VPC Network Architecture via NAT Gateway

This document provides detailed documentation for setting up a custom VPC with public and private subnets in AWS, using a NAT Gateway for secure internet access for private resources. It is based on the step-by-step implementation by Anurag Prasad.

Contents:--

1. Project Overview

The goal of this project is to design and implement a secure Virtual Private Cloud (VPC) in AWS with public and private subnets. A NAT Gateway is used to provide outbound internet access for private instances without exposing them directly to the internet. This project validates secure connectivity and controlled access between subnets, along with Apache server setup on a private EC2 instance.

2. Tools & Services Used

- AWS VPC – for creating a custom network.
- Subnets (Public & Private) – to segregate resources.
- Internet Gateway – for public internet access.
- NAT Gateway – to enable private instances to connect to the internet securely.
- AWS EC2 – to deploy public and private virtual machines.
- Apache httpd – web server installed on the private EC2 instance.

3. Architecture & Visual Representation

The architecture consists of the following components:

- VPC with a defined CIDR block.
- Two subnets placed in different Availability Zones (one public, one private).
- Internet Gateway attached to the VPC for public internet access.
- NAT Gateway in the public subnet to enable internet connectivity for the private subnet.
- Route Tables: Public subnet routes to IGW, private subnet routes to NAT Gateway.
- Two EC2 instances: one in the public subnet (jump host), and one in the private subnet (accessible via the public EC2).

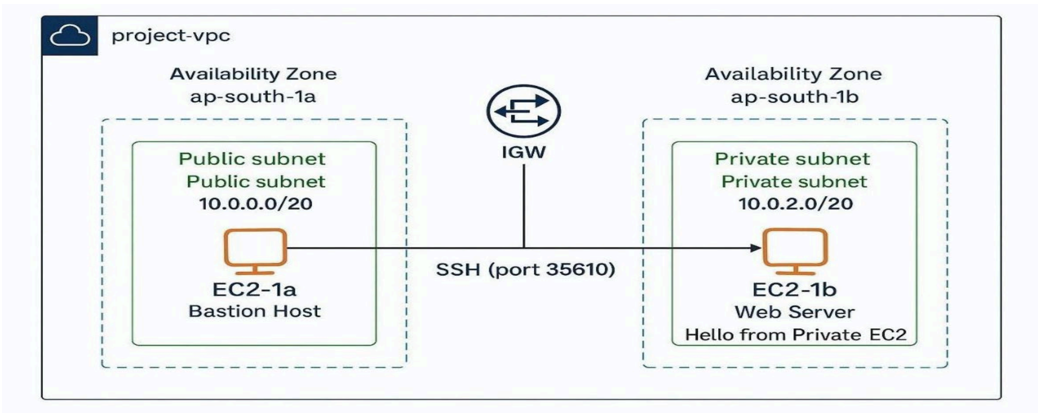
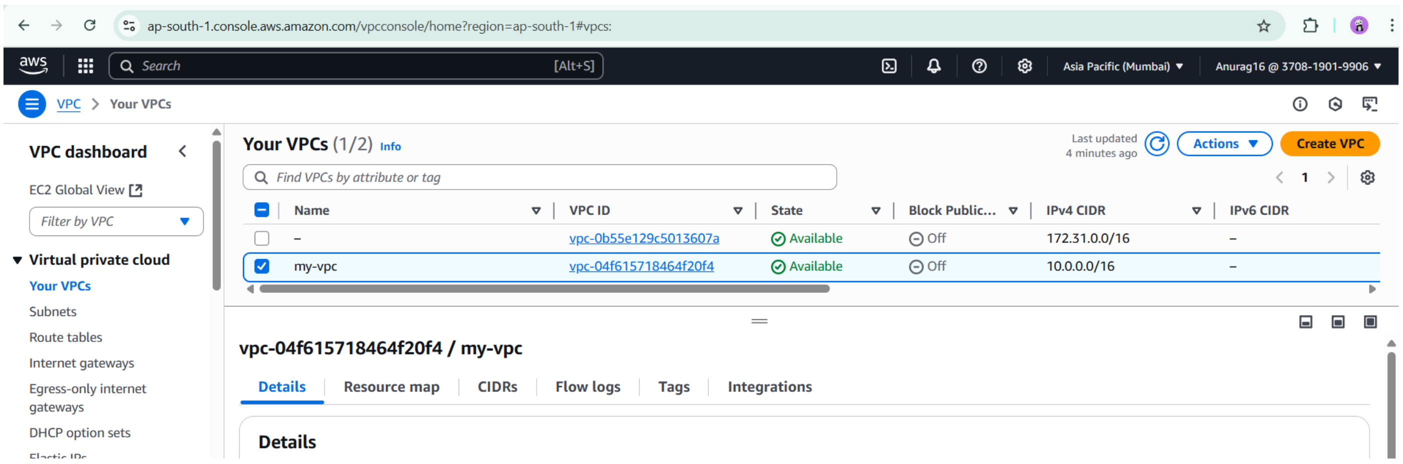


Fig: Visual Representation

4. Implementation Steps

Create a VPC with appropriate CIDR:



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#vpcs:

VPC > Your VPCs

Your VPCs (1/2) Info

Last updated 4 minutes ago

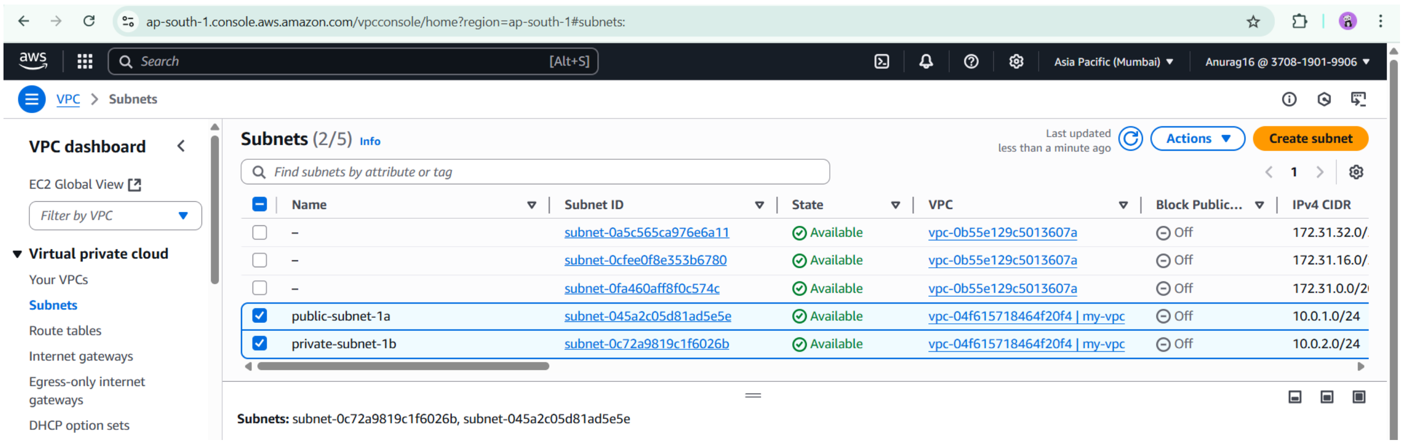
Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR
–	vpc-0b55e129c5013607a	Available	Off	172.31.0.0/16	–
my-vpc	vpc-04f615718464f20f4	Available	Off	10.0.0.0/16	–

vpc-04f615718464f20f4 / my-vpc

Details | Resource map | CIDRs | Flow logs | Tags | Integrations

Details

Create 2 subnets in different AZ:



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#subnets:

VPC > Subnets

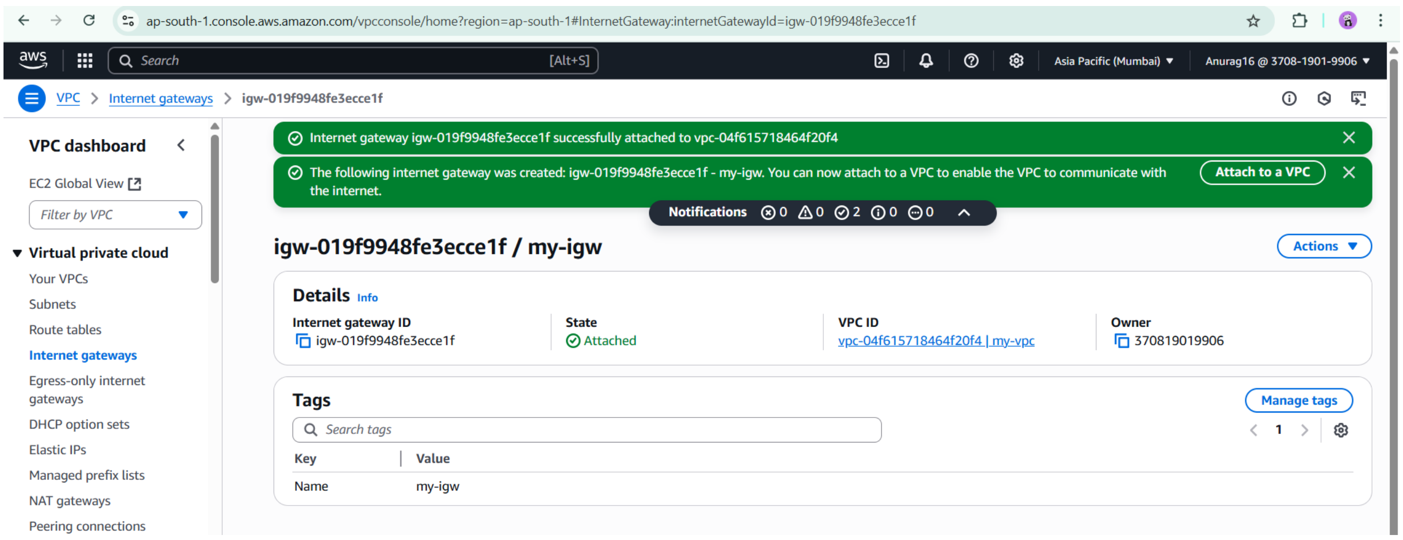
Subnets (2/5) Info

Last updated less than a minute ago

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
–	subnet-0a5c565ca976e6a11	Available	vpc-0b55e129c5013607a	Off	172.31.32.0/
–	subnet-0cfee0f8e353b6780	Available	vpc-0b55e129c5013607a	Off	172.31.16.0/
–	subnet-0fa460aff8f0c574c	Available	vpc-0b55e129c5013607a	Off	172.31.0.0/2
public-subnet-1a	subnet-045a2c05d81ad5e5e	Available	vpc-04f615718464f20f4 | my-vpc	Off	10.0.1.0/24
private-subnet-1b	subnet-0c72a9819c1f6026b	Available	vpc-04f615718464f20f4 | my-vpc	Off	10.0.2.0/24

Subnets: subnet-0c72a9819c1f6026b, subnet-045a2c05d81ad5e5e

Create an Internet Gateway and attach it to VPC:



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#InternetGateway:internetGatewayId=igw-019f9948fe3ecce1f

VPC > Internet gateways > igw-019f9948fe3ecce1f

Internet gateway igw-019f9948fe3ecce1f successfully attached to vpc-04f615718464f20f4

The following internet gateway was created: igw-019f9948fe3ecce1f - my-igw. You can now attach to a VPC to enable the VPC to communicate with the internet.

igw-019f9948fe3ecce1f / my-igw

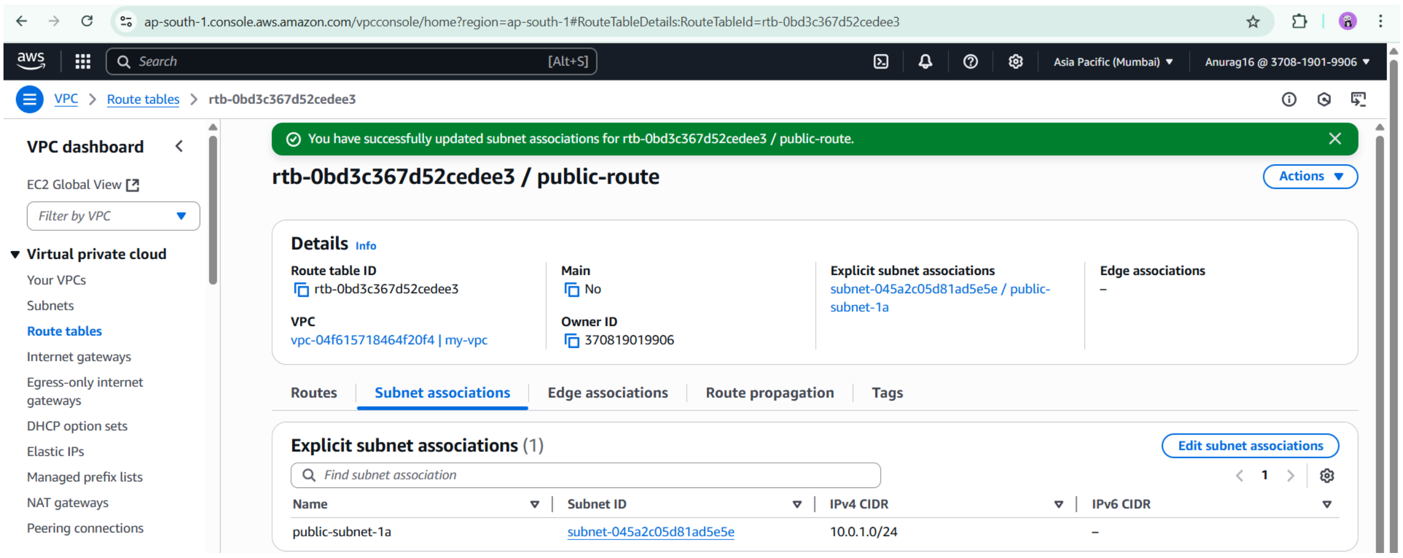
Details Info

Internet gateway ID	State	VPC ID	Owner
igw-019f9948fe3ecce1f	Attached	vpc-04f615718464f20f4 | my-vpc	370819019906

Tags

Key	Value
Name	my-igw

Configure the Public Subnet's route table to allow outbound internet access via the IGW:



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-0bd3c367d52cedee3

Asia Pacific (Mumbai) ▼ Anurag16 @ 3708-1901-9906 ▼

VPC > Route tables > rtb-0bd3c367d52cedee3

You have successfully updated subnet associations for rtb-0bd3c367d52cedee3 / public-route.

rtb-0bd3c367d52cedee3 / public-route

Details

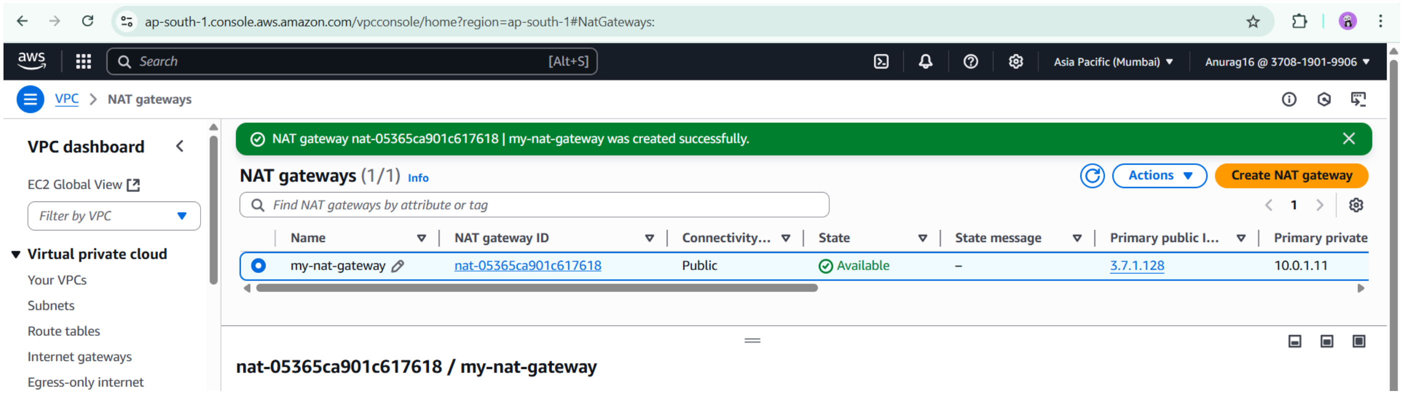
Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0bd3c367d52cedee3	No	subnet-045a2c05d81ad5e5e / public-subnet-1a	–
VPC: vpc-04f615718464f20f4 | my-vpc	Owner ID: 370819019906		

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

Explicit subnet associations (1)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
public-subnet-1a	subnet-045a2c05d81ad5e5e	10.0.1.0/24	–

Create a NAT Gateway:



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#NatGateways:

Asia Pacific (Mumbai) ▼ Anurag16 @ 3708-1901-9906 ▼

VPC > NAT gateways

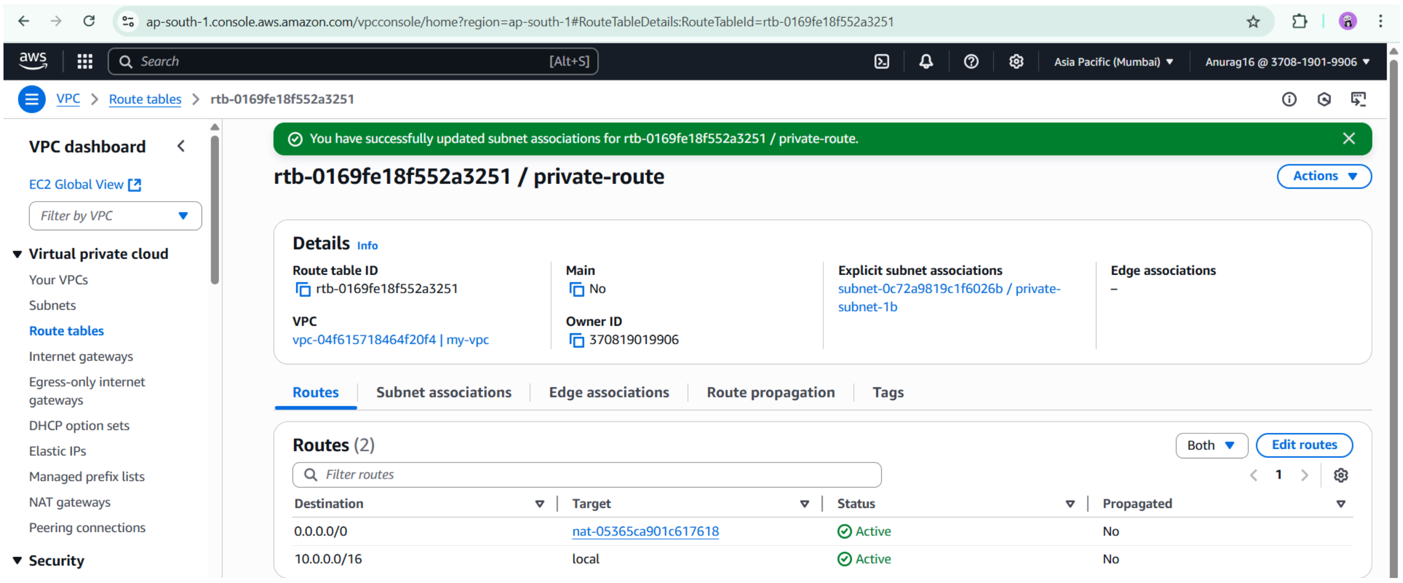
NAT gateway nat-05365ca901c617618 | my-nat-gateway was created successfully.

NAT gateways (1/1)

Name	NAT gateway ID	Connectivity...	State	State message	Primary public I...	Primary private
my-nat-gateway	nat-05365ca901c617618	Public	Available	–	3.7.1.128	10.0.1.11

nat-05365ca901c617618 / my-nat-gateway

Update the Private Subnet's route table to use the NAT for internet access (for outbound traffic only):



ap-south-1.console.aws.amazon.com/vpcconsole/home?region=ap-south-1#RouteTableDetails:RouteTableId=rtb-0169fe18f552a3251

Asia Pacific (Mumbai) ▼ Anurag16 @ 3708-1901-9906 ▼

VPC > Route tables > rtb-0169fe18f552a3251

You have successfully updated subnet associations for rtb-0169fe18f552a3251 / private-route.

rtb-0169fe18f552a3251 / private-route

Details

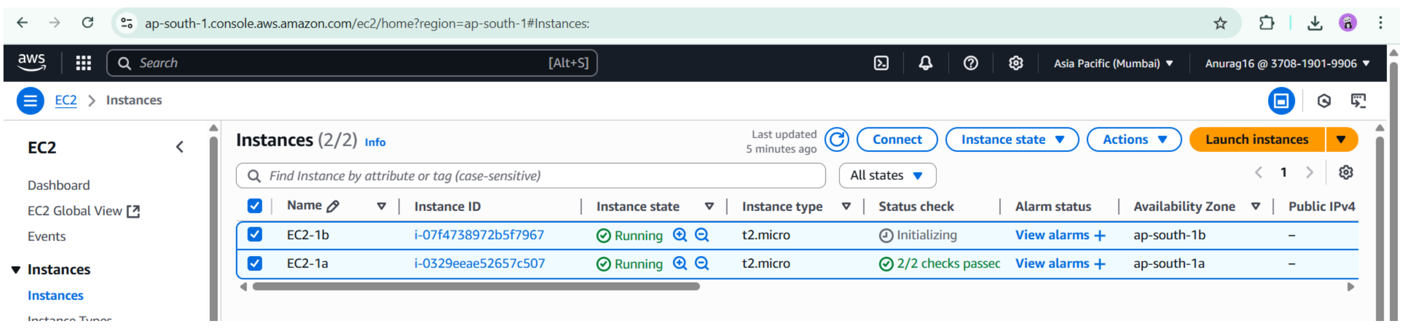
Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0169fe18f552a3251	No	subnet-0c72a9819c1f6026b / private-subnet-1b	–
VPC: vpc-04f615718464f20f4 | my-vpc	Owner ID: 370819019906		

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

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	nat-05365ca901c617618	Active	No
10.0.0.0/16	local	Active	No

Launch 2 VMs (one for each subnet):



ap-south-1.console.aws.amazon.com/ec2/home?region=ap-south-1#Instances:

Search [Alt+S] Asia Pacific (Mumbai) ▼ Anurag16 @ 3708-1901-9906 ▼

EC2 > Instances

EC2

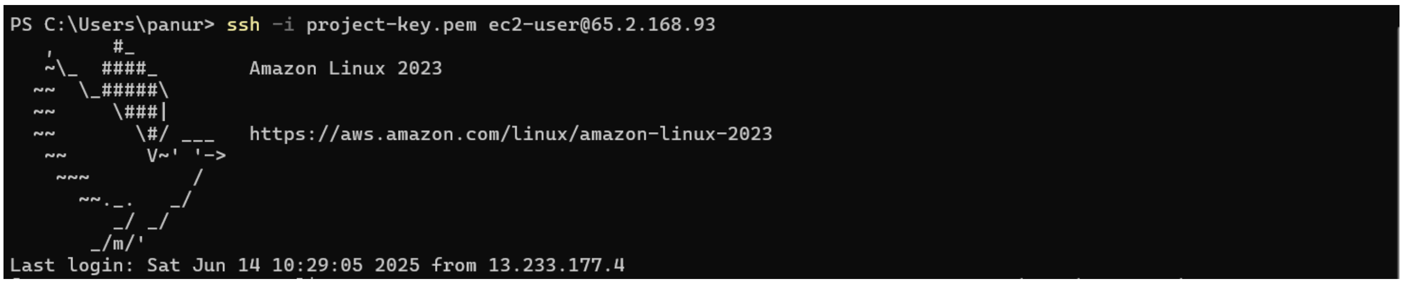
Dashboard
EC2 Global View
Events
▼ Instances
Instances

Instances (2/2) Info — Last updated 5 minutes ago — Connect — Instance state ▼ — Actions ▼ — Launch instances ▼

Find Instance by attribute or tag (case-sensitive) — All states ▼ — 1

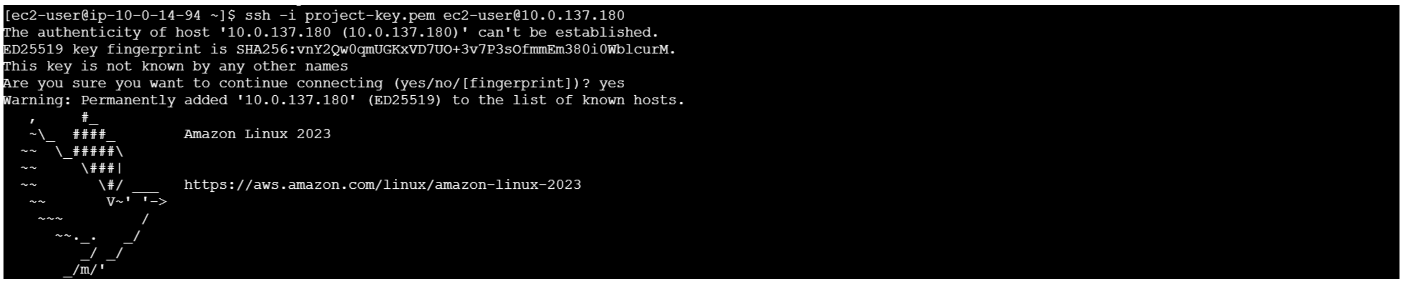
	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4
✓	EC2-1b	i-07f4738972b5f7967	Running	t2.micro	Initializing	View alarms +	ap-south-1b	–
✓	EC2-1a	i-0329eeae52657c507	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1a	–

SSH into public EC2:



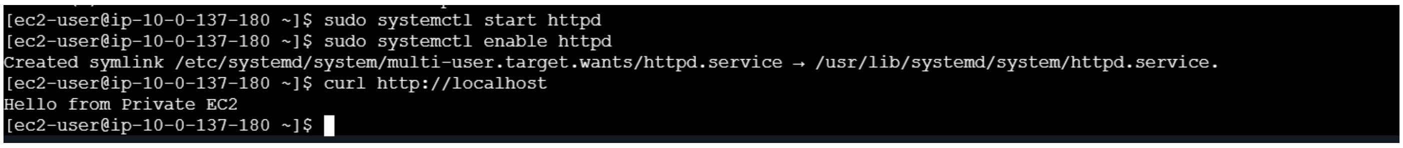
```
PS C:\Users\panur> ssh -i project-key.pem ec2-user@65.2.168.93
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Jun 14 10:29:05 2025 from 13.233.177.4
```

Now SSH into private EC2 by using the pubic one:



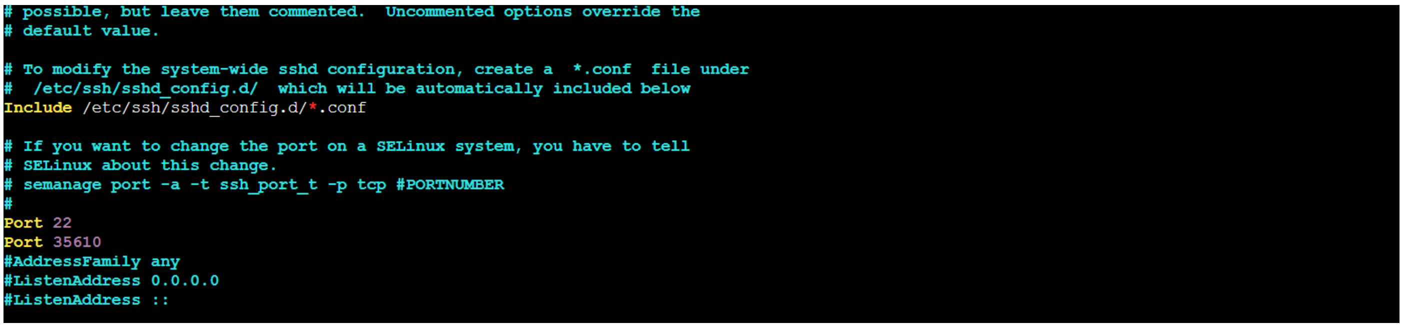
```
[ec2-user@ip-10-0-14-94 ~]$ ssh -i project-key.pem ec2-user@10.0.137.180
The authenticity of host '10.0.137.180 (10.0.137.180)' can't be established.
ED25519 key fingerprint is SHA256:vnY2Qw0qmUGKxVD7UO+3v7P3sOfmmEm380i0WblcurM.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '10.0.137.180' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
```

Install Apache httpd on private EC2 and create a simple webpage saying Hello from Private EC2:



```
[ec2-user@ip-10-0-137-180 ~]$ sudo systemctl start httpd
[ec2-user@ip-10-0-137-180 ~]$ sudo systemctl enable httpd
Created symlink /etc/systemd/system/multi-user.target.wants/httpd.service → /usr/lib/systemd/system/httpd.service.
[ec2-user@ip-10-0-137-180 ~]$ curl http://localhost
Hello from Private EC2
[ec2-user@ip-10-0-137-180 ~]$ █
```

Change the port from 22 to 35610 for private EC2:

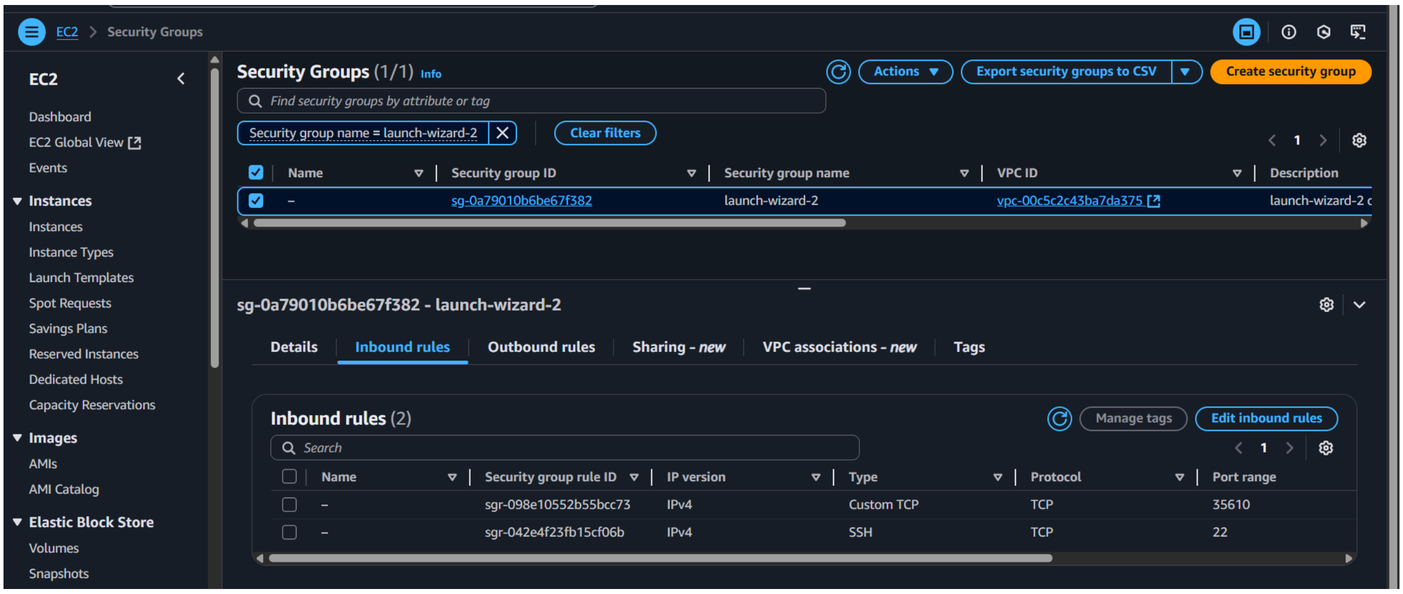


```
# possible, but leave them commented.  Uncommented options override the
# default value.

# To modify the system-wide sshd configuration, create a  *.conf  file under
#  /etc/ssh/sshd_config.d/  which will be automatically included below
Include /etc/ssh/sshd_config.d/*.conf

# If you want to change the port on a SELinux system, you have to tell
# SELinux about this change.
# semanage port -a -t ssh_port_t -p tcp #PORTNUMBER
#
Port 22
Port 35610
#AddressFamily any
#ListenAddress 0.0.0.0
#ListenAddress ::
```

Configure the private EC2's Security Group as well:



Security Groups (1/1) Info

Security group name = launch-wizard-2

Name	Security group ID	Security group name	VPC ID	Description
–	sg-0a79010b6be67f382	launch-wizard-2	vpc-00c5c2c43ba7da375	launch-wizard-2 c

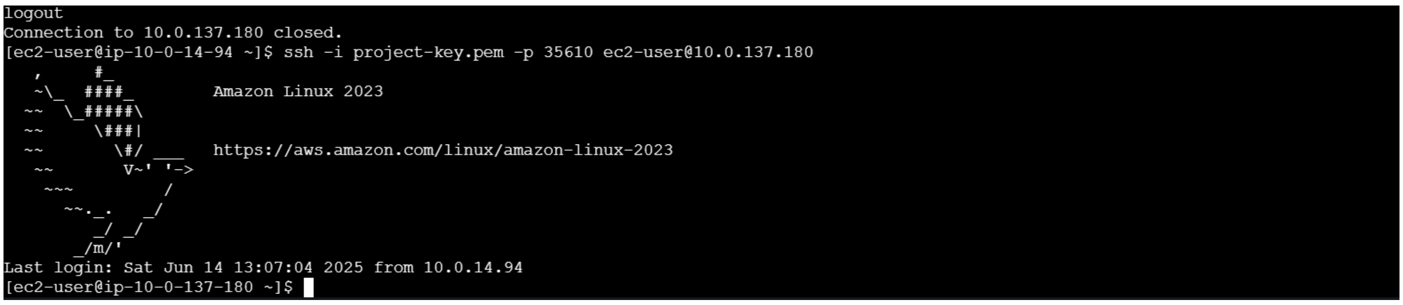
sg-0a79010b6be67f382 - launch-wizard-2

Details | Inbound rules | Outbound rules | Sharing - *new* | VPC associations - *new* | Tags

Inbound rules (2)

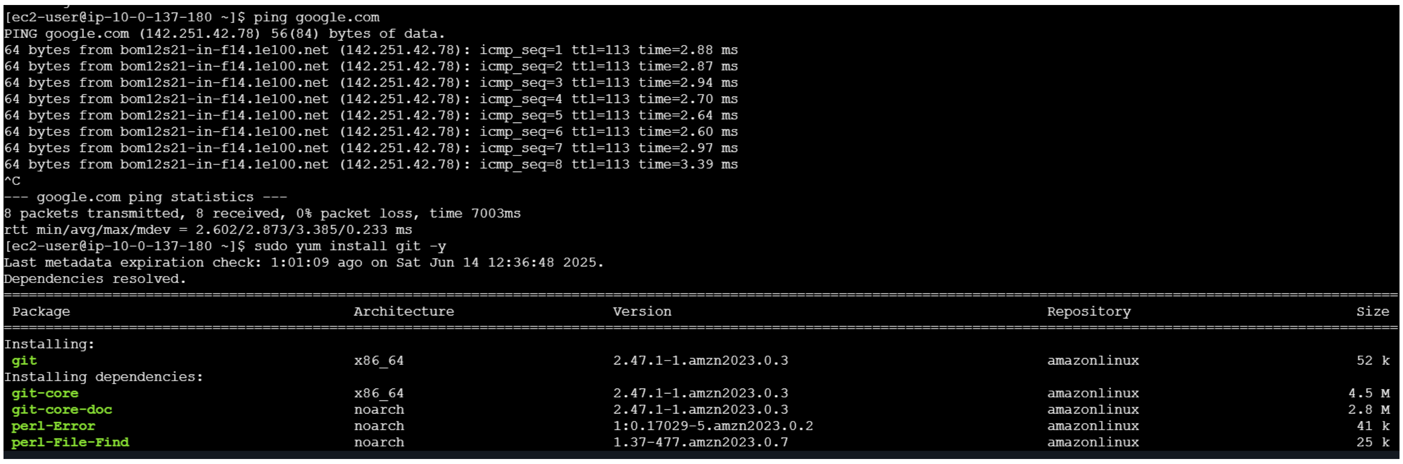
Name	Security group rule ID	IP version	Type	Protocol	Port range
–	sgr-098e10552b55bcc73	IPv4	Custom TCP	TCP	35610
–	sgr-042e4f23fb15cf06b	IPv4	SSH	TCP	22

SSH into private EC2 using port 35610 from public EC2:



```
logout
Connection to 10.0.137.180 closed.
[ec2-user@ip-10-0-14-94 ~]$ ssh -i project-key.pem -p 35610 ec2-user@10.0.137.180
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Sat Jun 14 13:07:04 2025 from 10.0.14.94
[ec2-user@ip-10-0-137-180 ~]$
```

Ensure the Private EC2 can access the internet via the NAT Gateway:



```
[ec2-user@ip-10-0-137-180 ~]$ ping google.com
PING google.com (142.251.42.78) 56(84) bytes of data.
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=1 ttl=113 time=2.88 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=2 ttl=113 time=2.87 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=3 ttl=113 time=2.94 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=4 ttl=113 time=2.70 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=5 ttl=113 time=2.64 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=6 ttl=113 time=2.60 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=7 ttl=113 time=2.97 ms
64 bytes from bom12s21-in-f14.1e100.net (142.251.42.78): icmp_seq=8 ttl=113 time=3.39 ms
^C
--- google.com ping statistics ---
8 packets transmitted, 8 received, 0% packet loss, time 7003ms
rtt min/avg/max/mdev = 2.602/2.873/3.385/0.233 ms
[ec2-user@ip-10-0-137-180 ~]$ sudo yum install git -y
Last metadata expiration check: 1:01:09 ago on Sat Jun 14 12:36:48 2025.
Dependencies resolved.
================================================================================================================
 Package                  Architecture          Version                             Repository            Size
================================================================================================================
Installing:
 git                      x86_64                2.47.1-1.amzn2023.0.3               amazonlinux           52 k
Installing dependencies:
 git-core                 x86_64                2.47.1-1.amzn2023.0.3               amazonlinux          4.5 M
 git-core-doc             noarch                2.47.1-1.amzn2023.0.3               amazonlinux          2.8 M
 perl-Error               noarch                1:0.17029-5.amzn2023.0.2            amazonlinux           41 k
 perl-File-Find           noarch                1.37-477.amzn2023.0.7               amazonlinux           25 k
```

5. Validation & Testing Validation

- Successfully connecting to the private EC2 instance via the public EC2 (jump host).
- Verifying Apache httpd is running and serving the test webpage from the private EC2.
- Ensuring SSH works on the updated port (35610).
- Confirming the private EC2 has outbound internet access through the NAT Gateway.

6. Challenges & Solutions

- SSH Restrictions: Resolved by configuring the private EC2 Security Group and updating the port.
- Private Internet Access: Ensured via NAT Gateway configuration and correct routing setup.
- Subnet Communication: Verified and tested by using the public EC2 as a jump host to access the private EC2.

7. Outcome

The project successfully demonstrates secure networking within AWS using VPC, public/private subnets, and a NAT Gateway. The setup ensures the private instance remains isolated while still being able to access the internet for updates and installations. The architecture adheres to best practices by restricting direct internet access to private resources.

AND DONE…