Challenge -3 (VPC)

1. Design and deploy a scalable network architecture using AWS Transit Gateway to simplify network connectivity between multiple VPCs.

Create Transit Gateway

- VPC → Transit Gateways → Create → default settings.

Create VPC Attachment

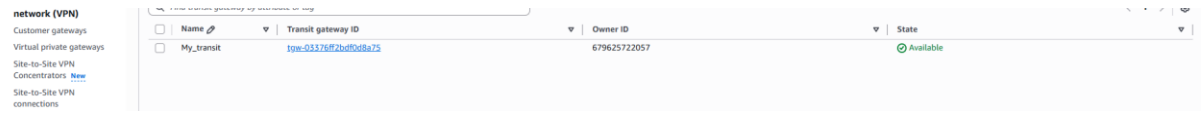
- Attach VPC to the TGW.
- Select 1 subnet per AZ.

Association happens automatically

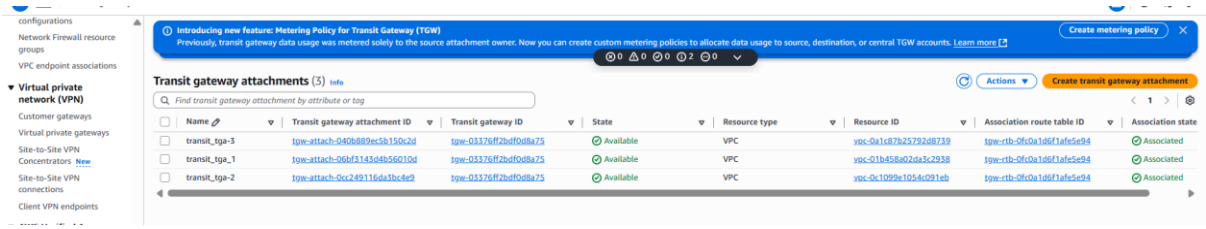
- No need to change anything here.

Update VPC Route Table

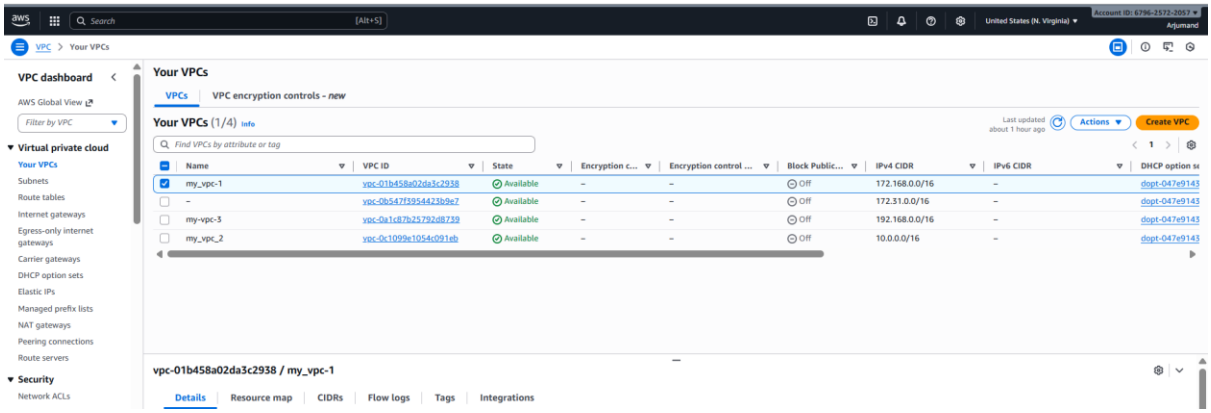
- Add route → Destination = other VPC CIDR
- Target = **Transit Gateway**



Created 3 transit gateway attachment with three different vpcs



Added all cidr and internet to the each routers (note: internet will be added to each vpc)



Connected with one of the three ec2 which has created with separate vcps using ssh command, then ping <private IP of other instance>.

```
[ec2-user@ip-172-168-0-196 ~]$ ping 192.168.0.59
PING 192.168.0.59 (192.168.0.59) 56(84) bytes of data.
64 bytes from 192.168.0.59: icmp_seq=1 ttl=126 time=2.63 ms
64 bytes from 192.168.0.59: icmp_seq=2 ttl=126 time=1.37 ms
64 bytes from 192.168.0.59: icmp_seq=3 ttl=126 time=1.21 ms
64 bytes from 192.168.0.59: icmp_seq=4 ttl=126 time=1.22 ms
64 bytes from 192.168.0.59: icmp_seq=5 ttl=126 time=1.27 ms
64 bytes from 192.168.0.59: icmp_seq=6 ttl=126 time=1.16 ms
64 bytes from 192.168.0.59: icmp_seq=7 ttl=126 time=1.29 ms
64 bytes from 192.168.0.59: icmp_seq=8 ttl=126 time=1.30 ms
64 bytes from 192.168.0.59: icmp_seq=9 ttl=126 time=1.40 ms
64 bytes from 192.168.0.59: icmp_seq=10 ttl=126 time=1.21 ms
64 bytes from 192.168.0.59: icmp_seq=11 ttl=126 time=1.18 ms
64 bytes from 192.168.0.59: icmp_seq=12 ttl=126 time=1.22 ms
64 bytes from 192.168.0.59: icmp_seq=13 ttl=126 time=1.19 ms
64 bytes from 192.168.0.59: icmp_seq=14 ttl=126 time=1.33 ms
64 bytes from 192.168.0.59: icmp_seq=15 ttl=126 time=1.24 ms
64 bytes from 192.168.0.59: icmp_seq=16 ttl=126 time=1.47 ms
64 bytes from 192.168.0.59: icmp_seq=17 ttl=126 time=1.27 ms
64 bytes from 192.168.0.59: icmp_seq=18 ttl=126 time=1.17 ms
64 bytes from 192.168.0.59: icmp_seq=19 ttl=126 time=1.16 ms
64 bytes from 192.168.0.59: icmp_seq=20 ttl=126 time=1.29 ms
64 bytes from 192.168.0.59: icmp_seq=21 ttl=126 time=1.22 ms
64 bytes from 192.168.0.59: icmp_seq=22 ttl=126 time=2.78 ms
64 bytes from 192.168.0.59: icmp_seq=23 ttl=126 time=1.19 ms
64 bytes from 192.168.0.59: icmp_seq=24 ttl=126 time=1.22 ms
64 bytes from 192.168.0.59: icmp_seq=25 ttl=126 time=1.16 ms
64 bytes from 192.168.0.59: icmp_seq=26 ttl=126 time=1.13 ms
64 bytes from 192.168.0.59: icmp_seq=27 ttl=126 time=1.15 ms
64 bytes from 192.168.0.59: icmp_seq=28 ttl=126 time=1.09 ms
64 bytes from 192.168.0.59: icmp_seq=29 ttl=126 time=1.33 ms
64 bytes from 192.168.0.59: icmp_seq=30 ttl=126 time=1.14 ms
64 bytes from 192.168.0.59: icmp_seq=31 ttl=126 time=1.12 ms
64 bytes from 192.168.0.59: icmp_seq=32 ttl=126 time=1.17 ms
64 bytes from 192.168.0.59: icmp_seq=33 ttl=126 time=1.16 ms
^C
--- 192.168.0.59 ping statistics ---
33 packets transmitted, 33 received, 0% packet loss, time 32048ms
rtt min/avg/max/mdev = 1.092/1.315/2.779/0.363 ms
[ec2-user@ip-172-168-0-196 ~]$ 10.0.0.143
-bash: 10.0.0.143: command not found
[ec2-user@ip-172-168-0-196 ~]$ ping 10.0.0.143
PING 10.0.0.143 (10.0.0.143) 56(84) bytes of data.
64 bytes from 10.0.0.143: icmp_seq=1 ttl=126 time=1.82 ms
64 bytes from 10.0.0.143: icmp_seq=2 ttl=126 time=0.899 ms
64 bytes from 10.0.0.143: icmp_seq=3 ttl=126 time=0.824 ms
64 bytes from 10.0.0.143: icmp_seq=4 ttl=126 time=0.822 ms
64 bytes from 10.0.0.143: icmp_seq=5 ttl=126 time=0.902 ms
64 bytes from 10.0.0.143: icmp_seq=6 ttl=126 time=0.875 ms
64 bytes from 10.0.0.143: icmp_seq=7 ttl=126 time=0.889 ms
64 bytes from 10.0.0.143: icmp_seq=8 ttl=126 time=0.822 ms
64 bytes from 10.0.0.143: icmp_seq=9 ttl=126 time=0.981 ms
64 bytes from 10.0.0.143: icmp_seq=10 ttl=126 time=0.742 ms
^C
--- 10.0.0.143 ping statistics ---
```

2. **Configure VPC endpoints to securely access AWS services without internet gateways or NAT gateways, ensuring data privacy and minimizing exposure to external threats.**

WHY THIS IS IMPORTANT (Security Reason)

Normally, a private EC2 instance needs a **NAT Gateway** to access S3.

With a VPC endpoint:

- Traffic **never goes to the Internet**
- Data stays **within AWS backbone network**
- No exposure to public IP addresses
- Zero attack surface from outside internet
- Helps meet security & compliance requirements

Steps:-

1. Created Two EC2 Instances

a) Public EC2 (Bastion Host)

- Launched in **public subnet** of the VPC
- **Auto-assign Public IP:** Enabled
- Security Group allowed **SSH (22)** from my local IP
- Used as a jump server to access the private EC2 instance

b) Private EC2

- Launched in **private subnet** of the same VPC
- **Auto-assign Public IP:** Disabled
- Security Group allowed **SSH ONLY from the Bastion Host's SG**
- No direct internet access

---

2. Created S3 VPC Endpoint

- Navigation: **VPC → Endpoints → Create Endpoint**
- Selected service: **com.amazonaws.<region>.s3** (Gateway type)
- Selected **my VPC**
- Attached endpoint to **private route table only**
- Verified route added:
- pl-xxxxxx → vpce-xxxxxxx (Active)

---

3. Prepared AWS CLI Access on Public EC2

- On my local machine, opened **Git Bash**
- Installed AWS CLI (if not installed)
- Configured credentials using:
- aws configure
    - Entered Access Key ID
    - Entered Secret Access Key
    - Region
    - Output format

These credentials allow S3 access.

---

4. Copied PEM Key to Public EC2

- Located .pem key in local Downloads
- Used SCP to copy the PEM file to the public EC2 instance:
- scp -i mykey.pem mykey.pem ec2-user@<public-ip>:/home/ec2-user/

---

5. SSH into Public EC2

- Connected using Git Bash:
- ssh -i mykey.pem ec2-user@<public-ec2-public-ip>

---

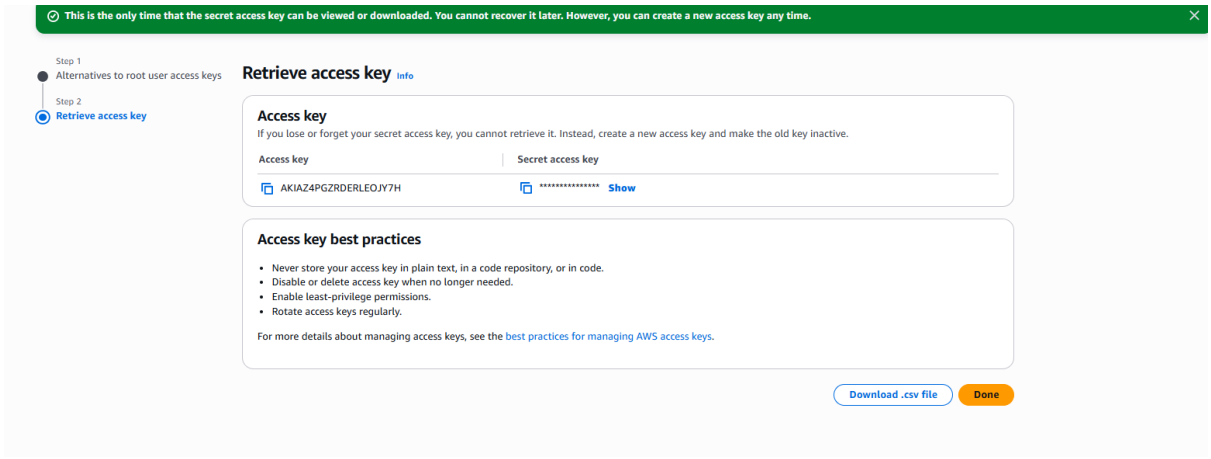
6. From Public EC2, SSH into Private EC2

- Used the private IP of the private instance:
- ssh -i mykey.pem ec2-user@<private-ec2-private-ip>
- Successfully entered the private instance **without any internet access**.

---

7. Tested S3 Access from the Private EC2

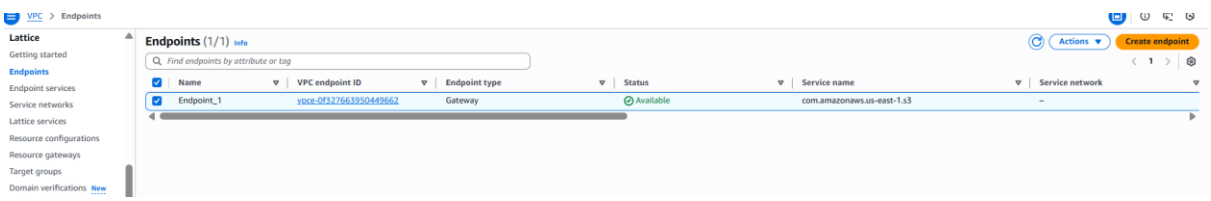
Inside the private EC2 terminal, ran command: aws s3 ls




```
[ec2-user@ip-172-168-0-137 ~]$ aws s3 ls

Unable to locate credentials. You can configure credentials by running "aws configure".
[ec2-user@ip-172-168-0-137 ~]$ aws configure
AWS Access Key ID [None]: AKIAZ4PGZRDERLEOJY7H
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
[ec2-user@ip-172-168-0-137 ~]$ aws s3 ls
2025-11-22 18:55:46 mmyflowlog-s3
[ec2-user@ip-172-168-0-137 ~]$
```

```
find: '/var/lib/sss/secrets': Permission denied
find: '/var/lib/chrony': Permission denied
find: '/var/lib/nfs/statd': Permission denied
find: '/var/lib/amazon/ssm/ipc': Permission denied
find: '/var/lib/amazon/ssm/runtimeconfig': Permission denied
find: '/var/lib/amazon/ssm/Vault': Permission denied
find: '/var/db/sudo': Permission denied
find: '/var/spool/at': Permission denied
find: '/var/tmp/systemd-private-086dd0afa93d47f1ac6180160729db3f-systemd-resolved.service-8ptsWC': Permission denied
find: '/var/tmp/systemd-private-086dd0afa93d47f1ac6180160729db3f-policy-routes@enX0.service-adp1Rm': Permission denied
find: '/var/tmp/systemd-private-086dd0afa93d47f1ac6180160729db3f-systemd-logind.service-Am6gg3': Permission denied
find: '/var/tmp/systemd-private-086dd0afa93d47f1ac6180160729db3f-dbus-broker.service-S2NF4W': Permission denied
find: '/var/tmp/systemd-private-086dd0afa93d47f1ac6180160729db3f-chronyd.service-Jto1Qg': Permission denied
/usr/lib/systemd/portable/profile
find: '/usr/share/empty.sshd': Permission denied
find: '/usr/libexec/initscripts/legacy-actions/auditd': Permission denied
find: '/root': Permission denied
[ec2-user@ip-172-168-0-137 ~]$ cd .aws
[ec2-user@ip-172-168-0-137 .aws]$ ls
config  credentials
[ec2-user@ip-172-168-0-137 .aws]$ cat config
[default]
region = us-east-1
output = json
[ec2-user@ip-172-168-0-137 .aws]$ cd ..
[ec2-user@ip-172-168-0-137 ~]$ vi Testing.pem
[ec2-user@ip-172-168-0-137 ~]$ chmod 400 Testing.pem
[ec2-user@ip-172-168-0-137 ~]$ ssh -i Testing.pem ec2-user@172.168.1.103
The authenticity of host '172.168.1.103 (172.168.1.103)' can't be established.
ED25519 key fingerprint is SHA256:Y/OCUeATFyUxj8GI2l9BGp2nEPFS1TMkFULsWua6+ZE.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.168.1.103' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-168-1-103 ~]$ sudo su -
[root@ip-172-168-1-103 ~]#
```



:b-0d37895265b0eef9e / my_roter_private

Details | Routes | Subnet associations | Edge associations | Route propagation | Tags

Routes (2)

Filter routes

Both | Edit routes

1

Destination	Target	Status	Propagated	Route Origin
pl-63a5400a	vpce-0f327663950449662	Active	No	Create Route
172.168.0.0/16	local	Active	No	Create Route Table


```
-bash: asw: command not found
[root@ip-172-168-1-103 ~]# aws s3 ls
2025-11-22 18:55:46 mmyflowlog-s3
[root@ip-172-168-1-103 ~]#
```