Contents

[Purpose 2](#_Toc51733457)

[General Diagram 2](#_Toc51733458)

[Prerequisites 2](#_Toc51733459)

[Lab 4B: VPC with VPC Peering and VPC Endpoint 2](#_Toc51733460)

[Lab 4B using Web Management Console 3](#_Toc51733461)

[Create VPC, subnets, IGW 3](#_Toc51733462)

[Create and accept VPC Peering 3](#_Toc51733463)

[Create Custom Routing Tables and associate to subnets 5](#_Toc51733464)

[Create S3 VPC Endpoint 7](#_Toc51733465)

[Modifying routing tables to reach VPC Peering 10](#_Toc51733466)

[Create instances 12](#_Toc51733467)

[Lab 4B using Command Line (Windows) 14](#_Toc51733468)

[Create VPC, VPC Peering, Subnets, IGW and Route Table 14](#_Toc51733469)

[Create Public Routing Table with VPC Peering, IGW. Create Keypair and Public Security Group 15](#_Toc51733470)

[Create Private Sec Group, Private Routing Table and S3 VPC Endpoint for Routing Subnet 16](#_Toc51733471)

[Create Instances 17](#_Toc51733472)

[Get Information about Instances 18](#_Toc51733473)

[Review Configurations using Putty, SFTP and Curl 19](#_Toc51733474)

[Clean resources 22](#_Toc51733475)

[For Web Management Console 22](#_Toc51733476)

[For Command Line (Windows) 23](#_Toc51733477)

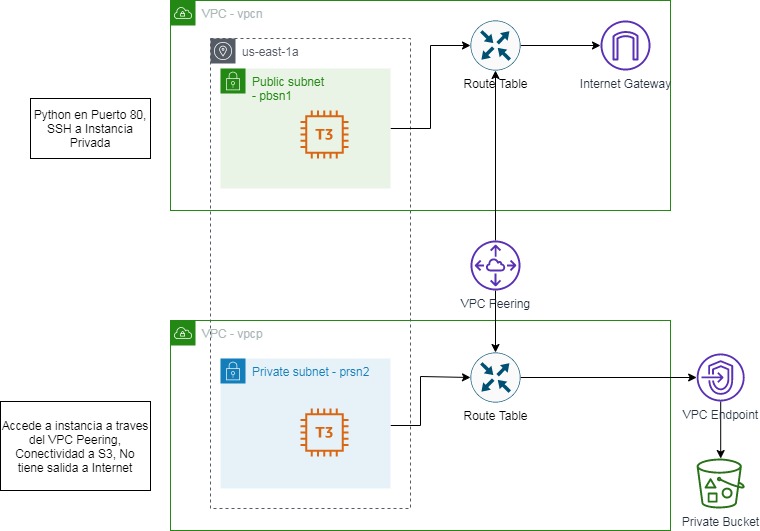
[Evidences to send 23](#_Toc51733478)

# 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.

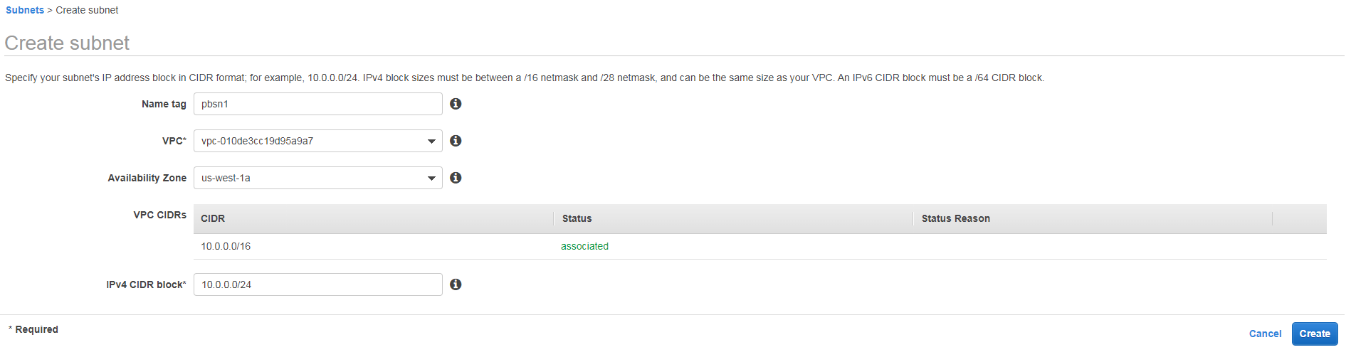
# Lab 4B: VPC with VPC Peering and VPC Endpoint

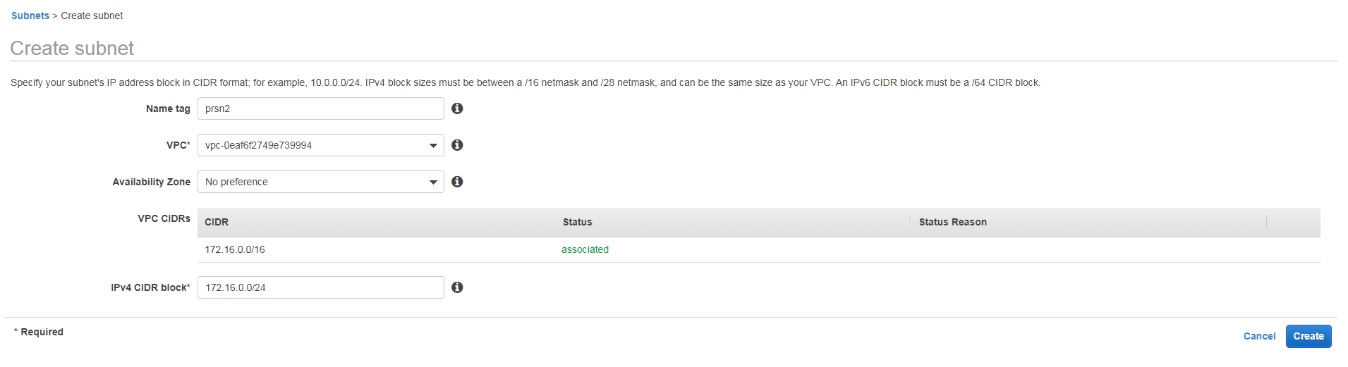
## Lab 4B using Web Management Console

### Create VPC, subnets, IGW

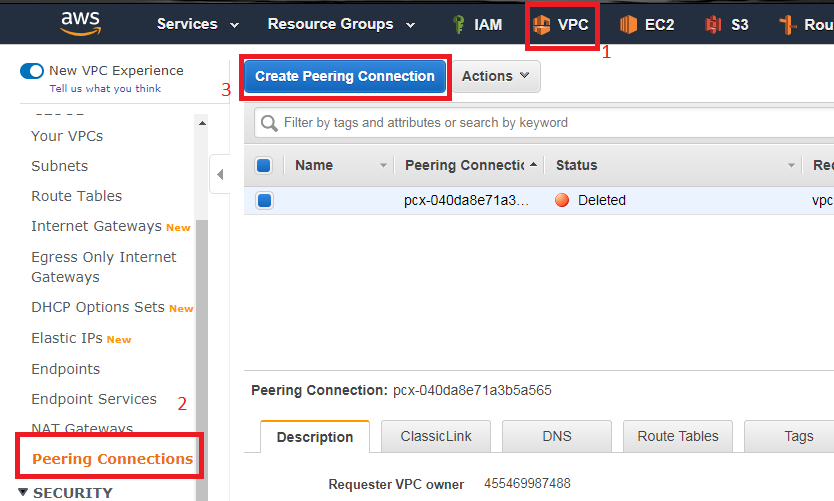
Prerequisite from previous Lab: Labs4c1. Some screenshoots.

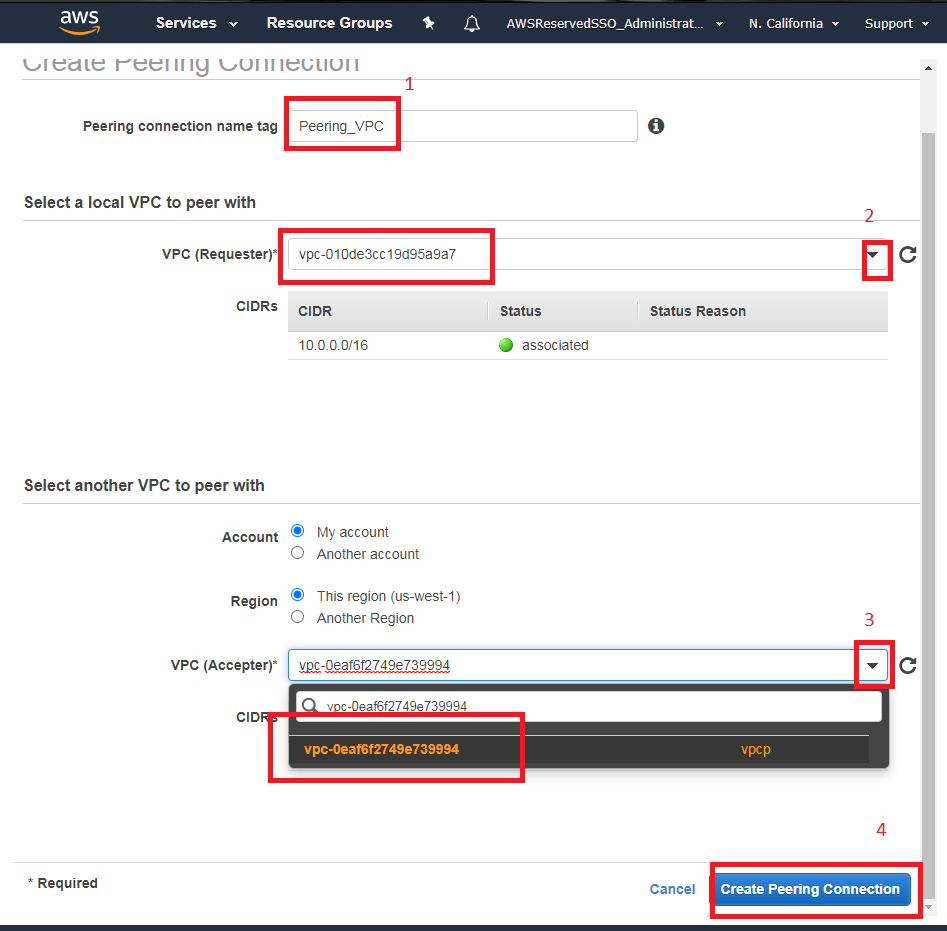
Remember to apply Public IP Mapping to Public Subnet.

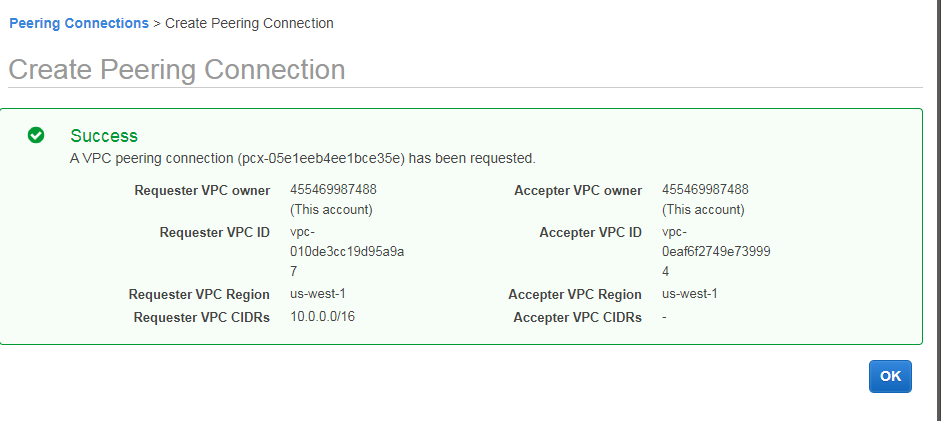


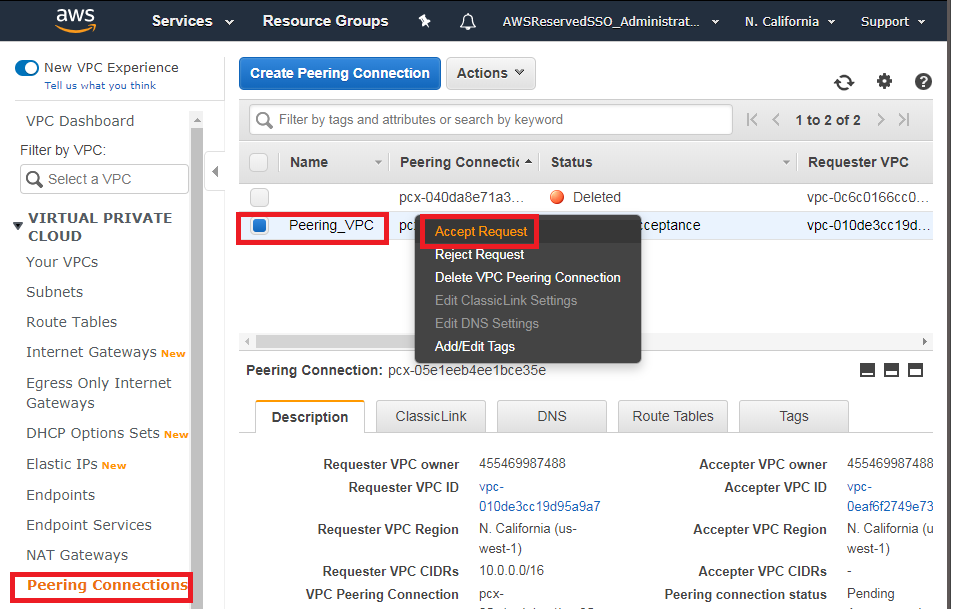


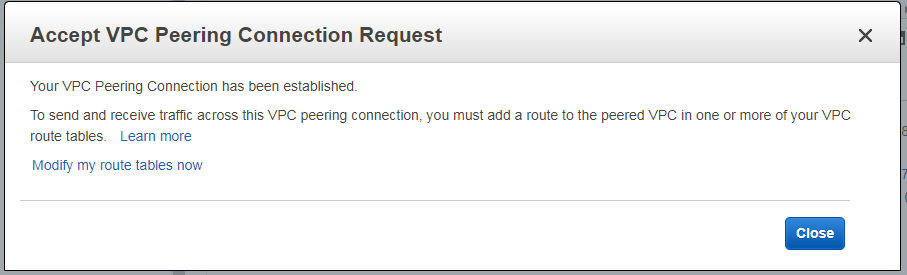
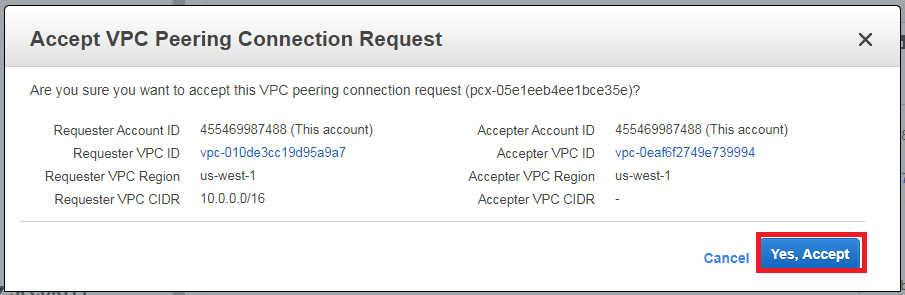
### Create and accept VPC Peering



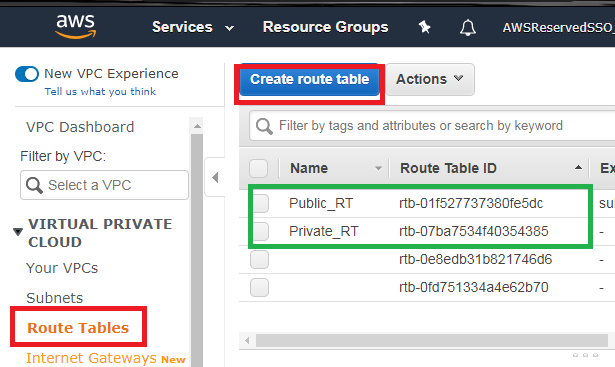


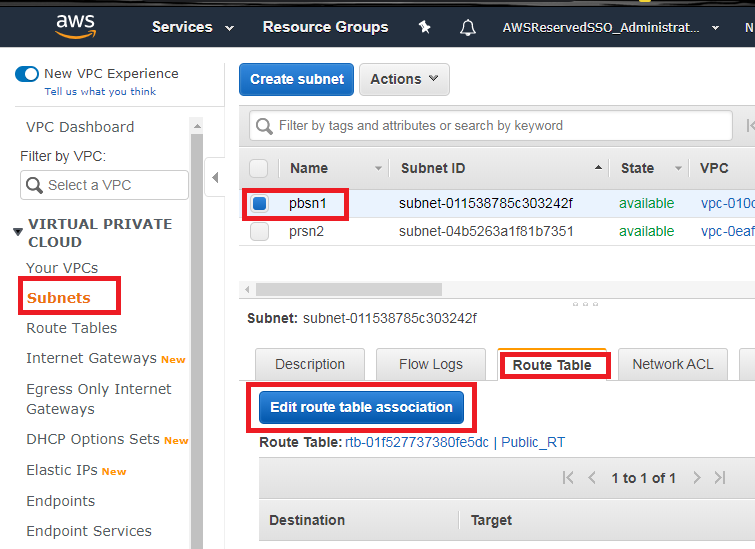


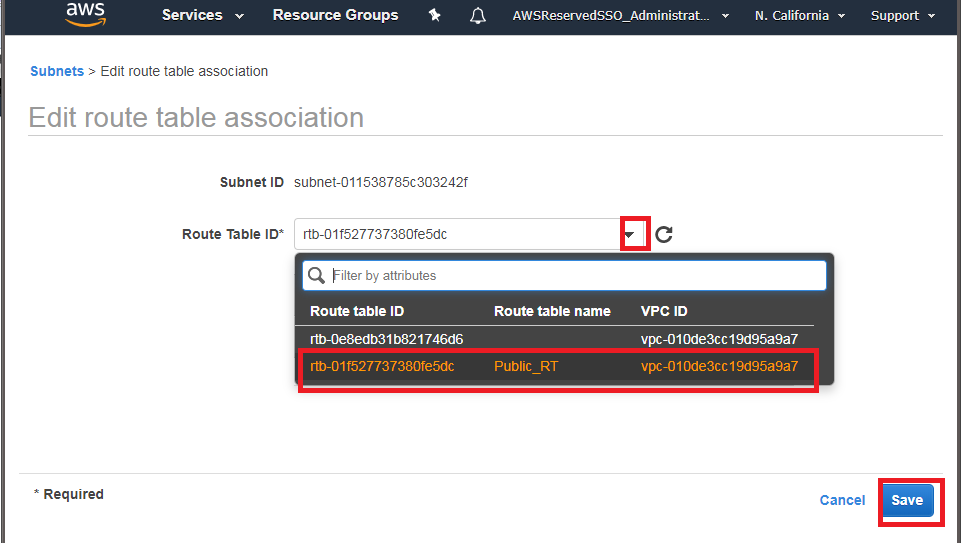




### Create Custom Routing Tables and associate to subnets

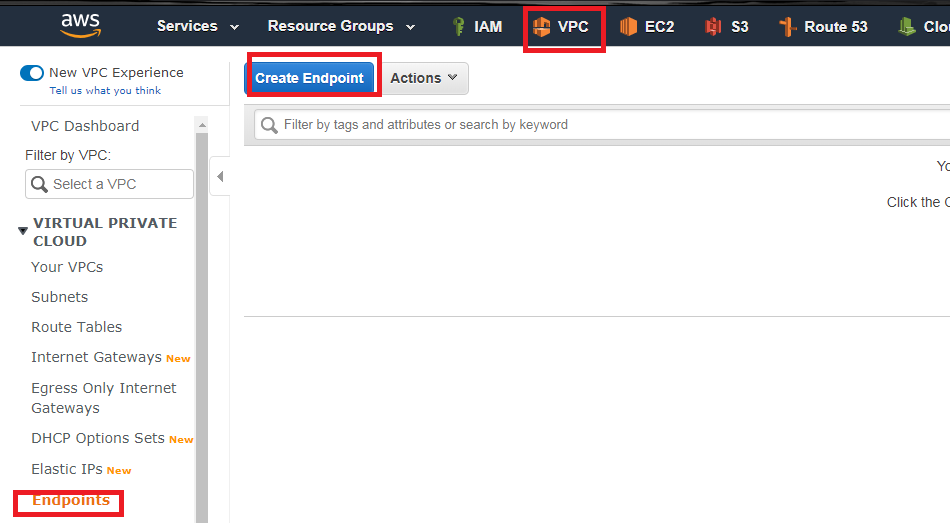


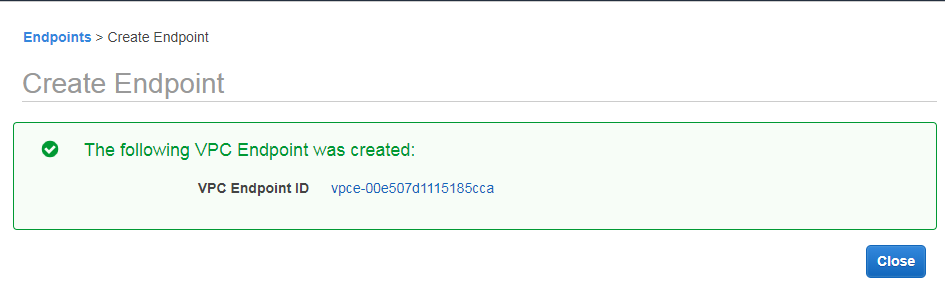
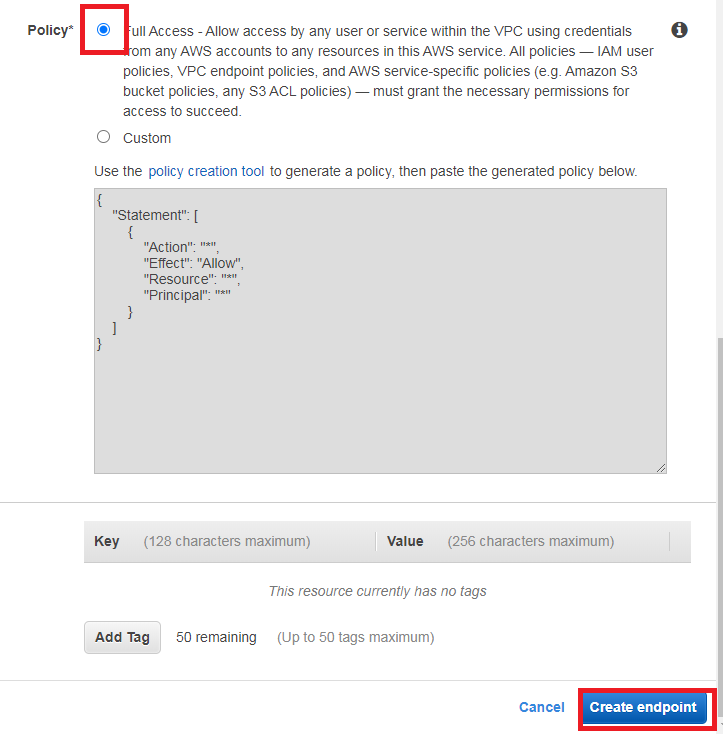
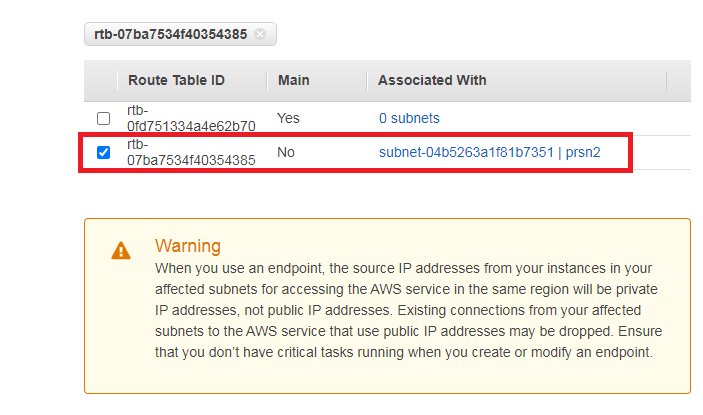
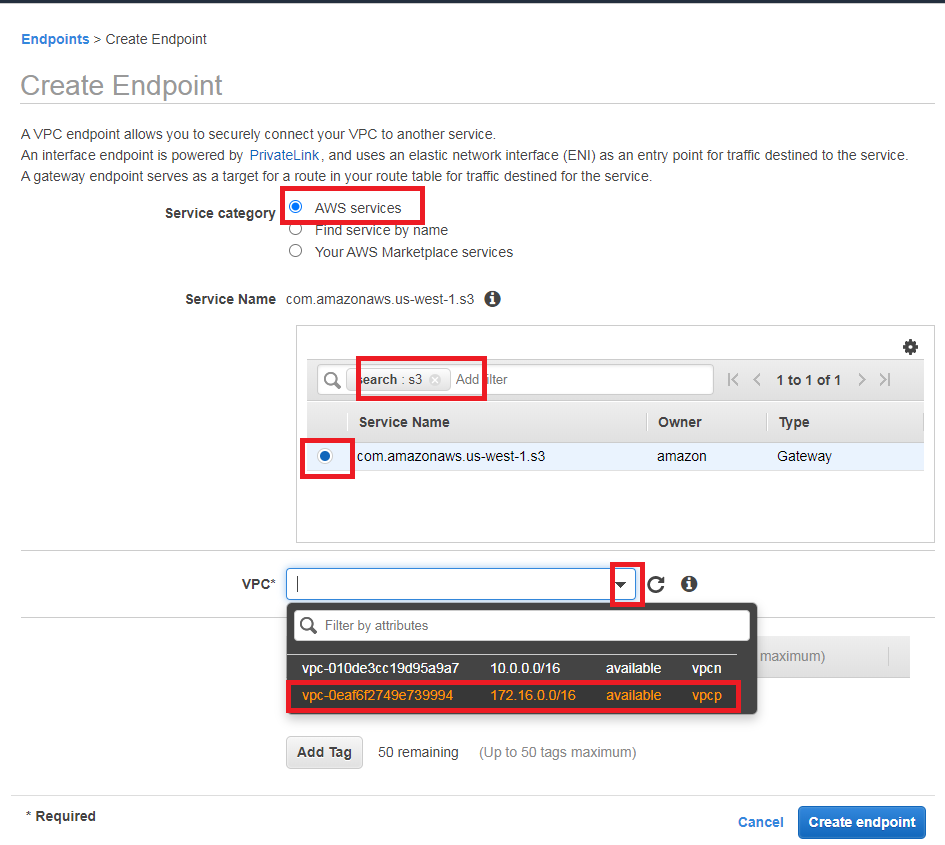




And the same case for the Private Subnet.

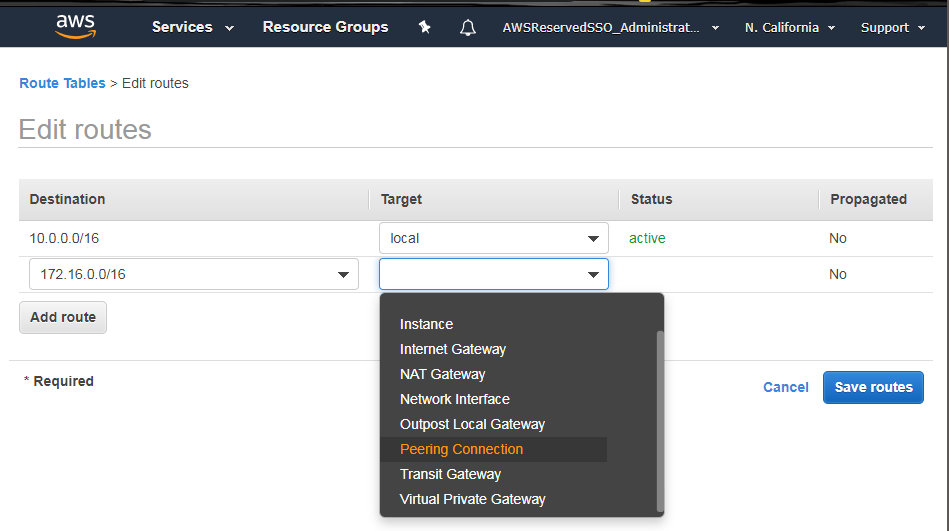
### Create S3 VPC Endpoint

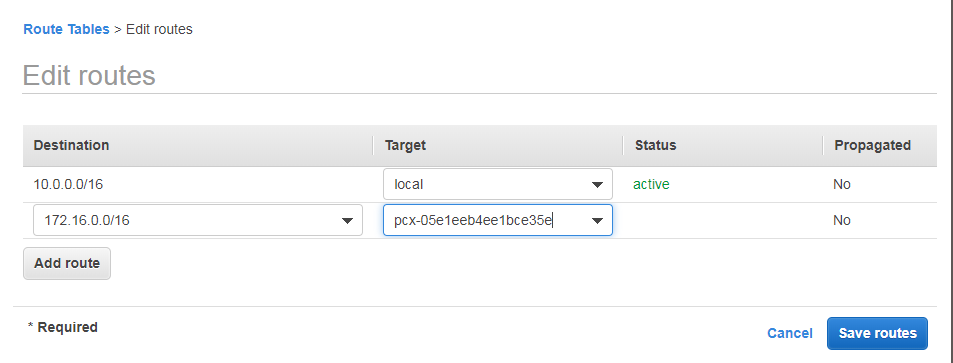




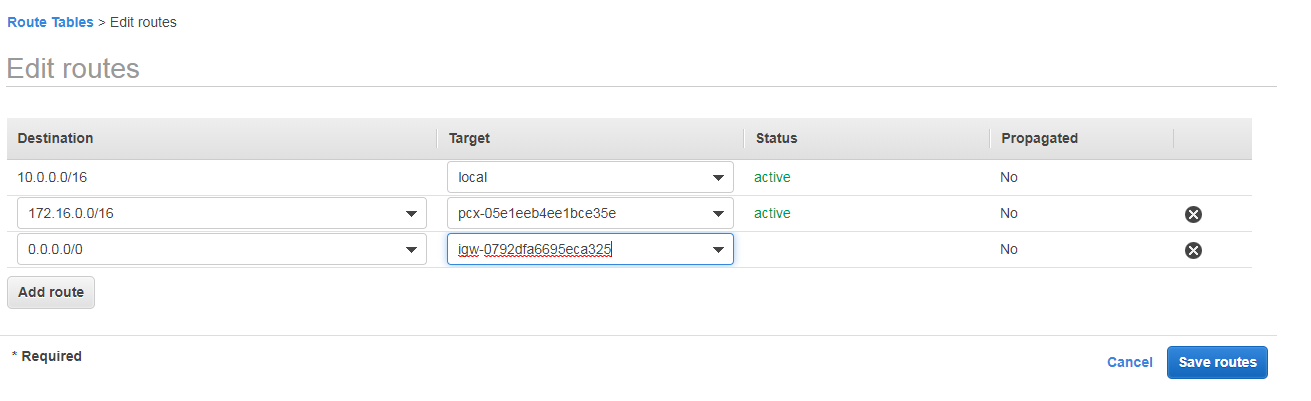
### Modifying routing tables to reach VPC Peering

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.

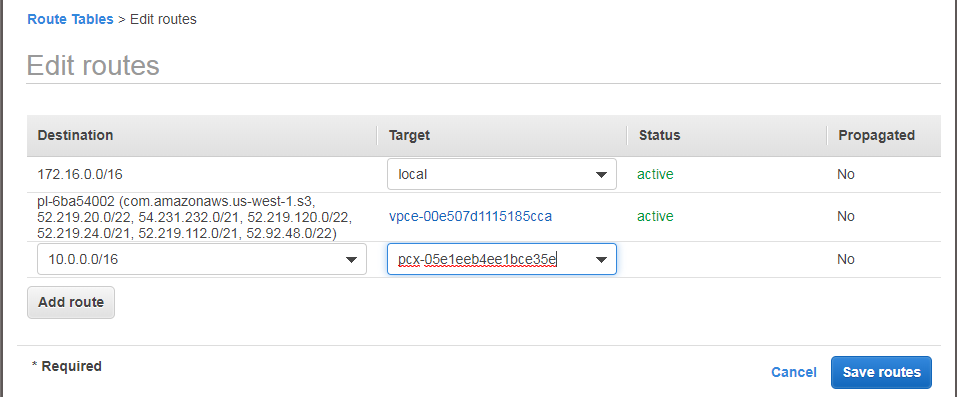




Add the IGW as default route on Public RT.

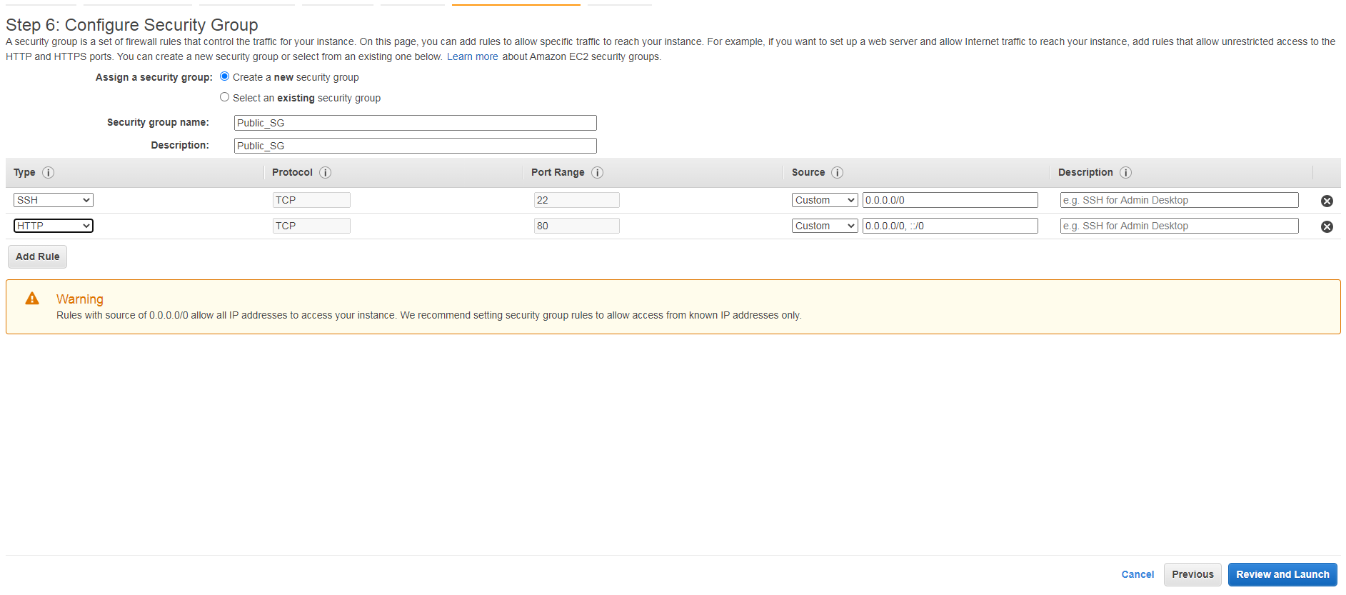


For the private RT is similar, however check vpcn range and the additional VPC Endpoint

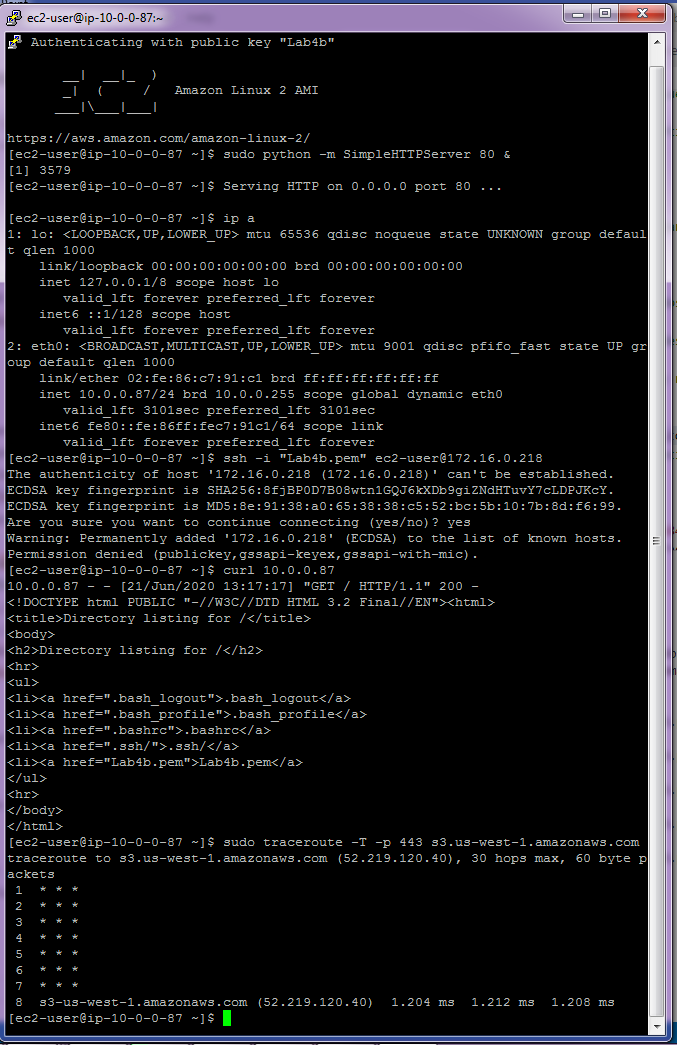


### Create instances

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



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



## 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

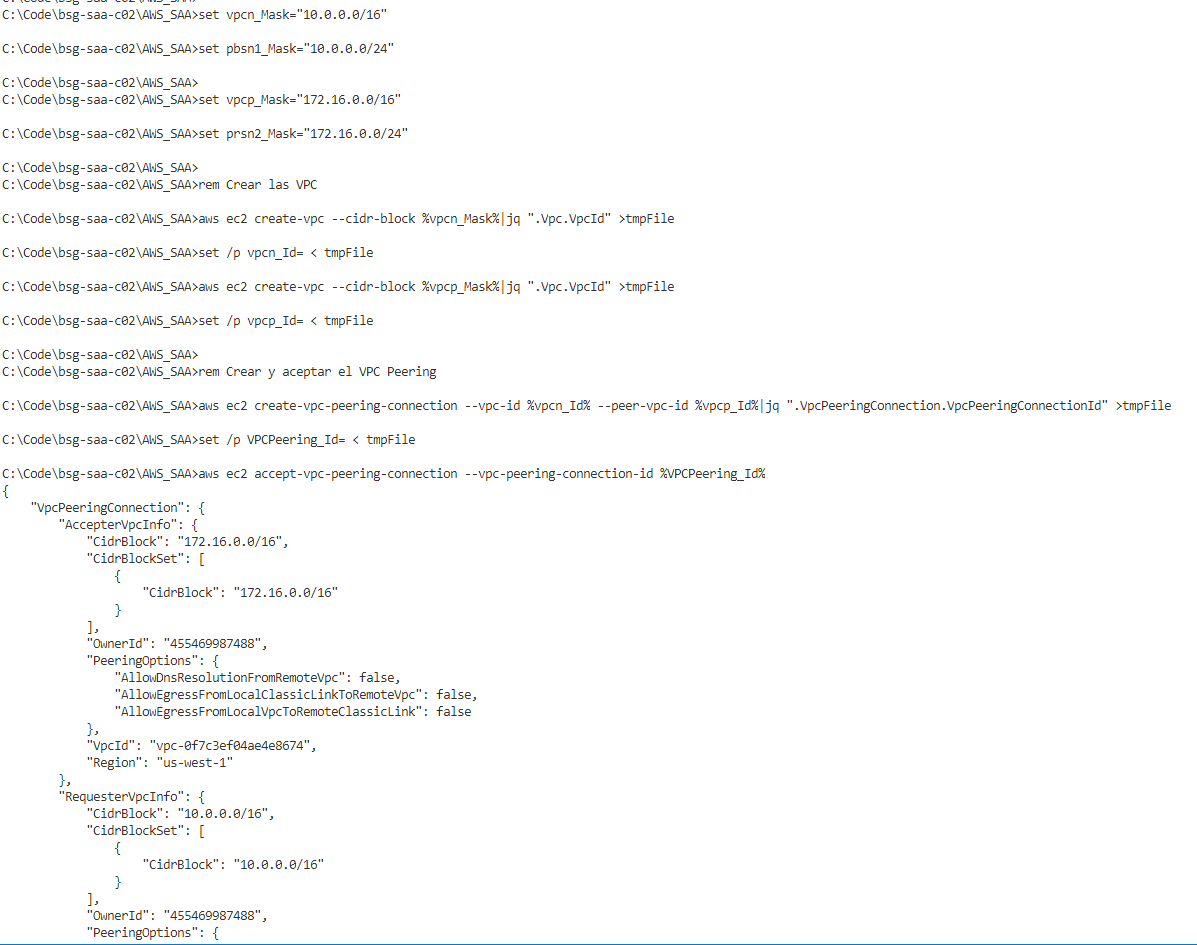
set /p prsn2\_Id= < tmpFile

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%



### 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 asignarle 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 publicas

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"

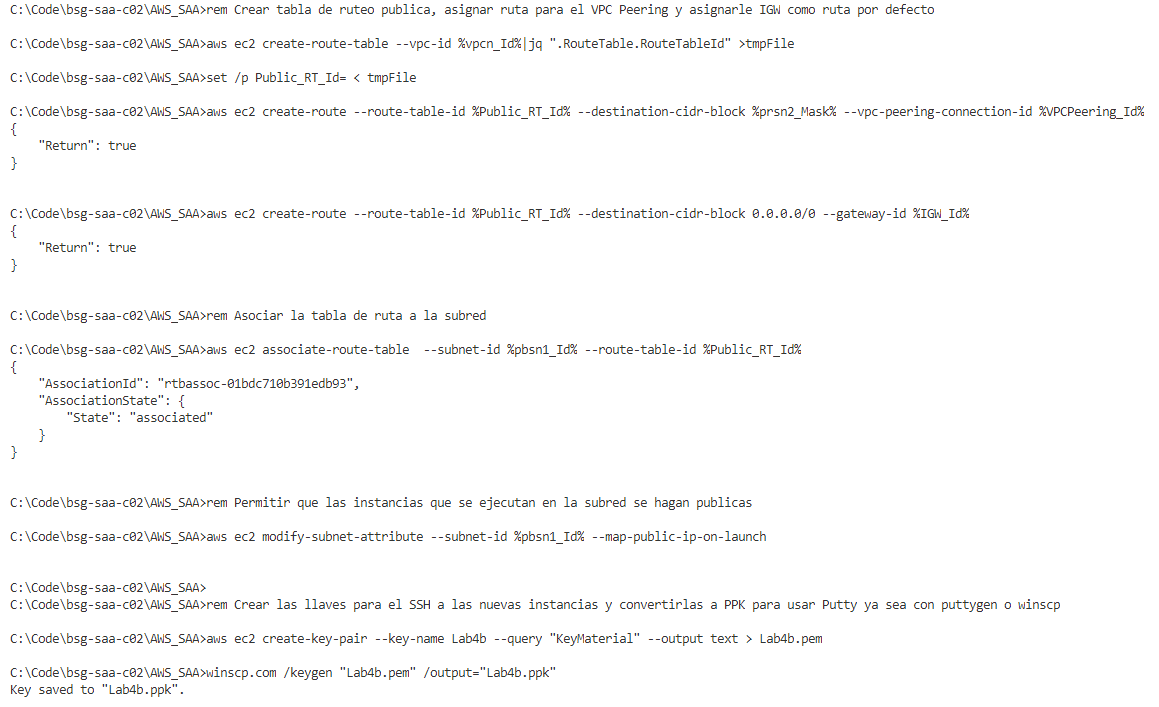
rem Crear los Security Groups para esas instancias

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



### 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 Peering 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

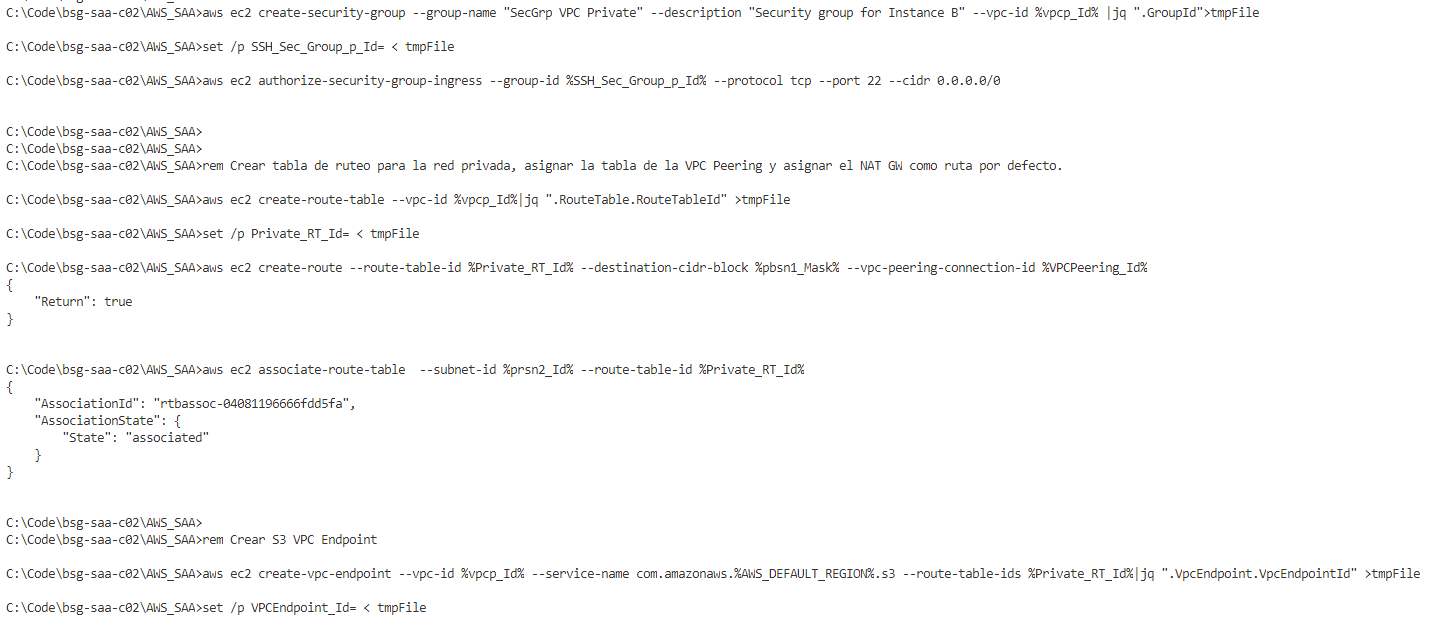
aws ec2 create-route --route-table-id %Private\_RT\_Id% --destination-cidr-block %pbsn1\_Mask% --vpc-peering-connection-id %VPCPeering\_Id%

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



### 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

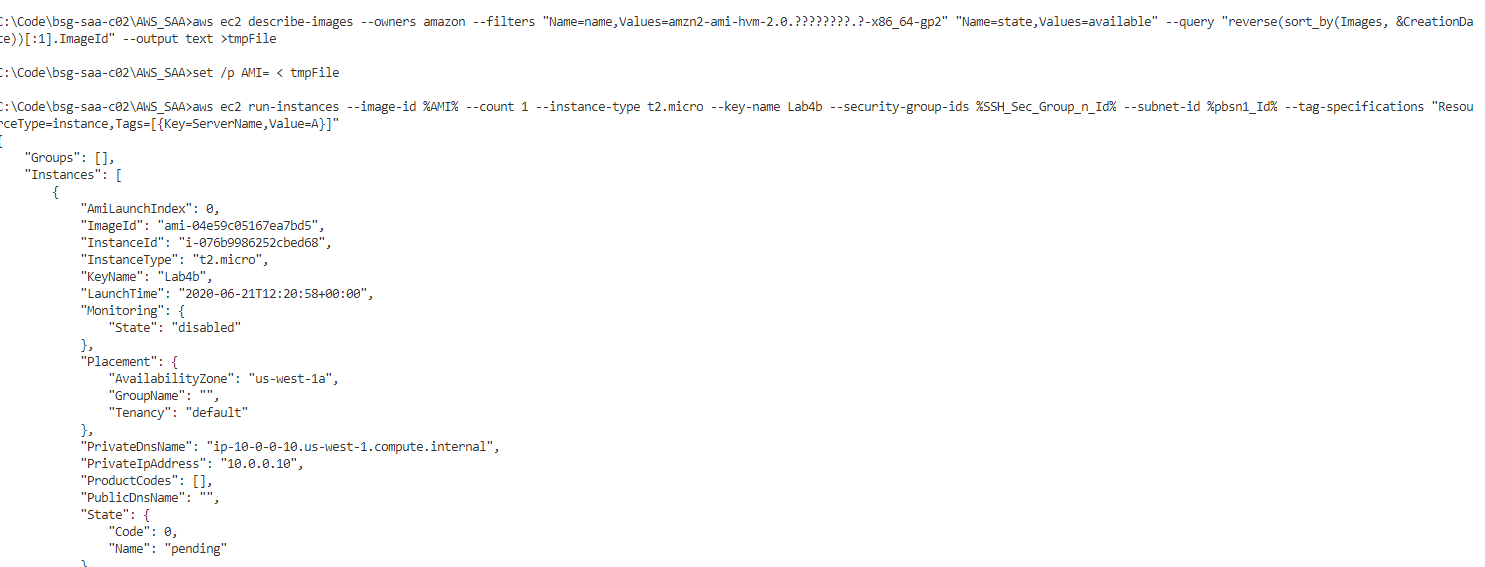
rem Puede utilizar Amazon Linux 2023 con Python 3

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}]"

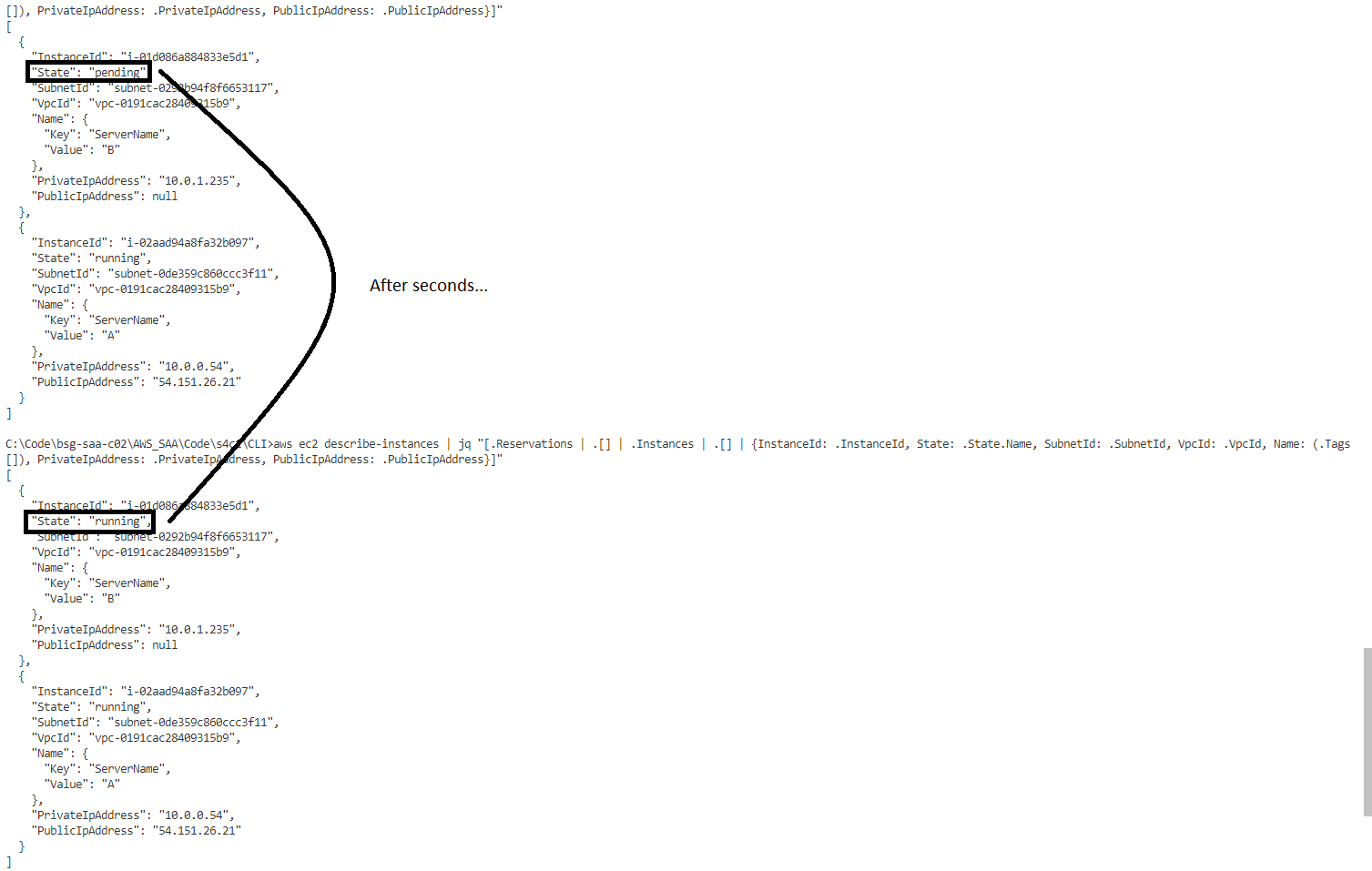
aws ec2 run-instances --image-id %AMI% --count 1 --instance-type t2.micro --key-name Lab4b --security-group-ids %SSH\_Sec\_Group\_p\_Id% --subnet-id %prsn2\_Id% --tag-specifications "ResourceType=instance,Tags=[{Key=ServerName,Value=B}]"



### 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}]"

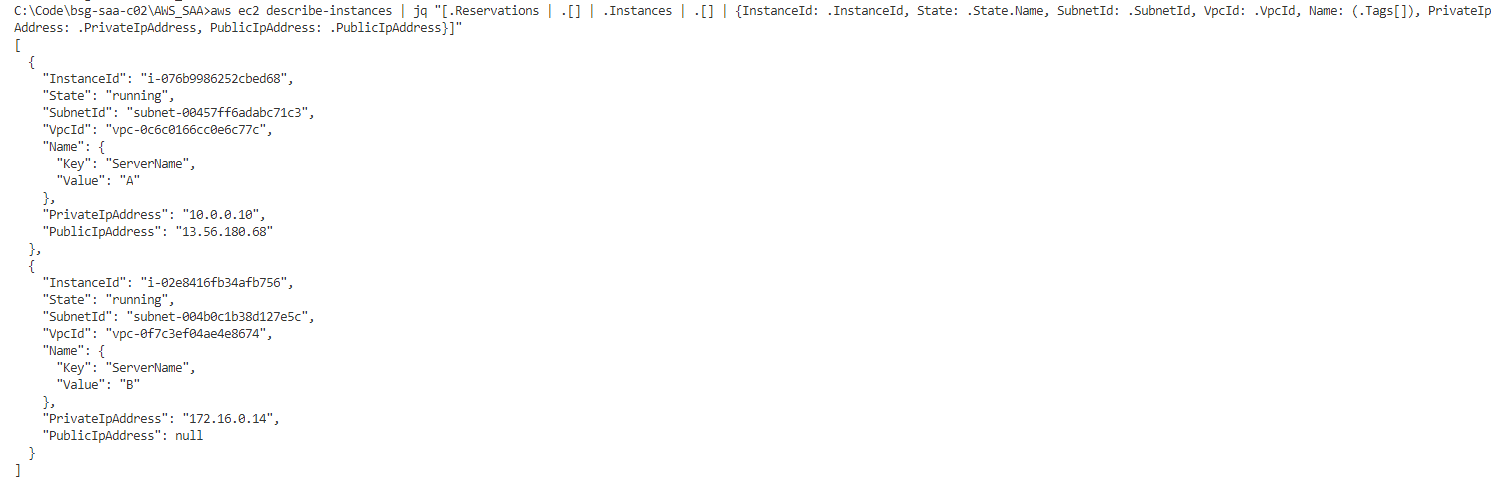


rem Traer Datos especificos de instancia A. Revisar contenido de Read\_A.jq

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



## Review Configurations using Putty, SFTP and Curl

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

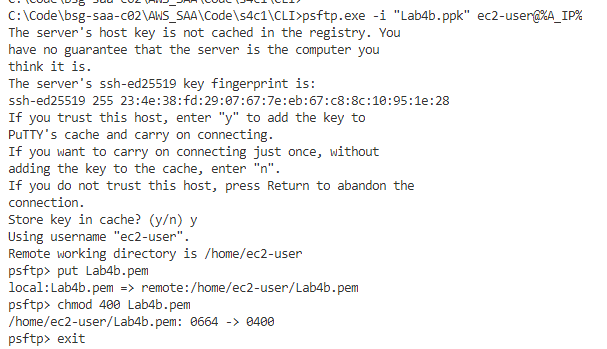
psftp.exe -i "Lab4b.ppk" ec2-user@%A\_IP%

rem Luego alli enviar el codigo para subir el certificado y salir

put Lab4b.pem

chmod 400 Lab4b.pem

exit



rem Ingresar a la instancia publica por SSH y dejar ejecutando en el SSH  "sudo 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

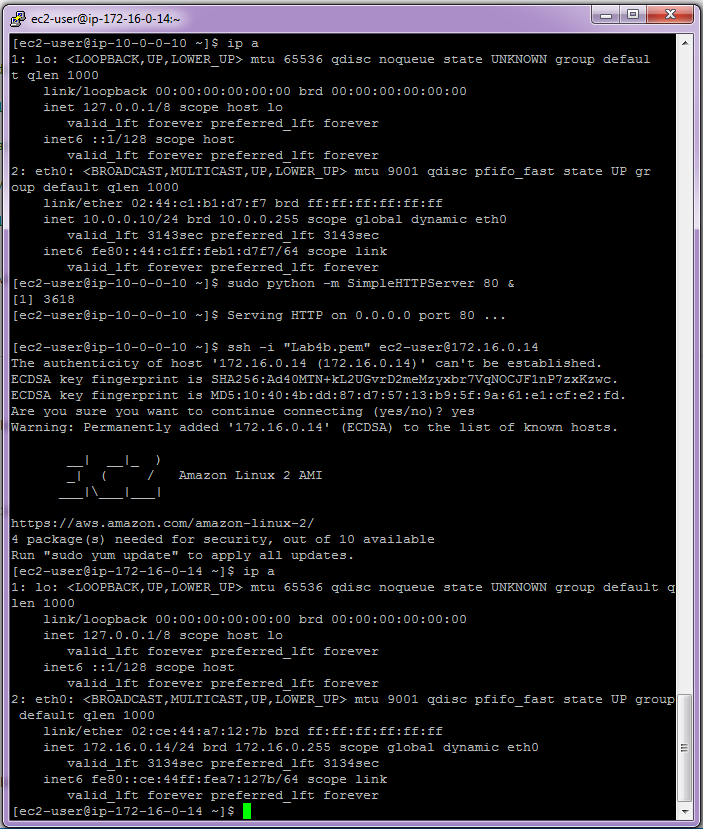
rem Mirar la configuracion de la maquina actual y revisar conectividad

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



rem Mirar la configuracion de la maquina actual y revisar conectividad

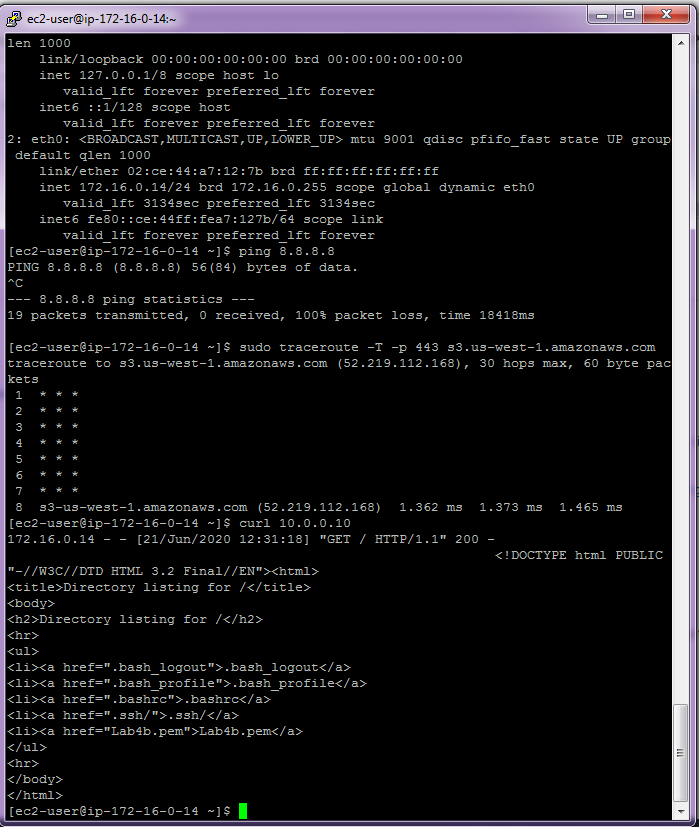
ip a

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



## 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](#_Review_Configurations_using), 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.