**GRUPO: 3 MESA: 2 e 4, Maria Aparecida, Samir, Vinicius Batista, Rodrigo Felix, Victor , Jose, Julio, Guilherme Rodrigues, Kauan, Davi Franco.**

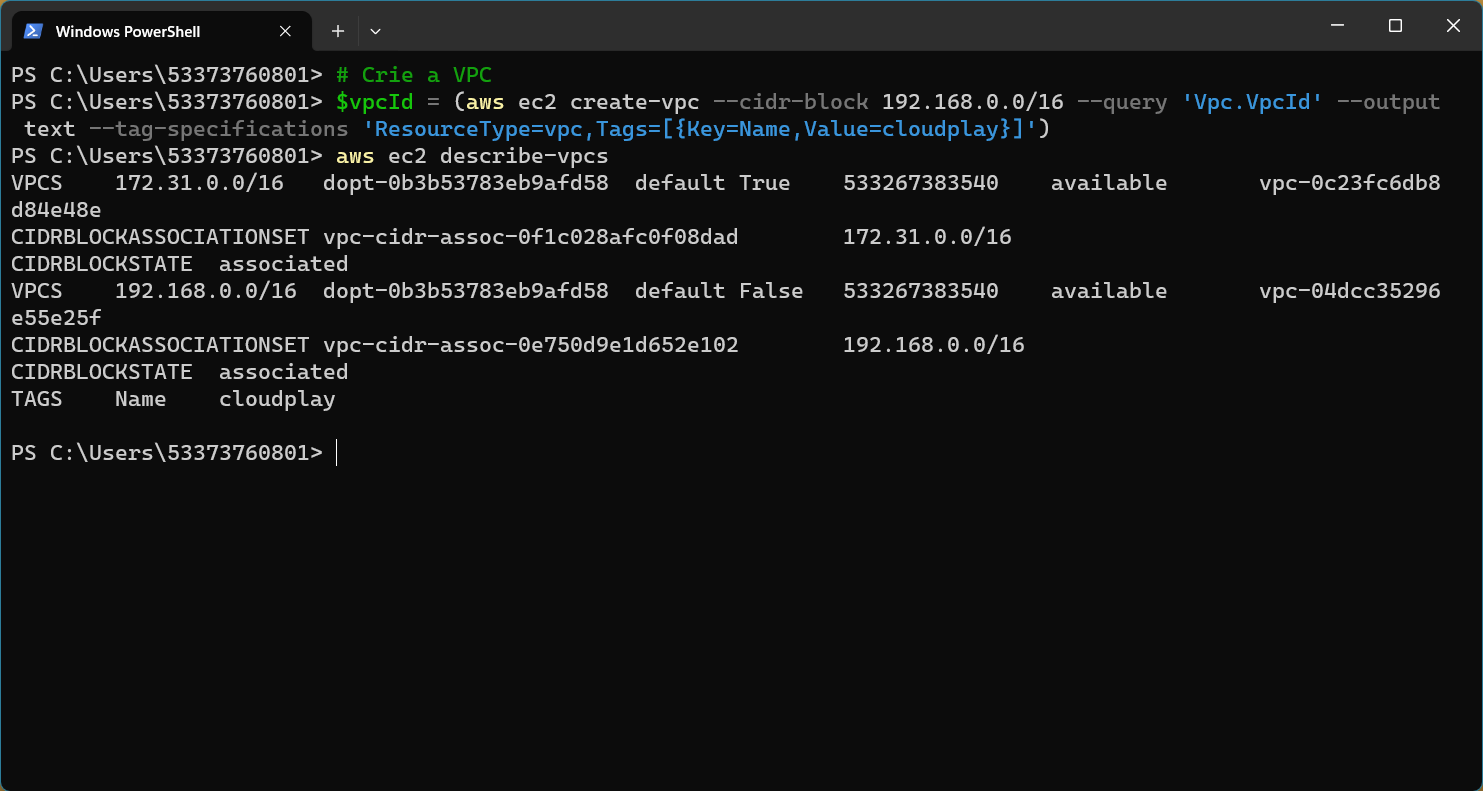
**PROF: Danilo, Fernando**

**Atividade:**

**VPC SCRIPT + PRINT:**

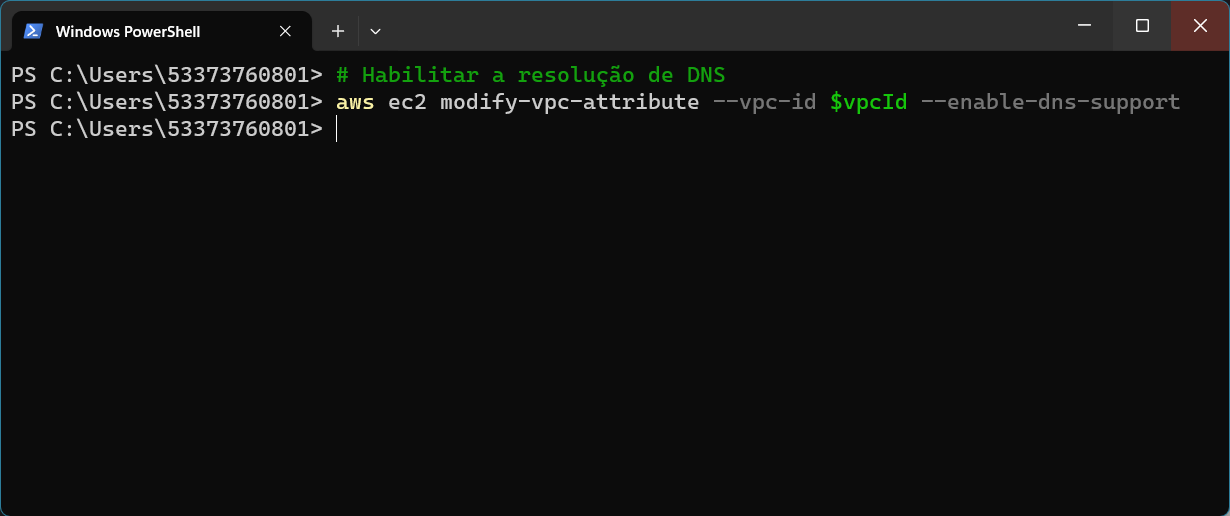
# Crie a VPC

$vpcId = (aws ec2 create-vpc --cidr-block 192.168.0.0/16 --query 'Vpc.VpcId' --output text --tag-specifications 'ResourceType=vpc,Tags=[{Key=Name,Value=cloudplay}]')

****

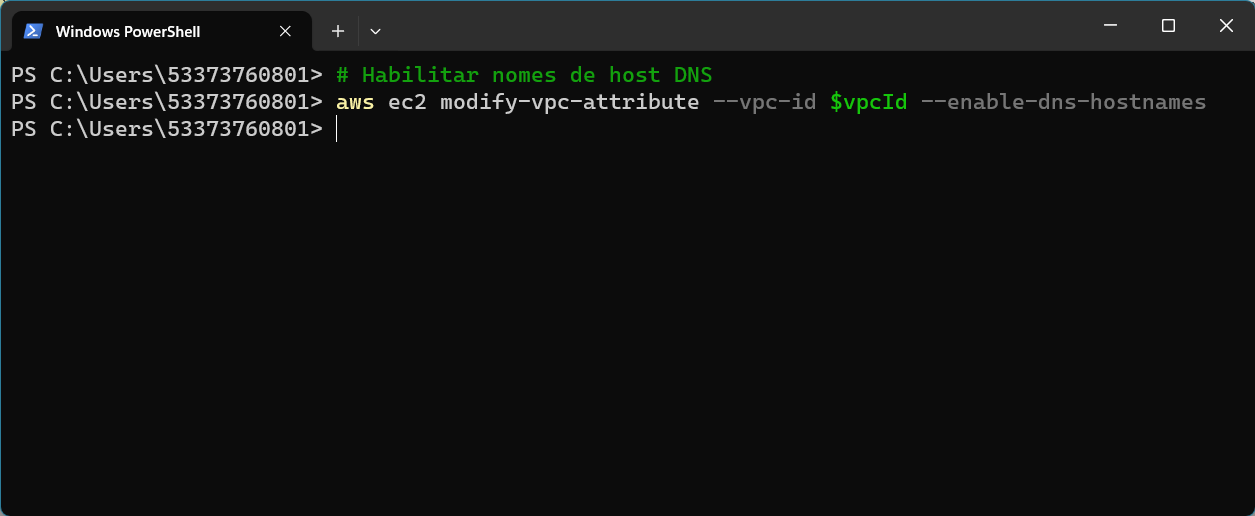
# Habilitar a resolução de DNS

aws ec2 modify-vpc-attribute --vpc-id $vpcId --enable-dns-support



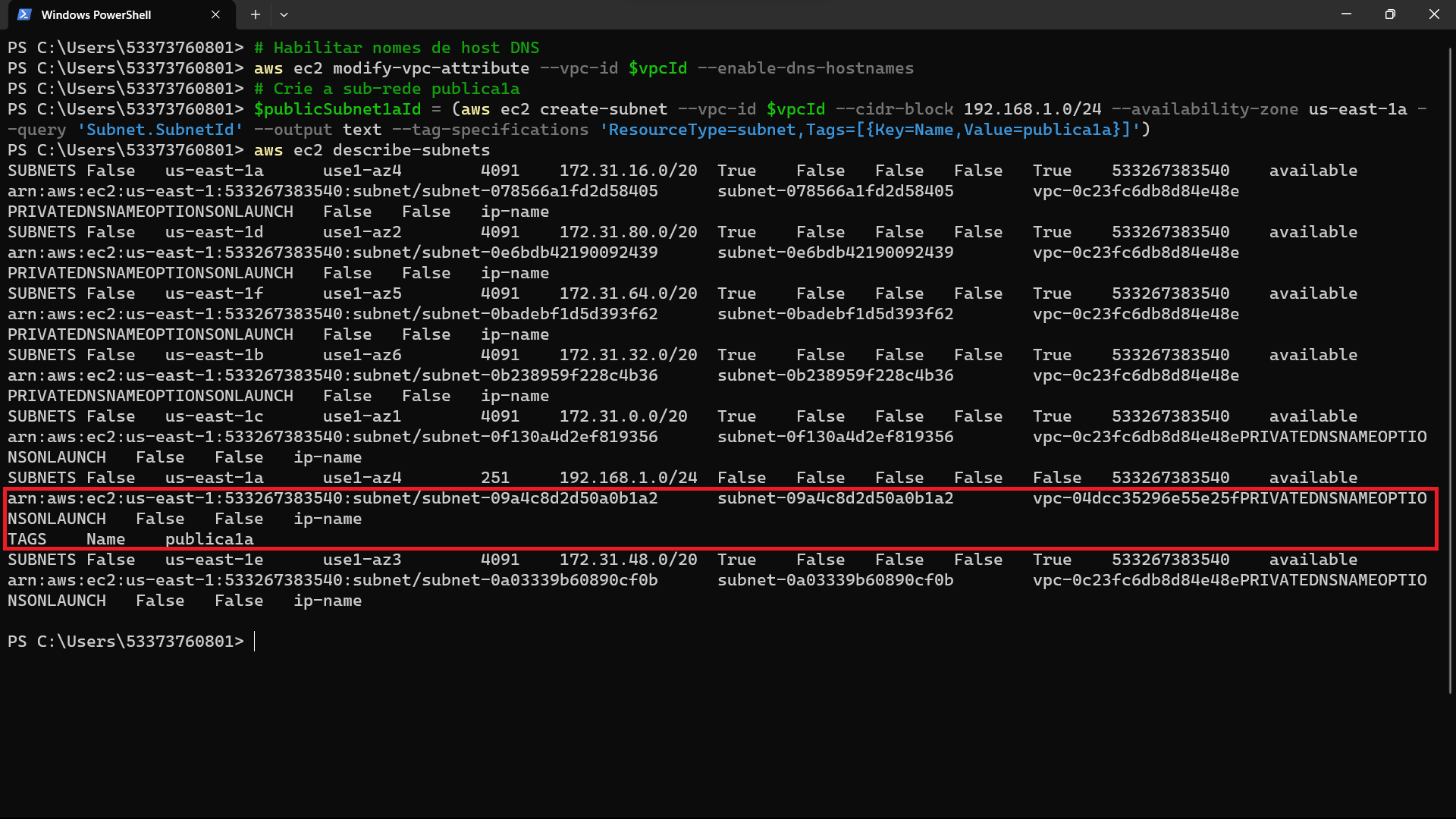
# Habilitar nomes de host DNS

aws ec2 modify-vpc-attribute --vpc-id $vpcId --enable-dns-hostnames



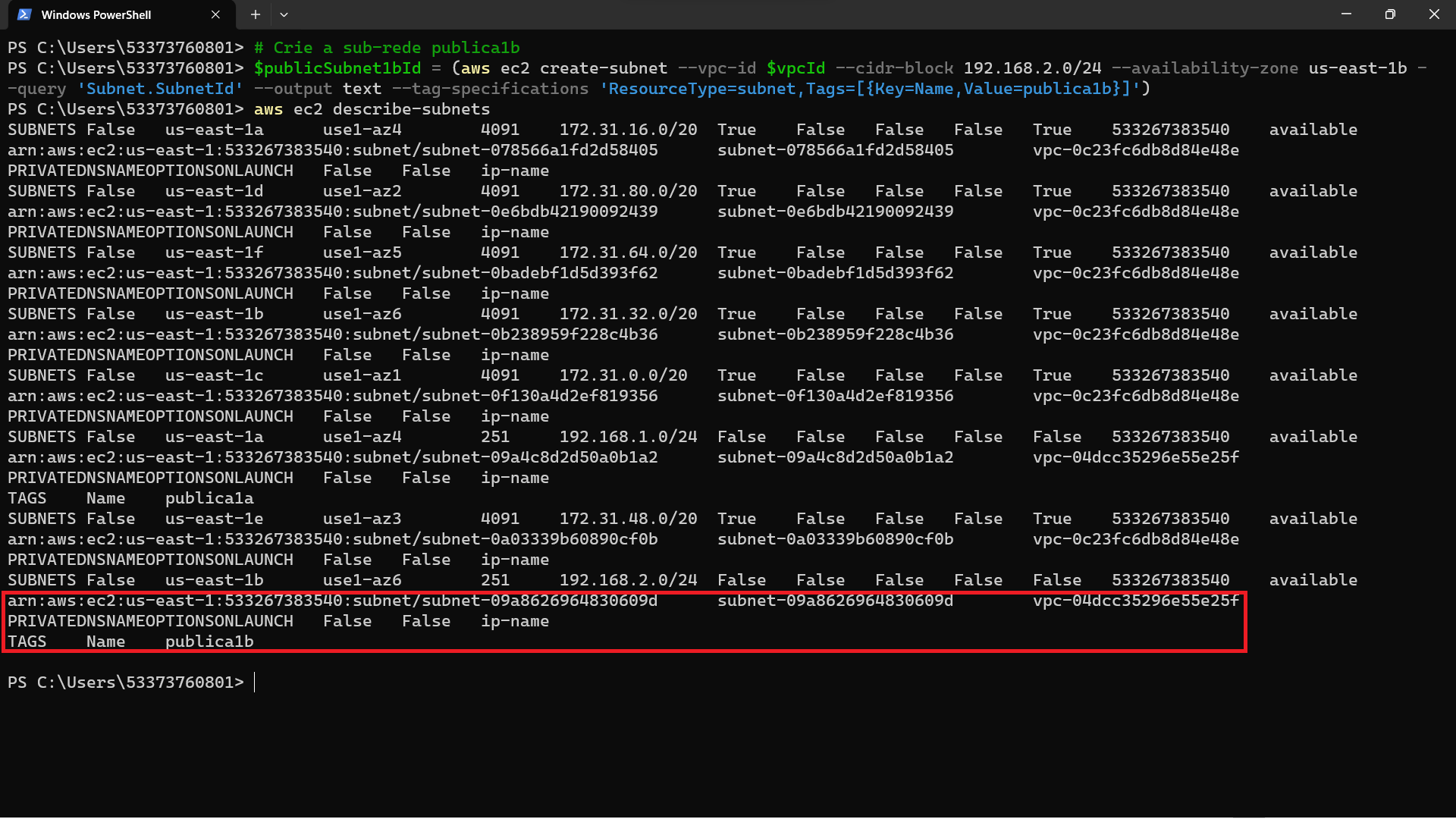
# Crie a sub-rede publica1a

$publicSubnet1aId = (aws ec2 create-subnet --vpc-id $vpcId --cidr-block 192.168.1.0/24 --availability-zone us-east-1a --query 'Subnet.SubnetId' --output text --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=publica1a}]')



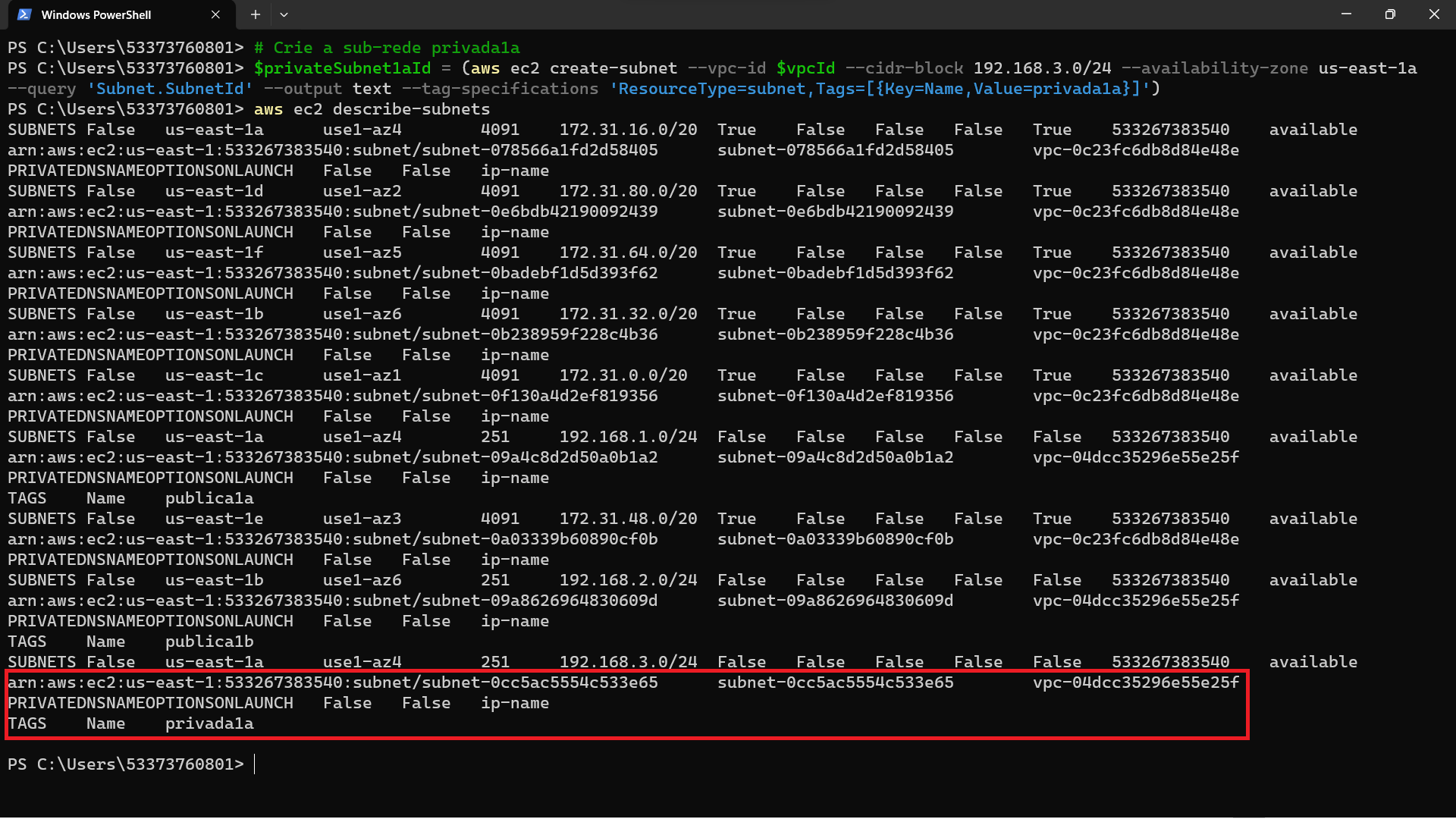
# Crie a sub-rede publica1b

$publicSubnet1bId = (aws ec2 create-subnet --vpc-id $vpcId --cidr-block 192.168.2.0/24 --availability-zone us-east-1b --query 'Subnet.SubnetId' --output text --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=publica1b}]')



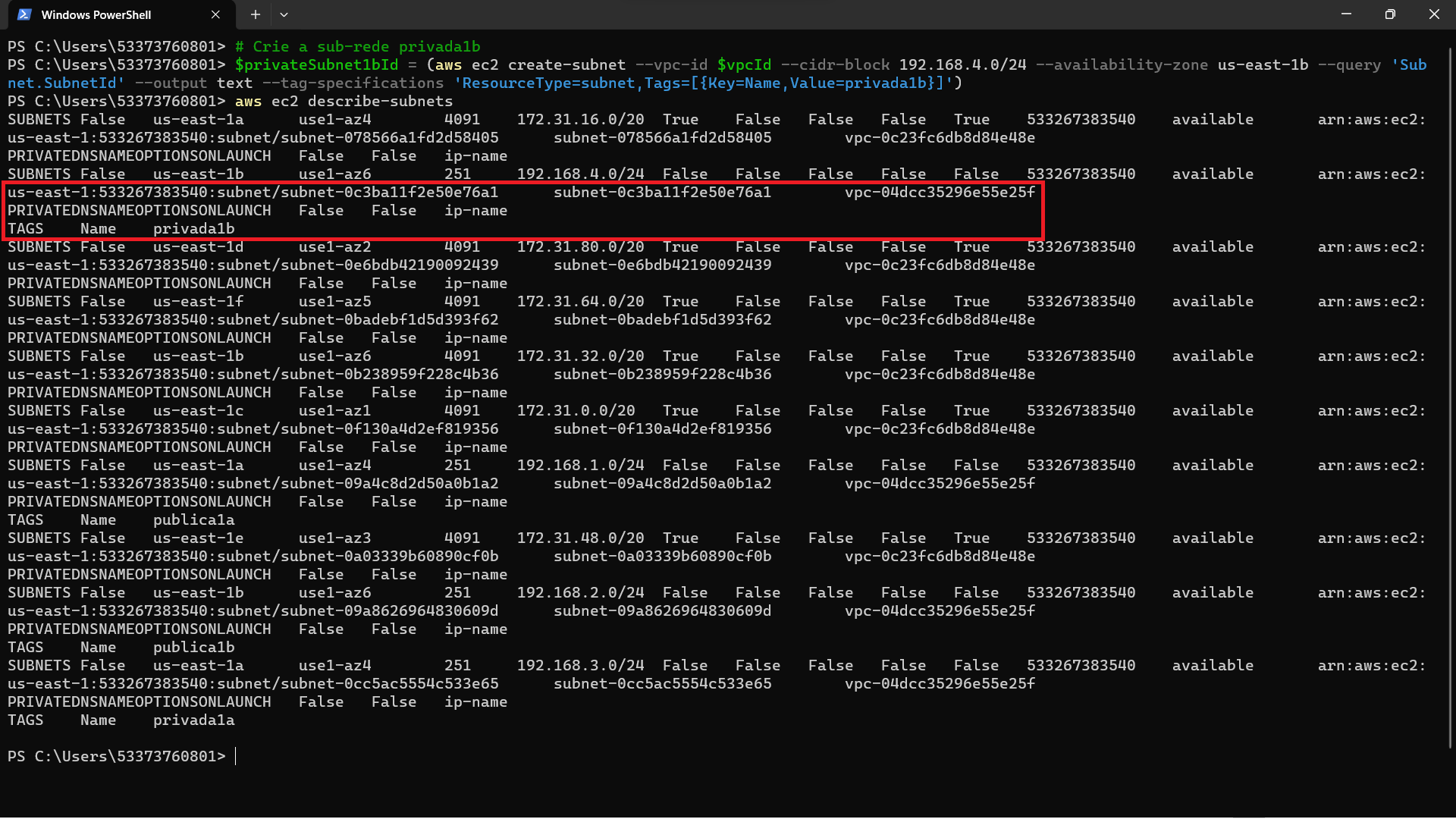
# Crie a sub-rede privada1a

$privateSubnet1aId = (aws ec2 create-subnet --vpc-id $vpcId --cidr-block 192.168.3.0/24 --availability-zone us-east-1a --query 'Subnet.SubnetId' --output text --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=privada1a}]')



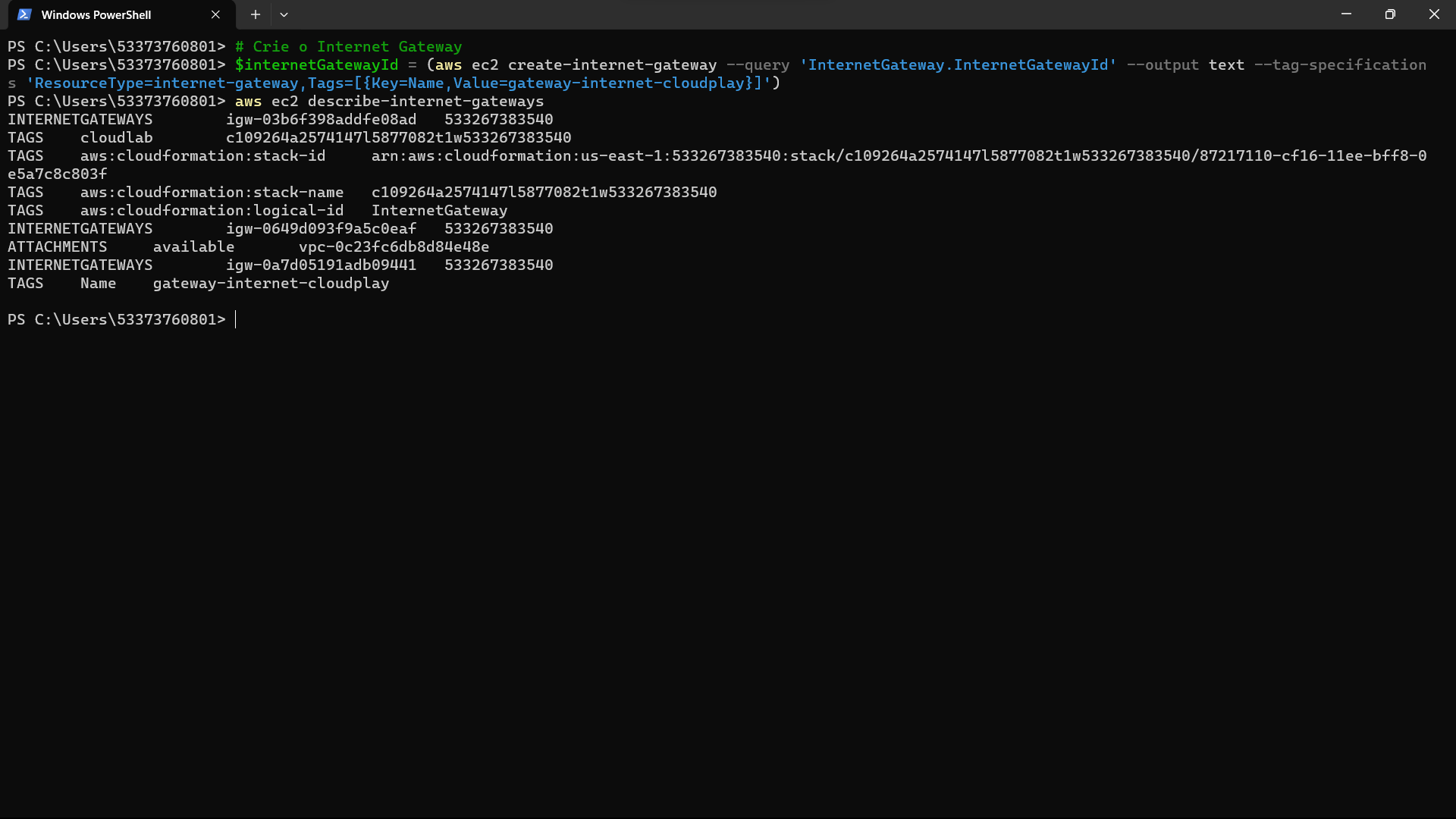
# Crie a sub-rede privada1b

$privateSubnet1bId = (aws ec2 create-subnet --vpc-id $vpcId --cidr-block 192.168.4.0/24 --availability-zone us-east-1b --query 'Subnet.SubnetId' --output text --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=privada1b}]')



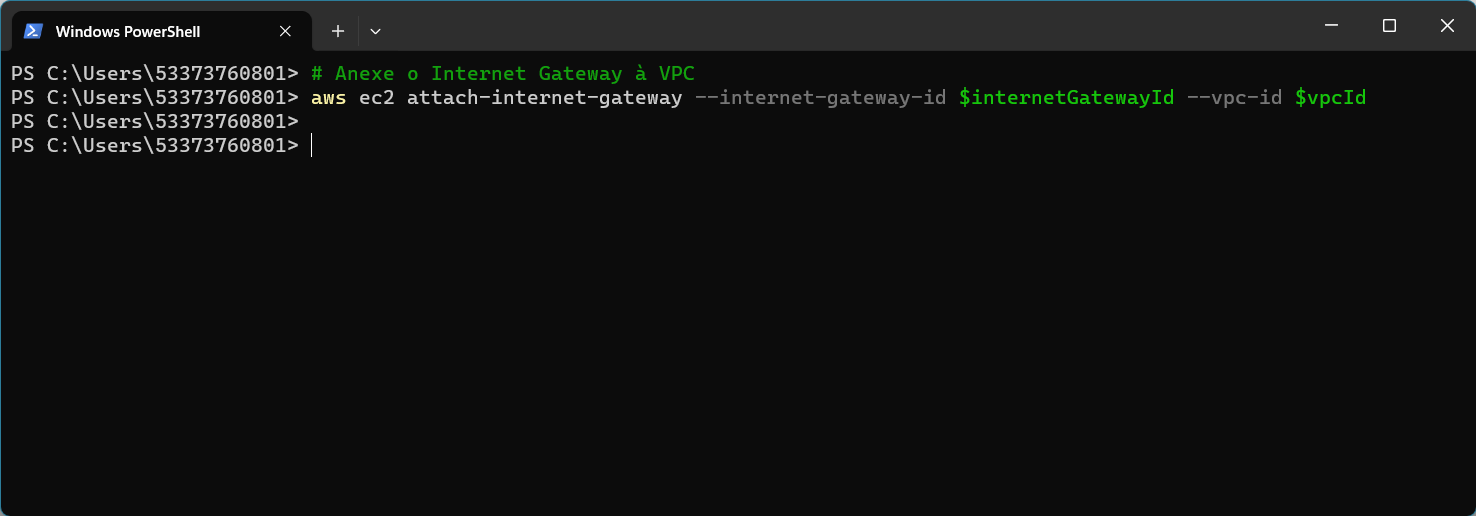
# Crie o Internet Gateway

$internetGatewayId = (aws ec2 create-internet-gateway --query 'InternetGateway.InternetGatewayId' --output text --tag-specifications 'ResourceType=internet-gateway,Tags=[{Key=Name,Value=gateway-internet-cloudplay}]')



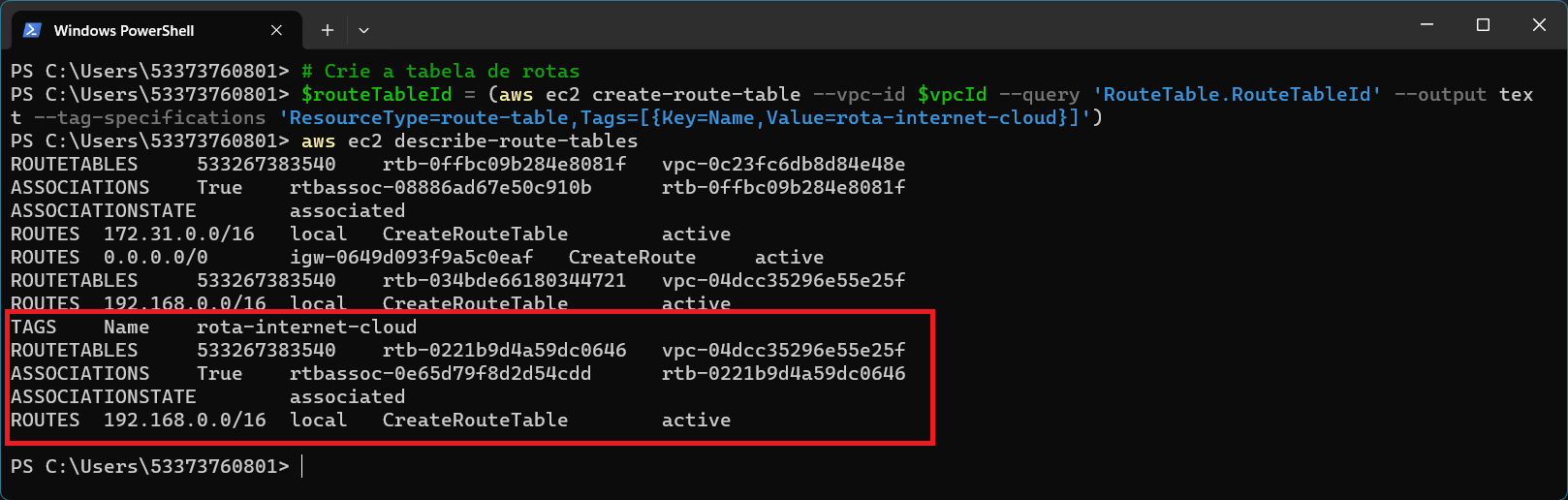
# Anexe o Internet Gateway à VPC

aws ec2 attach-internet-gateway --internet-gateway-id $internetGatewayId --vpc-id $vpcId



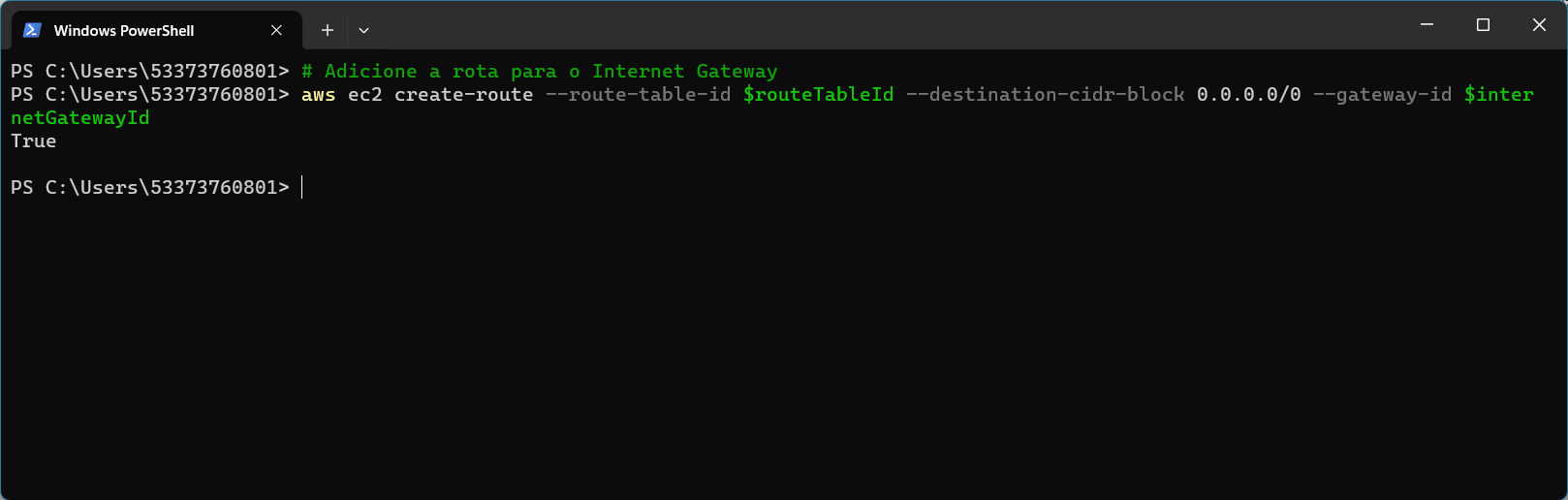
# Crie a tabela de rotas

$routeTableId = (aws ec2 create-route-table --vpc-id $vpcId --query 'RouteTable.RouteTableId' --output text --tag-specifications 'ResourceType=route-table,Tags=[{Key=Name,Value=rota-internet-cloud}]')



# Adicione a rota para o Internet Gateway

aws ec2 create-route --route-table-id $routeTableId --destination-cidr-block 0.0.0.0/0 --gateway-id $internetGatewayId

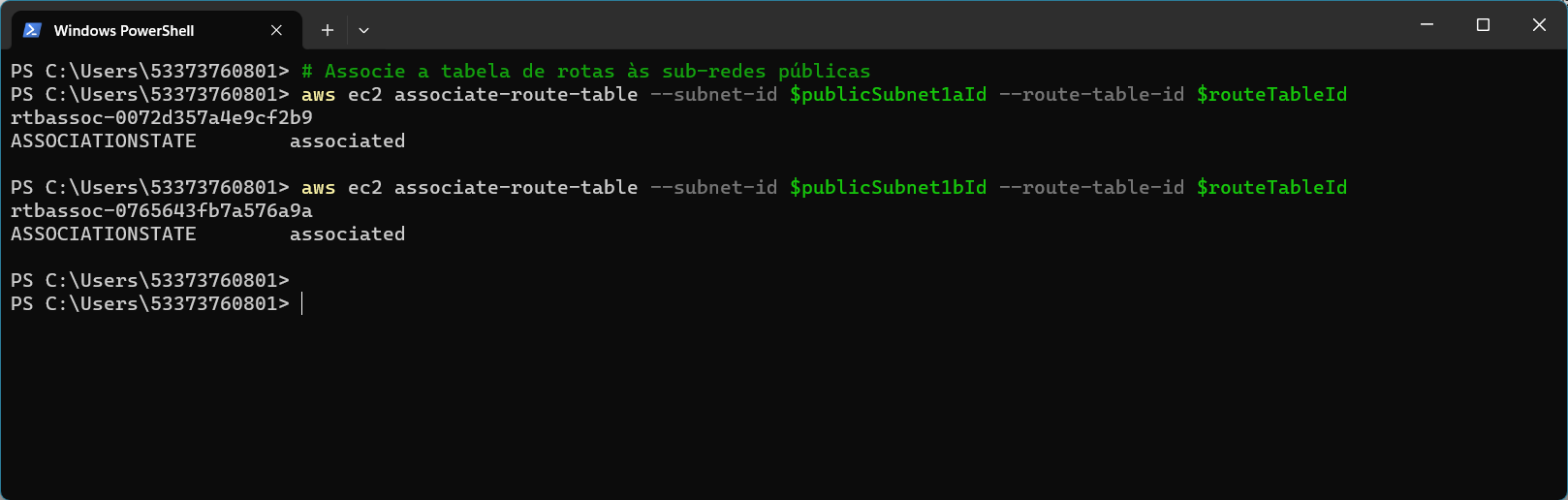


Cria

# Associe a tabela de rotas às sub-redes públicas

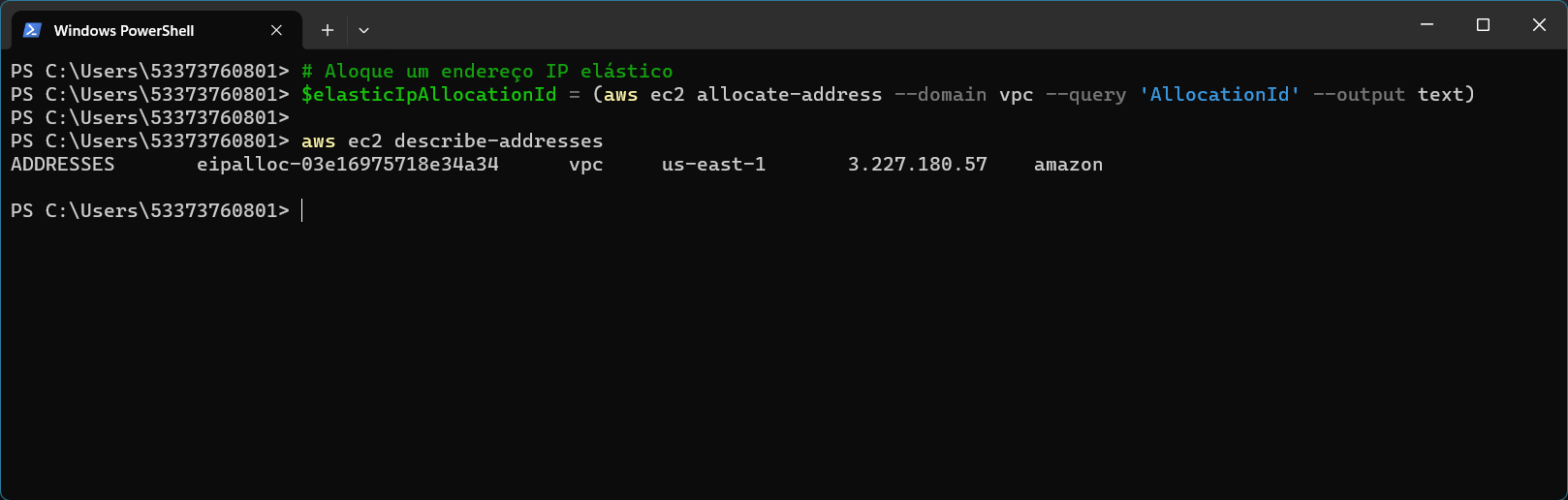
aws ec2 associate-route-table --subnet-id $publicSubnet1aId --route-table-id $routeTableId

aws ec2 associate-route-table --subnet-id $publicSubnet1bId --route-table-id $routeTableId



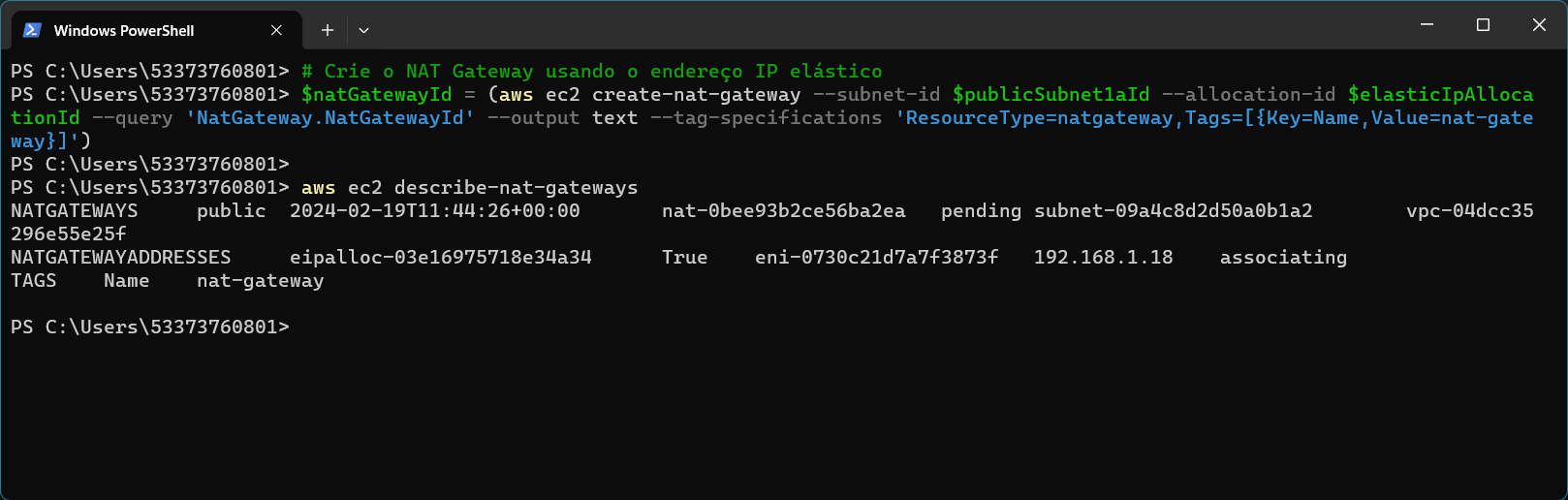
# Aloque um endereço IP elástico

$elasticIpAllocationId = (aws ec2 allocate-address --domain vpc --query 'AllocationId' --output text)



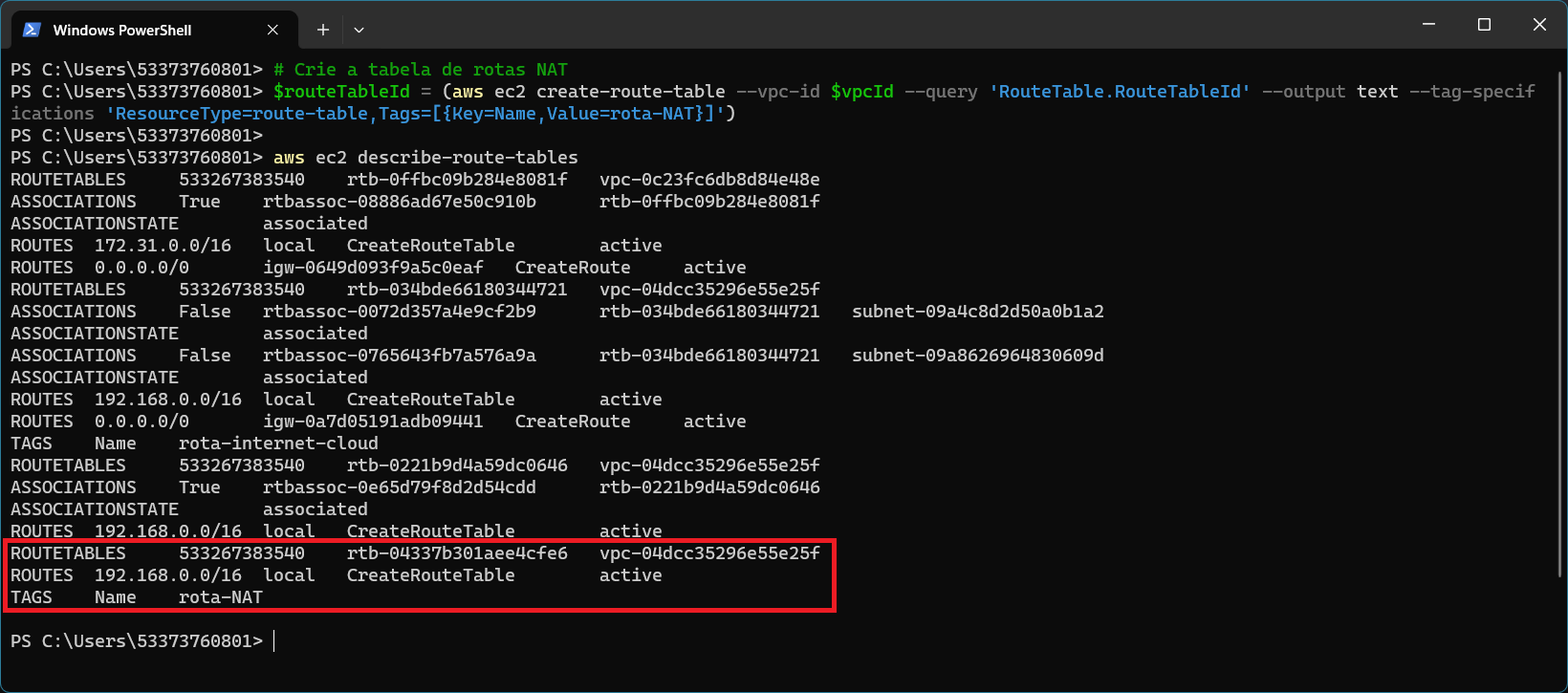
# Crie o NAT Gateway usando o endereço IP elástico

$natGatewayId = (aws ec2 create-nat-gateway --subnet-id $publicSubnet1aId --allocation-id $elasticIpAllocationId --query 'NatGateway.NatGatewayId' --output text --tag-specifications 'ResourceType=natgateway,Tags=[{Key=Name,Value=nat-gateway}]')



# Crie a tabela de rotas NAT

$routeTableId = (aws ec2 create-route-table --vpc-id $vpcId --query 'RouteTable.RouteTableId' --output text --tag-specifications 'ResourceType=route-table,Tags=[{Key=Name,Value=rota-NAT}]')

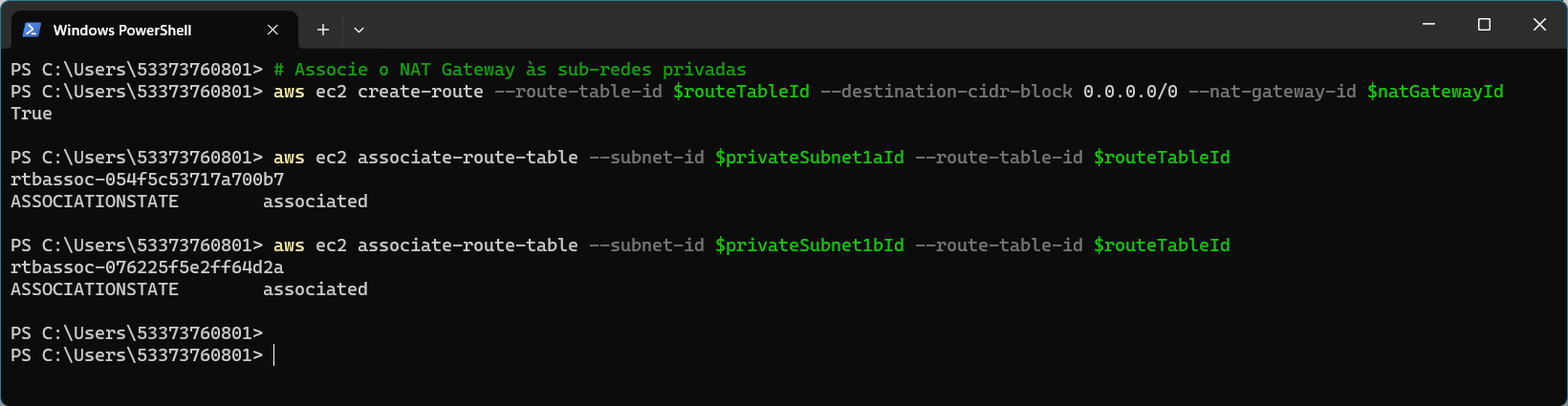


# Associe o NAT Gateway às sub-redes privadas

aws ec2 create-route --route-table-id $routeTableId --destination-cidr-block 0.0.0.0/0 --nat-gateway-id $natGatewayId

aws ec2 associate-route-table --subnet-id $privateSubnet1aId --route-table-id $routeTableId

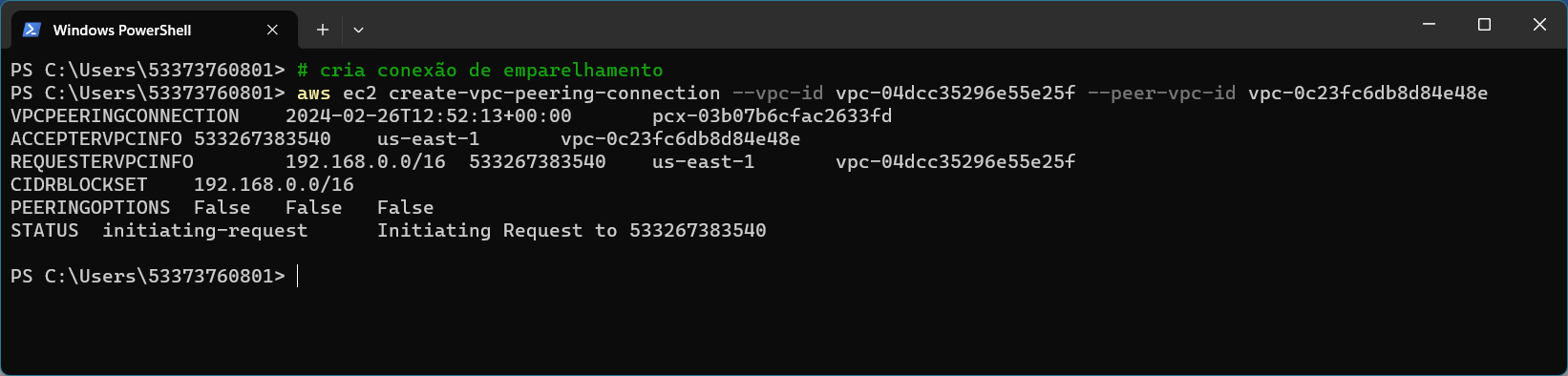
aws ec2 associate-route-table --subnet-id $privateSubnet1bId --route-table-id $routeTableId



**conexões de emparelhamento**

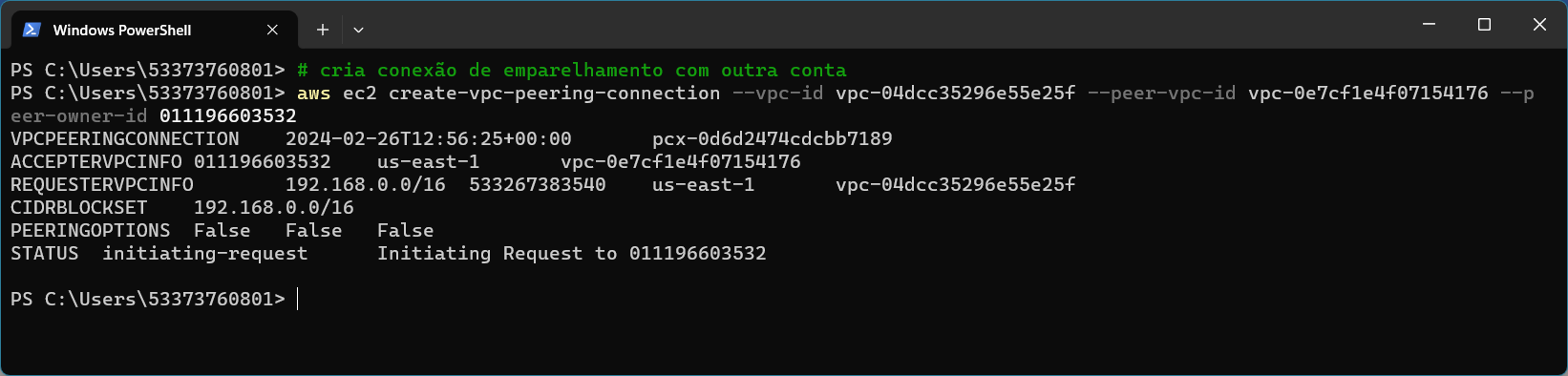
# cria conexão de emparelhamento com a mesma conta

aws ec2 create-vpc-peering-connection --vpc-id vpc-04dcc35296e55e25f --peer-vpc-id vpc-0c23fc6db8d84e48e



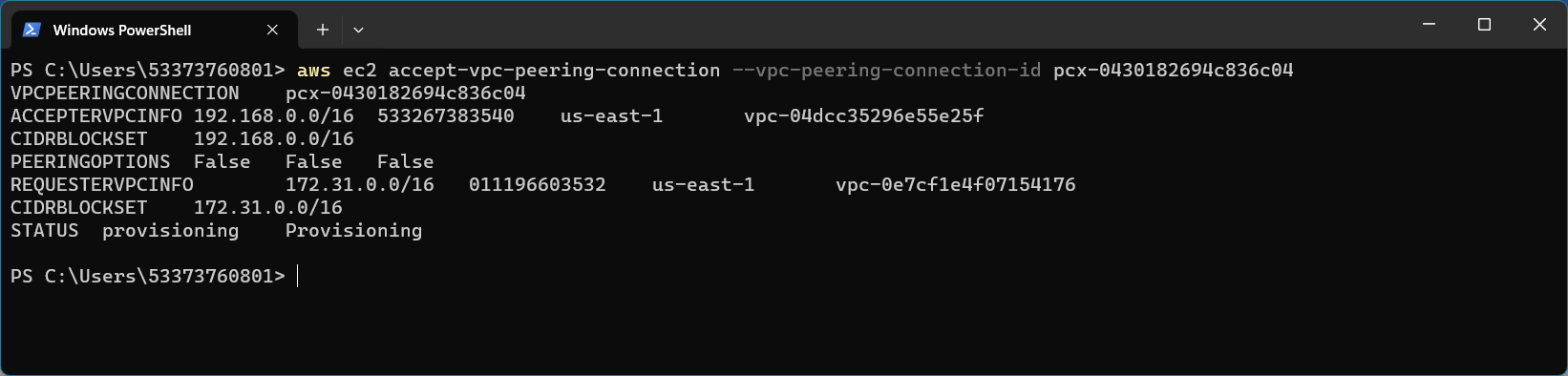
# cria conexão de emparelhamento com outra conta

aws ec2 create-vpc-peering-connection --vpc-id vpc-04dcc35296e55e25f --peer-vpc-id vpc-0e7cf1e4f07154176 --peer-owner-id 011196603532



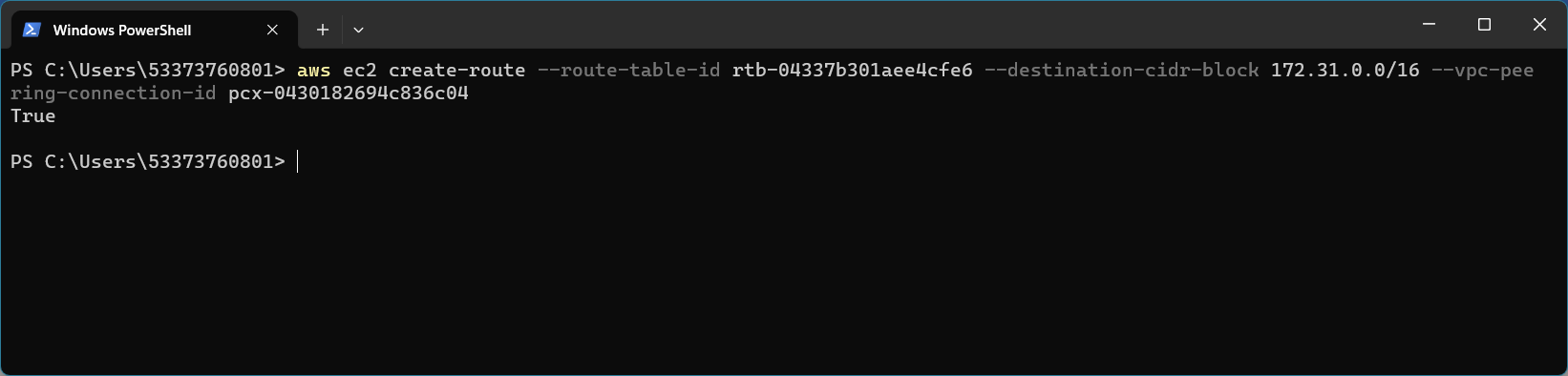
# aceita conexão de emparelhamento

aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-0430182694c836c04



# atualiza tabela de rotas

aws ec2 create-route --route-table-id rtb-04337b301aee4cfe6 --destination-cidr-block 172.31.0.0/16 --vpc-peering-connection-id pcx-0430182694c836c04



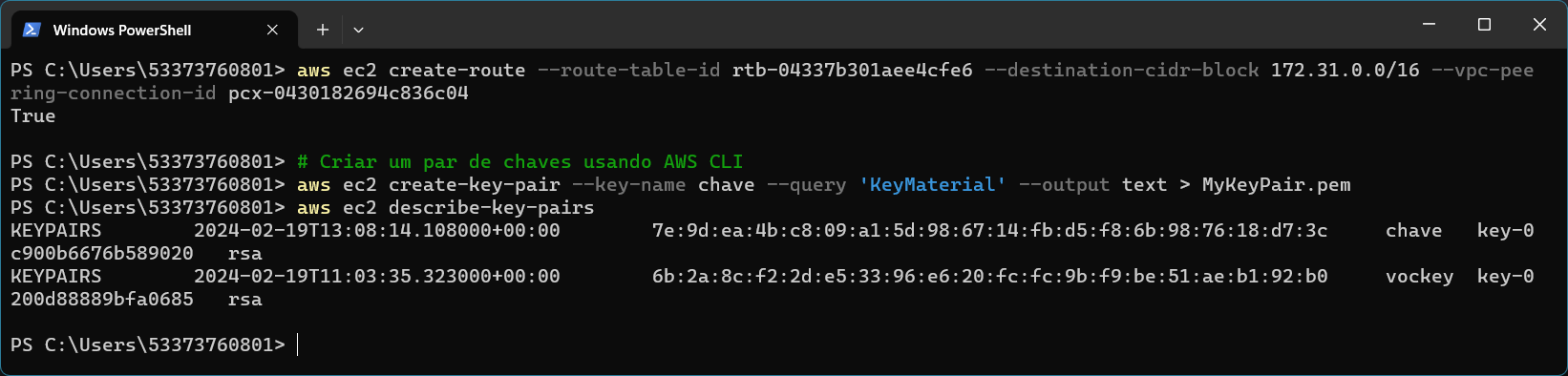
**EC2**

**Par de chaves**

# Criar um par de chaves usando AWS CLI

aws ec2 create-key-pair --key-name chave --query 'KeyMaterial' --output text > MyKeyPair.pem

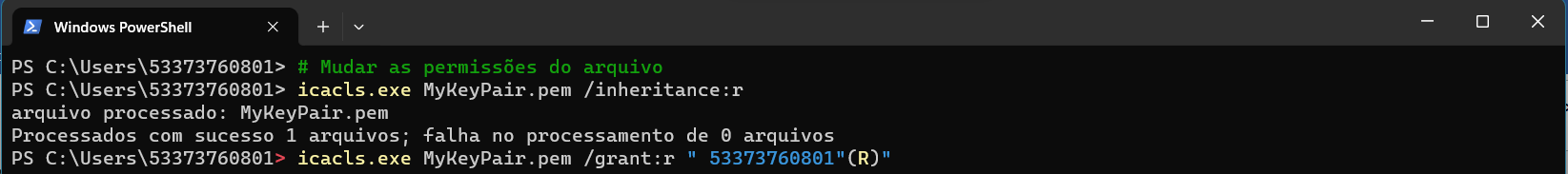
aws ec2 describe-key-pairs



# Mudar as permissões do arquivo

icacls.exe MyKeyPair.pem /inheritance:r

icacls.exe MyKeyPair.pem /grant:r " 53373760801"(R)"

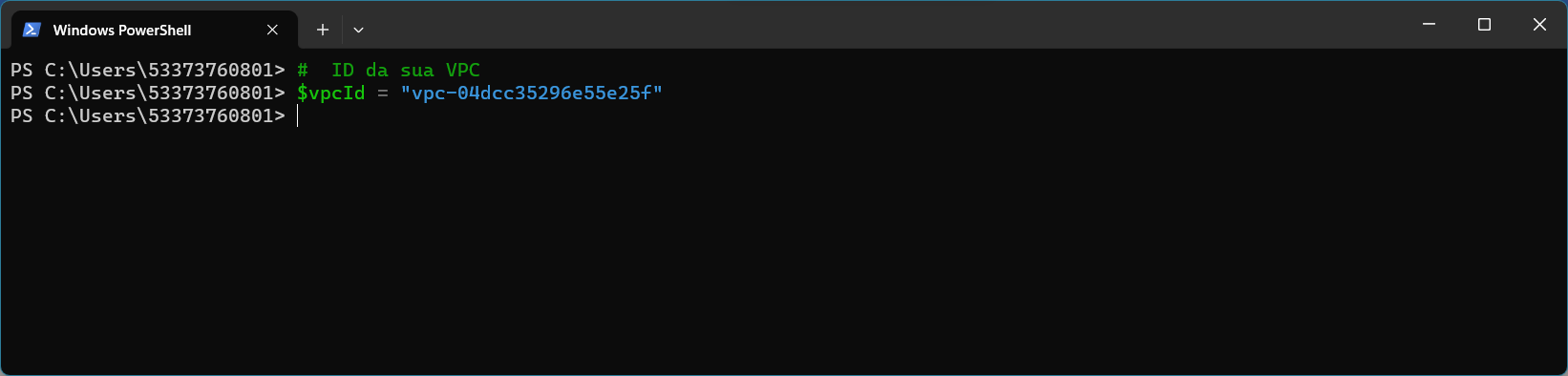


**Grupo de segurança**

**WINDOWS**

# ID da sua VPC

$vpcId = "vpc-04dcc35296e55e25f"



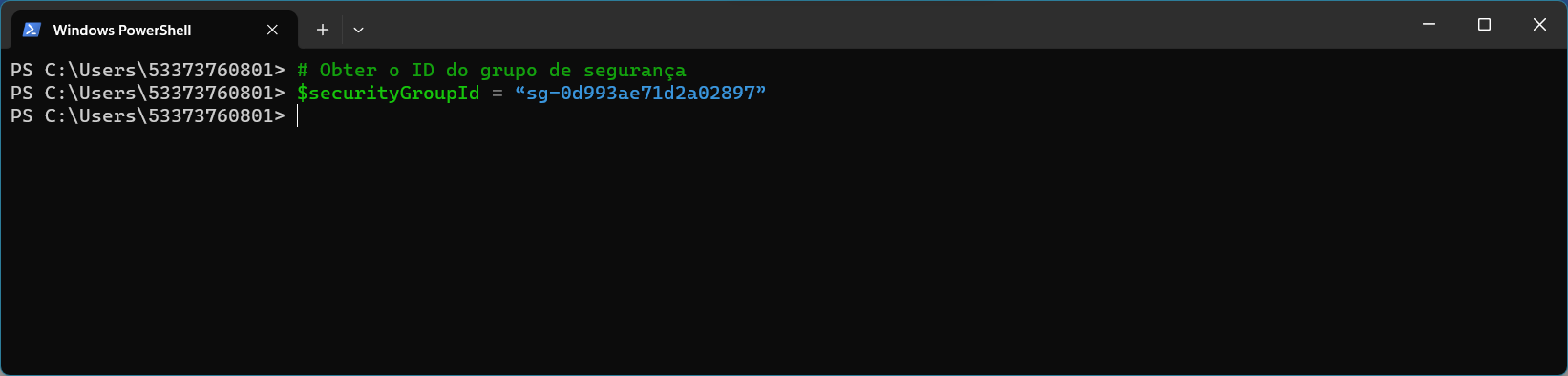
# Criar um grupo de segurança na VPC especificada

aws ec2 create-security-group --group-name groupsec-win --description "libera ping e rdp" --vpc-id vpc- $vpcId



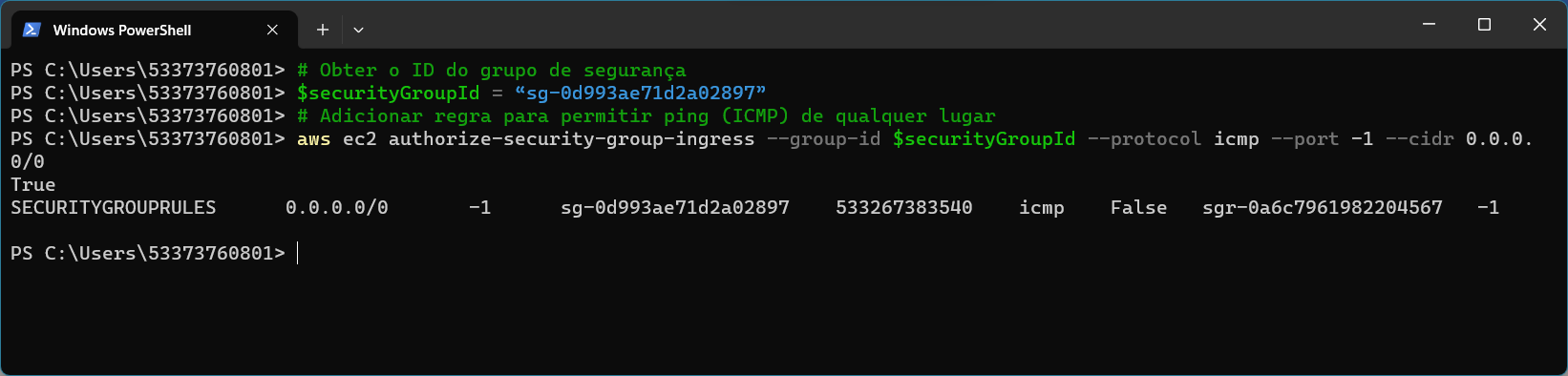
# Obter o ID do grupo de segurança

$securityGroupId = “sg-0d993ae71d2a02897”



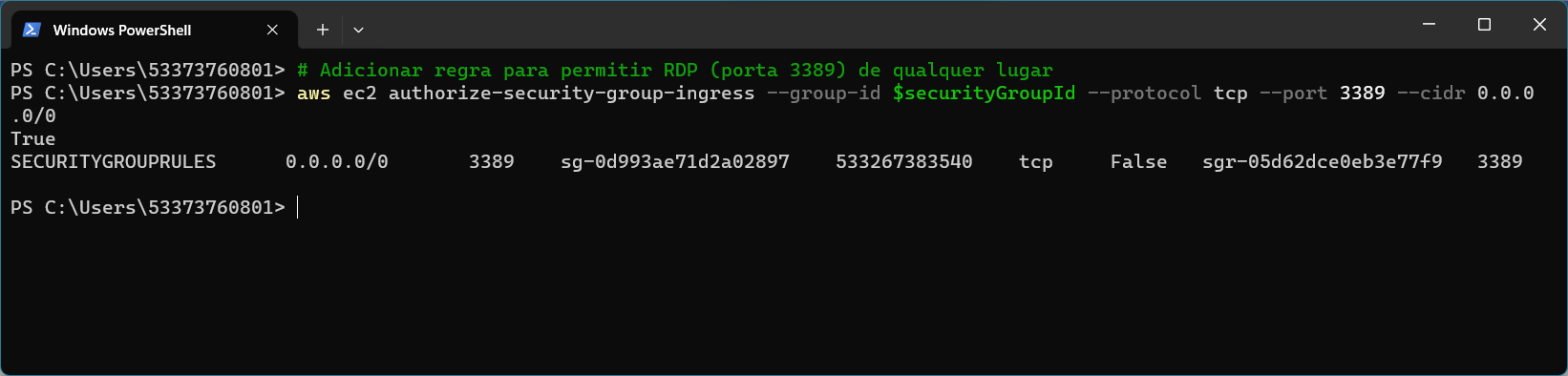
# Adicionar regra para permitir ping (ICMP) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol icmp --port -1 --cidr 0.0.0.0/0



# Adicionar regra para permitir RDP (porta 3389) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol tcp --port 3389 --cidr 0.0.0.0/0



**LINUX**

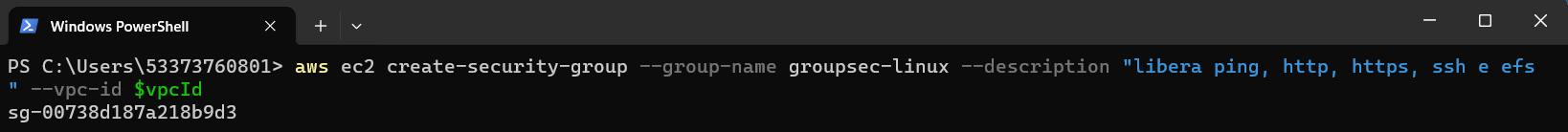
# ID da sua VPC

$vpcId = "vpc-04dcc35296e55e25f"



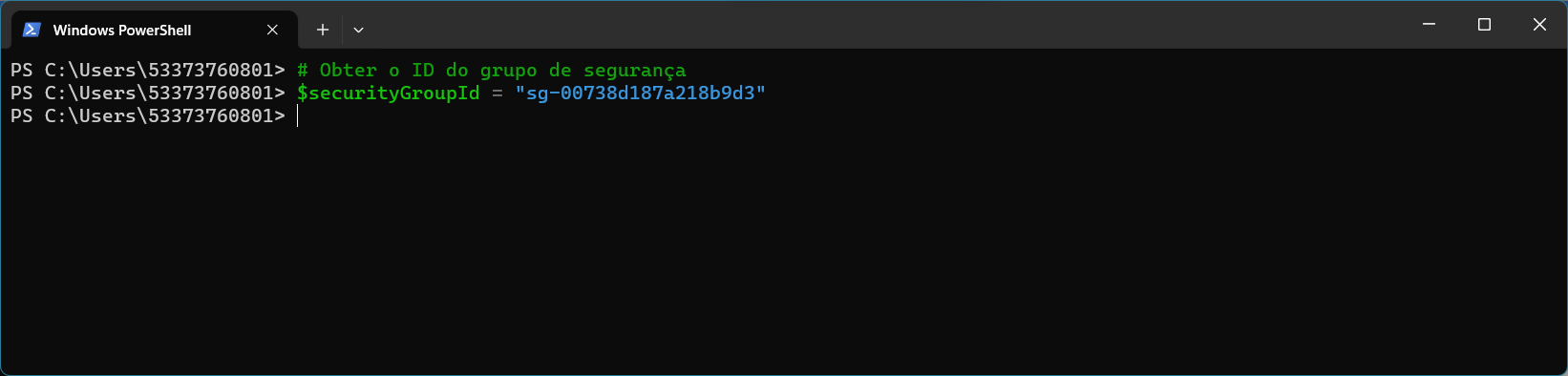
# Criar um grupo de segurança na VPC especificada

aws ec2 create-security-group --group-name groupsec-linux --description "libera ping, http, https, ssh e efs" --vpc-id $vpcId



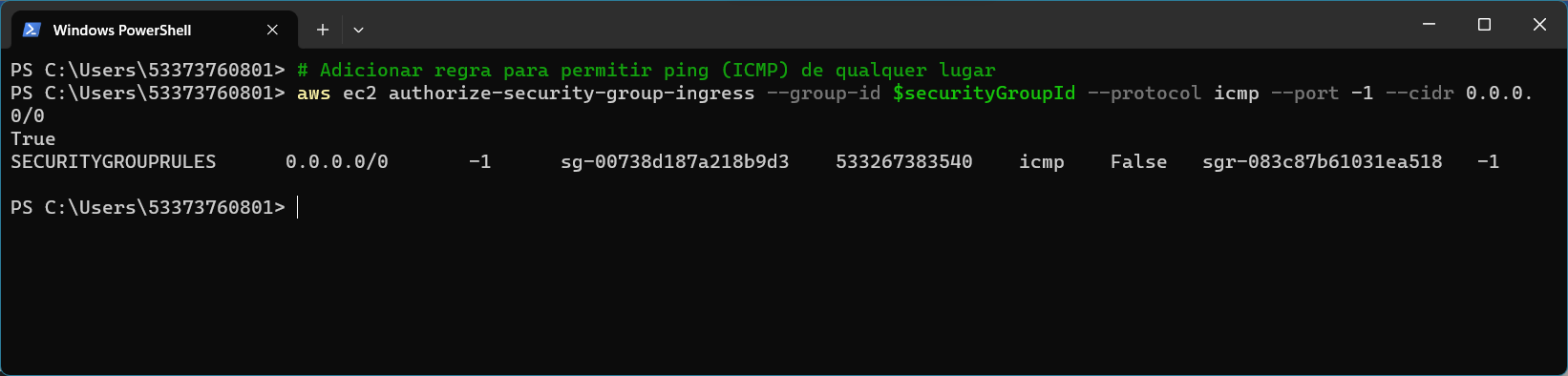
# Obter o ID do grupo de segurança

$securityGroupId = "sg-00738d187a218b9d3"



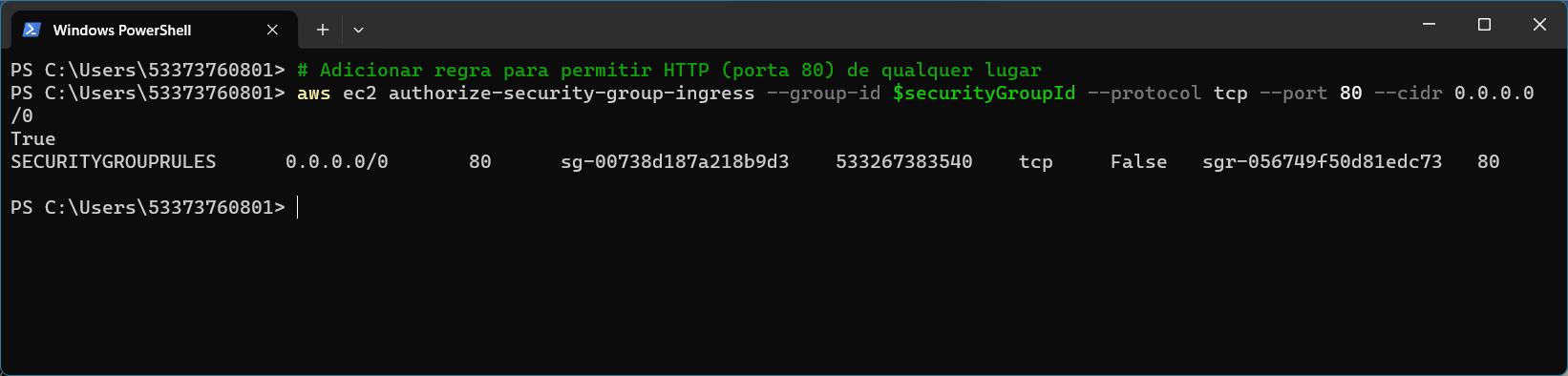
# Adicionar regra para permitir ping (ICMP) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol icmp --port -1 --cidr 0.0.0.0/0



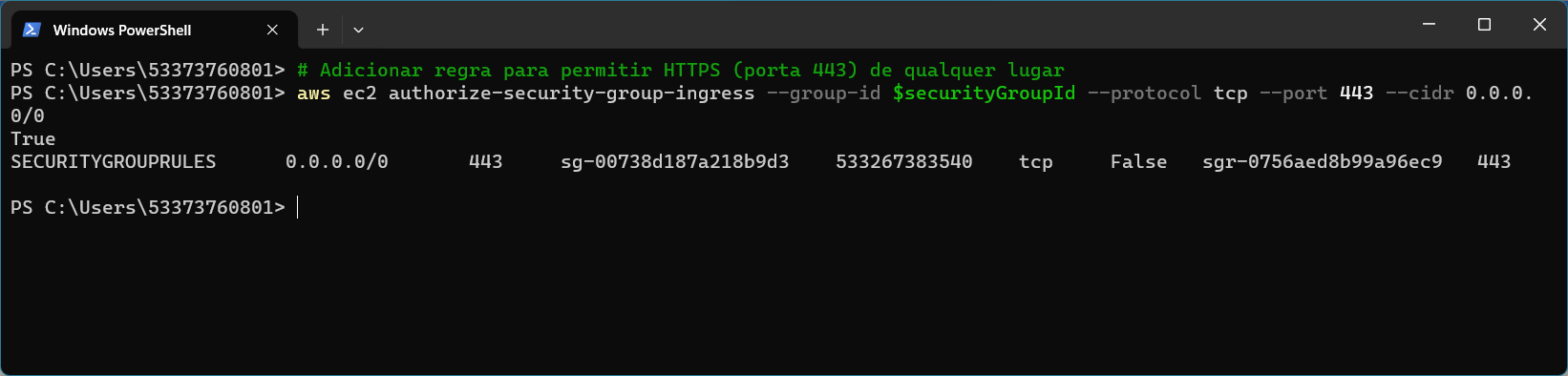
# Adicionar regra para permitir HTTP (porta 80) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol tcp --port 80 --cidr 0.0.0.0/0



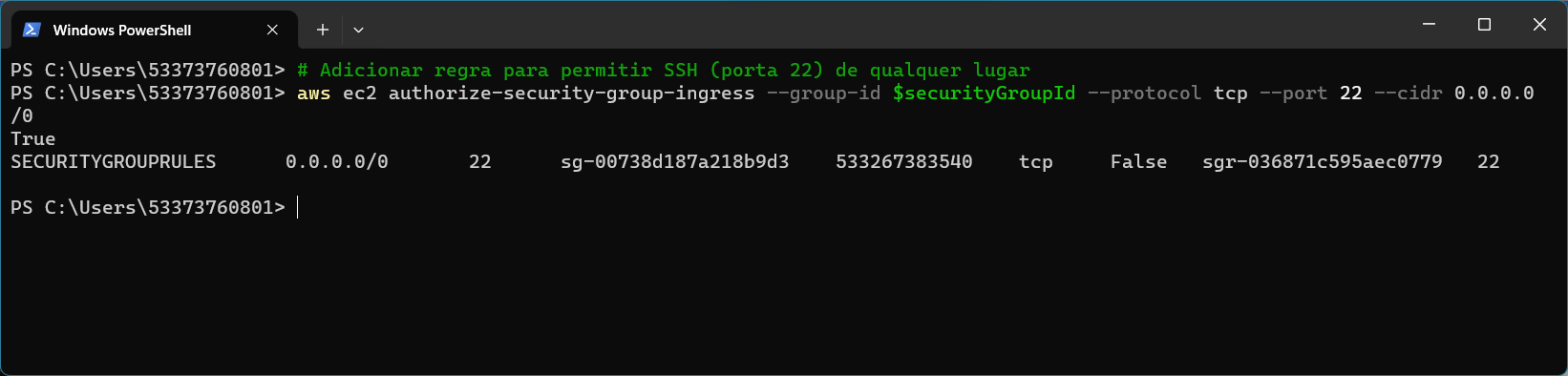
# Adicionar regra para permitir HTTPS (porta 443) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol tcp --port 443 --cidr 0.0.0.0/0



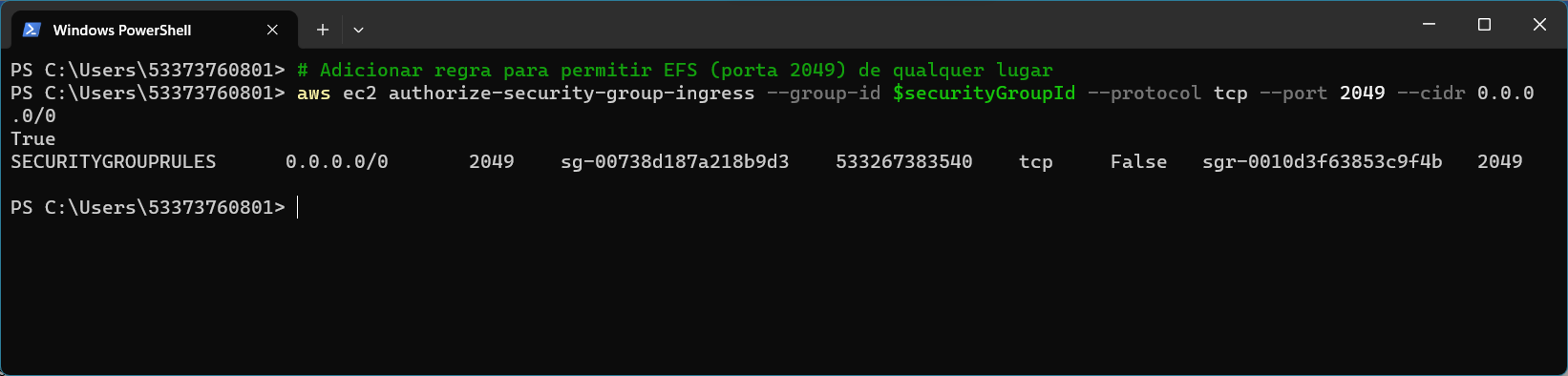
# Adicionar regra para permitir SSH (porta 22) de qualquer lugar

aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol tcp --port 22 --cidr 0.0.0.0/0



# Adicionar regra para permitir EFS (porta 2049) de qualquer lugar

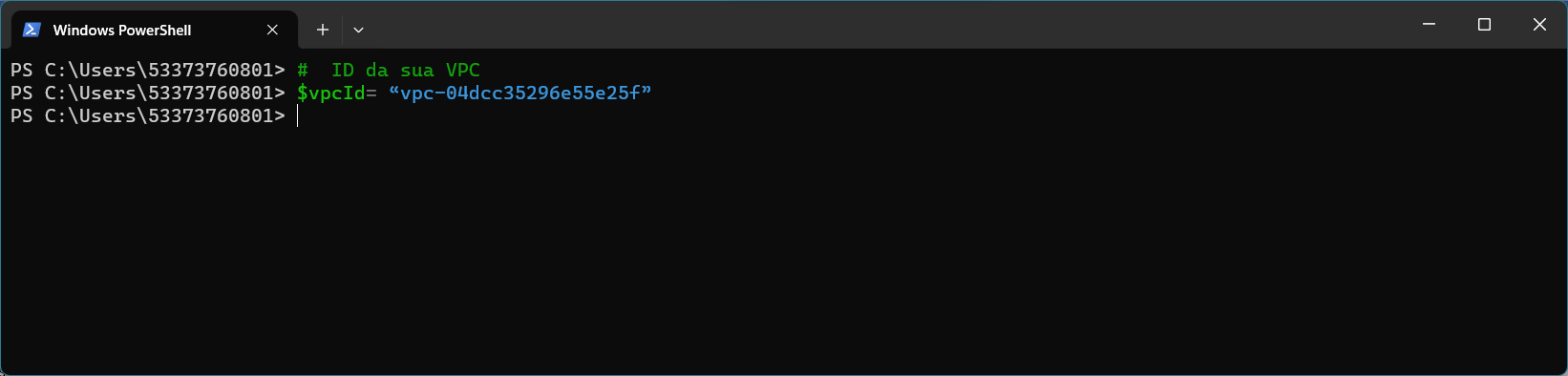
aws ec2 authorize-security-group-ingress --group-id $securityGroupId --protocol tcp --port 2049 --cidr 0.0.0.0/0



**EFS**

# ID da sua VPC

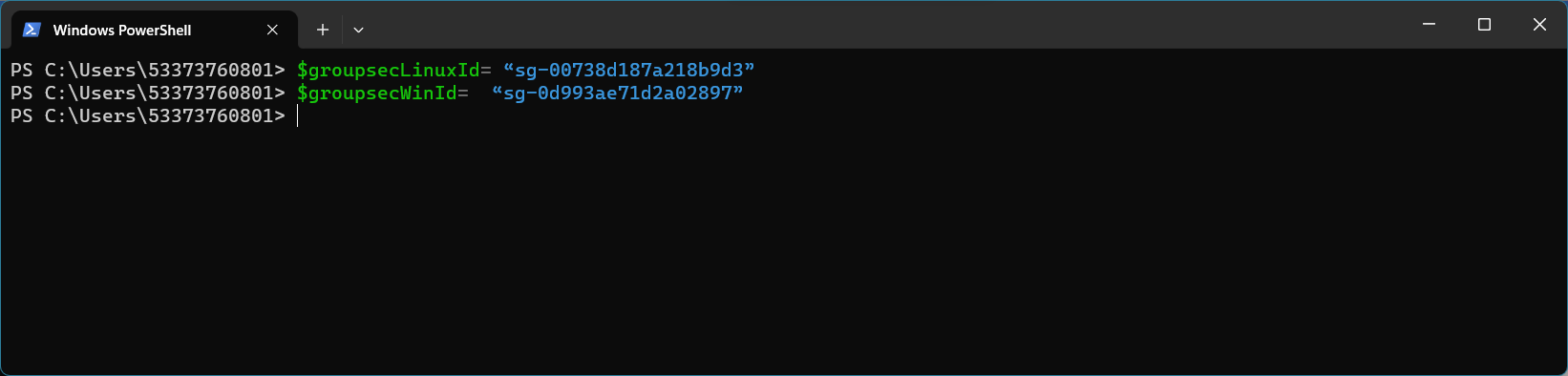
$vpcId= “vpc-04dcc35296e55e25f”



# Obter o ID do grupo de segurança

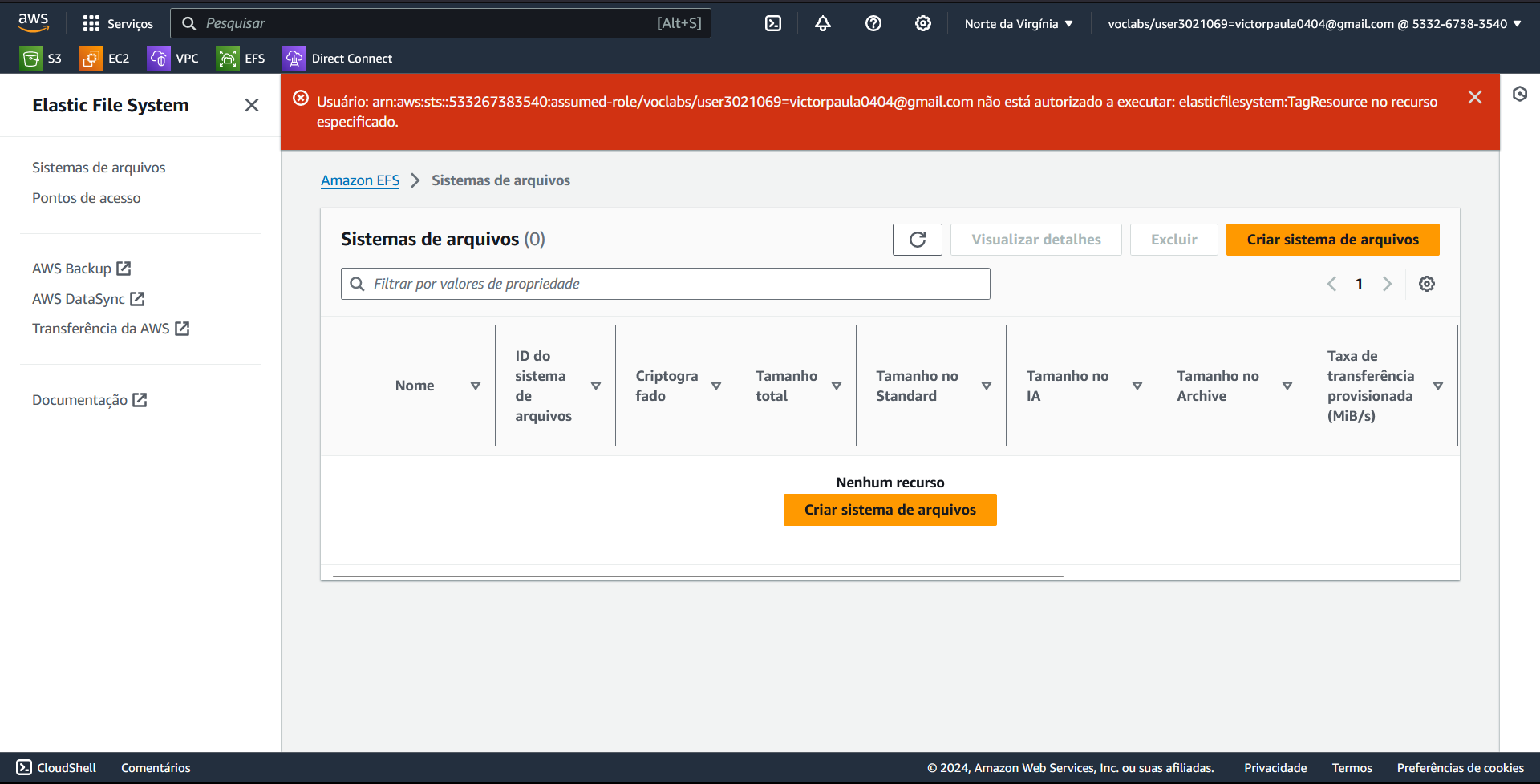
$groupsecLinuxId= “sg-00738d187a218b9d3”

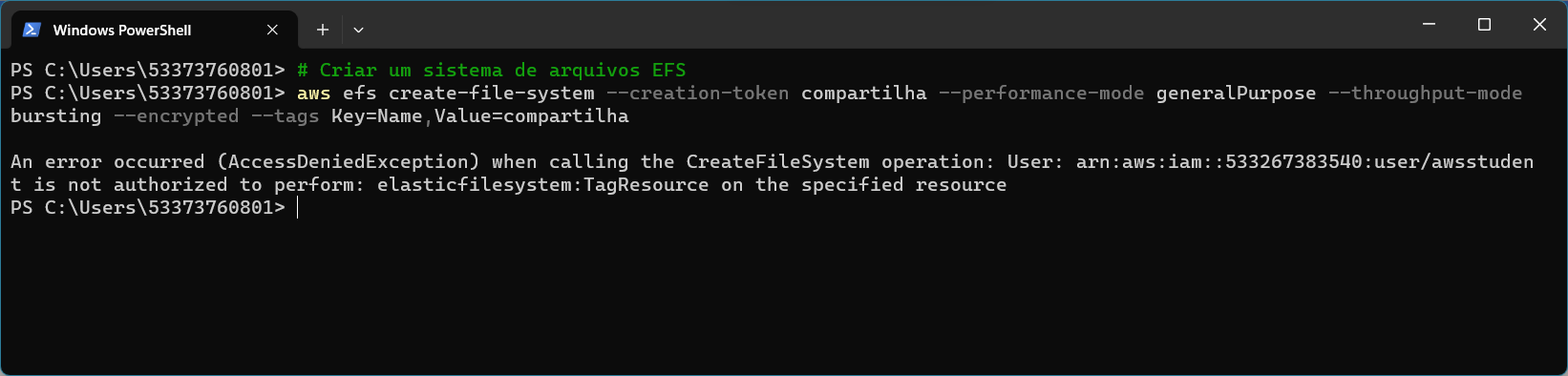
$groupsecWinId= “sg-0d993ae71d2a02897”



# Criar um sistema de arquivos EFS

aws efs create-file-system --creation-token compartilha --performance-mode generalPurpose --throughput-mode bursting --encrypted --tags Key=Name,Value=compartilha





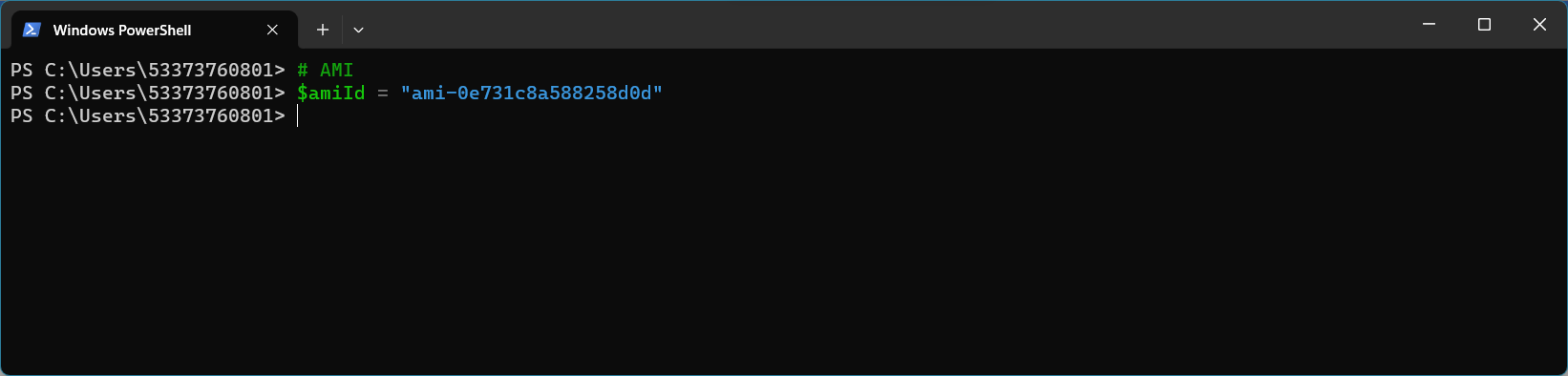
**Com a nossa conta de arquiteto não tem há permissão necessária para cria um sistema de arquivo**

**Instancias**

**rede privada**

# Consultar AMI no Marketplace

$amiId = "ami-0e731c8a588258d0d"



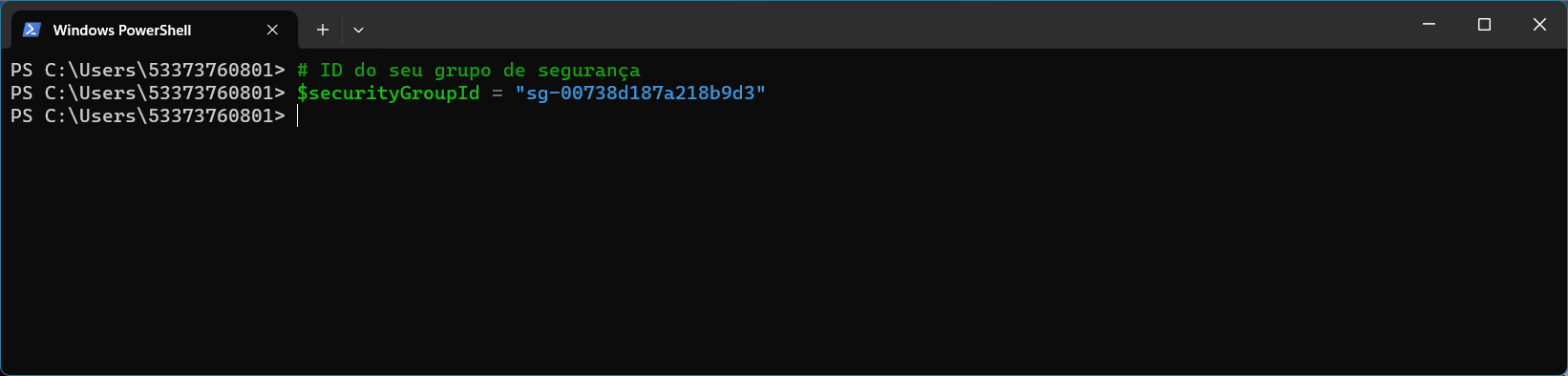
# ID da sua sub-rede

$subnetId = "subnet-0cc5ac5554c533e65"



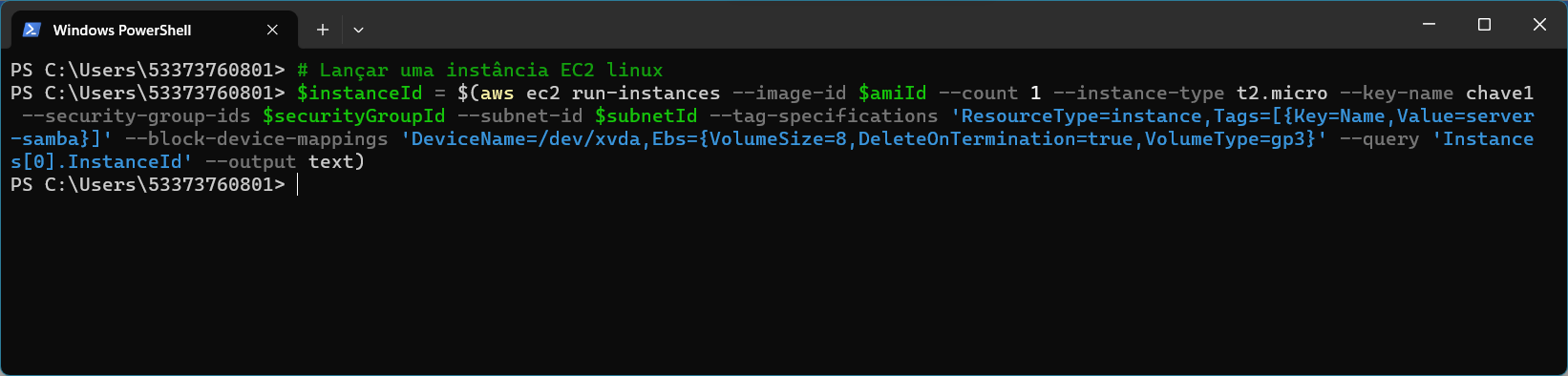
# ID do seu grupo de segurança

$securityGroupId = "sg-00738d187a218b9d3"

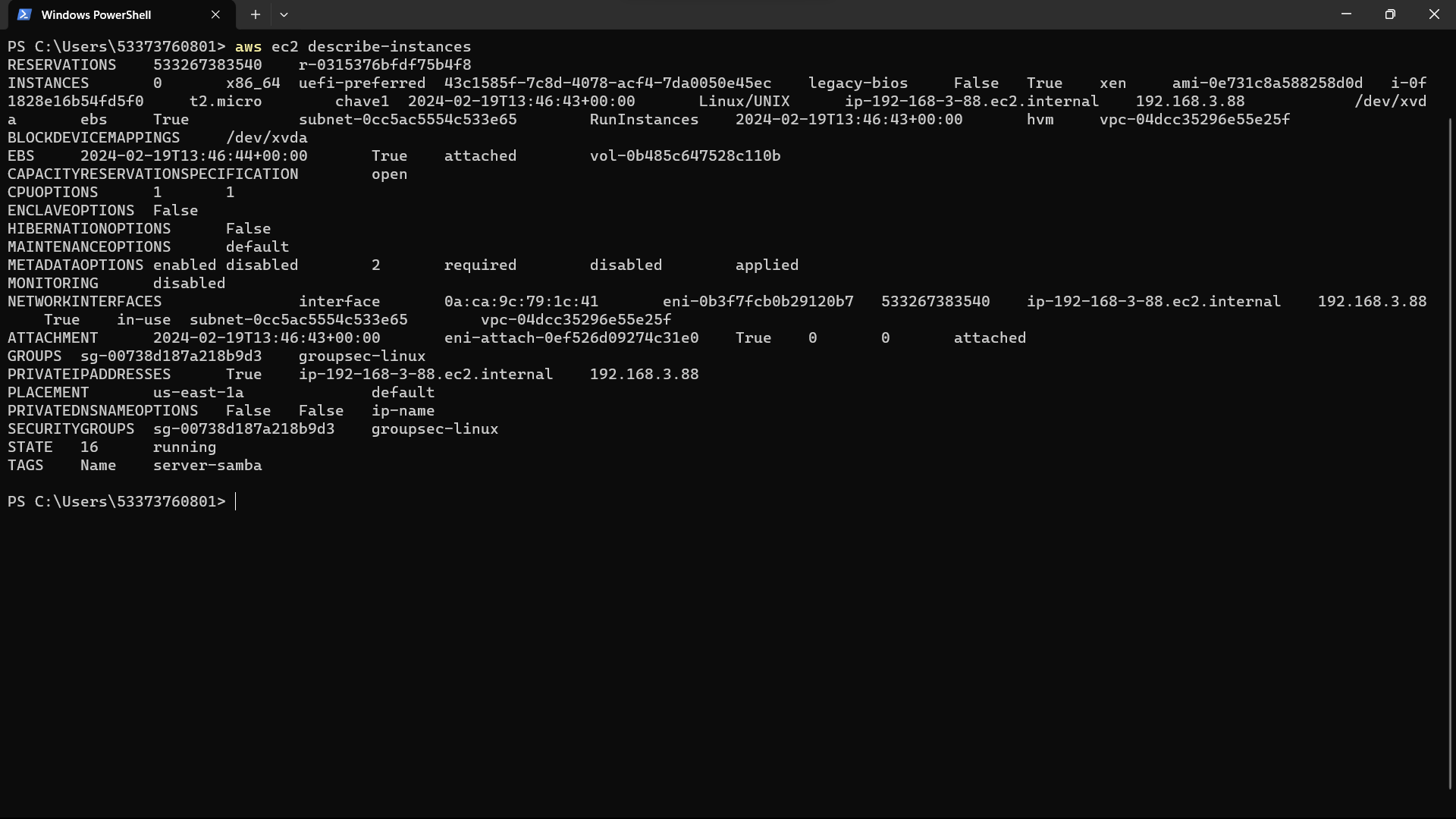


# Lançar uma instância EC2 linux

$instanceId = $(aws ec2 run-instances --image-id $amiId --count 1 --instance-type t2.micro --key-name chave1 --security-group-ids $securityGroupId --subnet-id $subnetId --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=server-samba}]' --block-device-mappings 'DeviceName=/dev/xvda,Ebs={VolumeSize=8,DeleteOnTermination=true,VolumeType=gp3}' --query 'Instances[0].InstanceId' --output text)

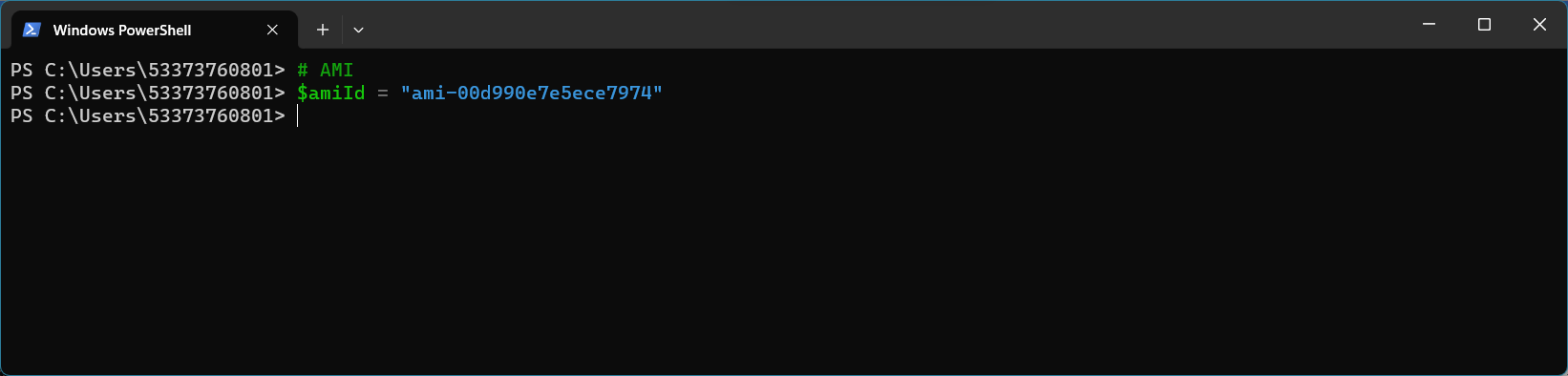


aws ec2 describe-instances



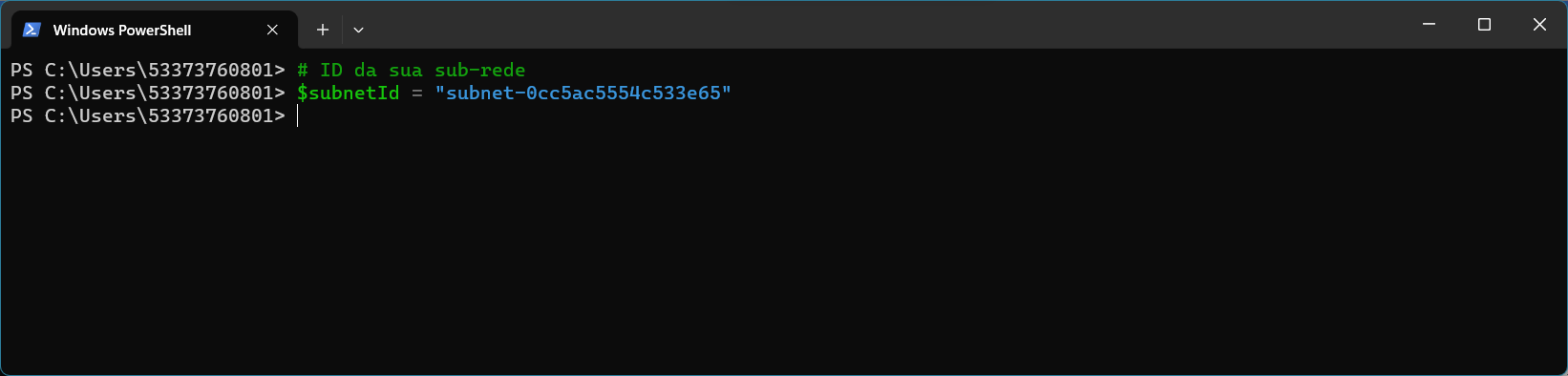
# AMI

$amiId = "ami-00d990e7e5ece7974"



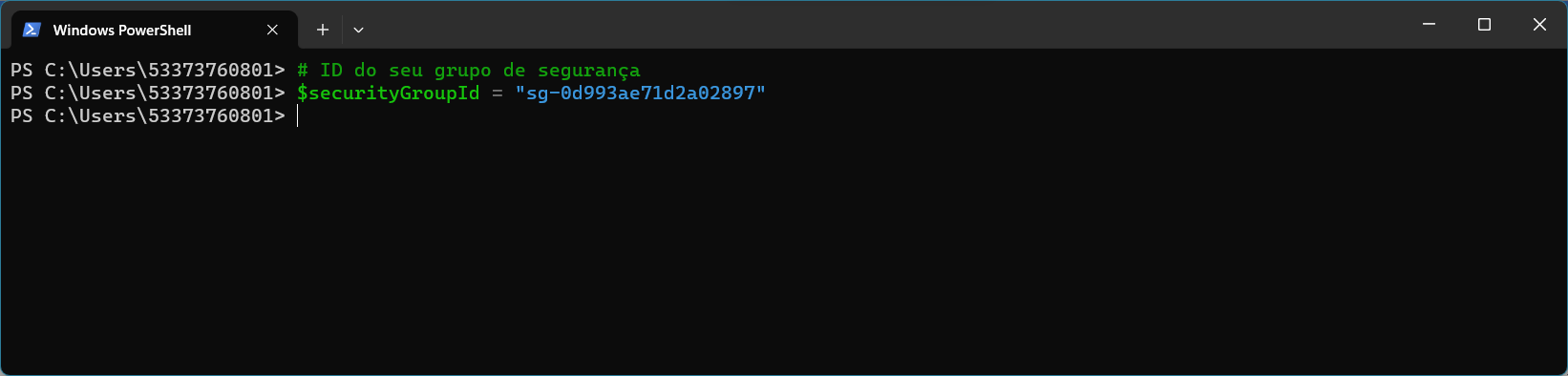
# ID da sua sub-rede

$subnetId = "subnet-0cc5ac5554c533e65"



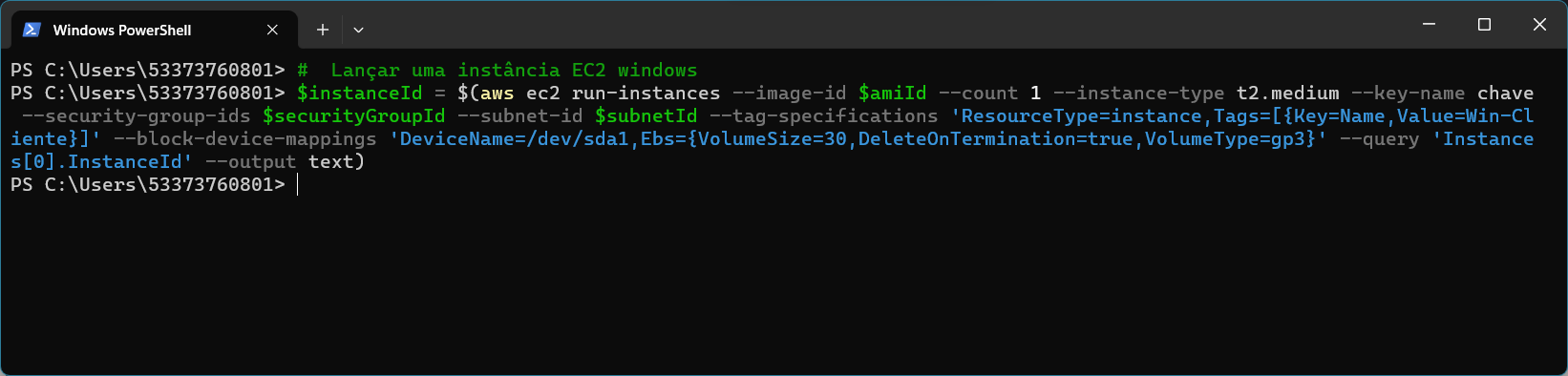
# ID do seu grupo de segurança

$securityGroupId = "sg-0d993ae71d2a02897"



# Lançar uma instância EC2 windows

$instanceId = $(aws ec2 run-instances --image-id $amiId --count 1 --instance-type t2.medium --key-name chave --security-group-ids $securityGroupId --subnet-id $subnetId --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=Win-Cliente}]' --block-device-mappings 'DeviceName=/dev/sda1,Ebs={VolumeSize=30,DeleteOnTermination=true,VolumeType=gp3}' --query 'Instances[0].InstanceId' --output text)



# AMI

$amiId = "ami-0440d3b780d96b29d"

# ID da sua sub-rede

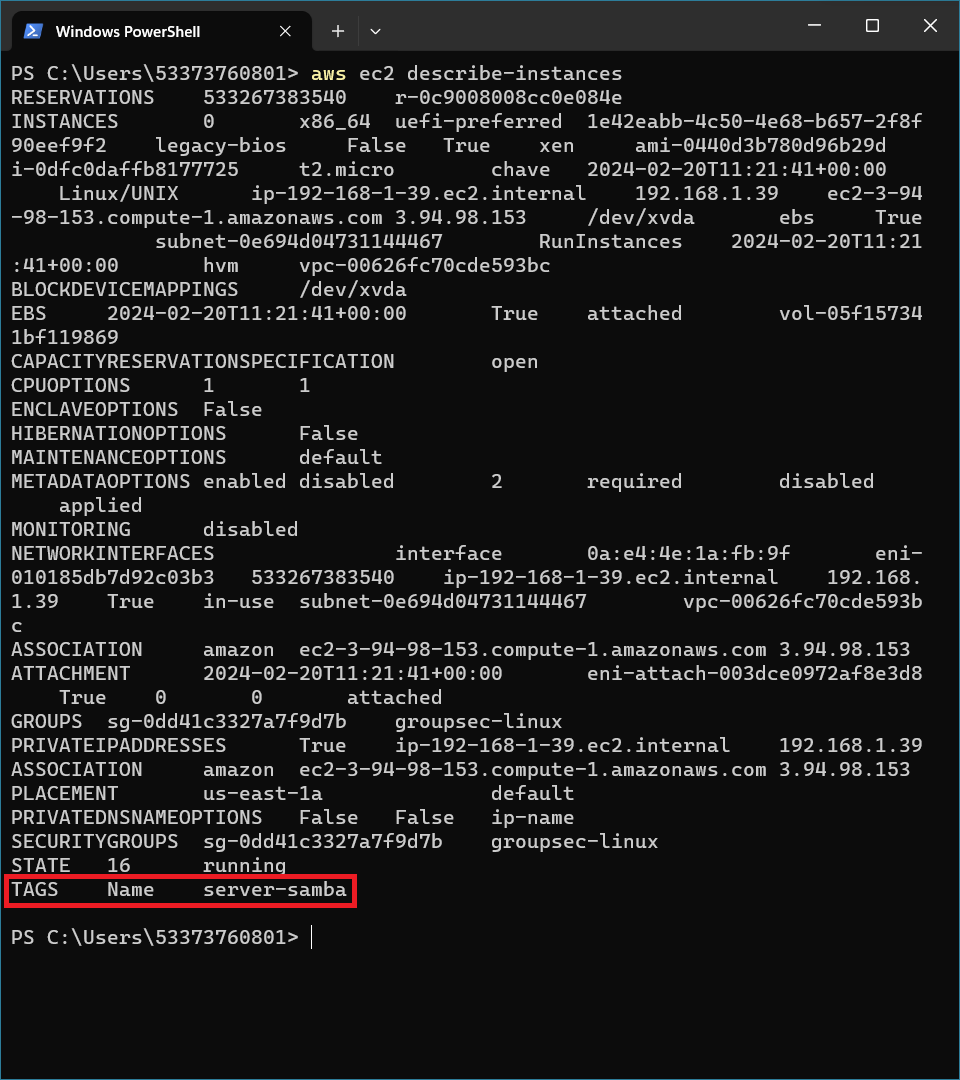
$subnetId = "subnet-0e694d04731144467"

# ID do seu grupo de segurança

$securityGroupId = "sg-0dd41c3327a7f9d7b"

# Lançar uma instância EC2 linux

$instanceId = $(aws ec2 run-instances --image-id $amiId --count 1 --instance-type t2.micro --key-name chave --security-group-ids $securityGroupId --subnet-id $subnetId --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=server-samba}]' --block-device-mappings 'DeviceName=/dev/xvda,Ebs={VolumeSize=8,DeleteOnTermination=true,VolumeType=gp3}' --associate-public-ip-address --query 'Instances[0].InstanceId' --output text)



# AMI

$amiId = "ami-0f9c44e98edf38a2b"

# ID da sua sub-rede

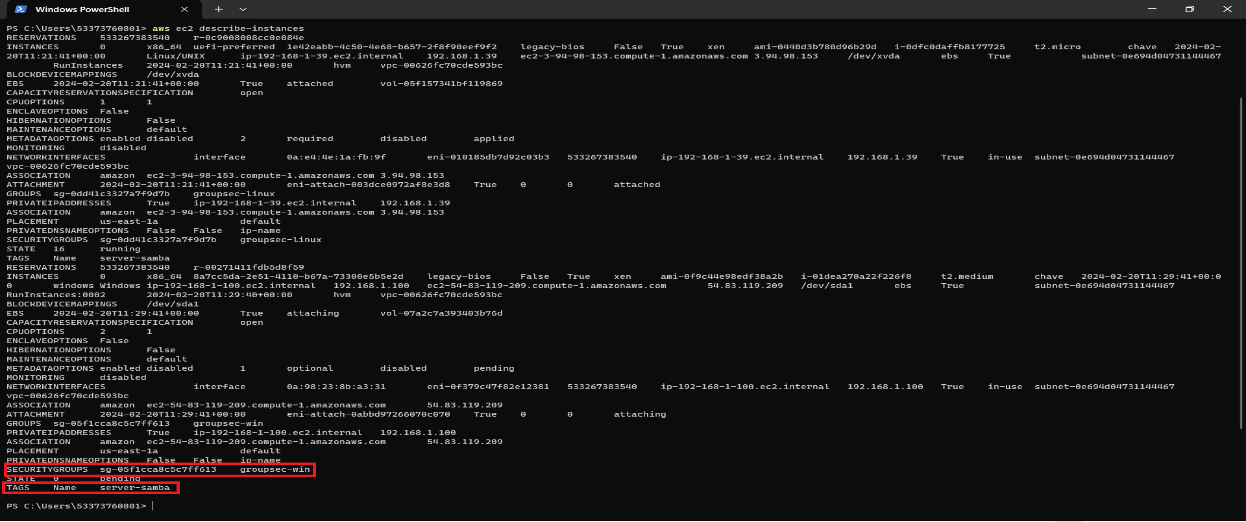
$subnetId = "subnet-0e694d04731144467"

# ID do seu grupo de segurança

$securityGroupId = "sg-05f1cca8c5c7ff613"

# Lançar uma instância EC2 windows

$instanceId = $(aws ec2 run-instances --image-id $amiId --count 1 --instance-type t2.medium --key-name chave --security-group-ids $securityGroupId --subnet-id $subnetId --tag-specifications 'ResourceType=instance,Tags=[{Key=Name,Value=server-samba}]' --block-device-mappings 'DeviceName=/dev/sda1,Ebs={VolumeSize=30,DeleteOnTermination=true,VolumeType=gp2}' --associate-public-ip-address --query 'Instances[0].InstanceId' --output text)



**EBS**

# Criar um volume EBS de 10GB na mesma zona de disponibilidade da instância

$volumeId = $(aws ec2 create-volume --size 10 --region us-east-1 --availability-zone us-east-1a --volume-type gp3 --query VolumeId --output text)

# ID da sua instância

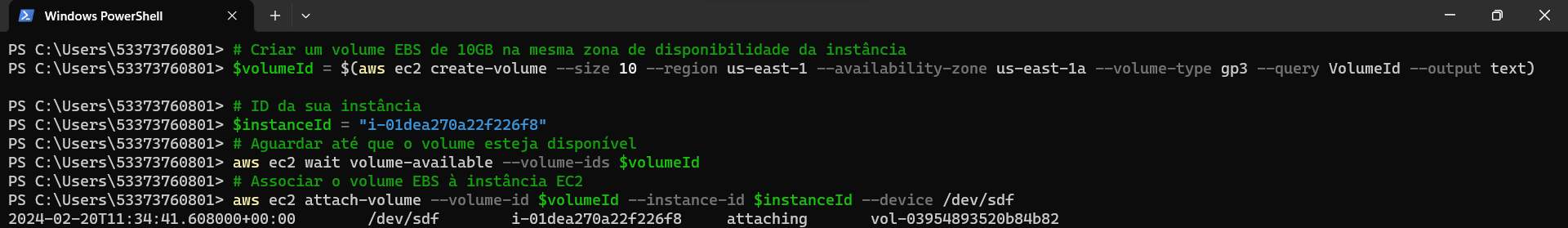
$instanceId = "i-01dea270a22f226f8"

# Aguardar até que o volume esteja disponível

aws ec2 wait volume-available --volume-ids $volumeId

# Associar o volume EBS à instância EC2

aws ec2 attach-volume --volume-id $volumeId --instance-id $instanceId --device /dev/sdf



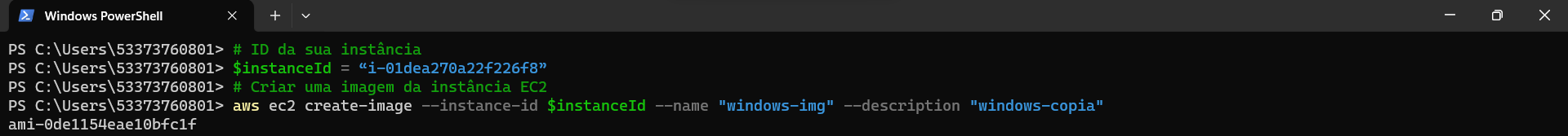
Imagem

# ID da sua instância

$instanceId = “i-01dea270a22f226f8”

# Criar uma imagem da instância EC2

aws ec2 create-image --instance-id $instanceId --name "windows-img" --description "windows-copia"

****

**Modelo de execução**

aws ec2 create-launch-template --launch-template-name my-template-for-auto-scaling --version-description version1 \

--launch-template-data '{"ImageId":"ami-04d5cc9b88example","InstanceType":"t2.micro"}'

**Grupo auto scaling**

# ID do seu modelo de execuçao

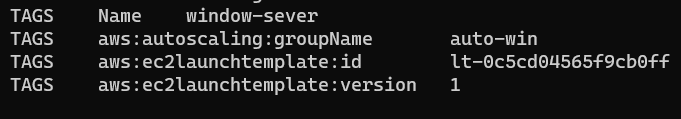
$launchTemplateId = "lt-0c5cd04565f9cb0ff"

# pelos IDs das suas zonas de disponibilidade, separados por vírgulas

$vpcZoneIdentifier = "subnet-0e694d04731144467, subnet-083bea507fa5688b7"

# Criar um grupo de Auto Scaling

aws autoscaling create-auto-scaling-group --auto-scaling-group-name "auto-win" --launch-template "LaunchTemplateId=$launchTemplateId" --min-size 2 --max-size 4 --desired-capacity 2 --vpc-zone-identifier $vpcZoneIdentifier --health-check-type EC2 --health-check-grace-period 300 --new-instances-protected-from-scale-in --tags "Key=Name,Value=window-sever "



**Grupo de destino**

# ID da sua VPC

$vpcId = "vpc-00626fc70cde593bc"

# Criar um grupo de destino

aws elbv2 create-target-group --name cloudplaydestino --protocol TCP --port 3389 --vpc-id $vpcId --target-type instance --health-check-protocol TCP

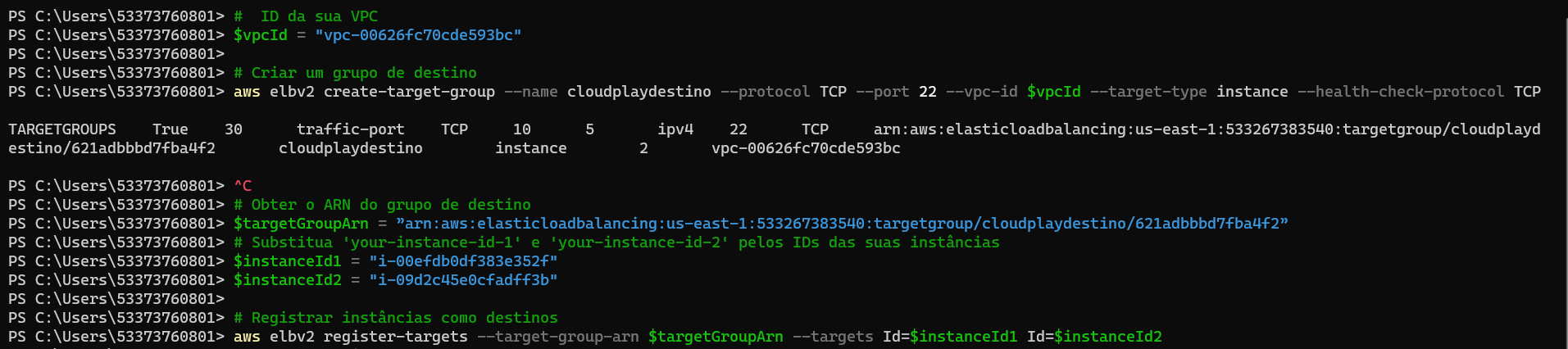
# Obter o ARN do grupo de destino

$targetGroupArn = ”arn:aws:elasticloadbalancing:us-east-1:533267383540:targetgroup/cloudplaydestino/621adbbbd7fba4f2”

# Substitua 'your-instance-id-1' e 'your-instance-id-2' pelos IDs das suas instâncias

$instanceId1 = "i-00efdb0df383e352f"

$instanceId2 = "i-09d2c45e0cfadff3b"



**Network load balancer**

# Criar o Network Load Balancer

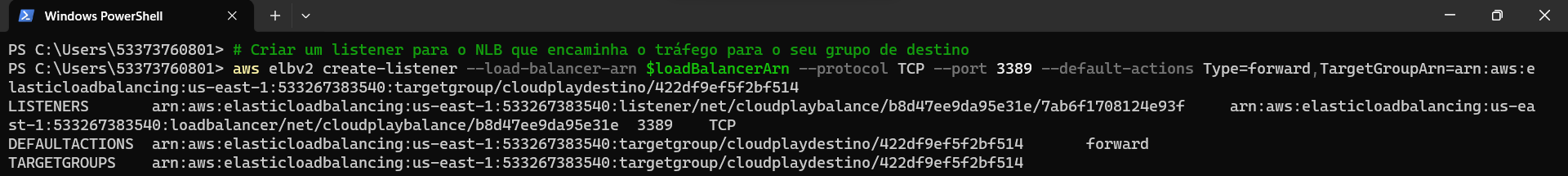
$loadBalancerArn = $(aws elbv2 create-load-balancer --name cloudplaybalance --type network --scheme internet-facing --ip-address-type ipv4 --subnets subnet-0e694d04731144467 subnet-083bea507fa5688b7 --security-groups sg-05f1cca8c5c7ff613 --query 'LoadBalancers[0].LoadBalancerArn' --output text)

# Aguardar até que o NLB esteja disponível

aws elbv2 wait load-balancer-available --load-balancer-arns $loadBalancerArn

# Criar um listener para o NLB que encaminha o tráfego para o seu grupo de destino

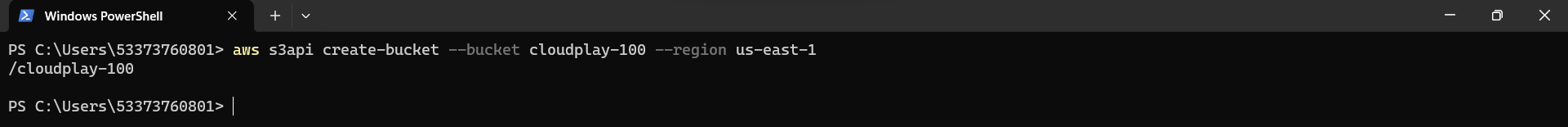
aws elbv2 create-listener --load-balancer-arn $loadBalancerArn --protocol TCP --port 3389 --default-actions Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:us-east-1:533267383540:targetgroup/cloudplaydestino/422df9ef5f2bf514

****

**S3**

criar

aws s3api create-bucket --bucket cloudplay-100 --region us-east-1



excluir

aws s3 rb s3://cloudplay-100 –force

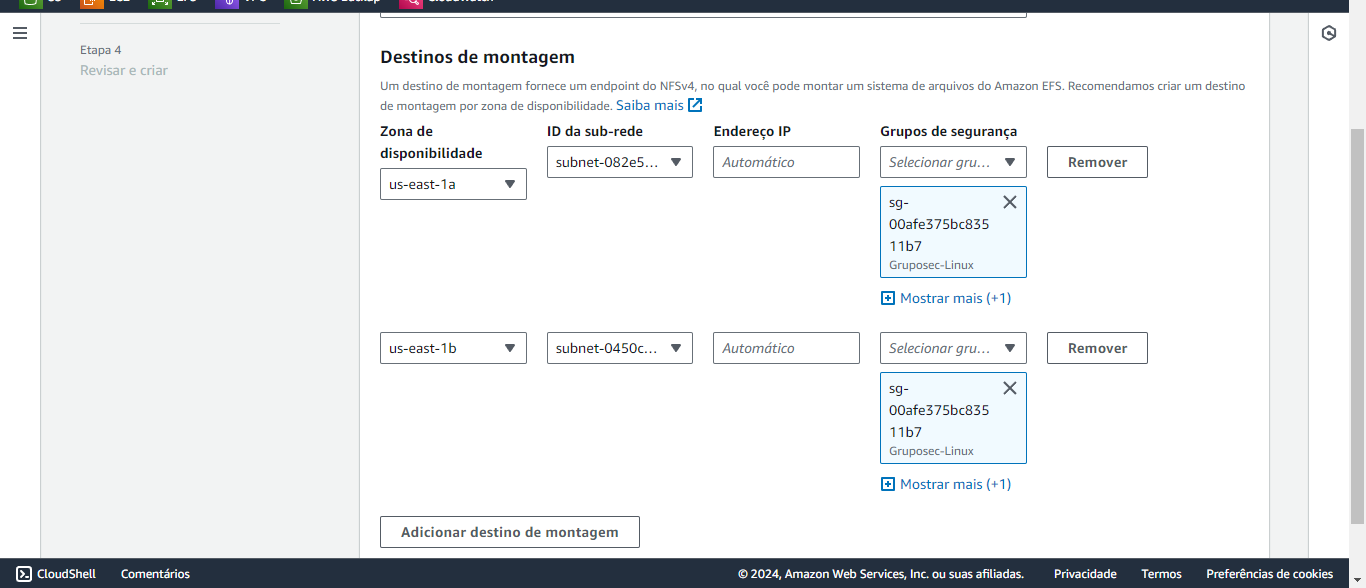


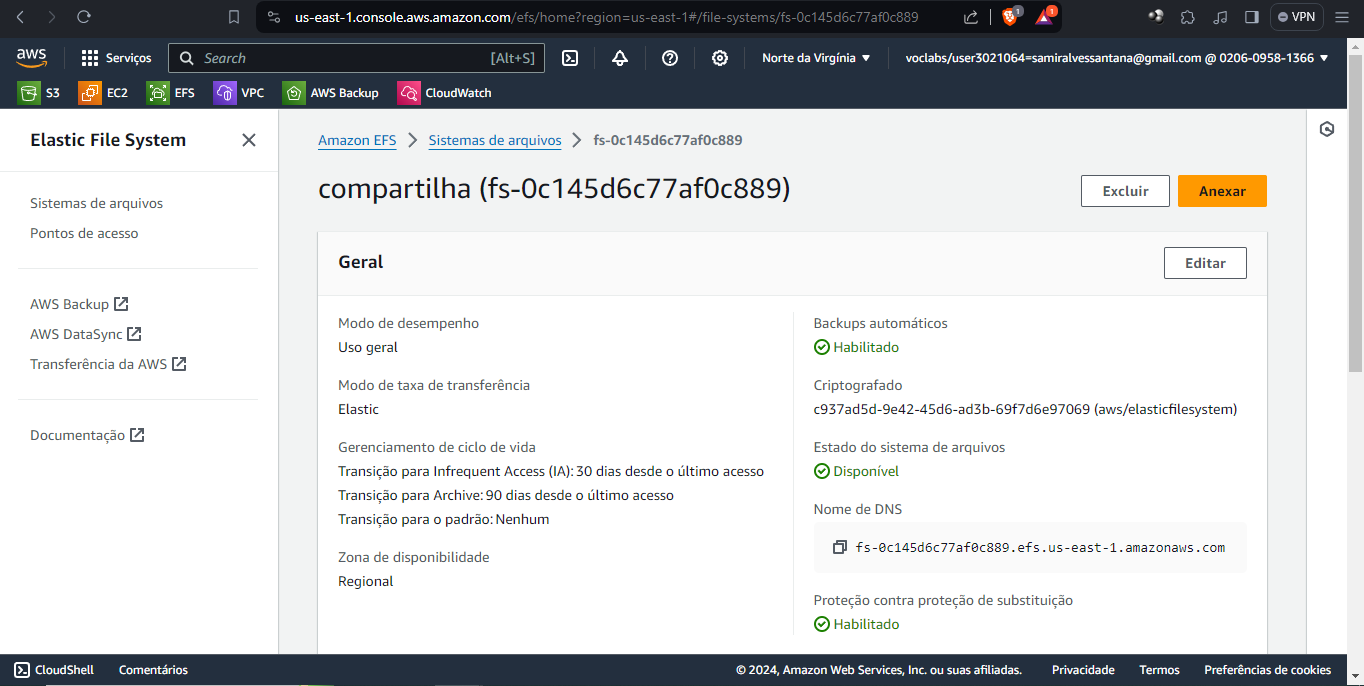
**Compartilhamento**

**Obs.: feito na AWS Academy Learner Lab [69190] pois, a AWS Academy Cloud Architecting [71296] não permite a criação de efs**

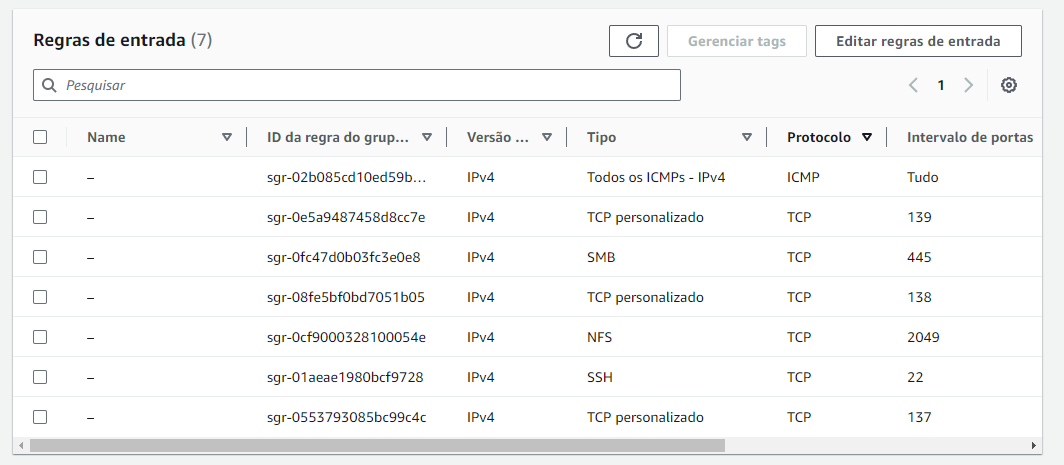
Criar o efs e coloca lo em, pelo menos, duas az







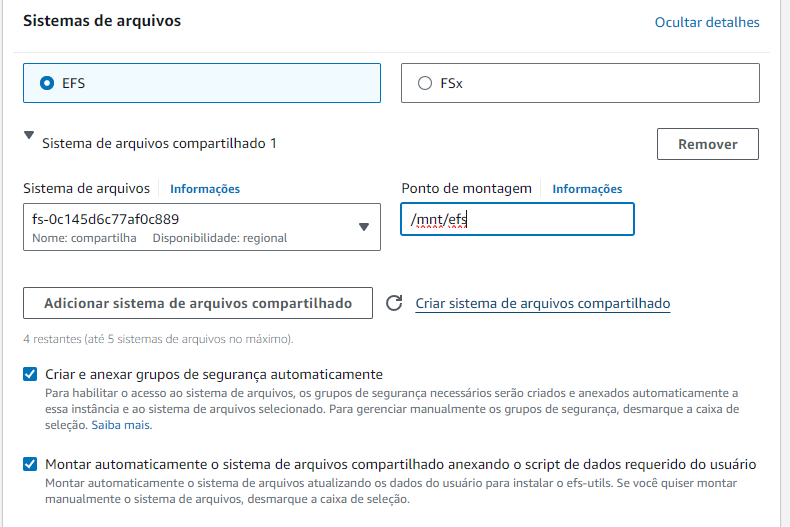
No grupo de segurança liberar as portas nfs 2049 e TCP 137, 138, 139 e 445



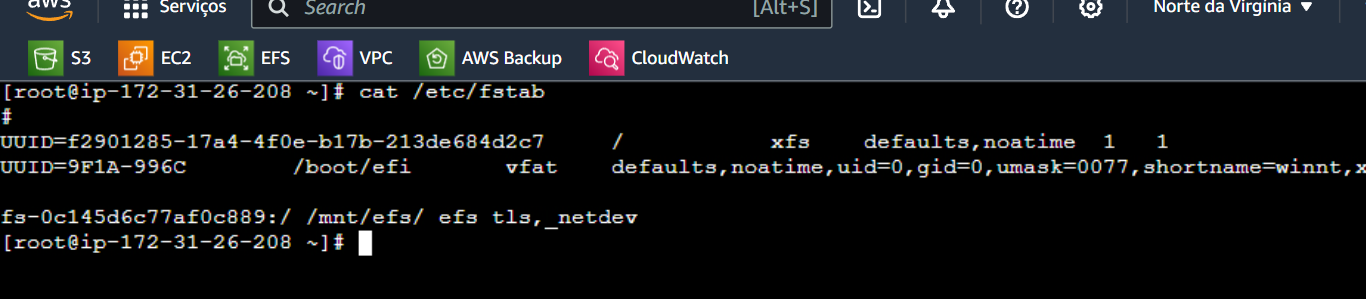


Criar instância linux amazon

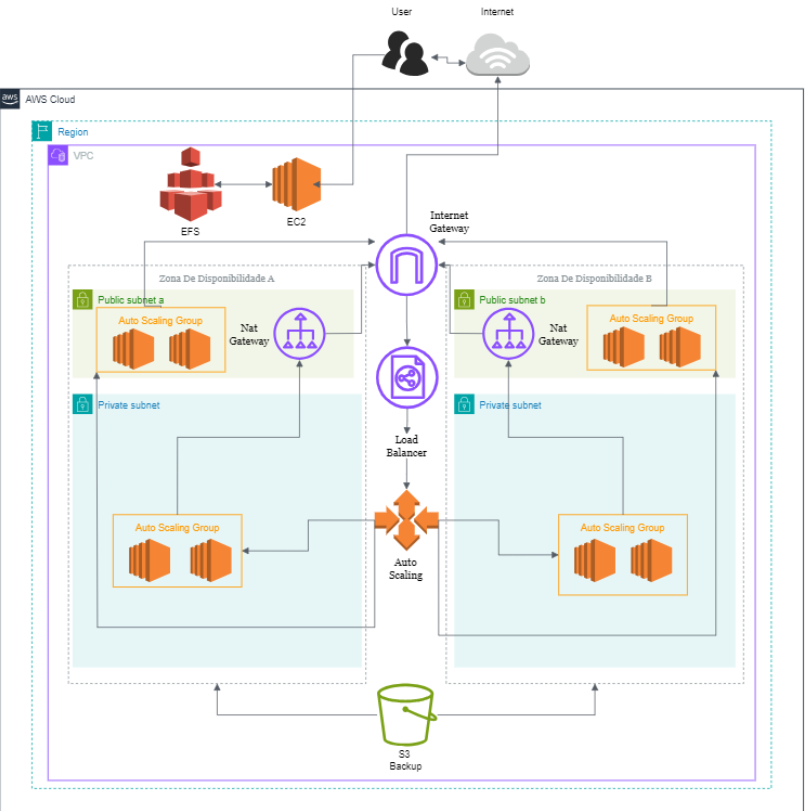
Colocar o efs para montar na inicialização



Ponto de montagem do EFS



**DIAGRAMA**

****