**AWS VPC last part:**

VPC2

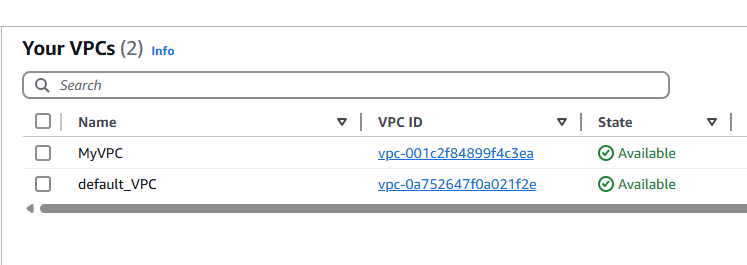
VPC1

VM2

VM1

Lets say we have two VPCs with VMs. These VPCs could be in the same AWS account or different AWS account.

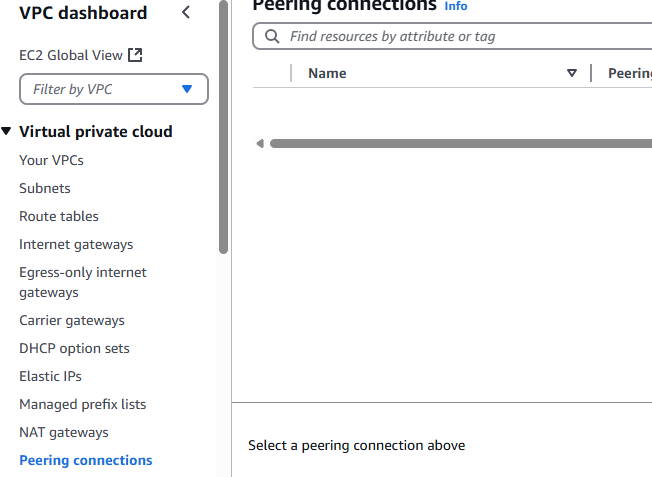
Open VPCs



We have 2 VPCs here

Click on Peering connections:

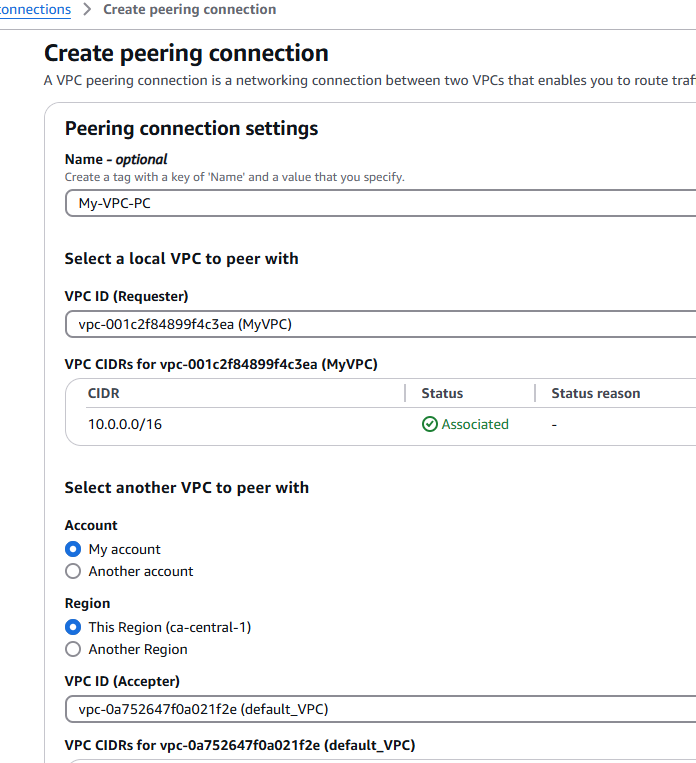
No Peering connections found

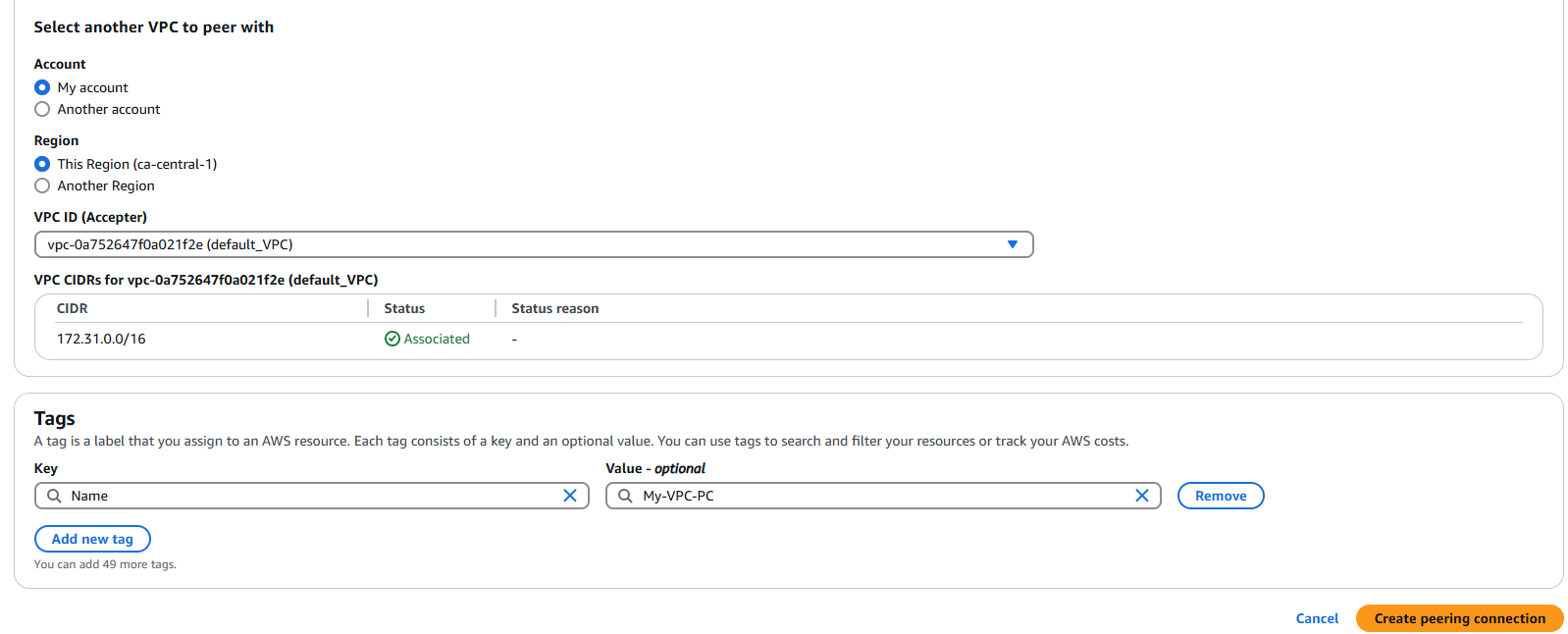


What’s the use of Peering connections?

VPC Peering is a networking connection that allows two Amazon Virtual Private Clouds (VPCs) to communicate privately as if they were in the same network.

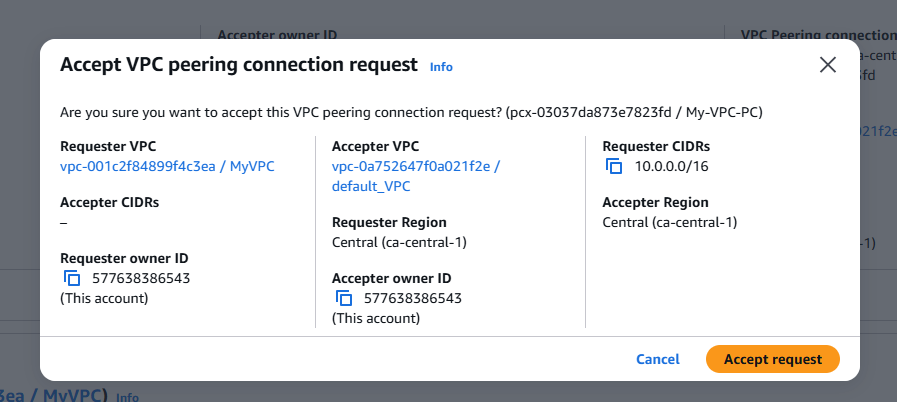
Select Requester as MyVPC and I want to connect with default-VPC

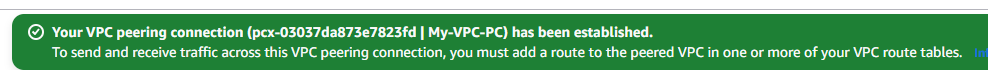




Click Create Peering connection

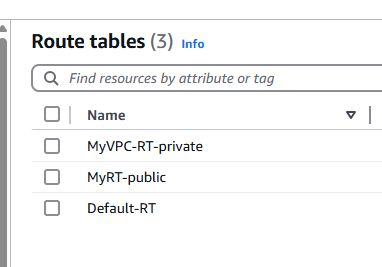
Actions ---> Accept request

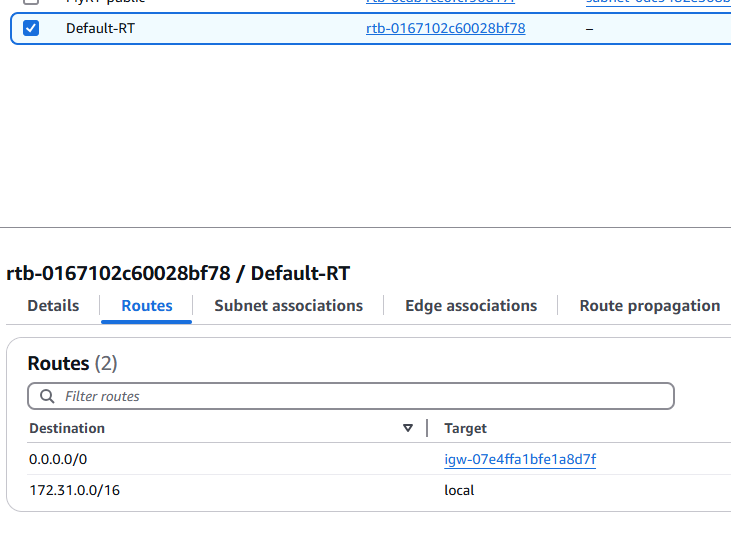




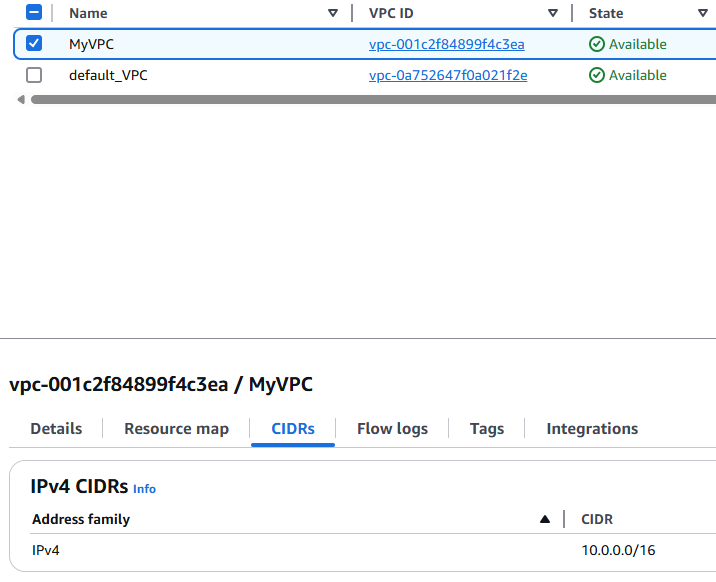
We have to go and enable in the route tables that VMs can interact with each other

Go to Route tables



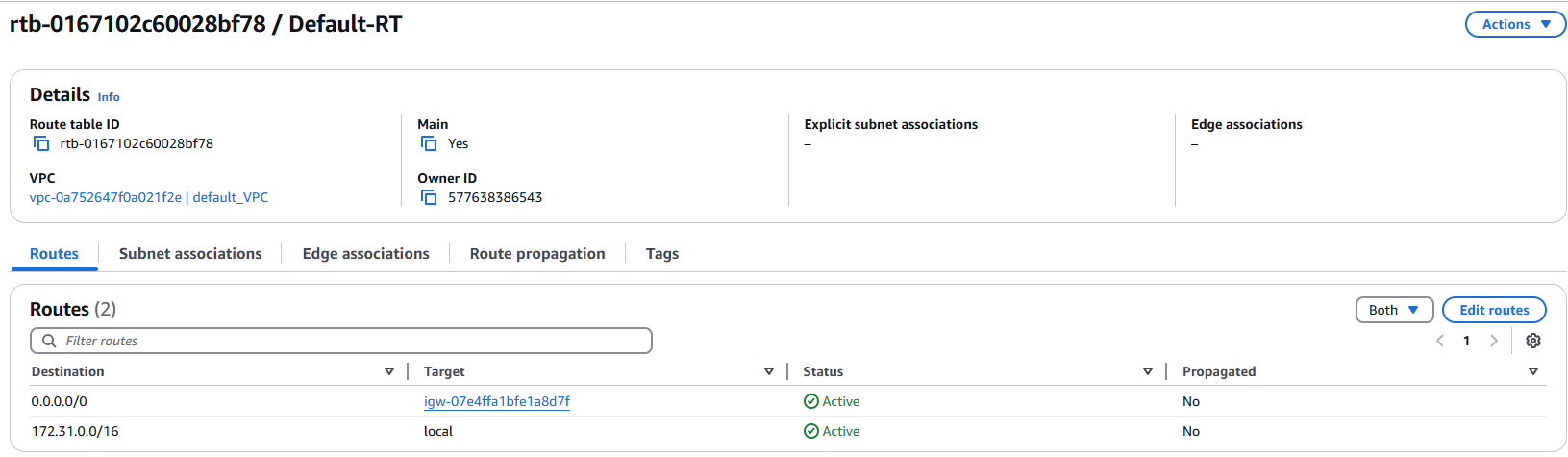


Go back to My\_VPC



Copy CIDR 10.0.0.0/16

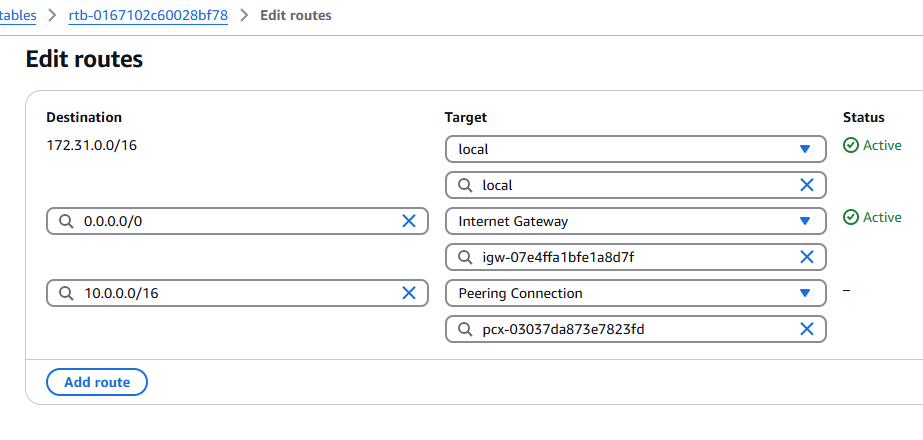
Go back to Default-RT



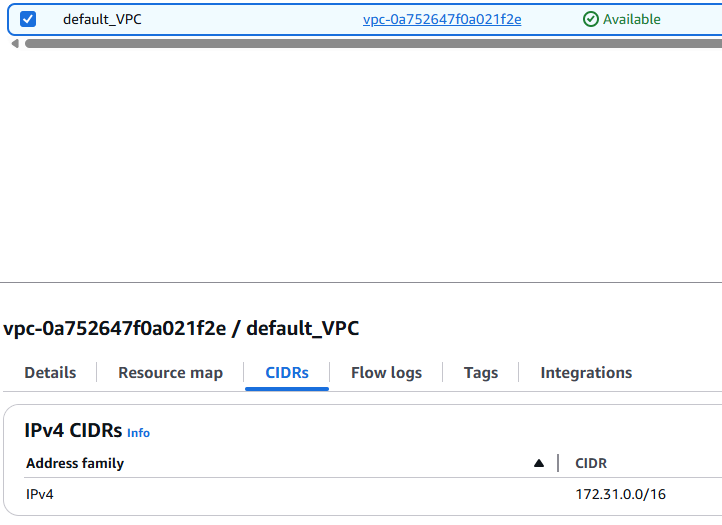
What it means is, allow requests from this IP: 10.0.0.0/16. there will be a request from MyVPC into default-VPC that’s the meaning

From this particular CIDR block there will be a request

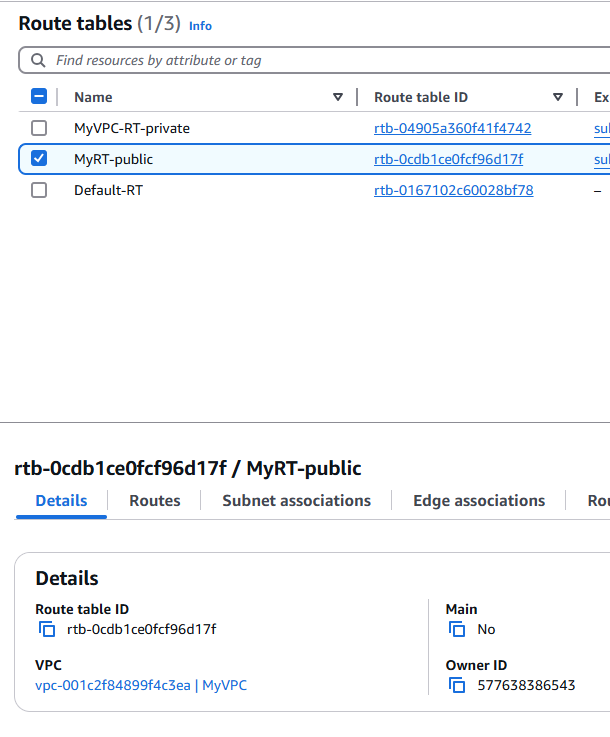
Add route



For two way, update default-VPC CIDR in My-Route table

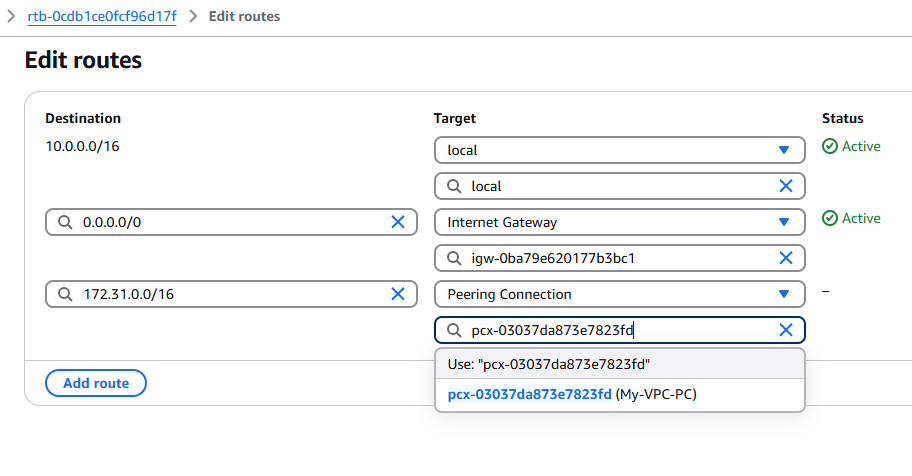


Click on public Route table

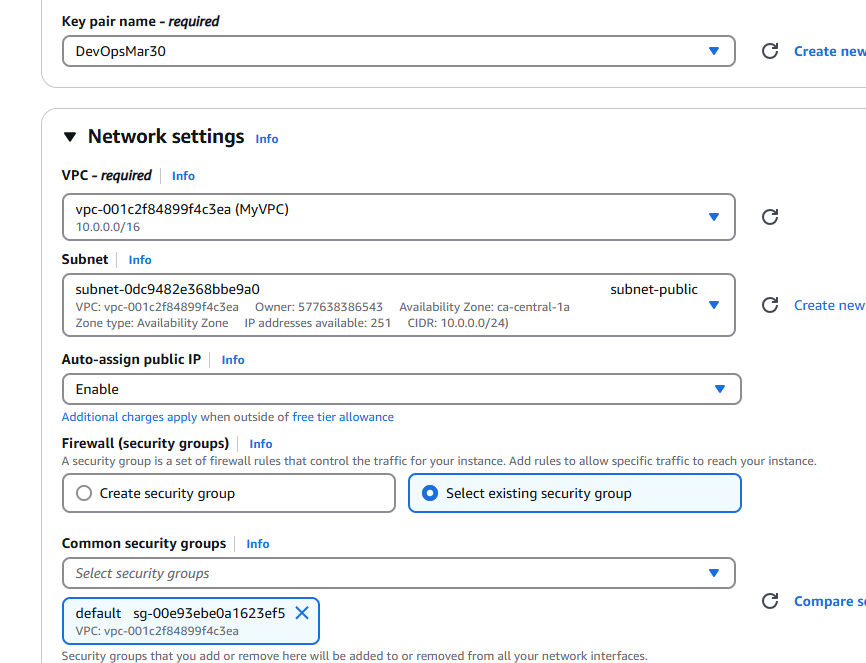


Routes ---> Edit route --> Add route

Added default-VPC CIDR into Public-RT

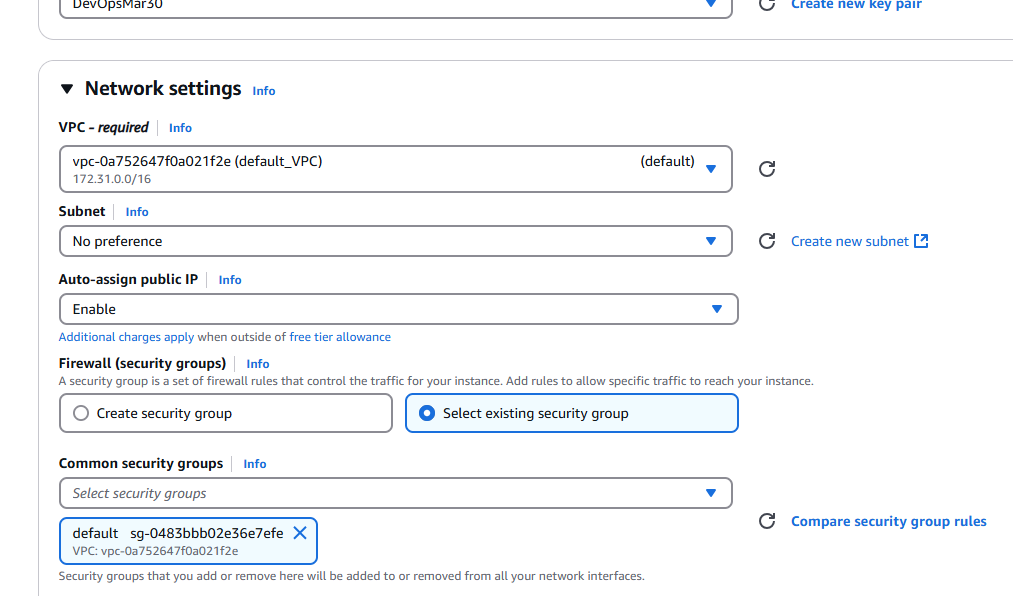


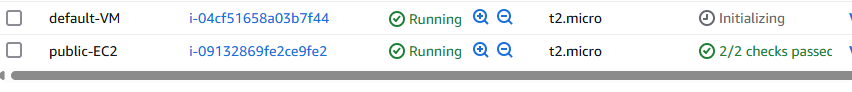
Launch an EC2 instance



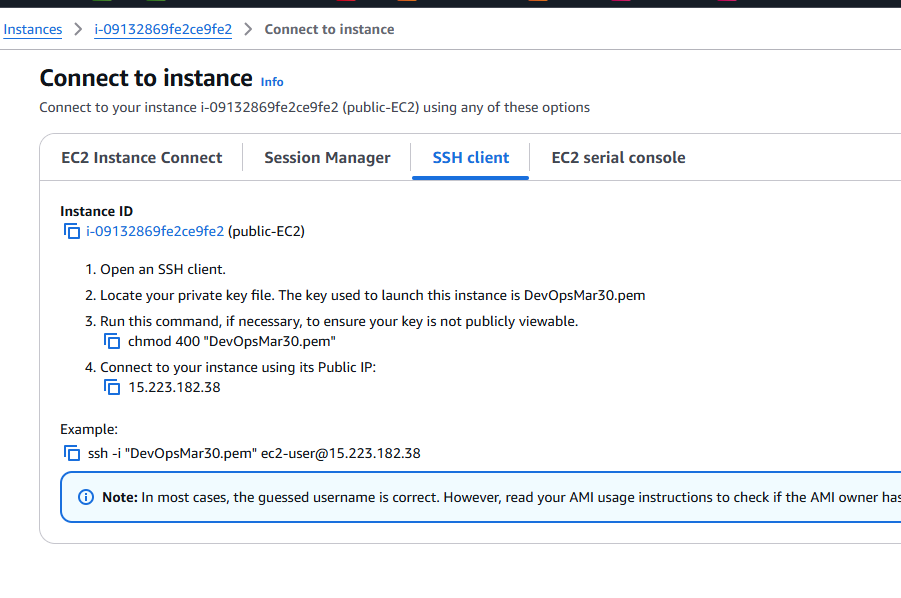
subnet-public is there

Create another VM into default-VPC

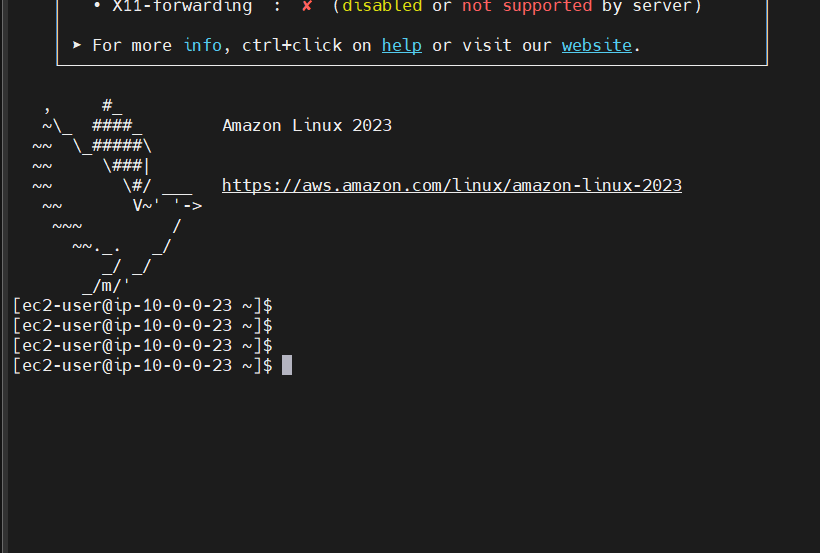




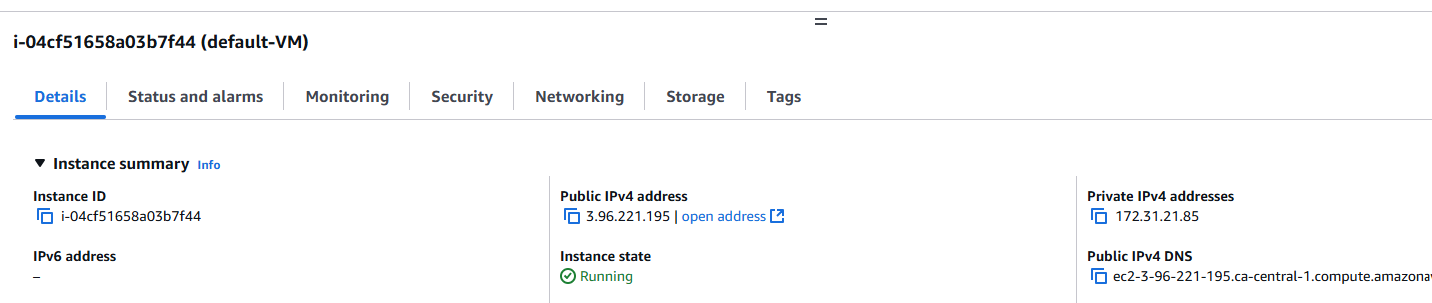
Click on Connect in public-EC2



Connect to public-EC2



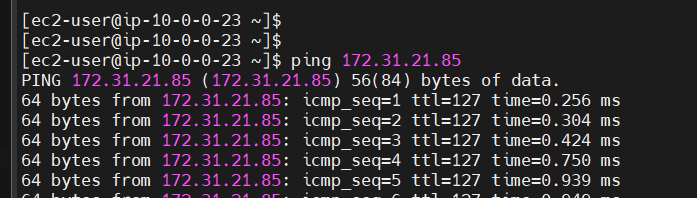
Copy Private IPv4 address from default-VM



Private IP: 172.31.21.85

Connect from Public-EC2 to default-VM

**Make sure in the Security groups All Traffic is enabled**



In this example, both VPCs are in the same account

Yes, we can establish connection between two VPCs in different accounts as well. This is called as VPC peering

Security Groups:

Why do we have security groups?

Acts like a firewall to secure our resources, decides incoming traffic and outgoing traffic

SG contains two important parts -> inbound rules and outbound rules

Inbound rules: incoming traffic to your VM

Outgoing rules: outgoing traffic from your VM

In One Security group, we can add 50 rules

By default, all traffic is denied in security groups, unless and until you make the request for what kind of traffic is allowed etc

--> Security group only allows rules, by default all rules are denied

--> We cannot configure deny rules in security group

--> Security groups are applicable at resource level (not at subnet or VPC) and manually we have to add Security group to a resource

--> Security groups are stateful (any changes we apply to the incoming rules it is applicable to outgoing rules also)

NACL: Network Access Control List / Network ACL

NACL acts as a Firewall for our subnets in VPC

It is applicable at the subnet level

NACL rules are applicable resources, which are part of a subnet

NACL is Stateless (Any rule for the incoming traffic must be manually added to outgoing traffic)

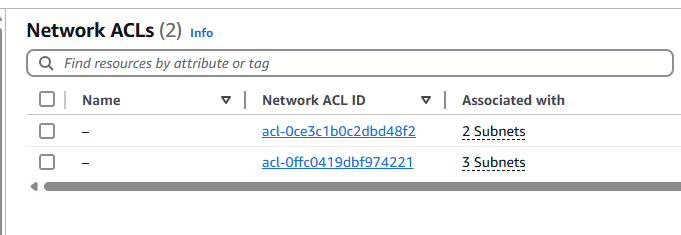
In NACL, we can configure both allow and deny rules

One Subnet can have only one NACL

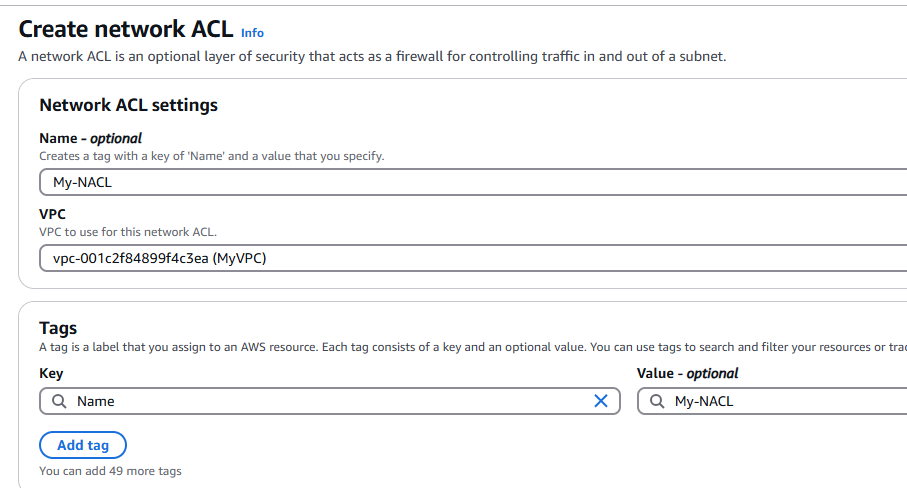
However, one NACL can be added to multiple subnets

NACL acts as first level of defense for incoming traffic while security group acts as first level of defense for outgoing traffic

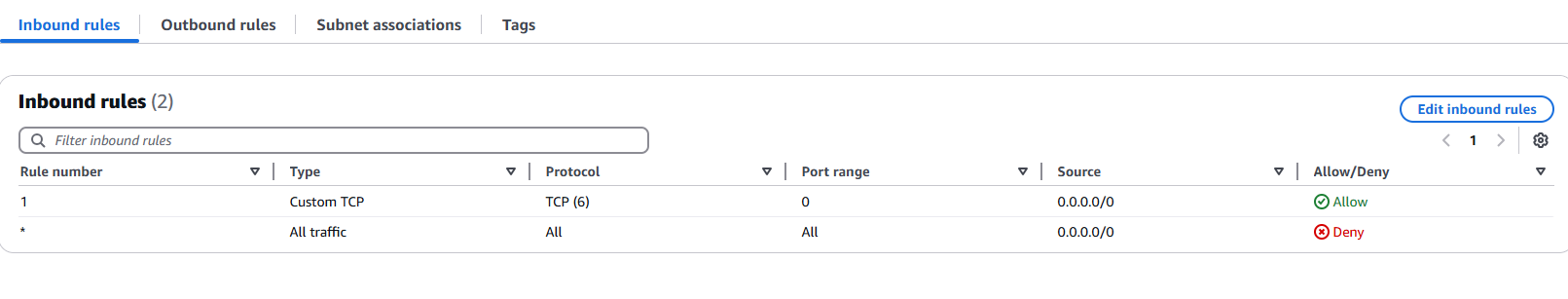
We have Subnet first then VM inside, so first level of defense for incoming is NACL but for outgoing is Security group

In VPC, we can find NACL  


By default, 2 NACLs are created. We are using only 2 NACLs for all the subnets. Multiple NACLs cannot be used in one subnet but vice-versa is possible.

Create NACL  


In NACL, we have both options: Allow and Deny



**Elastic File System (EFS):**

EBS --> Elastic Block Store

S3 --> unlimited storage

EFS --> File system storage --> shared that means this particular file can work with multiple resources

EC2 VM2

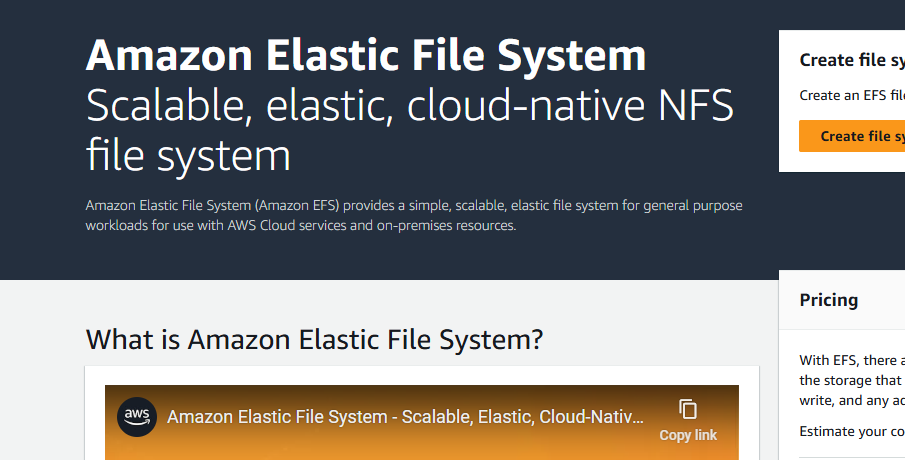
EC2 VM1

EBS

EBS

File on EFS

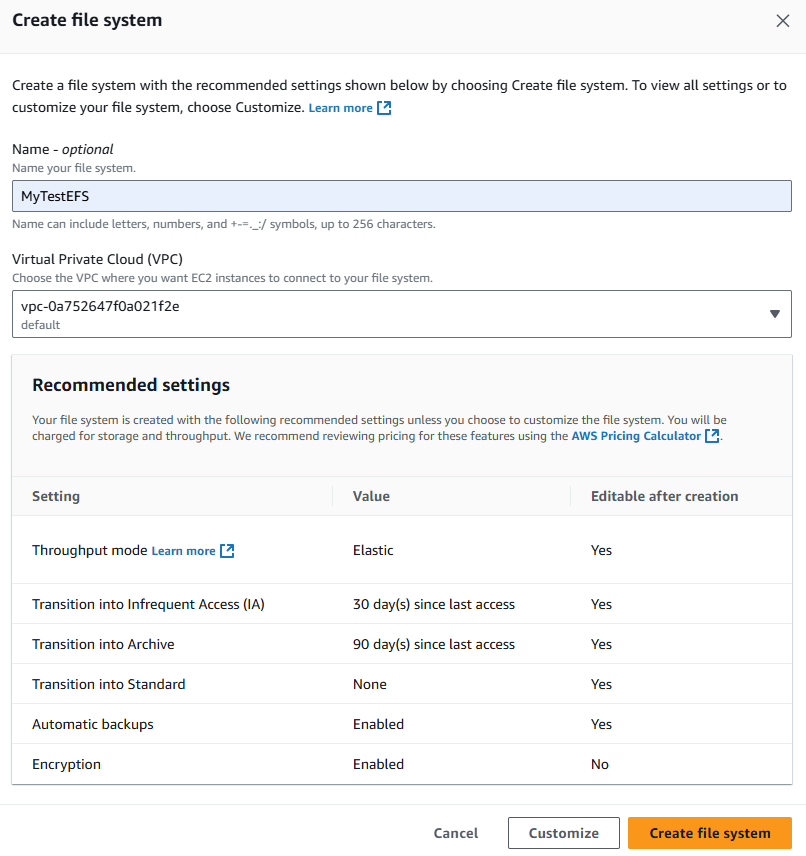
This file on EFS can be accessed by multiple resources at the same time



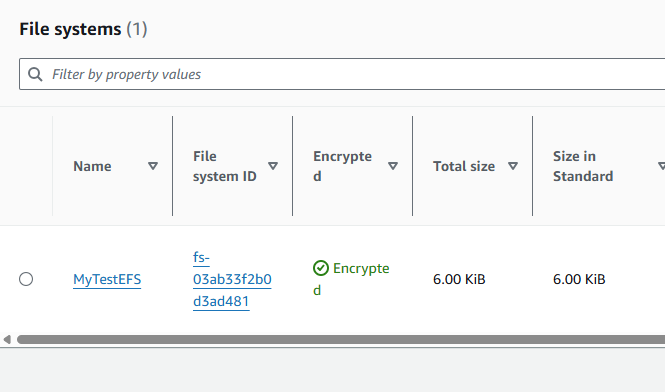
Advantage:

Create scalable file storage to be used on EC2

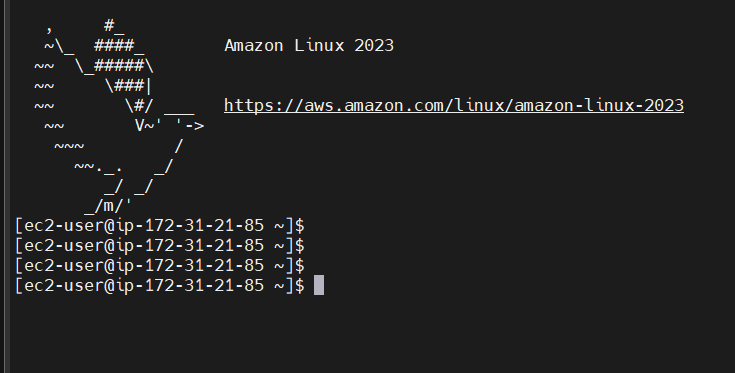
Fully managed by AWS, Low cost, pay for what you use, highly available and scalable performance



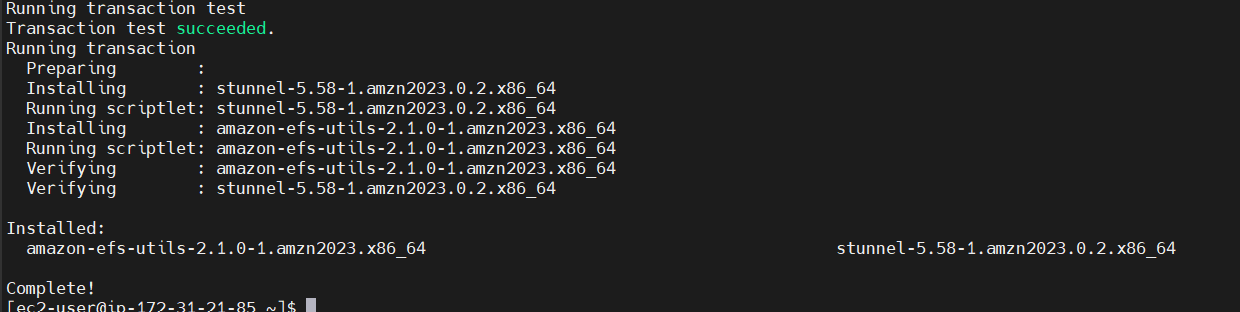
File ID that’s required

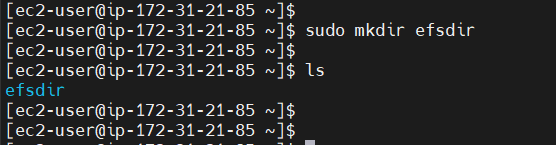


I am using the defaultVM



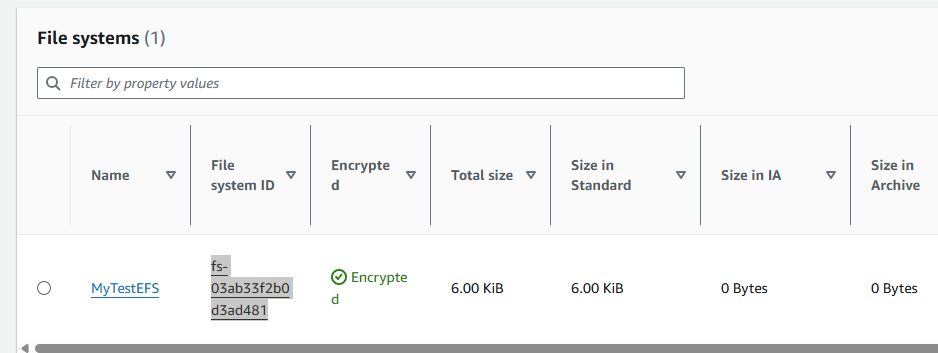
[ec2-user@ip-172-31-21-85 ~]$ sudo yum install -y amazon-efs-utils





Command for mounting

Copy the EFS file system ID



Attach EFS File system ID to VM

[ec2-user@ip-172-31-21-85 ~]$

[ec2-user@ip-172-31-21-85 ~]$ sudo mount -t efs -o tls fs-03ab33f2b0d3ad481:/ efsdir

[ec2-user@ip-172-31-21-85 ~]$ sudo mount -t efs -o tls fs-03ab33f2b0d3ad481:/ efsdir

[ec2-user@ip-172-31-21-85 ~]$ cd efsdir/

[ec2-user@ip-172-31-21-85 efsdir]$ ls

[ec2-user@ip-172-31-21-85 efsdir]$ sudo touch demo.txt

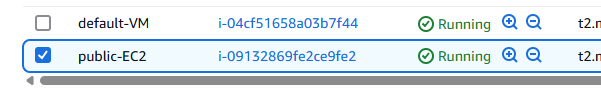
[ec2-user@ip-172-31-21-85 efsdir]$ ls -l

total 4

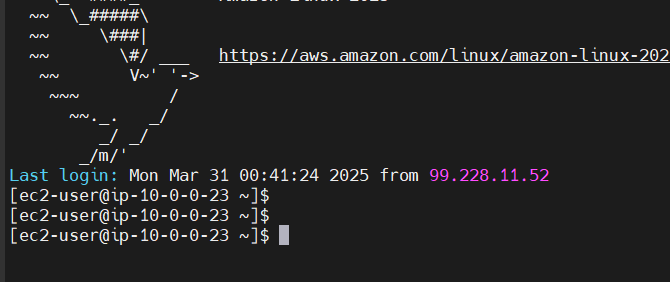
-rw-r--r--. 1 root root 0 Mar 31 03:10 demo.txt

We have attached EFS to default-VM

Now opening public-EC2



Open public-EC2 in a different terminal window



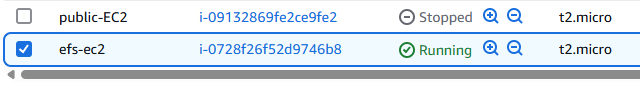
[ec2-user@ip-10-0-0-23 ~]$ sudo yum install -y amazon-efs-utils

[ec2-user@ip-10-0-0-23 ~]$ sudo mount -t efs -o tls fs-03ab33f2b0d3ad481:/ efsdir

Failed to resolve "fs-03ab33f2b0d3ad481.efs.ca-central-1.amazonaws.com" - check that your file system ID is correct, and ensure that the VPC has an EFS mount target for this file system ID.

Your **EC2 instance and EFS must be in the same VPC**.

Lets create one more EC2 in the same VPC



[ec2-user@ip-172-31-24-90 ~]$

[ec2-user@ip-172-31-24-90 ~]$ sudo yum install -y amazon-efs-utils

We mount EFS once again on this new EC2

[ec2-user@ip-172-31-24-90 ~]$ sudo mkdir efsdir

[ec2-user@ip-172-31-24-90 ~]$ sudo mount -t efs -o tls fs-03ab33f2b0d3ad481:/ efsdir

[ec2-user@ip-172-31-24-90 ~]$ cd efsdir

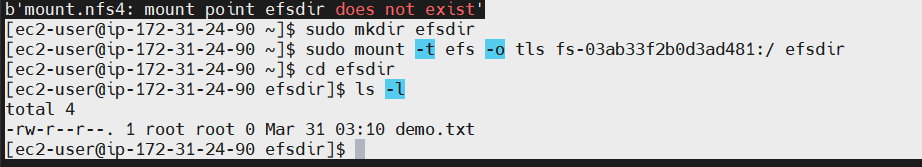
[ec2-user@ip-172-31-24-90 efsdir]$ ls -l

total 4

-rw-r--r--. 1 root root 0 Mar 31 03:10 demo.txt

[ec2-user@ip-172-31-24-90 efsdir]$

We see the same demo.txt in this efsdir



[ec2-user@ip-172-31-24-90 efsdir]$ sudo touch newFile.txt

[ec2-user@ip-172-31-24-90 efsdir]$ ls -l

total 8

-rw-r--r--. 1 root root 0 Mar 31 03:10 demo.txt

-rw-r--r--. 1 root root 0 Mar 31 03:26 newFile.txt

We have two files on efs-ec2 new VM

Go to the other EC2:

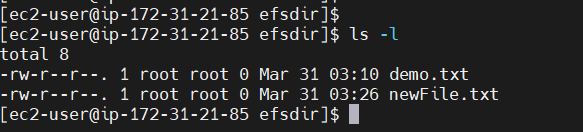
[ec2-user@ip-172-31-21-85 efsdir]$ ls -l

total 8

-rw-r--r--. 1 root root 0 Mar 31 03:10 demo.txt

-rw-r--r--. 1 root root 0 Mar 31 03:26 newFile.txt

We can see newFile.txt in the default-VM, which we created actually in efs-ec2



Steps to work with EFS practicals:

1. Login into AWS console ---> Services ---> EFS (it is in Storage also) --> Create FileSystem
2. Get File System ID
3. Create EC2 instances --> two different instances
4. Login into EC2 instance using Moba
5. Sudo yum install -y amazon-efs-utils
6. Create a folder/directory
7. sudo mkdir efsdir
8. Mounting File system:
9. sudo mount -t efs -o tls <file-id> :/ efsdir
10. cd efsdir
11. Create files in the directory
12. Then Connect to the second EC2 instance
13. Repeat the same steps
14. Check behavior of shared file system check efsdir