cc0...
☑	Peering_VPC	pc	ceptance	vpc-010de3cc19d...

Peering Connection: pcx-05e1eeb4ee1bce35e

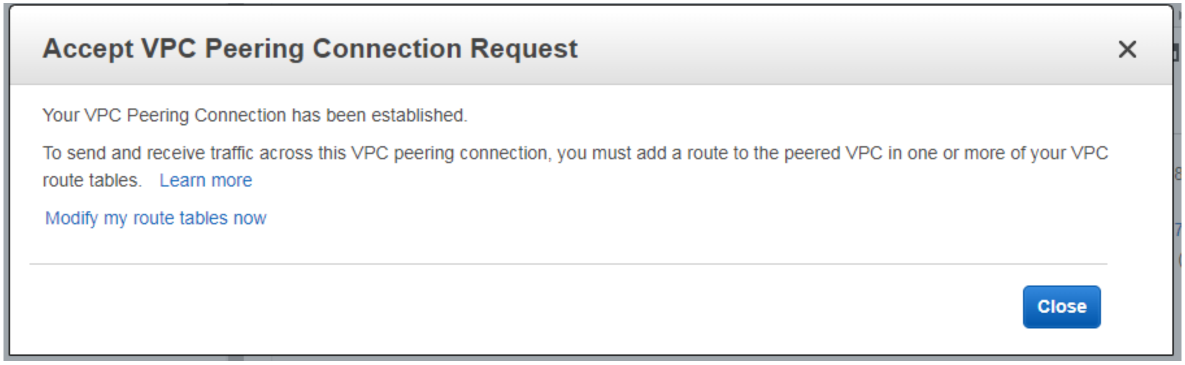
Requester VPC owner	455469987488	Accepter VPC owner	455469987488
Requester VPC ID	vpc-010de3cc19d95a9a7	Accepter VPC ID	vpc-0eaf6f2749e73
Requester VPC Region	N. California (us-west-1)	Accepter VPC Region	N. California (u west-1)
Requester VPC CIDRs	10.0.0.0/16	Accepter VPC CIDRs	-
VPC Peering Connection	pcx-	Peering connection status	Pending



Accept VPC Peering Connection Request

Are you sure you want to accept this VPC peering connection request (pcx-05e1eeb4ee1bce35e)?

Requester Account ID	455469987488 (This account)	Accepter Account ID	455469987488 (This account)
Requester VPC ID	vpc-010de3cc19d95a9a7	Accepter VPC ID	vpc-0eaf6f2749e739994
Requester VPC Region	us-west-1	Accepter VPC Region	us-west-1
Requester VPC CIDR	10.0.0.0/16	Accepter VPC CIDR	-



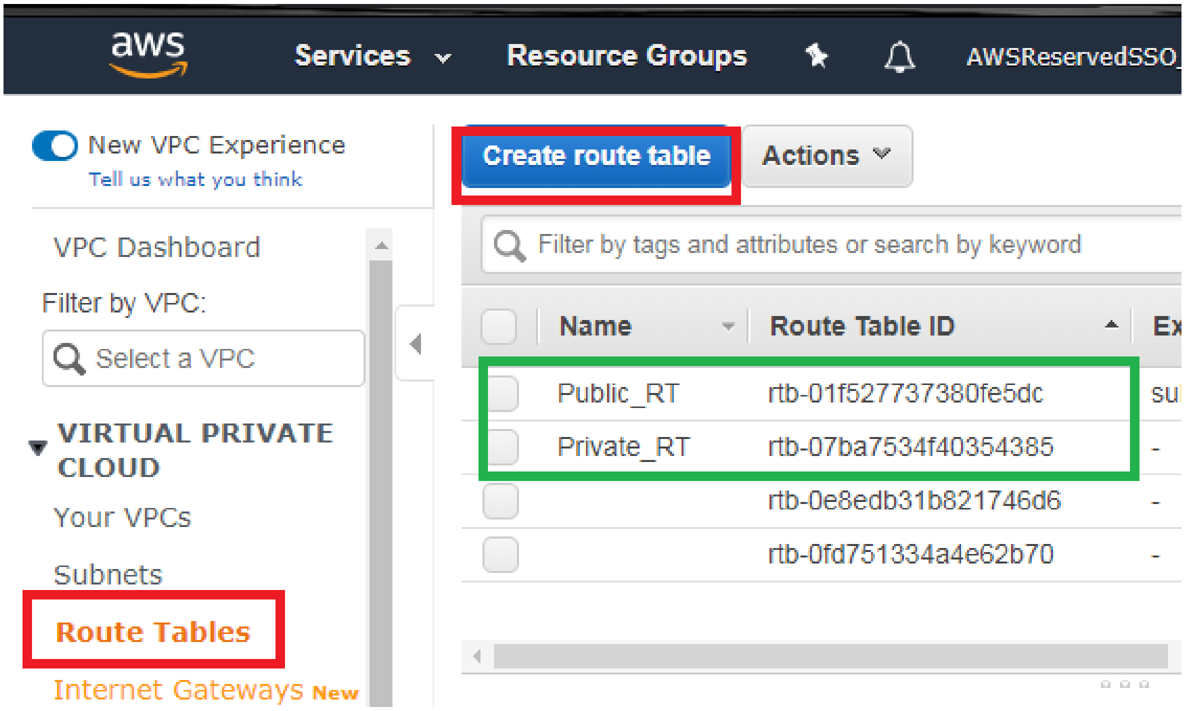
Accept VPC Peering Connection Request

Your VPC Peering Connection has been established.

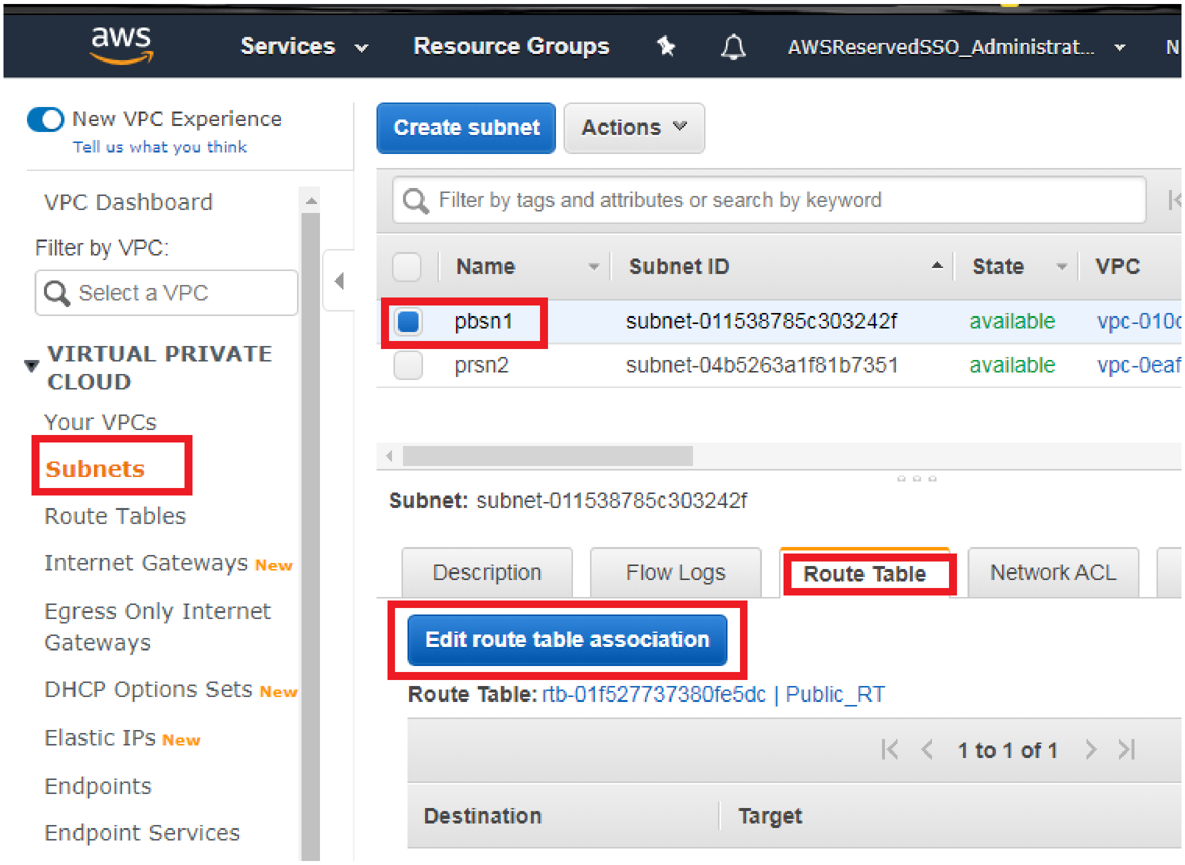
To send and receive traffic across this VPC peering connection, you must add a route to the peered VPC in one or more of your VPC route tables. Learn more

Modify my route tables now

Create Custom Routing Tables and associate to subnets



Name	Route Table ID	Ex
Public_RT	rtb-01f527737380fe5dc	su
Private_RT	rtb-07ba7534f40354385	-
	rtb-0e8edb31b821746d6	-
	rtb-0fd751334a4e62b70	-



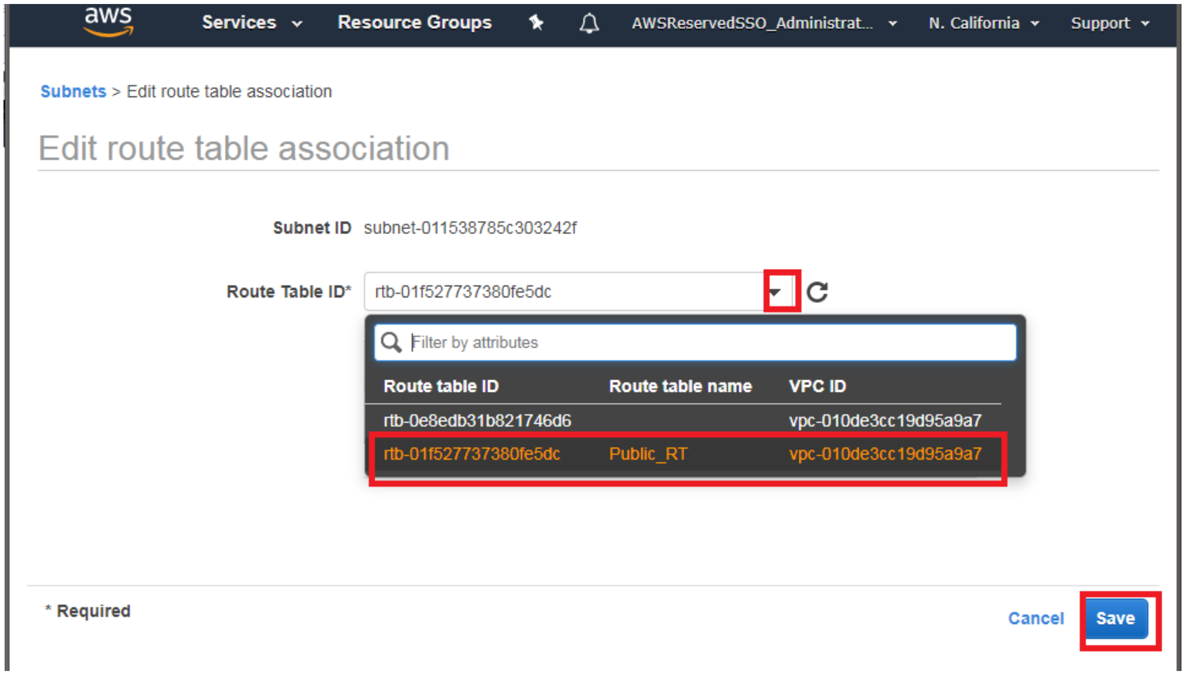
Name	Subnet ID	State	VPC
pbsn1	subnet-011538785c303242f	available	vpc-010c
prsn2	subnet-04b5263a1f81b7351	available	vpc-0eaf

Subnet: subnet-011538785c303242f

Route Table: rtb-01f527737380fe5dc | Public_RT

1 to 1 of 1

Destination	Target



Subnets > Edit route table association

Edit route table association

Subnet ID subnet-011538785c303242f

Route Table ID* rtb-01f527737380fe5dc

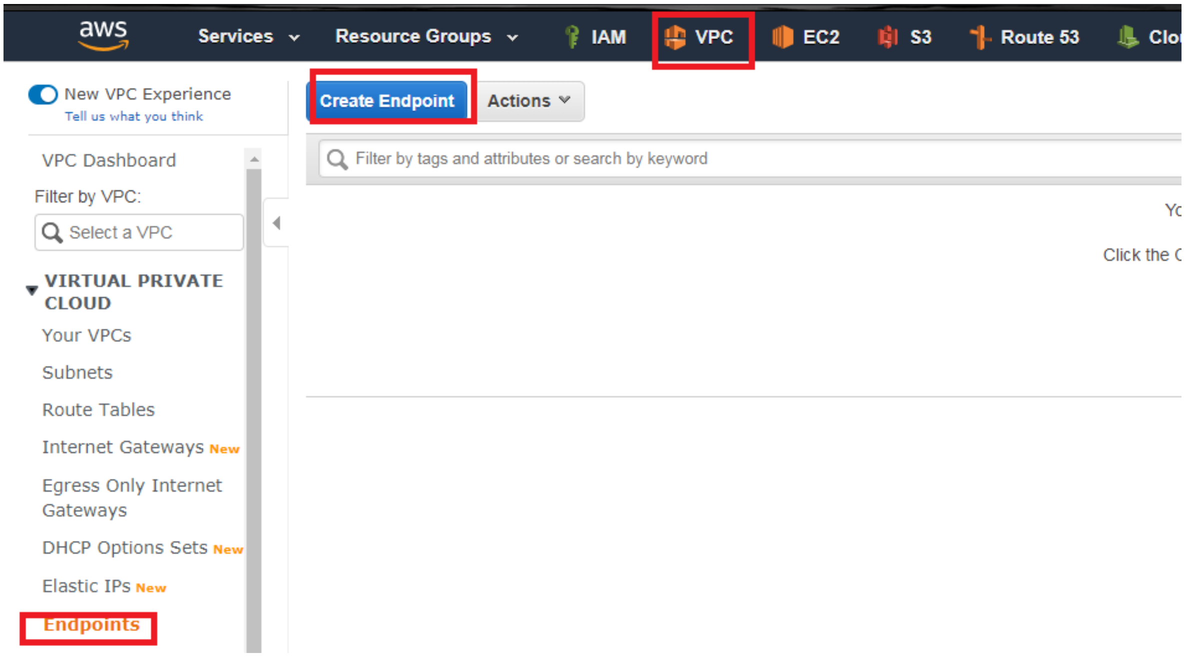
Filter by attributes

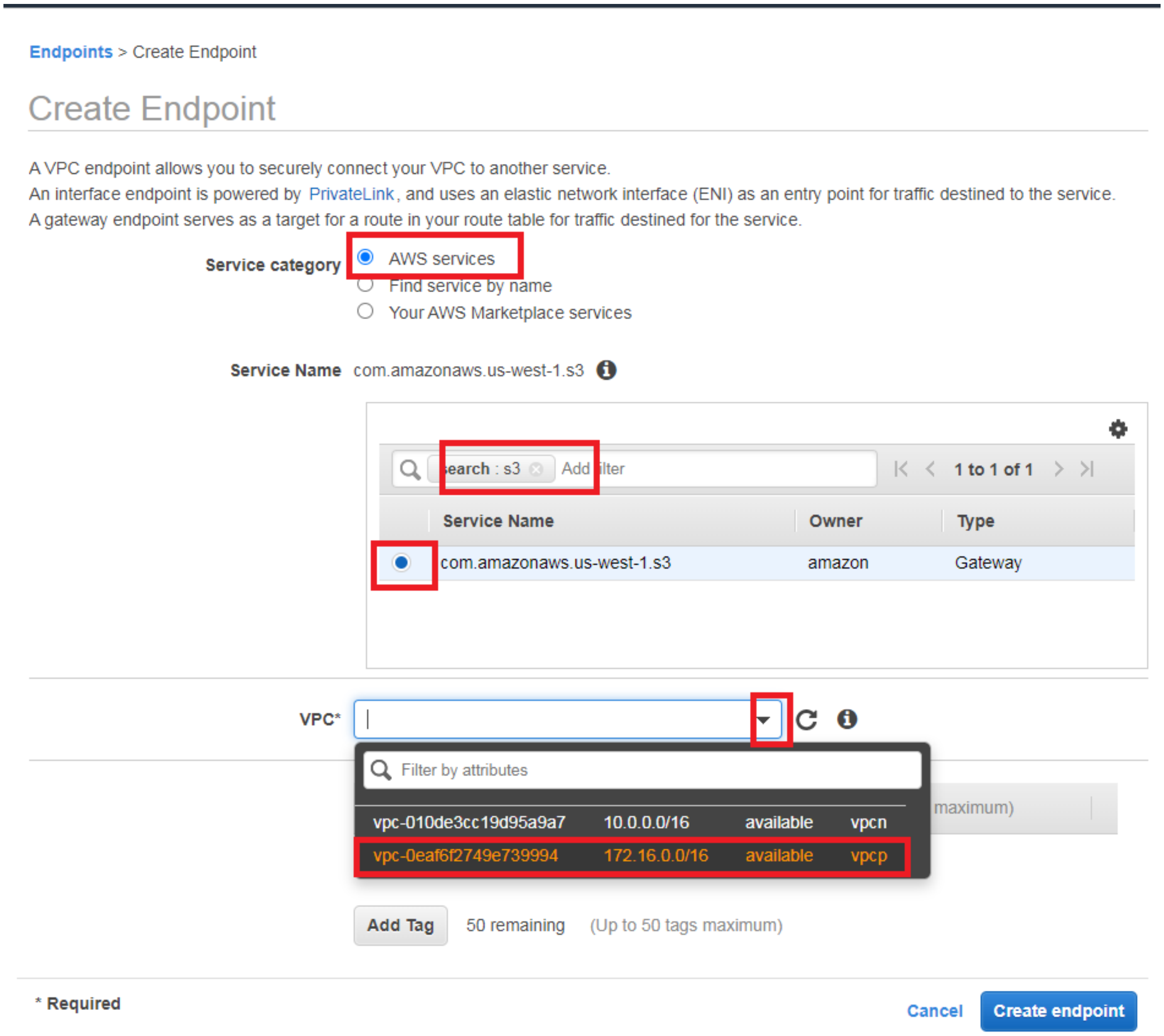
Route table ID	Route table name	VPC ID
rtb-0e8edb31b821746d6		vpc-010de3cc19d95a9a7
rtb-01f527737380fe5dc	Public_RT	vpc-010de3cc19d95a9a7

* Required

And the same case for the Private Subnet.

Create S3 VPC Endpoint





Endpoints > Create Endpoint

Create Endpoint

A VPC endpoint allows you to securely connect your VPC to another service.
An interface endpoint is powered by PrivateLink, and uses an elastic network interface (ENI) as an entry point for traffic destined to the service.
A gateway endpoint serves as a target for a route in your route table for traffic destined for the service.

Service category
● AWS services
○ Find service by name
○ Your AWS Marketplace services

Service Name com.amazonaws.us-west-1.s3

search : s3 Add filter — 1 to 1 of 1

	Service Name	Owner	Type
●	com.amazonaws.us-west-1.s3	amazon	Gateway

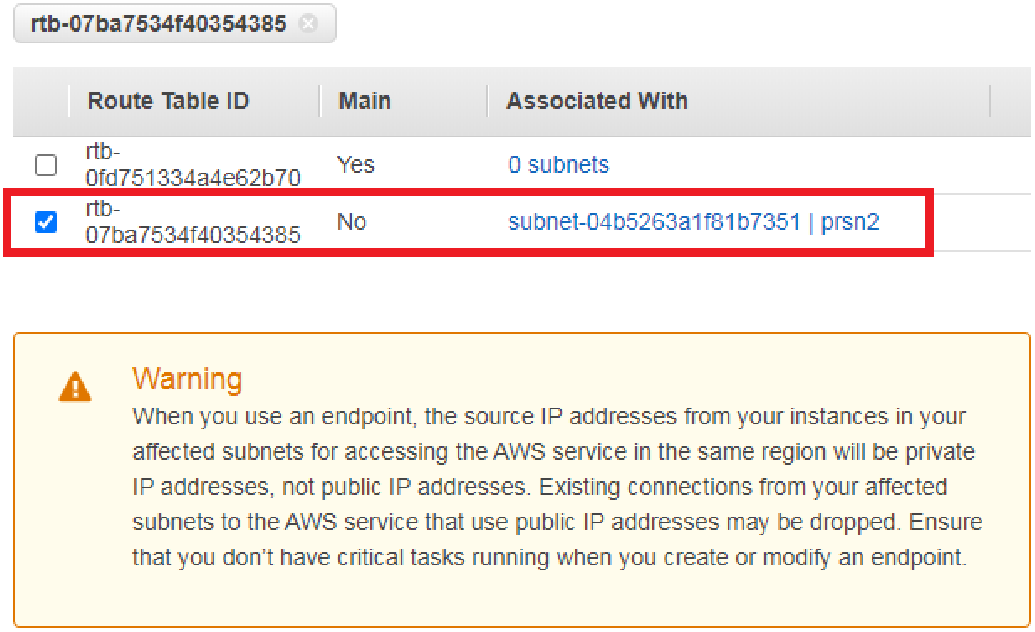
VPC*

Filter by attributes

```
vpc-010de3cc19d95a9a7   10.0.0.0/16    available   vpcn
vpc-0eaf6f2749e739994   172.16.0.0/16  available   vpcp
```

Add Tag 50 remaining (Up to 50 tags maximum)

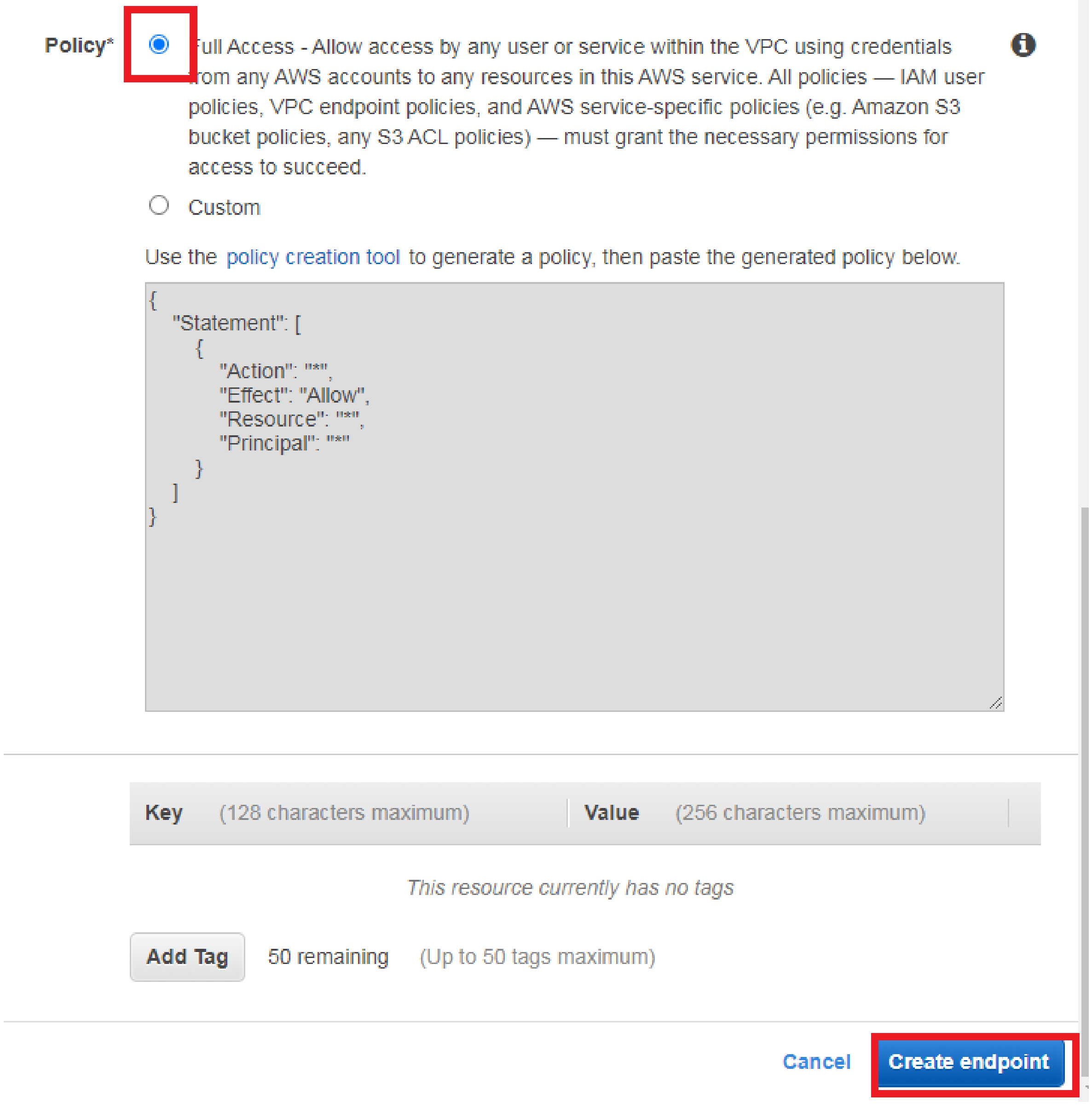
* Required



	Route Table ID	Main	Associated With
□	rtb-0fd751334a4e62b70	Yes	0 subnets
☑	rtb-07ba7534f40354385	No	subnet-04b5263a1f81b7351 | prsn2

Warning

When you use an endpoint, the source IP addresses from your instances in your affected subnets for accessing the AWS service in the same region will be private IP addresses, not public IP addresses. Existing connections from your affected subnets to the AWS service that use public IP addresses may be dropped. Ensure that you don't have critical tasks running when you create or modify an endpoint.



Policy* ● Full Access - Allow access by any user or service within the VPC using credentials from any AWS accounts to any resources in this AWS service. All policies — IAM user policies, VPC endpoint policies, and AWS service-specific policies (e.g. Amazon S3 bucket policies, any S3 ACL policies) — must grant the necessary permissions for access to succeed.

○ Custom

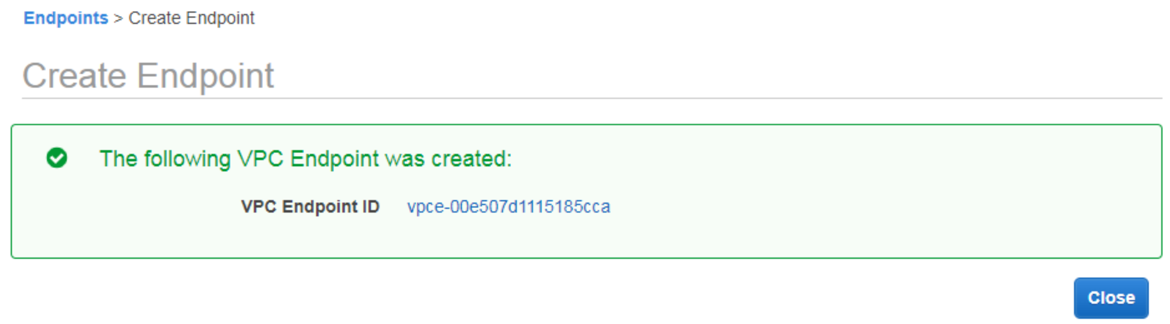
Use the policy creation tool to generate a policy, then paste the generated policy below.

```
{
  "Statement": [
    {
      "Action": "*",
      "Effect": "Allow",
      "Resource": "*",
      "Principal": "*"
    }
  ]
}
```

Key (128 characters maximum)	Value (256 characters maximum)

This resource currently has no tags

Add Tag
50 remaining (Up to 50 tags maximum)
Cancel
Create endpoint



Endpoints > Create Endpoint

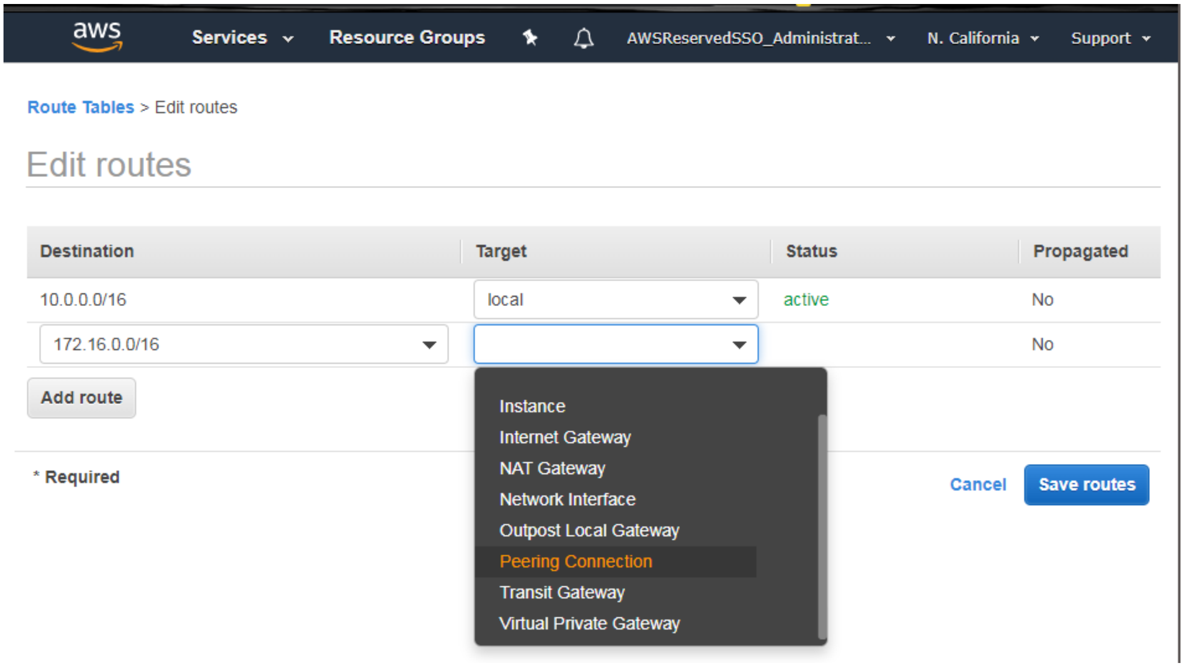
Create Endpoint

The following VPC Endpoint was created:

VPC Endpoint ID vpce-00e507d1115185cca

Modifying routing tables to reach VPC Peering

For the public RT, you have to edit the RT and the destination is the IP Range from vpcp and the target is the VPC Peering Connection.

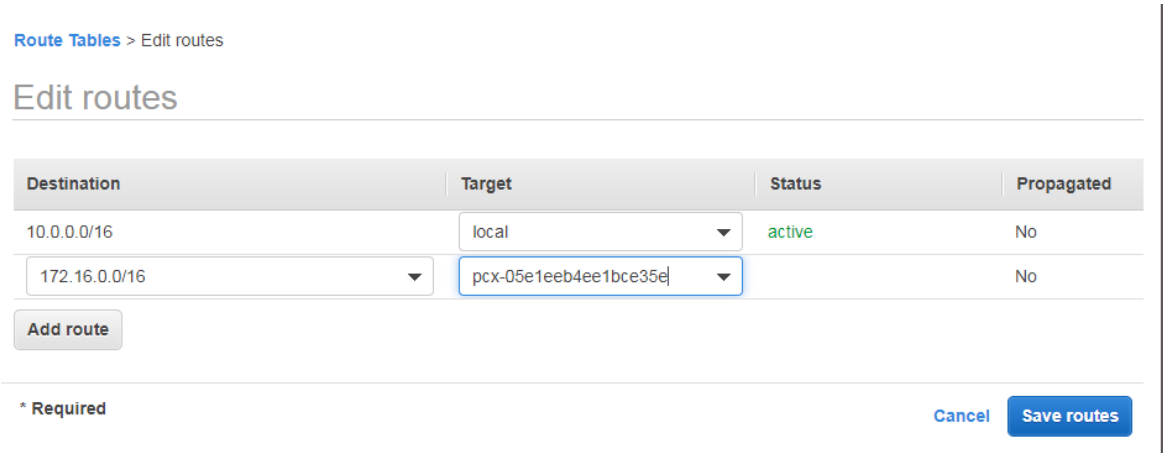


aws Services Resource Groups AWSReservedSSO_Administrat... N. California Support

Route Tables > Edit routes

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
172.16.0.0/16			No

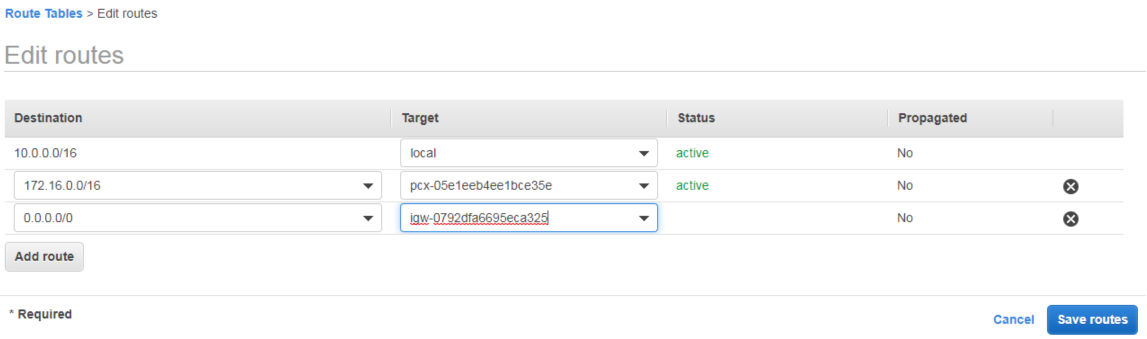


Route Tables > Edit routes

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
172.16.0.0/16	pcx-05e1eeb4ee1bce35e		No

Add the IGW as default route on Public RT.

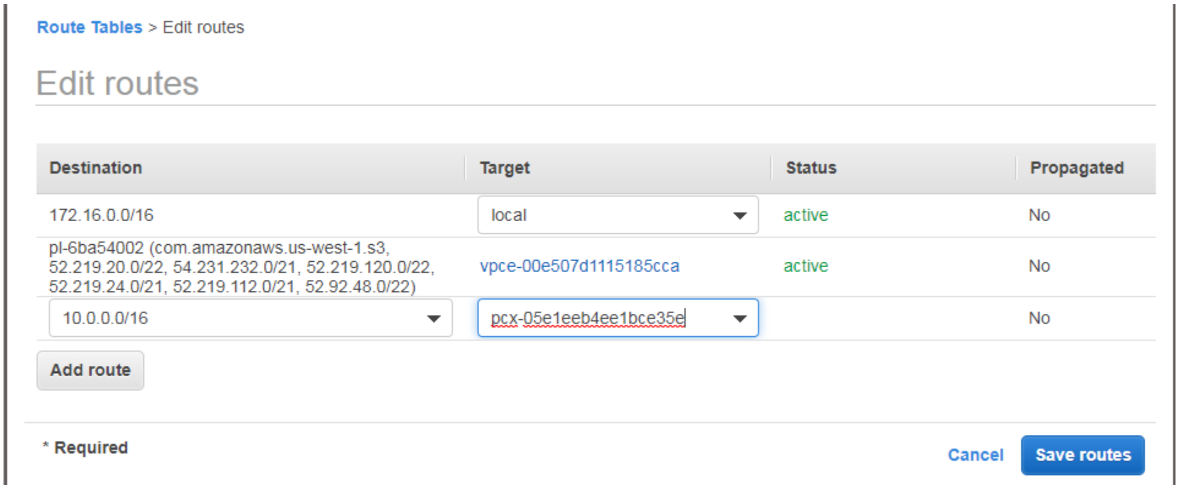


Route Tables > Edit routes

Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	active	No
172.16.0.0/16	pcx-05e1eeb4ee1bce35e	active	No
0.0.0.0/0	igw-0792dfa6695eca325		No

For the private RT is similar, however check vpcn range and the additional VPC Endpoint



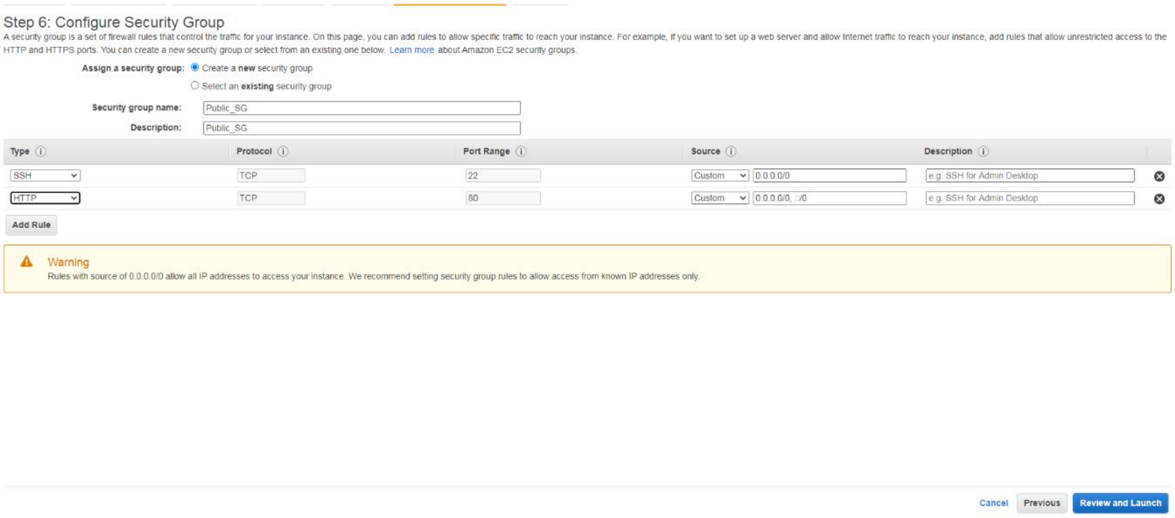
Route Tables > Edit routes

Edit routes

Destination	Target	Status	Propagated
172.16.0.0/16	local	active	No
pl-6ba54002 (com.amazonaws.us-west-1.s3, 52.219.20.0/22, 54.231.232.0/21, 52.219.120.0/22, 52.219.24.0/21, 52.219.112.0/21, 52.92.48.0/22)	vpce-00e507d1115185cca	active	No
10.0.0.0/16	pcx-05e1eeb4ee1bce35e		No

Create instances

You have to create 2 similar instances, however remember that Sec Group from Public Instances allows HTTP inbound connections.



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

Assign a security group: ● Create a **new** security group
○ Select an **existing** security group

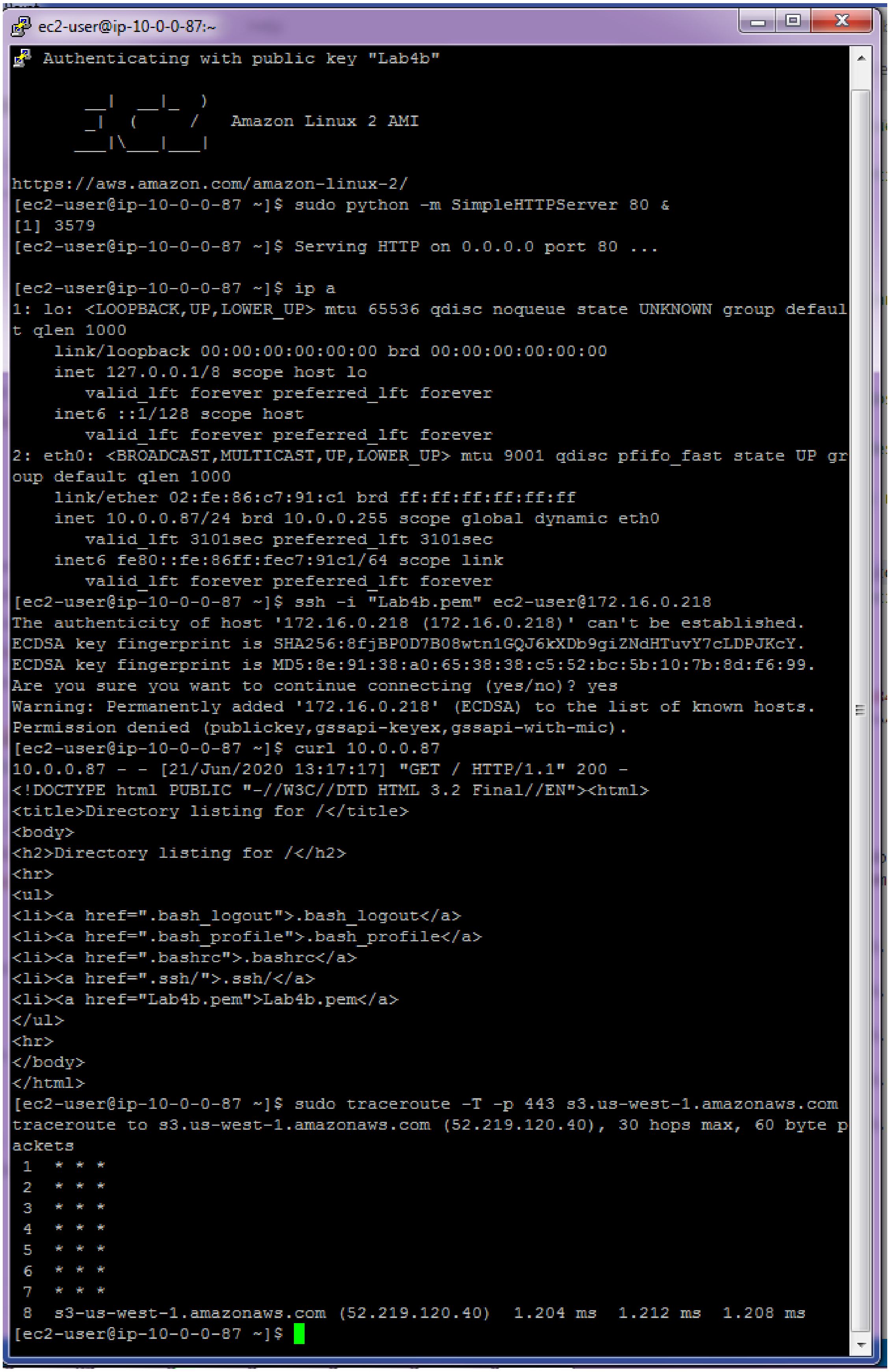
Security group name: Public_SG

Description: Public_SG

Type	Protocol	Port Range	Source		Description
SSH	TCP	22	Custom	0.0.0.0/0	e.g. SSH for Admin Desktop
HTTP	TCP	80	Custom	0.0.0.0/0, ::/0	e.g. SSH for Admin Desktop

Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

And go to “Review Configurations” Section. Here is come the evidence of working using Web Management Console.



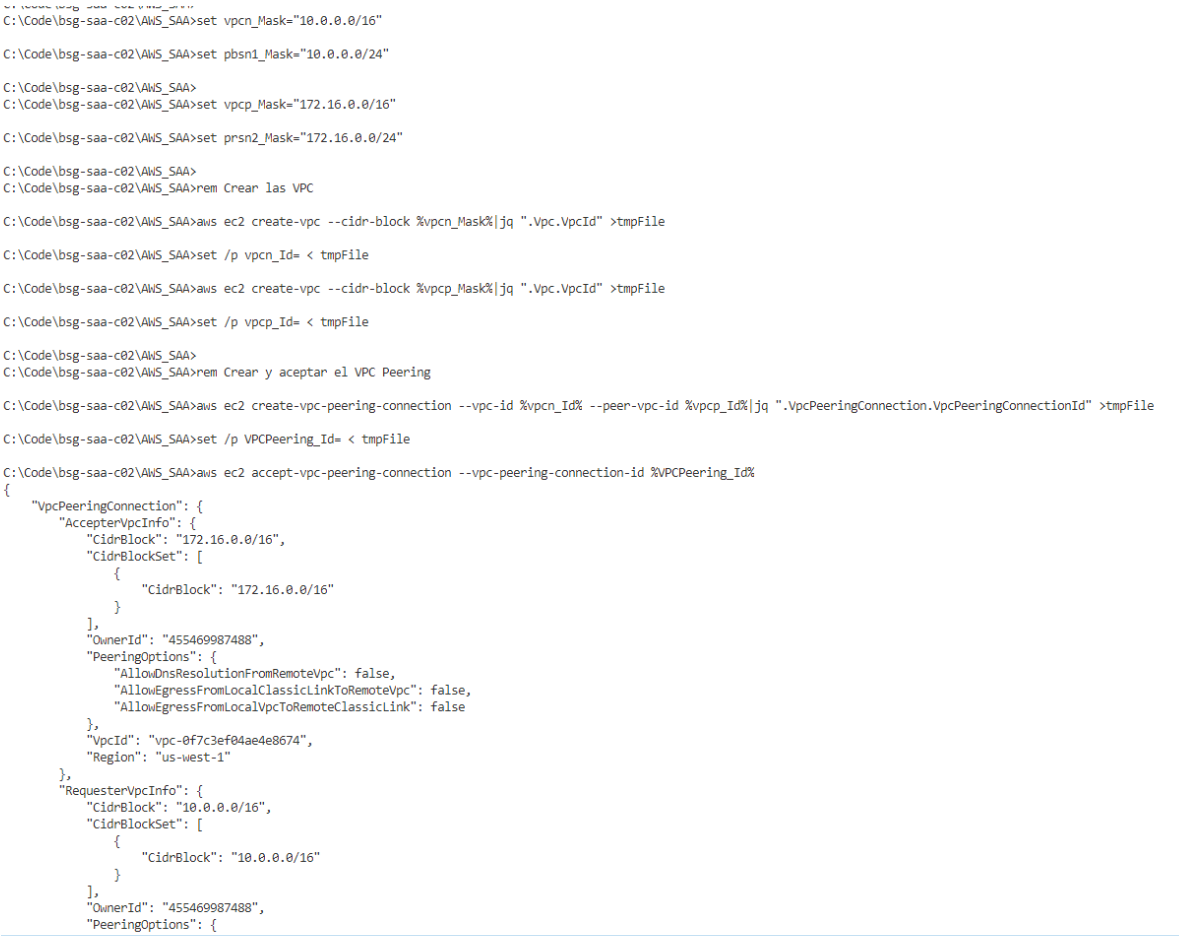
ec2-user@ip-10-0-0-87:~

```
Authenticating with public key "Lab4b"

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

https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-10-0-0-87 ~]$ sudo python -m SimpleHTTPServer 80 &
[1] 3579
[ec2-user@ip-10-0-0-87 ~]$ Serving HTTP on 0.0.0.0 port 80 ...

[ec2-user@ip-10-0-0-87 ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP gr
oup default qlen 1000
    link/ether 02:fe:86:c7:91:c1 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.87/24 brd 10.0.0.255 scope global dynamic eth0
       valid_lft 3101sec preferred_lft 3101sec
    inet6 fe80::fe:86ff:fec7:91c1/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@ip-10-0-0-87 ~]$ ssh -i "Lab4b.pem" ec2-user@172.16.0.218
The authenticity of host '172.16.0.218 (172.16.0.218)' can't be established.
ECDSA key fingerprint is SHA256:8fjBP0D7B08wtn1GQJ6kXDb9giZNdHTuvY7cLDPJKcY.
ECDSA key fingerprint is MD5:8e:91:38:a0:65:38:38:c5:52:bc:5b:10:7b:8d:f6:99.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.0.218' (ECDSA) to the list of known hosts.
Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-10-0-0-87 ~]$ curl 10.0.0.87
10.0.0.87 - - [21/Jun/2020 13:17:17] "GET / HTTP/1.1" 200 -
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_logout">.bash_logout</a>
<li><a href=".bash_profile">.bash_profile</a>
<li><a href=".bashrc">.bashrc</a>
<li><a href=".ssh/">.ssh/</a>
<li><a href="Lab4b.pem">Lab4b.pem</a>
</ul>
<hr>
</body>
</html>
[ec2-user@ip-10-0-0-87 ~]$ sudo traceroute -T -p 443 s3.us-west-1.amazonaws.com
traceroute to s3.us-west-1.amazonaws.com (52.219.120.40), 30 hops max, 60 byte p
ackets
 1  * * *
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  s3-us-west-1.amazonaws.com (52.219.120.40)  1.204 ms  1.212 ms  1.208 ms
[ec2-user@ip-10-0-0-87 ~]$
```

Lab 4B using Command Line (Windows)

Create VPC, VPC Peering, Subnets, IGW and Route Table

```
rem Crear las VPC
aws ec2 create-vpc --cidr-block %vpcn_Mask%|jq ".Vpc.VpcId" >tmpFile
set /p vpcn_Id= < tmpFile
aws ec2 create-vpc --cidr-block %vpcp_Mask%|jq ".Vpc.VpcId" >tmpFile
set /p vpcp_Id= < tmpFile

rem Crear y aceptar el VPC Peering
aws ec2 create-vpc-peering-connection --vpc-id %vpcn_Id% --peer-vpc-id %vpcp_Id%|jq ".VpcPeeringConnection.VpcPeeringConnectionId" >tmpFile
set /p VPCPeering_Id= < tmpFile
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id %VPCPeering_Id%


rem Crear subredes
aws ec2 create-subnet --vpc-id %vpcn_Id% --cidr-block %pbsn1_Mask% --availability-zone %first_az%|jq ".Subnet.SubnetId" >tmpFile
set /p pbsn1_Id= < tmpFile
aws ec2 create-subnet --vpc-id %vpcp_Id% --cidr-block %prsn2_Mask% --availability-zone %first_az%|jq ".Subnet.SubnetId" >tmpFile
set /p prsn2_Id= < tmpFile

rem Crear el Internet Gateway IGW y asignarlo a la VPC
aws ec2 create-internet-gateway|jq ".InternetGateway.InternetGatewayId"  >tmpFile
set /p IGW_Id= < tmpFile
aws ec2 attach-internet-gateway --vpc-id %vpcn_Id% --internet-gateway-id %IGW_Id%
```

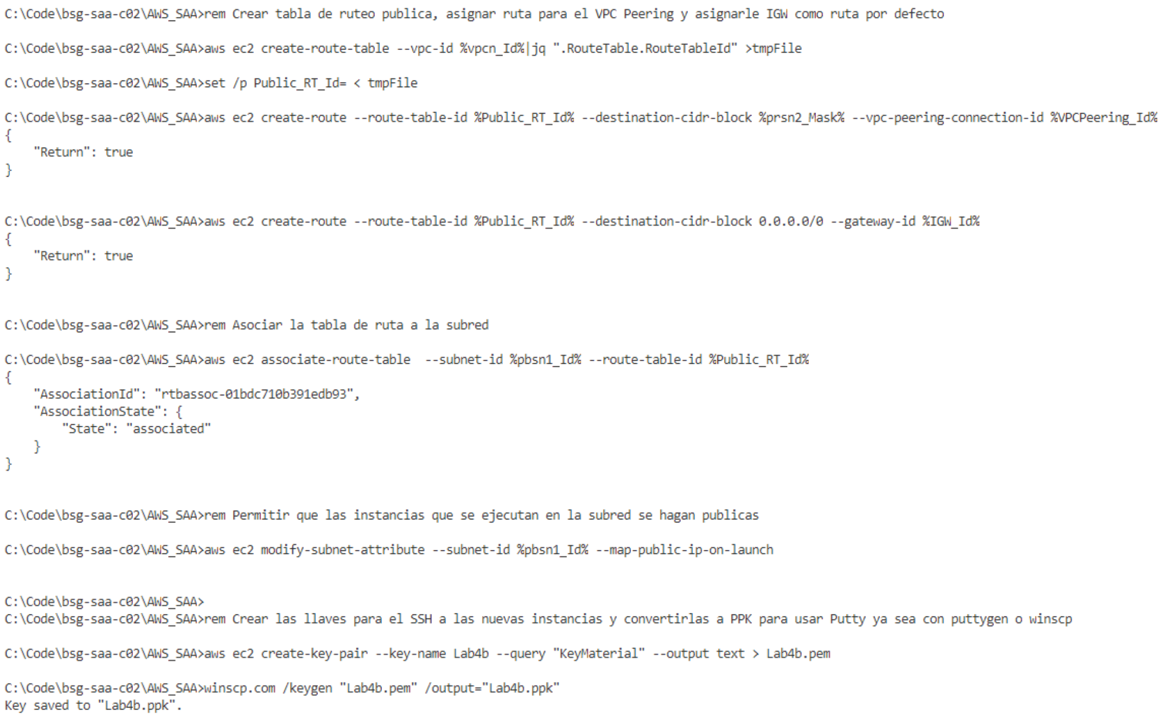


```
C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>set vpcn_Mask="10.0.0.0/16"

C:\Code\bsg-saa-c02\AWS_SAA>set pbsn1_Mask="10.0.0.0/24"

C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>set vpcp_Mask="172.16.0.0/16"

C:\Code\bsg-saa-c02\AWS_SAA>set prsn2_Mask="172.16.0.0/24"

C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear las VPC

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-vpc --cidr-block %vpcn_Mask%|jq ".Vpc.VpcId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p vpcn_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-vpc --cidr-block %vpcp_Mask%|jq ".Vpc.VpcId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p vpcp_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear y aceptar el VPC Peering

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-vpc-peering-connection --vpc-id %vpcn_Id% --peer-vpc-id %vpcp_Id%|jq ".VpcPeeringConnection.VpcPeeringConnectionId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p VPCPeering_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id %VPCPeering_Id%
{
    "VpcPeeringConnection": {
        "AccepterVpcInfo": {
            "CidrBlock": "172.16.0.0/16",
            "CidrBlockSet": [
                {
                    "CidrBlock": "172.16.0.0/16"
                }
            ],
            "OwnerId": "455469987488",
            "PeeringOptions": {
                "AllowDnsResolutionFromRemoteVpc": false,
                "AllowEgressFromLocalClassicLinkToRemoteVpc": false,
                "AllowEgressFromLocalVpcToRemoteClassicLink": false
            },
            "VpcId": "vpc-0f7c3ef04ae4e8674",
            "Region": "us-west-1"
        },
        "RequesterVpcInfo": {
            "CidrBlock": "10.0.0.0/16",
            "CidrBlockSet": [
                {
                    "CidrBlock": "10.0.0.0/16"
                }
            ],
            "OwnerId": "455469987488",
            "PeeringOptions": {
```

Create Public Routing Table with VPC Peering, IGW. Create Keypair and Public Security Group

```
rem Crear tabla de ruteo publica, asignar ruta para el VPC Peering y asignar
le IGW como ruta por defecto
aws ec2 create-route-table --vpc-
id %vpcn_Id%|jq ".RouteTable.RouteTableId" >tmpFile
set /p Public_RT_Id= < tmpFile
aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-
block %prsn2_Mask% --vpc-peering-connection-id %VPCPeering_Id%
aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-
block 0.0.0.0/0 --gateway-id %IGW_Id%
rem Asociar la tabla de ruta a la subred
aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-
id %Public_RT_Id%
rem Permitir que las instancias que se ejecutan en la subred se hagan public
as
aws ec2 modify-subnet-attribute --subnet-id %pbsn1_Id% --map-public-ip-on-
launch
```

```
rem Crear las llaves para el SSH a las nuevas instancias y convertirlas a PPK para usar Putty ya sea con puttygen o winscp
aws ec2 create-key-pair --key-name Lab4b --query "KeyMaterial" --output text > Lab4b.pem
winscp.com /keygen "Lab4b.pem" /output="Lab4b.ppk"


rem Crear los Security Groups para esas instancias
aws ec2 create-security-group --group-name "SecGrp VPC Public" --description "Security group for Instance A" --vpc-id %vpcn_Id% |jq ".GroupId">tmpFile
set /p SSH_Sec_Group_n_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SSH_Sec_Group_n_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0
aws ec2 authorize-security-group-ingress --group-id %SSH_Sec_Group_n_Id% --protocol tcp --port 80 --cidr 0.0.0.0/0
```

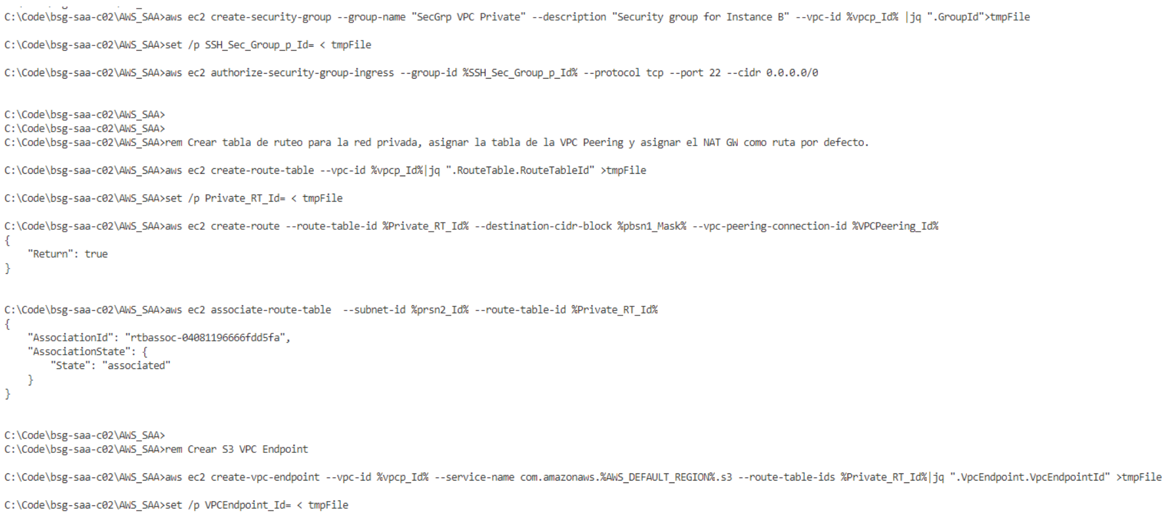


```
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear tabla de ruteo publica, asignar ruta para el VPC Peering y asignarle IGW como ruta por defecto

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-route-table --vpc-id %vpcn_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p Public_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-block %prsn2_Mask% --vpc-peering-connection-id %VPCPeering_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-route --route-table-id %Public_RT_Id% --destination-cidr-block 0.0.0.0/0 --gateway-id %IGW_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA>rem Asociar la tabla de ruta a la subred

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 associate-route-table  --subnet-id %pbsn1_Id% --route-table-id %Public_RT_Id%
{
    "AssociationId": "rtbassoc-01bdc710b391edb93",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA>rem Permitir que las instancias que se ejecutan en la subred se hagan publicas

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 modify-subnet-attribute --subnet-id %pbsn1_Id% --map-public-ip-on-launch


C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear las llaves para el SSH a las nuevas instancias y convertirlas a PPK para usar Putty ya sea con puttygen o winscp

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-key-pair --key-name Lab4b --query "KeyMaterial" --output text > Lab4b.pem

C:\Code\bsg-saa-c02\AWS_SAA>winscp.com /keygen "Lab4b.pem" /output="Lab4b.ppk"
Key saved to "Lab4b.ppk".
```

Create Private Sec Group, Private Routing Table and S3 VPC Endpoint for Routing Subnet

```
aws ec2 create-security-group --group-name "SecGrp VPC Private" --description "Security group for Instance B" --vpc-id %vpcp_Id% |jq ".GroupId">tmpFile
set /p SSH_Sec_Group_p_Id= < tmpFile
aws ec2 authorize-security-group-ingress --group-id %SSH_Sec_Group_p_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0
```

```
rem Crear tabla de ruteo para la red privada, asignar la tabla de la VPC Pee
ring y asignar el NAT GW como ruta por defecto.
aws ec2 create-route-table --vpc-
id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile
set /p Private_RT_Id= < tmpFile
aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-
block %pbsn1_Mask% --vpc-peering-connection-id %VPCPeering_Id%
aws ec2 associate-route-table  --subnet-id %prsn2_Id% --route-table-
id %Private_RT_Id%

rem Crear S3 VPC Endpoint
aws ec2 create-vpc-endpoint --vpc-id %vpcp_Id% --service-
name com.amazonaws.%AWS_DEFAULT_REGION%.s3 --route-table-
ids %Private_RT_Id%|jq ".VpcEndpoint.VpcEndpointId" >tmpFile
set /p VPCEndpoint_Id= < tmpFile
```

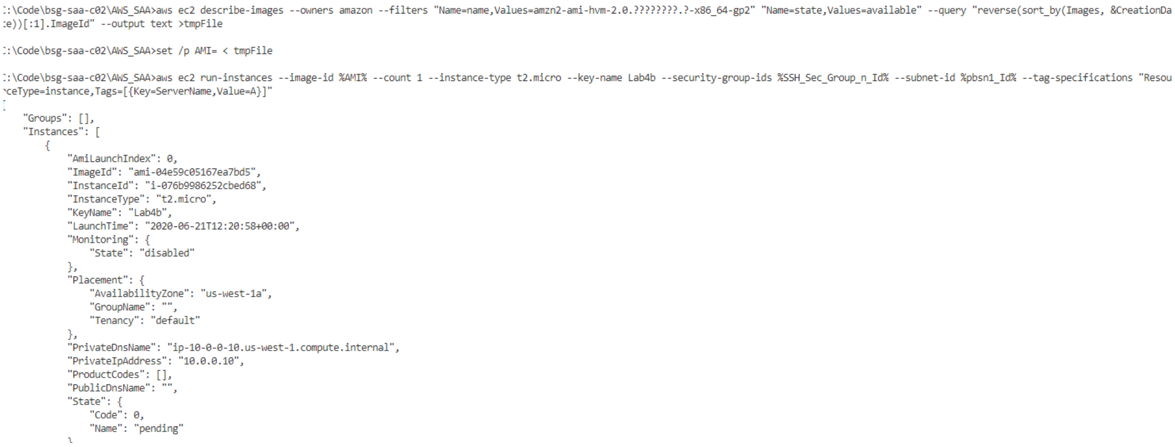


```
C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-security-group --group-name "SecGrp VPC Private" --description "Security group for Instance B" --vpc-id %vpcp_Id% |jq ".GroupId">tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p SSH_Sec_Group_p_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 authorize-security-group-ingress --group-id %SSH_Sec_Group_p_Id% --protocol tcp --port 22 --cidr 0.0.0.0/0


C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear tabla de ruteo para la red privada, asignar la tabla de la VPC Peering y asignar el NAT GW como ruta por defecto.

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-route-table --vpc-id %vpcp_Id%|jq ".RouteTable.RouteTableId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p Private_RT_Id= < tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-route --route-table-id %Private_RT_Id% --destination-cidr-block %pbsn1_Mask% --vpc-peering-connection-id %VPCPeering_Id%
{
    "Return": true
}


C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 associate-route-table  --subnet-id %prsn2_Id% --route-table-id %Private_RT_Id%
{
    "AssociationId": "rtbassoc-04081196666fdd5fa",
    "AssociationState": {
        "State": "associated"
    }
}


C:\Code\bsg-saa-c02\AWS_SAA>
C:\Code\bsg-saa-c02\AWS_SAA>rem Crear S3 VPC Endpoint

C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 create-vpc-endpoint --vpc-id %vpcp_Id% --service-name com.amazonaws.%AWS_DEFAULT_REGION%.s3 --route-table-ids %Private_RT_Id%|jq ".VpcEndpoint.VpcEndpointId" >tmpFile

C:\Code\bsg-saa-c02\AWS_SAA>set /p VPCEndpoint_Id= < tmpFile
```

Create Instances

```
rem Crear instancias. Este es el commando para Amazon Linux 2 con Python 2
aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-
ami-hvm-2.0.????????.?-x86_64-gp2" "Name=state,Values=available" --
query "reverse(sort_by(Images, &CreationDate))[:1].ImageId" --
output text >tmpFile
rem Puede utilizar Amazon Linux 2023 con Python 3
aws ec2 describe-images --owners amazon --filters "Name=name,Values=al2023-
ami-2023*-x86_64" "Name=state,Values=available" --query
"reverse(sort_by(Images, &CreationDate))[:1].ImageId" --output text >tmpFile
set /p AMI= < tmpFile
```

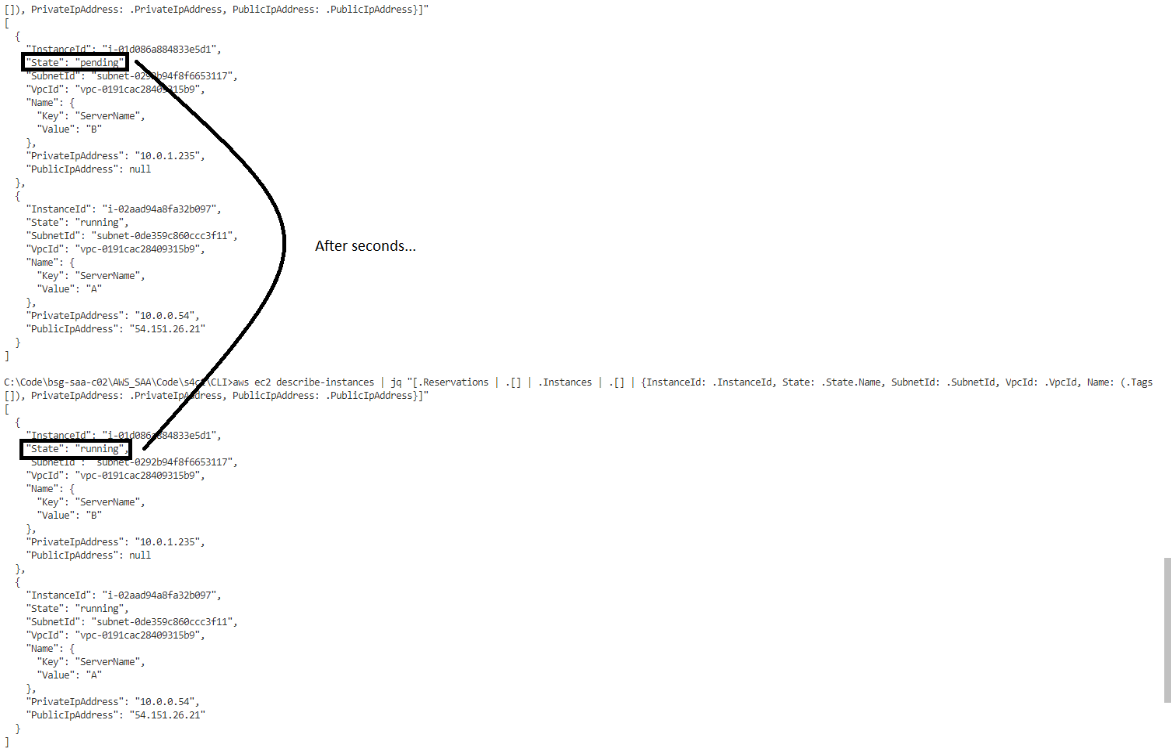
```
aws ec2 run-instances --image-id %AMI% --count 1 --instance-type t2.micro --key-name Lab4b --security-group-ids %SSH_Sec_Group_n_Id% --subnet-id %pbsn1_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=ServerName,Value=A}]"
aws ec2 run-instances --image-id %AMI% --count 1 --instance-type t2.micro --key-name Lab4b --security-group-ids %SSH_Sec_Group_p_Id% --subnet-id %prsn2_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=ServerName,Value=B}]"
```



```
:\Code\bsg-saa-c02\AWS_SAA>aws ec2 describe-images --owners amazon --filters "Name=name,Values=amzn2-ami-hvm-2.0.????????.?-x86_64-gp2" "Name=state,Values=available" --query "reverse(sort_by(Images, &CreationDa
:e))[:1].ImageId" --output text >tmpFile

:\Code\bsg-saa-c02\AWS_SAA>set /p AMI= < tmpFile

:\Code\bsg-saa-c02\AWS_SAA>aws ec2 run-instances --image-id %AMI% --count 1 --instance-type t2.micro --key-name Lab4b --security-group-ids %SSH_Sec_Group_n_Id% --subnet-id %pbsn1_Id% --tag-specifications "Resou
rceType=instance,Tags=[{Key=ServerName,Value=A}]"
{
    "Groups": [],
    "Instances": [
        {
            "AmiLaunchIndex": 0,
            "ImageId": "ami-04e59c05167ea7bd5",
            "InstanceId": "i-076b9986252cbed68",
            "InstanceType": "t2.micro",
            "KeyName": "Lab4b",
            "LaunchTime": "2020-06-21T12:20:58+00:00",
            "Monitoring": {
                "State": "disabled"
            },
            "Placement": {
                "AvailabilityZone": "us-west-1a",
                "GroupName": "",
                "Tenancy": "default"
            },
            "PrivateDnsName": "ip-10-0-0-10.us-west-1.compute.internal",
            "PrivateIpAddress": "10.0.0.10",
            "ProductCodes": [],
            "PublicDnsName": "",
            "State": {
                "Code": 0,
                "Name": "pending"
            }
```

Get Information about Instances

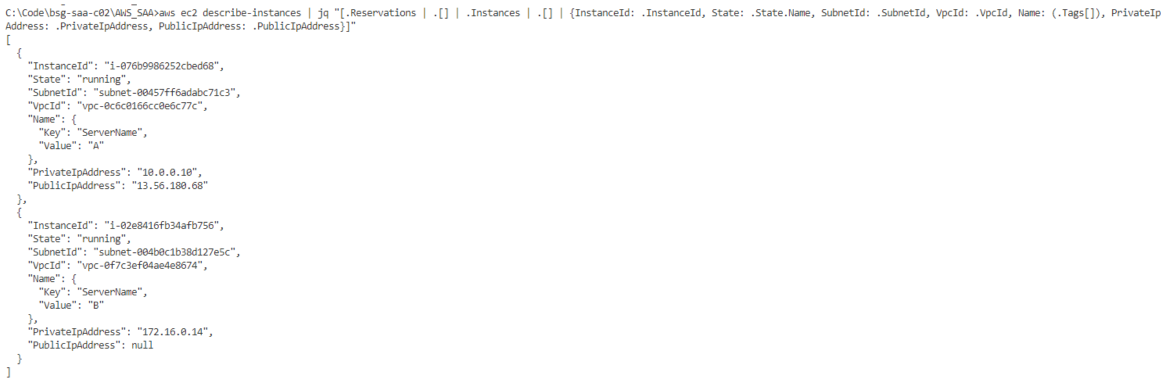
```
rem Traer estados de la Instancias
aws ec2 describe-instances | jq "[.Reservations | .[] | .Instances | .[] | {InstanceId: .InstanceId, State: .State.Name, SubnetId: .SubnetId, VpcId: .VpcId, Name: (.Tags[]), PrivateIpAddress: .PrivateIpAddress, PublicIpAddress: .PublicIpAddress}]"
```



```
[]), PrivateIpAddress: .PrivateIpAddress, PublicIpAddress: .PublicIpAddress}]"
[
  {
    "InstanceId": "i-01d086a884833e5d1",
    "State": "pending"
    "SubnetId": "subnet-0292b94f8f6653117",
    "VpcId": "vpc-0191cac28409315b9",
    "Name": {
      "Key": "ServerName",
      "Value": "B"
    },
    "PrivateIpAddress": "10.0.1.235",
    "PublicIpAddress": null
  },
  {
    "InstanceId": "i-02aad94a8fa32b097",
    "State": "running",
    "SubnetId": "subnet-0de359c860ccc3f11",
    "VpcId": "vpc-0191cac28409315b9",
    "Name": {
      "Key": "ServerName",
      "Value": "A"
    },
    "PrivateIpAddress": "10.0.0.54",
    "PublicIpAddress": "54.151.26.21"
  }
]

C:\Code\bsg-saa-c02\AWS_SAA\Code\s4c[illegible]\CLI>aws ec2 describe-instances | jq "[.Reservations | .[] | .Instances | .[] | {InstanceId: .InstanceId, State: .State.Name, SubnetId: .SubnetId, VpcId: .VpcId, Name: (.Tags
[]), PrivateIpAddress: .PrivateIpAddress, PublicIpAddress: .PublicIpAddress}]"
[
  {
    "InstanceId": "i-01d086a884833e5d1",
    "State": "running",
    "SubnetId": "subnet-0292b94f8f6653117",
    "VpcId": "vpc-0191cac28409315b9",
    "Name": {
      "Key": "ServerName",
      "Value": "B"
    },
    "PrivateIpAddress": "10.0.1.235",
    "PublicIpAddress": null
  },
  {
    "InstanceId": "i-02aad94a8fa32b097",
    "State": "running",
    "SubnetId": "subnet-0de359c860ccc3f11",
    "VpcId": "vpc-0191cac28409315b9",
    "Name": {
      "Key": "ServerName",
      "Value": "A"
    },
    "PrivateIpAddress": "10.0.0.54",
    "PublicIpAddress": "54.151.26.21"
  }
]
```

After seconds...

```
rem Traer Datos especificos de instancia A. Revisar contenido de Read_A.jq
aws ec2 describe-instances | jq -f Read_A.jq
aws ec2 describe-instances | jq -
f Read_A.jq|jq ".[].PublicIpAddress" >tmpFile
set /p A_IP= < tmpFile
```

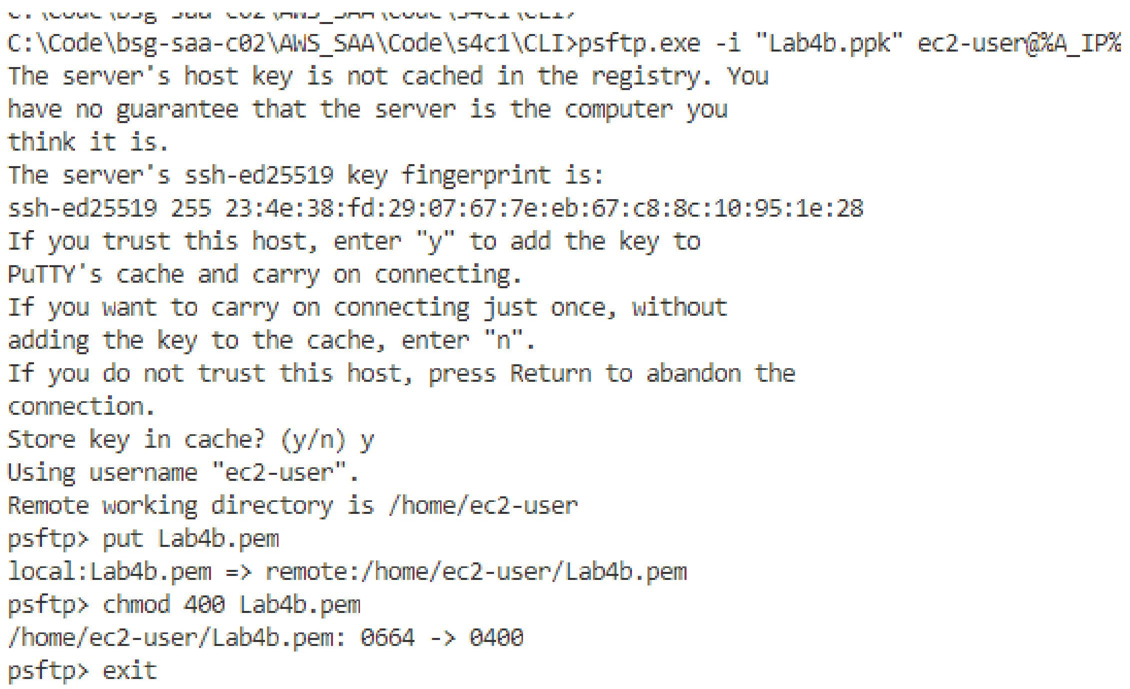


```
C:\Code\bsg-saa-c02\AWS_SAA>aws ec2 describe-instances | jq "[.Reservations | .[] | .Instances | .[] | {InstanceId: .InstanceId, State: .State.Name, SubnetId: .SubnetId, VpcId: .VpcId, Name: (.Tags[]), PrivateIp
Address: .PrivateIpAddress, PublicIpAddress: .PublicIpAddress}]"
[
  {
    "InstanceId": "i-076b9986252cbed68",
    "State": "running",
    "SubnetId": "subnet-00457ff6adabc71c3",
    "VpcId": "vpc-0c6c0166cc0e6c77c",
    "Name": {
      "Key": "ServerName",
      "Value": "A"
    },
    "PrivateIpAddress": "10.0.0.10",
    "PublicIpAddress": "13.56.180.68"
  },
  {
    "InstanceId": "i-02e8416fb34afb756",
    "State": "running",
    "SubnetId": "subnet-004b0c1b38d127e5c",
    "VpcId": "vpc-0f7c3ef04ae4e8674",
    "Name": {
      "Key": "ServerName",
      "Value": "B"
    },
    "PrivateIpAddress": "172.16.0.14",
    "PublicIpAddress": null
  }
]
```

Review Configurations using Putty, SFTP and Curl

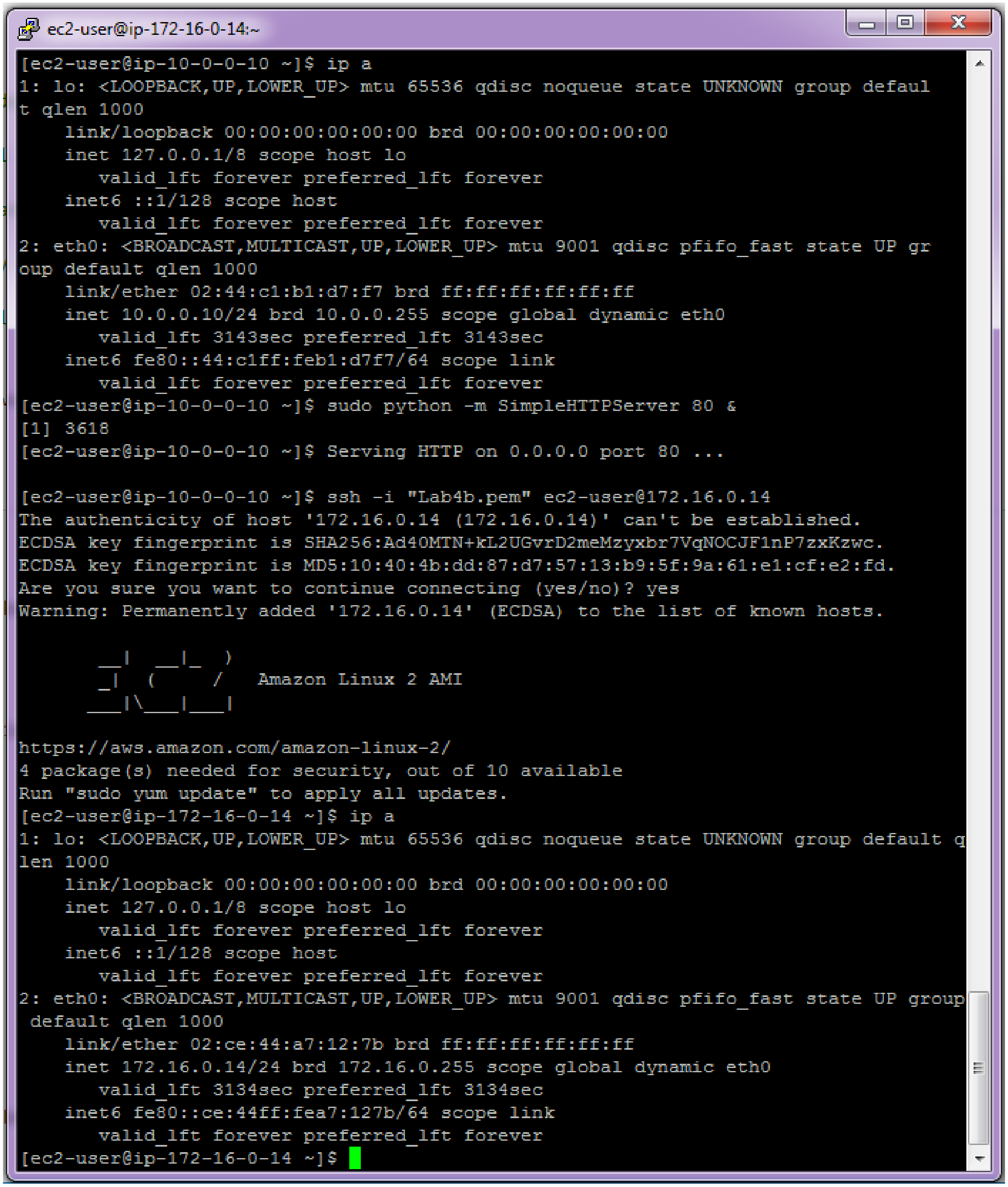
```
rem Enviar la llave a la Instancia Publica para luego desde alli conectarse
a la IP Privada
```

```
psftp.exe -i "Lab4b.ppk" ec2-user@%A_IP%
rem Luego alli enviar el codigo para subir el certificado y salir
put Lab4b.pem
chmod 400 Lab4b.pem
exit
```



```
C:\Code\bsg-saa-c02\AWS_SAA\Code\s4c1\CLI>
C:\Code\bsg-saa-c02\AWS_SAA\Code\s4c1\CLI>psftp.exe -i "Lab4b.ppk" ec2-user@%A_IP%
The server's host key is not cached in the registry. You
have no guarantee that the server is the computer you
think it is.
The server's ssh-ed25519 key fingerprint is:
ssh-ed25519 255 23:4e:38:fd:29:07:67:7e:eb:67:c8:8c:10:95:1e:28
If you trust this host, enter "y" to add the key to
PuTTY's cache and carry on connecting.
If you want to carry on connecting just once, without
adding the key to the cache, enter "n".
If you do not trust this host, press Return to abandon the
connection.
Store key in cache? (y/n) y
Using username "ec2-user".
Remote working directory is /home/ec2-user
psftp> put Lab4b.pem
local:Lab4b.pem => remote:/home/ec2-user/Lab4b.pem
psftp> chmod 400 Lab4b.pem
/home/ec2-user/Lab4b.pem: 0664 -> 0400
psftp> exit
```

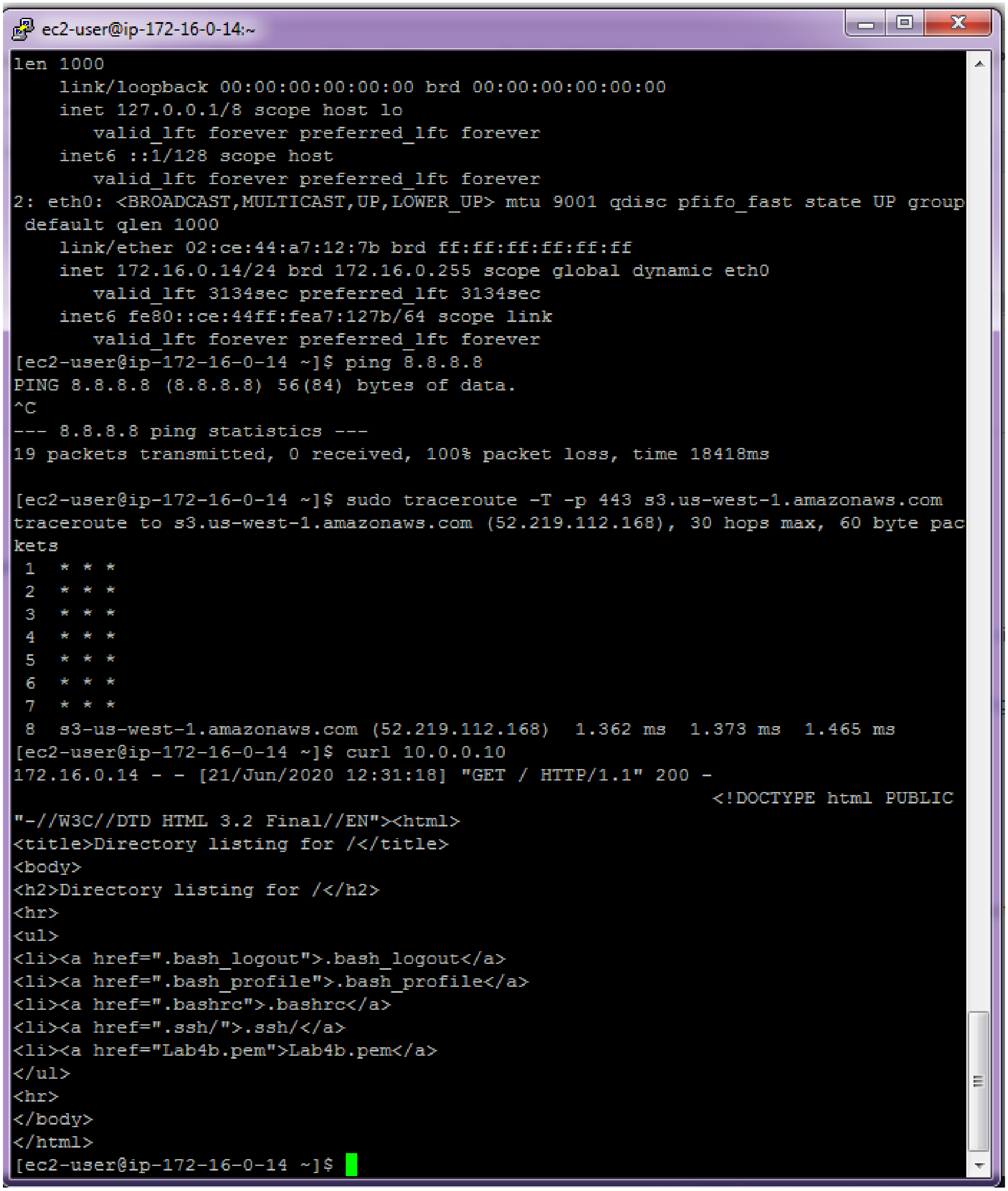
```
rem Ingresar a la instancia publica por SSH y dejar ejecutando en el SSH  "s
udo python -m SimpleHTTPServer 80"
putty.exe -i "Lab4b.ppk" ec2-user@%A_IP%
rem Mirar la configuracion de la maquina actual
ip a
rem Ejecutar para dejar un servidor web ejecutándose para Python 2
sudo python -m SimpleHTTPServer 80 &
rem Dentro de la instancia ejecutar para Python 3
sudo python3 -m http.server 80

rem Conectarse por SSH a la Instancia Privada y desde alli escribir la IP de
 la instancia privada
ssh -i "Lab4b.pem" ec2-user@172.16.0.14
rem Mirar la configuracion de la maquina actual y revisar conectividad
ip a
ping 8.8.8.8
sudo traceroute -T -p 443 s3.us-west-1.amazonaws.com
sudo traceroute -T -p 443 eltiempo.com
```



```
[ec2-user@ip-10-0-0-10 ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP gr
oup default qlen 1000
    link/ether 02:44:c1:b1:d7:f7 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.10/24 brd 10.0.0.255 scope global dynamic eth0
       valid_lft 3143sec preferred_lft 3143sec
    inet6 fe80::44:c1ff:feb1:d7f7/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@ip-10-0-0-10 ~]$ sudo python -m SimpleHTTPServer 80 &
[1] 3618
[ec2-user@ip-10-0-0-10 ~]$ Serving HTTP on 0.0.0.0 port 80 ...

[ec2-user@ip-10-0-0-10 ~]$ ssh -i "Lab4b.pem" ec2-user@172.16.0.14
The authenticity of host '172.16.0.14 (172.16.0.14)' can't be established.
ECDSA key fingerprint is SHA256:Ad40MTN+kL2UGvrD2meMzyxbr7VqNOCJF1nP7zxKzwc.
ECDSA key fingerprint is MD5:10:40:4b:dd:87:d7:57:13:b9:5f:9a:61:e1:cf:e2:fd.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '172.16.0.14' (ECDSA) to the list of known hosts.

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

https://aws.amazon.com/amazon-linux-2/
4 package(s) needed for security, out of 10 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-16-0-14 ~]$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default q
len 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group
 default qlen 1000
    link/ether 02:ce:44:a7:12:7b brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.14/24 brd 172.16.0.255 scope global dynamic eth0
       valid_lft 3134sec preferred_lft 3134sec
    inet6 fe80::ce:44ff:fea7:127b/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@ip-172-16-0-14 ~]$
```

```
rem Mirar la configuracion de la maquina actual y revisar conectividad
ip a
ping 8.8.8.8
sudo traceroute -T -p 443 s3.us-west-1.amazonaws.com
rem Verificar acceso a la IP Privada de la Instancia Publica.
```

```
curl 10.0.0.10
```



```
len 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP group
 default qlen 1000
    link/ether 02:ce:44:a7:12:7b brd ff:ff:ff:ff:ff:ff
    inet 172.16.0.14/24 brd 172.16.0.255 scope global dynamic eth0
       valid_lft 3134sec preferred_lft 3134sec
    inet6 fe80::ce:44ff:fea7:127b/64 scope link
       valid_lft forever preferred_lft forever
[ec2-user@ip-172-16-0-14 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
19 packets transmitted, 0 received, 100% packet loss, time 18418ms

[ec2-user@ip-172-16-0-14 ~]$ sudo traceroute -T -p 443 s3.us-west-1.amazonaws.com
traceroute to s3.us-west-1.amazonaws.com (52.219.112.168), 30 hops max, 60 byte pac
kets
 1  * * *
 2  * * *
 3  * * *
 4  * * *
 5  * * *
 6  * * *
 7  * * *
 8  s3-us-west-1.amazonaws.com (52.219.112.168)  1.362 ms  1.373 ms  1.465 ms
[ec2-user@ip-172-16-0-14 ~]$ curl 10.0.0.10
172.16.0.14 - - [21/Jun/2020 12:31:18] "GET / HTTP/1.1" 200 -
                                                              <!DOCTYPE html PUBLIC
"-//W3C//DTD HTML 3.2 Final//EN"><html>
<title>Directory listing for /</title>
<body>
<h2>Directory listing for /</h2>
<hr>
<ul>
<li><a href=".bash_logout">.bash_logout</a>
<li><a href=".bash_profile">.bash_profile</a>
<li><a href=".bashrc">.bashrc</a>
<li><a href=".ssh/">.ssh/</a>
<li><a href="Lab4b.pem">Lab4b.pem</a>
</ul>
<hr>
</body>
</html>
[ec2-user@ip-172-16-0-14 ~]$
```

Clean resources

For Web Management Console

EC2: Terminate Instances

EC2: Security Groups

EC2: KeyPairs

VPC: IGW (Detach and then Delete)

VPC: Peering Connections

VPC: Endpoints

VPC: Subnets

VPC: RT

VPC: VPC

For Command Line (Windows)

```
rem ----- ELIMINAR RECURSOS ----
aws ec2 terminate-instances --instance-ids "i-02e8416fb34afb756" "i-076b9986252cbed68"
aws ec2 delete-vpc-peering-connection --vpc-peering-connection-id %VPCPeering_Id%
aws ec2 delete-vpc-endpoints --vpc-endpoint-ids %VPCEndpoint_Id%
aws ec2 delete-security-group --group-id %SSH_Sec_Group_p_Id%
aws ec2 delete-security-group --group-id %SSH_Sec_Group_n_Id%
aws ec2 delete-subnet --subnet-id %prsn2_Id%
aws ec2 delete-route-table --route-table-id %Private_RT_Id%

aws ec2 detach-internet-gateway --internet-gateway-id %IGW_Id% --vpc-id %vpcn_Id%
aws ec2 delete-internet-gateway --internet-gateway-id %IGW_Id%
aws ec2 delete-subnet --subnet-id %pbsn1_Id%
aws ec2 delete-route-table --route-table-id %Public_RT_Id%

aws ec2 delete-vpc --vpc-id %vpcp_Id%
aws ec2 delete-vpc --vpc-id %vpcn_Id%

aws ec2 delete-key-pair --key-name Lab4b
```

Evidences to send (Optional)

To have a review, the student has to send some screenshots to instructor email:

1. All images from Review Configurations using Putty, SFTP and Curl, because it show the copy of authorization key (pem), SSH connection to instance on intranet layer from public layer, simple HTTP server on public instances, S3 VPC endpoint connection.