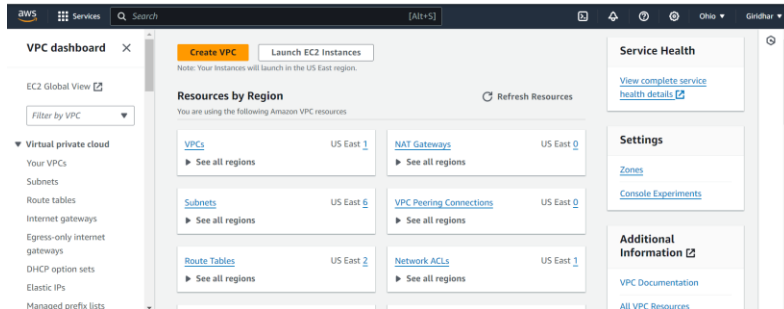


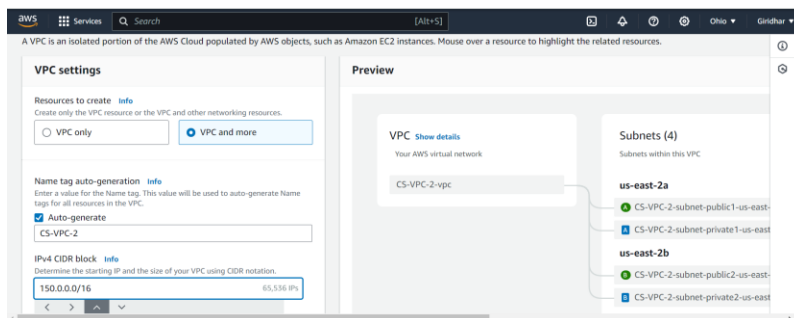
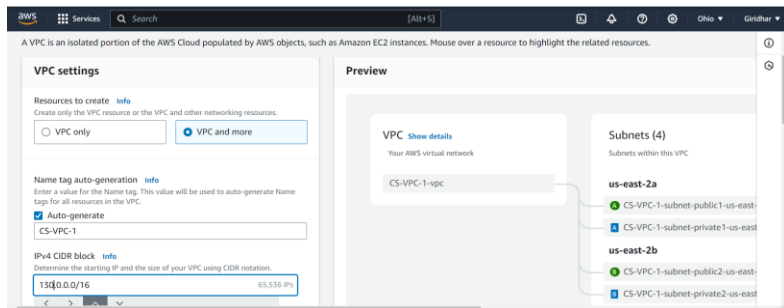
Three VPCs using Transit Gateway;

Step1 - creating VPCs

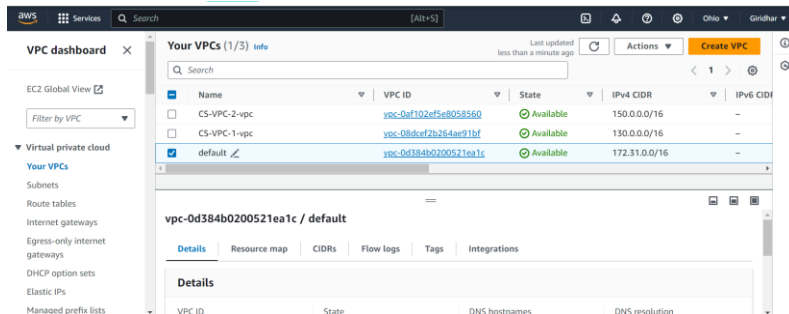
Go to vpc and click on create vpc



Create two vpc by selecting option vpc and more, so system only will create subnets and they will associate to route tables and also it will create igw and attaches to vpcs.

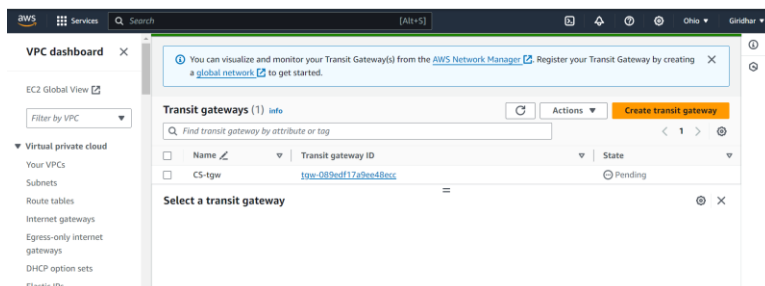
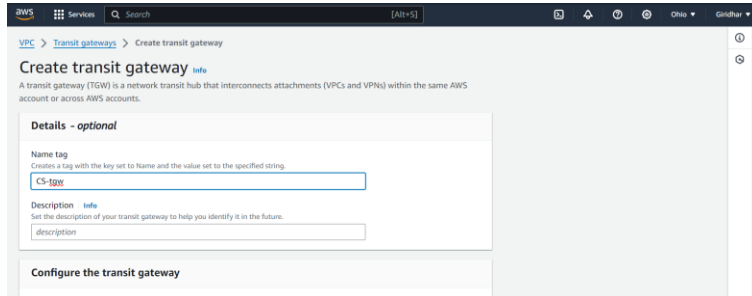
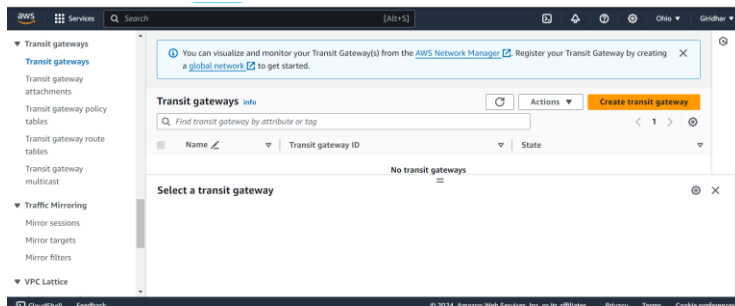


And we have default vpc, total three VPCs are available

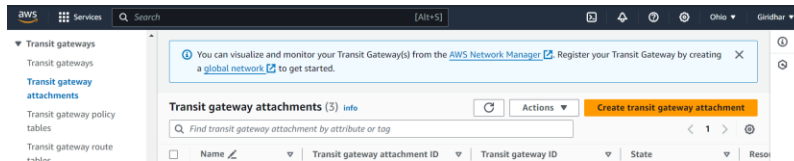


Step2 – creating transit gateway

Go to Transit Gateways and create transit gateway (CS-tgw)

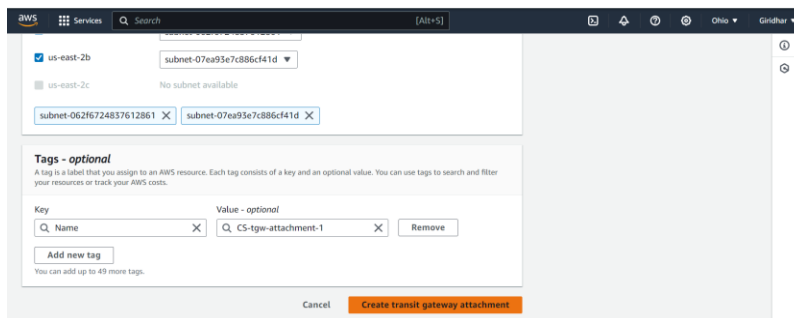
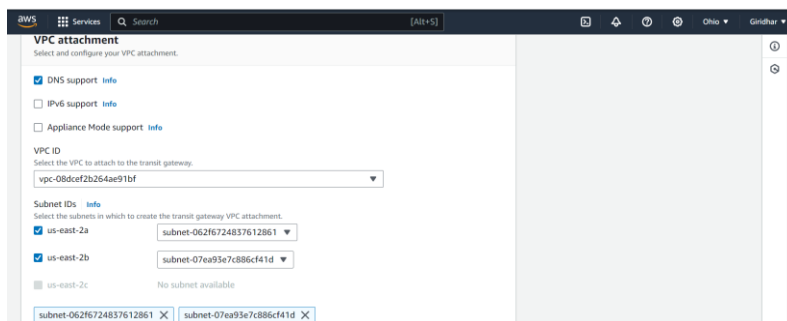
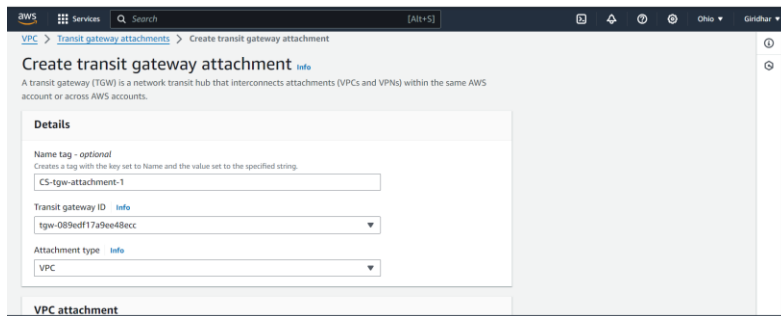


Step3 -create transit gateway attachments for three VPCs one by one



create transit gateway attachment one (CS-tgw-attachment-1) by selecting transit gateway id(CS-tgw), which we created before and

select attachment type is VPC and select VPC id any one of three vpcs (CS-VPC-1) we created earlier and select subnets



Now create transit gateway attachments for another two vpcs follow the same process

Create tgw attachment (CS-tgw-attachment-2) for (CS-VPC-2)

Create transit gateway attachment [Info](#)

A transit gateway (TGW) is a network transit hub that interconnects attachments (VPCs and VPNs) within the same AWS account or across AWS accounts.

Details

Name tag - optional
Creates a tag with the key set to Name and the value set to the specified string.

CS-tgw-attachment-2

Transit gateway ID [Info](#)

tgw-089edf17a9ee48ecc

Attachment type [Info](#)

VPC

VPC attachment

VPC attachment

Select and configure your VPC attachment.

☒ **DNS support** [Info](#)

☐ **IPv6 support** [Info](#)

☐ **Appliance Mode support** [Info](#)

VPC ID
Select the VPC to attach to the transit gateway.
vpc-0af102ef5e8058560

Subnet IDs [Info](#)
Select the subnets in which to create the transit gateway VPC attachment.

☒ us-east-2a

subnet-0f51f1dcf0fe0793d

☒ us-east-2b

subnet-053bad9109e3c368e

☐ us-east-2c

No subnet available

subnet-0f51f1dcf0fe0793d X subnet-053bad9109e3c368e X

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key **Value - optional**

You can add up to 49 more tags.

Create tgw attachment (CS-tgw-attachment-3) for default vpc

Create transit gateway attachment [Info](#)

A transit gateway (TGW) is a network transit hub that interconnects attachments (VPCs and VPNs) within the same AWS account or across AWS accounts.

Details

Name tag - optional
Creates a tag with the key set to Name and the value set to the specified string.

CS-tgw-attachment-3

Transit gateway ID [Info](#)

tgw-089edf17a9ee48ecc

Attachment type [Info](#)

VPC

VPC attachment

VPC attachment

Select and configure your VPC attachment.

☒ **DNS support** [Info](#)

☐ **IPv6 support** [Info](#)

☐ **Appliance Mode support** [Info](#)

VPC ID
Select the VPC to attach to the transit gateway.
vpc-0d384b0200521ea1c

Subnet IDs [Info](#)
Select the subnets in which to create the transit gateway VPC attachment.

☒ us-east-2a

subnet-01137ae8602ea409b

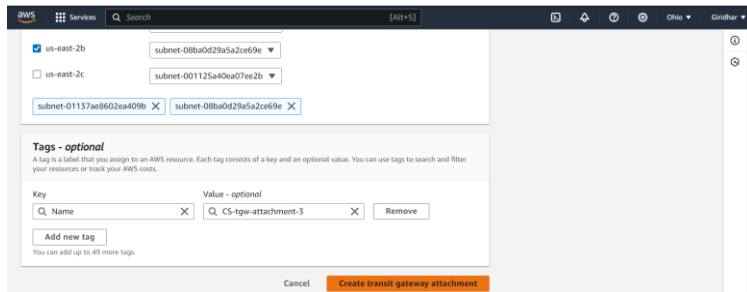
☒ us-east-2b

subnet-08ba0d29a5a2ce69e

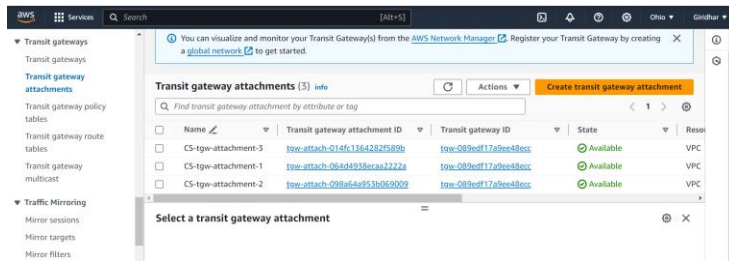
☐ us-east-2c

subnet-001125a40ea07ee2b

subnet-01137ae8602ea409b X subnet-08ba0d29a5a2ce69e X

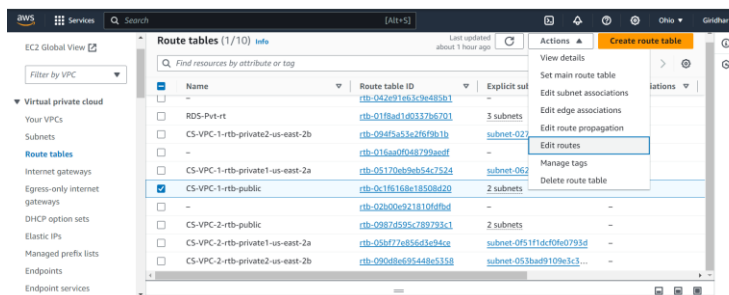
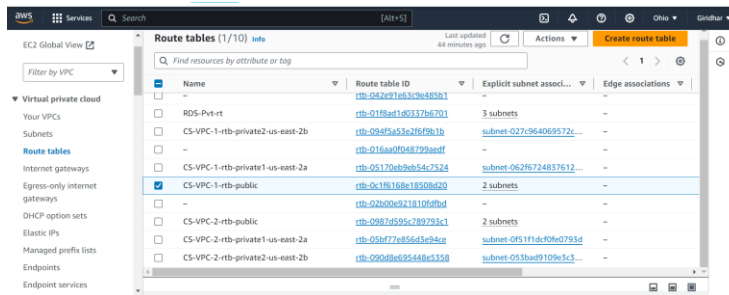


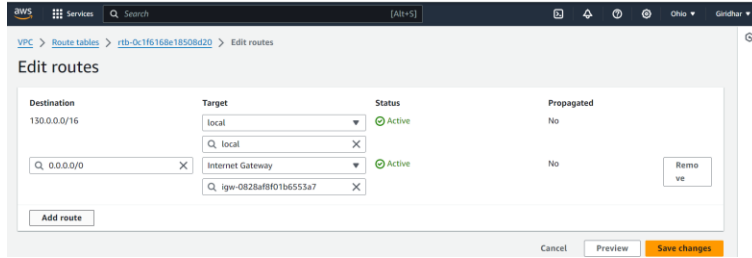
Now the three tgw attachments are available



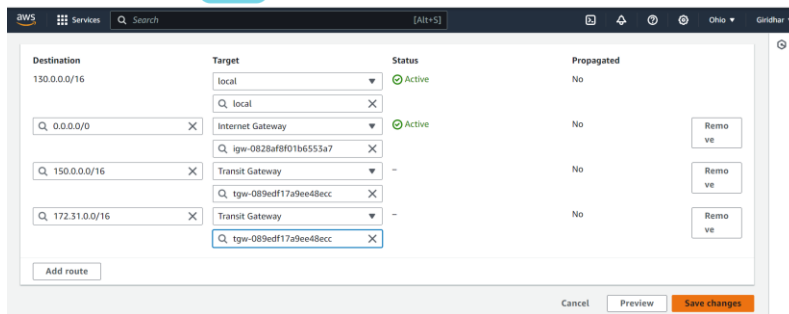
Step4 – Route table

Go to route tables and select CS-VPC-1 public and click on actions --> edit route --> add route

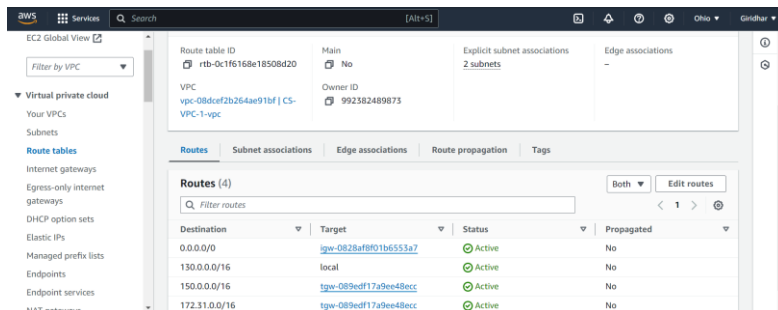




Now there will be CS-VPC-1 IP address 130.0.0.0/16 as destination, we need to add the other two vpcs ip address in destination and in target select transit gateway and select attachment (CS-tgw-attachment-1), click on save changes.

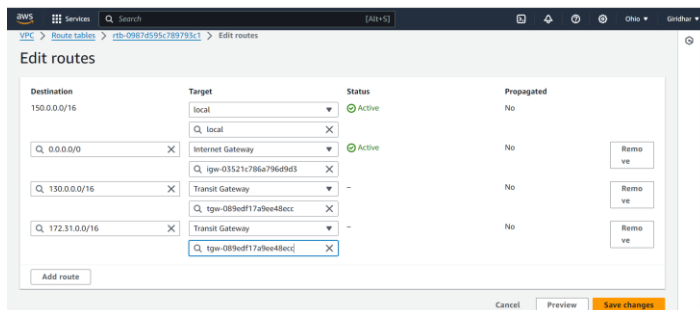
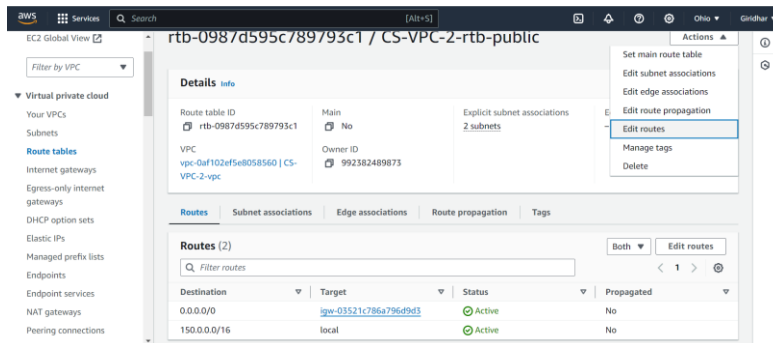
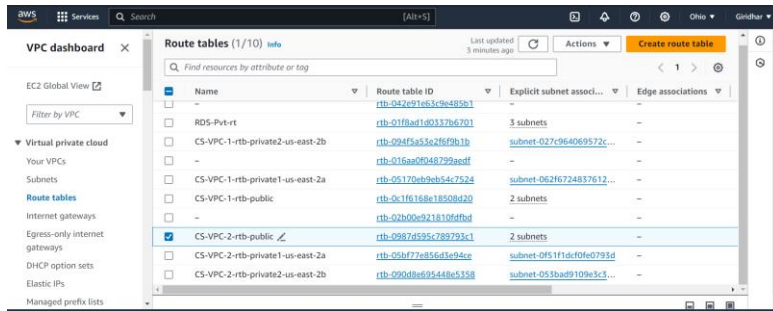


We can see that, added two ip address are active for CS-VPC-1

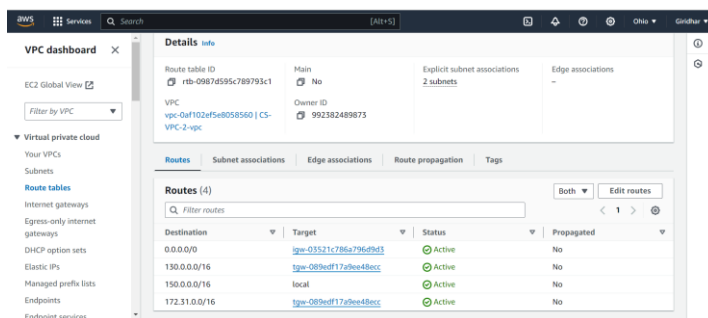


Same process for remaining two vpcs, we should add other two ip address which are not in destination

Go to CS-VPC-2 public and repeat the process of adding ip addressess



Now added two ip address are active for CS-VPC-2



And also, for Default vpc, follow same process of adding ip's

The first screenshot shows the 'Route tables (1/10)' list in the AWS VPC console. The table lists various route tables, including 'rtb-042e91e63c9e485b1' which is highlighted. The second screenshot shows the 'Details' page for 'rtb-042e91e63c9e485b1'. It shows the route table is the main table for VPC 'vpc-d638460200521ea7c'. The 'Routes' tab shows two routes: one for destination '0.0.0.0/0' targeting 'igw-0ee7fa4ef2c88cd88' (Active), and another for '172.31.0.0/16' targeting 'local' (Active). The third screenshot shows the 'Edit routes' page for the same route table. It displays the existing routes and allows adding new ones. The 'Add route' button is visible at the bottom.

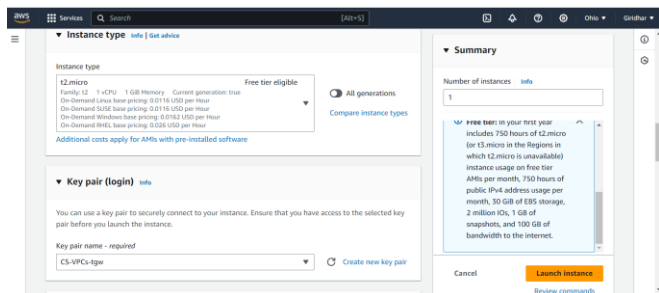
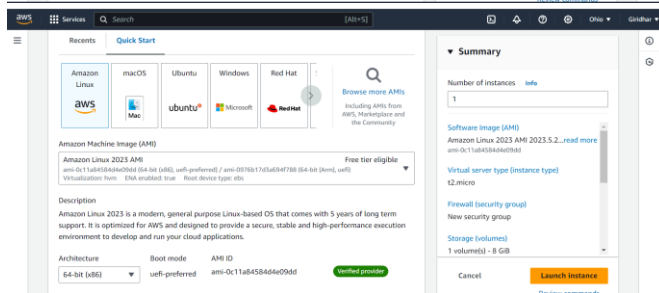
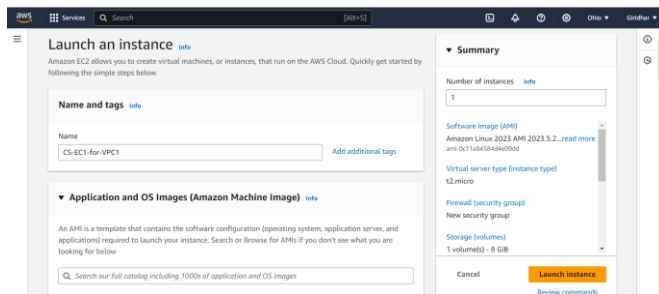
added two ip address are active for default VPC

This screenshot shows the 'Details' page for the route table 'rtb-042e91e63c9e485b1' after updates. The 'Routes' tab now shows four routes: '0.0.0.0/0' to 'igw-0ee7fa4ef2c88cd88' (Active), '130.0.0.0/16' to 'tgw-089edf17a9ee48ecc' (Active), '150.0.0.0/16' to 'tgw-089edf17a9ee48ecc' (Active), and '172.31.0.0/16' to 'local' (Active). All routes are marked as 'Active'.

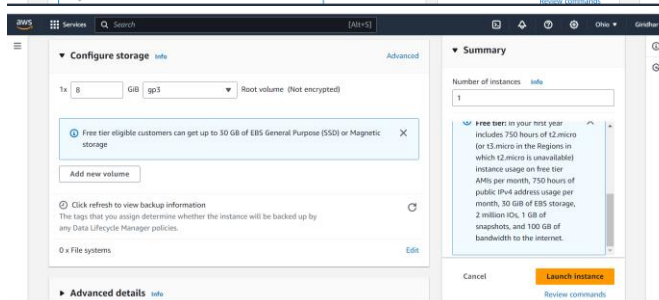
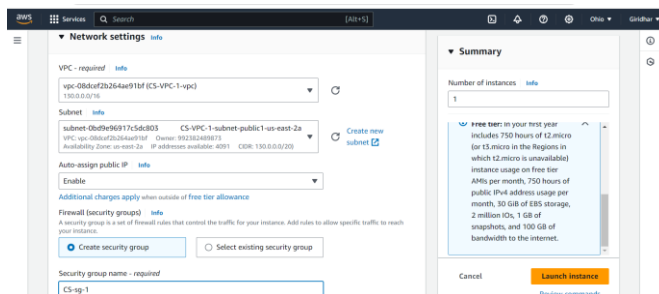
Step5 – creating EC2 instance for three VPCs

The screenshot shows the 'Instances' page in the AWS Management Console. It displays a table with columns for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. Below the table, it states 'No instances' and 'You do not have any instances in this region'. There is a 'Launch instances' button and a 'Select an instance' dropdown menu.

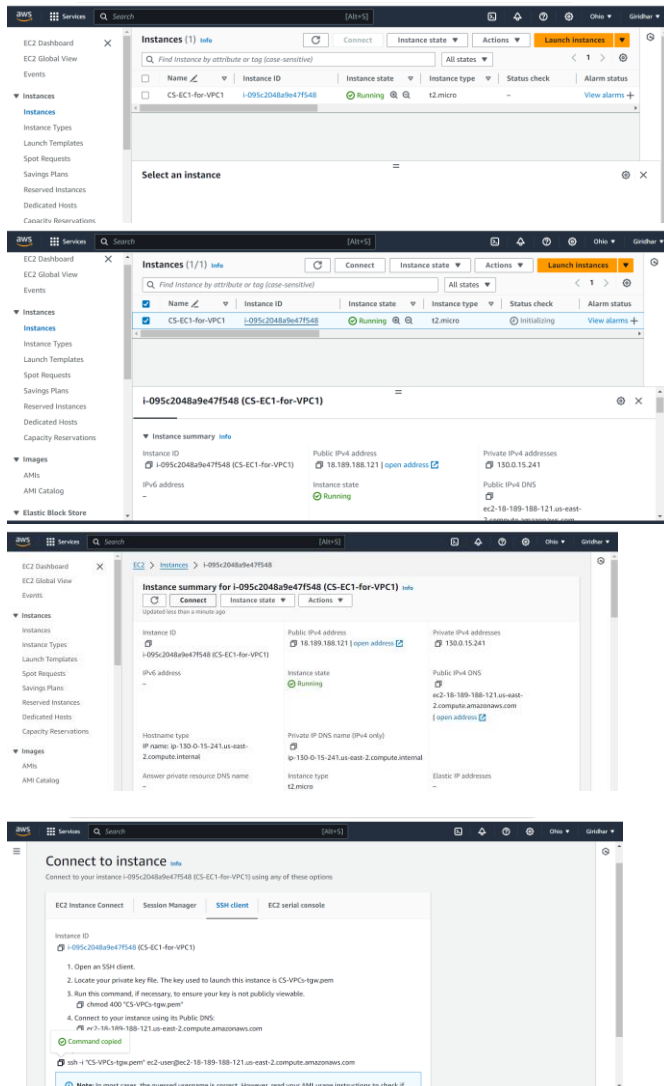
Create instance (CS-EC1-for-VPC1) and select amazon linux, create new key pair



Now select Vpc1 and subnet, create new security grp



Now instance is created and running



Connect to the instance to git bash by ssh and change to root user and install nginx

```
ec2-user@ip-130-0-15-241:~$ ssh -i "CS-VPcs-tgw.pem" ec2-user@ec2-18-189-188-121.us-east-2.compute.amazonaws.com
The authenticity of host 'ec2-18-189-188-121.us-east-2.compute.amazonaws.com (18.189.188.121)' can't be established.
ED25519 key fingerprint is SHA256:kzALdE3XOYzg29e8W46kbvqxHdQaYwLP846Dycx21s.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-18-189-188-121.us-east-2.compute.amazonaws.com' (ED25519) to the list of known hosts.

Amazon Linux 2023
https://aws.amazon.com/linux/amazon-linux-2023

[ec2-user@ip-130-0-15-241 ~]$ sudo -i
[ec2-user@ip-130-0-15-241 ~]$ sudo -i
[root@ip-130-0-15-241 ~]# yum update -y && yum install nginx -y
```

After installing nginx change directory to html, remove the index.html file which was already present and create file index.html again to have our own text data in that

```

root@ip-130-0-15-241:/usr/share/nginx/html
[root@ip-130-0-15-241 ~]# cd /usr/share/nginx/html
[root@ip-130-0-15-241 html]# ls
404.html  50x.html  icons  index.html  nginx-logo.png  poweredby.png
[root@ip-130-0-15-241 html]# rm index.html
rm: remove regular file 'index.html'? yes
[root@ip-130-0-15-241 html]# ls
404.html  50x.html  icons  nginx-logo.png  poweredby.png
[root@ip-130-0-15-241 html]# vi index.html

root@ip-130-0-15-241:/usr/share/nginx/html
[root@ip-130-0-15-241 ~]# cd /usr/share/nginx/html
[root@ip-130-0-15-241 html]# ls
404.html  50x.html  icons  index.html  nginx-logo.png  poweredby.png
[root@ip-130-0-15-241 html]# rm index.html
rm: remove regular file 'index.html'? yes
[root@ip-130-0-15-241 html]# ls
404.html  50x.html  icons  nginx-logo.png  poweredby.png
[root@ip-130-0-15-241 html]# vi index.html
404.html  50x.html  icons  index.html  nginx-logo.png  poweredby.png
[root@ip-130-0-15-241 html]# systemctl restart nginx
Unknown command verb restart.
[root@ip-130-0-15-241 html]# systemctl restart nginx
[root@ip-130-0-15-241 html]# clear

```

Go to instance and in security grp add inbound rule http which has port 80, so we can see the text file in chrome by adding (:80) in public ip of instance, which is created in git bash.

The first screenshot shows the AWS Management Console 'Security details' page for an EC2 instance. It lists the security group 'sg-0d8ccb1c3fc3b992d (CS-sg-1)' and shows the existing inbound rule for SSH (port 22).

The second screenshot shows the 'Details' page for the security group 'sg-0d8ccb1c3fc3b992d - CS-sg-1'. It shows the security group name, ID, description, owner, and the current inbound rule for SSH.

The third screenshot shows the 'Edit inbound rules' page. It displays the existing SSH rule and allows adding a new rule. A new rule for HTTP (port 80) is being added with the source set to 'Anywhere' (0.0.0.0/0).

Hi their this is case study of VPC - 1

Now create another instance (CS-EC2-for-VP2)

The first screenshot shows the 'Launch instance' wizard in the AWS Management Console. The 'Network settings' tab is selected, showing the VPC (vpc-daf102ef5e8058560 [CS-VPC-2-vpc]), Subnet (subnet-0b241e7c19c461abf CS-VPC-2-subnet-public2-us-east-2b), and Auto-assign public IP (Enabled). The 'Summary' tab shows the instance type (t2.micro) and the 'Launch instance' button.

The second screenshot shows the 'Instances' page in the AWS Management Console. The instance 'CS-EC2-for-VP2' (i-0117c644df4b79061) is listed with a status of 'Running'. The 'Connect' button is visible.

The third screenshot shows the 'Connect to instance' page in the AWS Management Console. The 'SSH client' tab is selected, showing the instance ID (i-0117c644df4b79061) and the public DNS (ec2-52-14-109-131.us-east-2.compute.amazonaws.com). The 'Command copied' message is visible.

Same process as done for instance one

The screenshot shows a web browser window with a terminal session. The terminal output is as follows:

```
[root@ip-150-0-29-194 html]# cd  
[root@ip-150-0-29-194 ~]# curl 150.0.29.194  
Hi buddies this is case study of VPC2  
[root@ip-150-0-29-194 ~]# |
```

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name

CS-EC3-for-defaultVPC

Add additional tags

Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Summary

Number of instances

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.5.2...read more

ami-0c11a843d846e09dd

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel

Launch instance

Network settings

VPC - required

vpc-0d384b0200521ea1c (default)

172.31.0.0/16

Subnet

subnet-096a776770c95c5a8

VPC: vpc-0d384b0200521ea1c Owner: 902382489873 Availability Zone: us-east-2a IP addresses available: 4091 CIDR: 172.31.0.0/20

Create new subnet

Auto-assign public IP

Enable

Additional charges apply when outside of free tier allowance

Firewall (security groups)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

Create security group

Select existing security group

Security group name - required

CS-sg-3

Summary

Number of instances

1

Software Image (AMI)

Amazon Linux 2023 AMI 2023.5.2...read more

ami-0c11a843d846e09dd

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

1 volume(s) - 8 GiB

Cancel

Launch instance

Review commands

EC2 Dashboard

EC2 Global View

Events

Instances

Instance Types

Launch Templates

Spot Requests

Savings Plans

Reserved Instances

Dedicated Hosts

Capacity Reservations

Images

AMIs

AMI Catalog

Instances (1/3)

Find Instance by attribute or tag (case-sensitive)

All states

Launch instances

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status
<input type="checkbox"/>	CS-EC2-for-VPC2	i-0117644df4b79061	Running	t2.micro	2/2 checks pass	View alarms
<input checked="" type="checkbox"/>	CS-EC3-for-def...	i-00083cc2ed7ebd00d	Running	t2.micro	Initializing	View alarms
<input type="checkbox"/>	CS-EC1-for-VPC1	i-095c2048a9e47f548	Running	t2.micro	2/2 checks pass	View alarms

I-00083cc2ed7ebd00d (CS-EC3-for-defaultVPC)

Details

Status and alarms

Monitoring

Security

Networking

Storage

Tags

Instance summary

The image shows the AWS Management Console 'Edit inbound rules' page for a security group. It lists two rules: one for SSH (port 22) and one for HTTP (port 80), both allowing traffic from 0.0.0.0/0. Below this is a terminal window showing an SSH session from a Windows machine to an Amazon Linux instance. The terminal output includes the SSH command, the host's fingerprint, and the user's login. After logging in, the user runs 'sudo -i' to become root, then updates the system and installs nginx. Finally, the user runs 'curl 172.31.12.115', and the output shows the default nginx welcome message. At the bottom, a web browser window displays the same message at the IP address 18.119.128.192.

We have created communication to all three VPCs with Transit Gateway step by step, now we can access any data of any vpc from any one of three

```

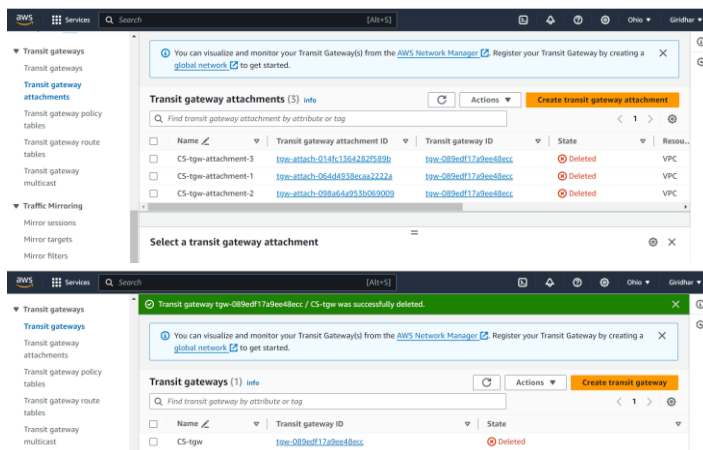
root@ip-130-0-15-241:~
[root@ip-130-0-15-241 ~]# curl 130.0.15.241
Hi their this is case study of VPC - 1
[root@ip-130-0-15-241 ~]# curl 150.0.29.194
Hi buddies this is case study of VPC2
[root@ip-130-0-15-241 ~]# curl 172.31.12.115
Hello All this is case study default vpc
[root@ip-130-0-15-241 ~]#

```

```
root@ip-150-0-29-194:~  
[root@ip-150-0-29-194 ~]# curl 150.0.29.194  
Hi buddies this is case study of VPC2  
[root@ip-150-0-29-194 ~]# curl 130.0.15.241  
Hi their this is case study of VPC - 1  
[root@ip-150-0-29-194 ~]# curl 172.31.12.115  
Hello All this is case study default vpc  
[root@ip-150-0-29-194 ~]#
```

```
root@ip-172-31-12-115:~  
[root@ip-172-31-12-115 ~]# curl 172.31.12.115  
Hello All this is case study default vpc  
[root@ip-172-31-12-115 ~]# curl 150.0.29.194  
Hi buddies this is case study of VPC2  
[root@ip-172-31-12-115 ~]# curl 130.0.15.241  
Hi their this is case study of VPC - 1  
[root@ip-172-31-12-115 ~]# |
```

Delete all in reverse process



Instances (5) info

Find instance by attribute or tag (case-sensitive)

All states

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability zone
<input type="checkbox"/>	CS-EC2-for-VPC2	i-0117d44d4b79061	Terminated	t2.micro	-	View alarms	us-east-1a
<input type="checkbox"/>	CS-EC3-for-def...	i-00083cc2ed7eb00d	Terminated	t2.micro	-	View alarms	us-east-1a
<input type="checkbox"/>	CS-EC1-for-VPC1	i-095c2048a9e47548	Terminated	t2.micro	-	View alarms	us-east-1a

VPC dashboard

EC2 Global View

Filter by VPC

Virtual private cloud

Your VPCs

Subnets

Route tables

Internet gateways

Egress-only internet gateways

DHCP option sets

Elastic IPs

You successfully deleted vpc-0bdcf2b264ae913f / CS-VPC-1-vpc and 10 other resources.

Details

Last updated less than a minute ago

Your VPCs (1/2) info

Search

	Name	VPC ID	State
<input checked="" type="checkbox"/>	CS-VPC-2-vpc	vpc-0af102ef5e8058560	Available
<input type="checkbox"/>	default	vpc-0af584b0209521ea1e	Available

Actions

- Create default VPC
- Create flow log
- Edit VPC settings
- Edit CIDRs
- Manage middlebox routes
- Manage tags
- Delete VPC

vpc-0af102ef5e8058560 / CS-VPC-2-vpc

Details Resource map CIDRs Flow logs Tags Integrations