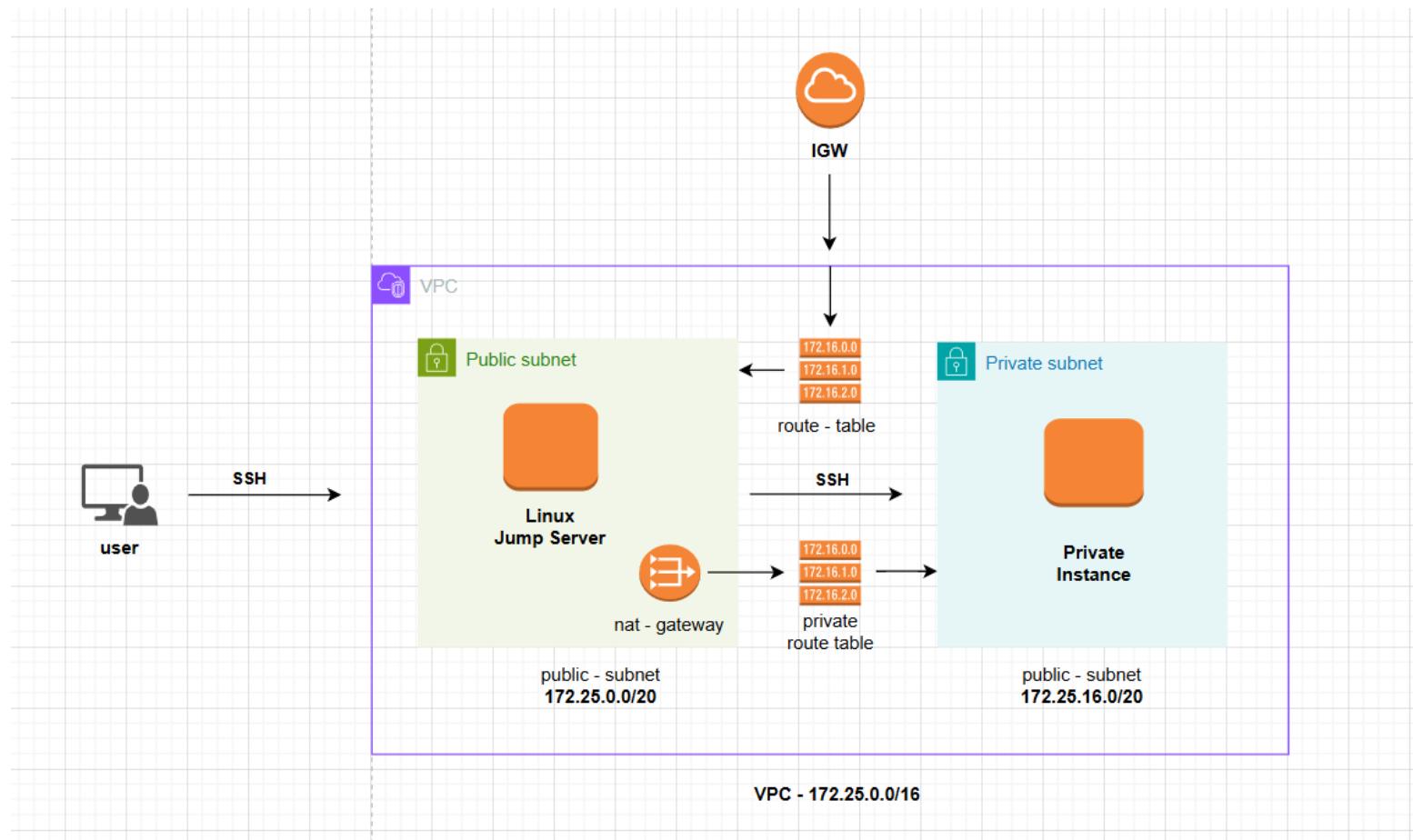


Project Title: Designing and Implementing a Private Network using VPC

Designing and Implementing a Private Network using AWS VPC:

This project demonstrates the design and implementation of a secure, scalable, and isolated private network architecture using Amazon Web Services (AWS) Virtual Private Cloud (VPC). The setup ensures efficient traffic routing, secure remote access, and controlled internet exposure for EC2 instances through strategic subnet design and access control mechanisms.



Key Components and Implementation

- **Virtual Private Cloud (VPC):**
 - Designed a custom VPC with CIDR block allocation to logically isolate network resources.
 - Created both **public** and **private subnets** to separate externally exposed and internal workloads.
- **Subnets & Route Tables:**
 - Configured separate route tables for public and private subnets to control traffic flow.
 - Enabled communication between subnets and managed routing to the internet where applicable.
- **Internet Gateway & NAT Gateway:**
 - Attached an Internet Gateway to the public subnet to allow internet traffic.
 - Set up a **NAT Gateway** to allow instances in the private subnet to access the internet **without exposing them publicly**.
- **EC2 Instances:**
 - Deployed an EC2 instance in the **private subnet** for secure and isolated computing.
 - Used AMIs, security groups, and IAM roles as per security and access requirements.
- **Bastion Host (Jump Box):**
 - Launched a Bastion Host in the public subnet to act as a secure bridge for accessing private instances.
 - Configured SSH key-based access, restricting remote access only via the Bastion Host to enhance security.

Security Configuration

- Configured **Security Groups** to:
 - Allow SSH only from trusted IPs to the Bastion Host.
 - Deny direct access to private subnet instances from the internet.
 - Permit only necessary inbound/outbound traffic per instance role.

Skills & Tools Used

- **AWS Services:** EC2, VPC, Subnets, Internet Gateway, NAT Gateway, Route Tables, Bastion Host, Security Groups
- **Networking Concepts:** CIDR, NAT, SSH Tunneling, Network Isolation
- **Security Best Practices:** Principle of least privilege, restricted SSH access, private instance isolation

Outcome

Successfully created a secure and robust private network environment in AWS. This design allows internal applications to run securely without public exposure while maintaining necessary external access through controlled and monitored pathways.

1. **EC2 Instance Setup:** Launched a **Jump Server in the public subnet** to enable SSH access, and a **Private EC2 instance in the private subnet**, configured with security groups to allow access only via the Jump Server for enhanced security.

The screenshot shows the AWS EC2 Instances page. The left sidebar is collapsed. The main area displays two instances:

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone
private - instance	i-0f41936eae9c42373	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1b
jump - server	i-074551944d573345a	Running	t2.micro	2/2 checks passed	View alarms	ap-south-1a

The 'private - instance' row is selected. Below it, the instance details for 'i-0f41936eae9c42373 (private - instance)' are shown:

Details			Status and alarms			Monitoring			Security			Networking			Storage			Tags			
Instance summary																					
Instance ID	i-0f41936eae9c42373		Public IPv4 address	-		Private IPv4 addresses	172.25.17.128		Public IPv4 DNS	-		Private IP DNS name (IPv4 only)	ip-172-25-17-128.ap-south-1.compute.internal		Hostname type	IP name: ip-172-25-17-128.ap-south-1.compute.internal		IP name: ip-172-25-17-128.ap-south-1.compute.internal	IP name: ip-172-25-17-128.ap-south-1.compute.internal	IP name: ip-172-25-17-128.ap-south-1.compute.internal	IP name: ip-172-25-17-128.ap-south-1.compute.internal

At the bottom of the page, there are various AWS service icons and a footer with copyright information and navigation links.

2. **VPC Setup:** Created a custom VPC (my-vpc) with CIDR block 172.25.0.0/16, enabling DNS resolution and subnet segmentation for secure and organized cloud networking.

The screenshot shows the AWS VPC Console interface. The top navigation bar includes tabs for 'Inbox', 'vpcs | VPC Console', 'Instances | EC2 | ap-south...', 'Gaurav9540/ansible', 'SCP Permission Denied', and 'Installing Ansible on specif...'. Below the navigation bar is the AWS navigation bar with links to IAM, S3, EC2, VPC, CloudFront, Aurora and RDS, Route 53, CloudWatch, Lambda, Amazon Simple Email Service, Simple Notification Service, and Simple Queue Service. The region is set to 'Asia Pacific (Mumbai)' and the user is 'bala'.

The main content area displays the 'VPC dashboard' under 'Virtual private cloud'. On the left, there's a sidebar with 'Your VPCs' (1/2) and a search bar. The main table lists two VPCs:

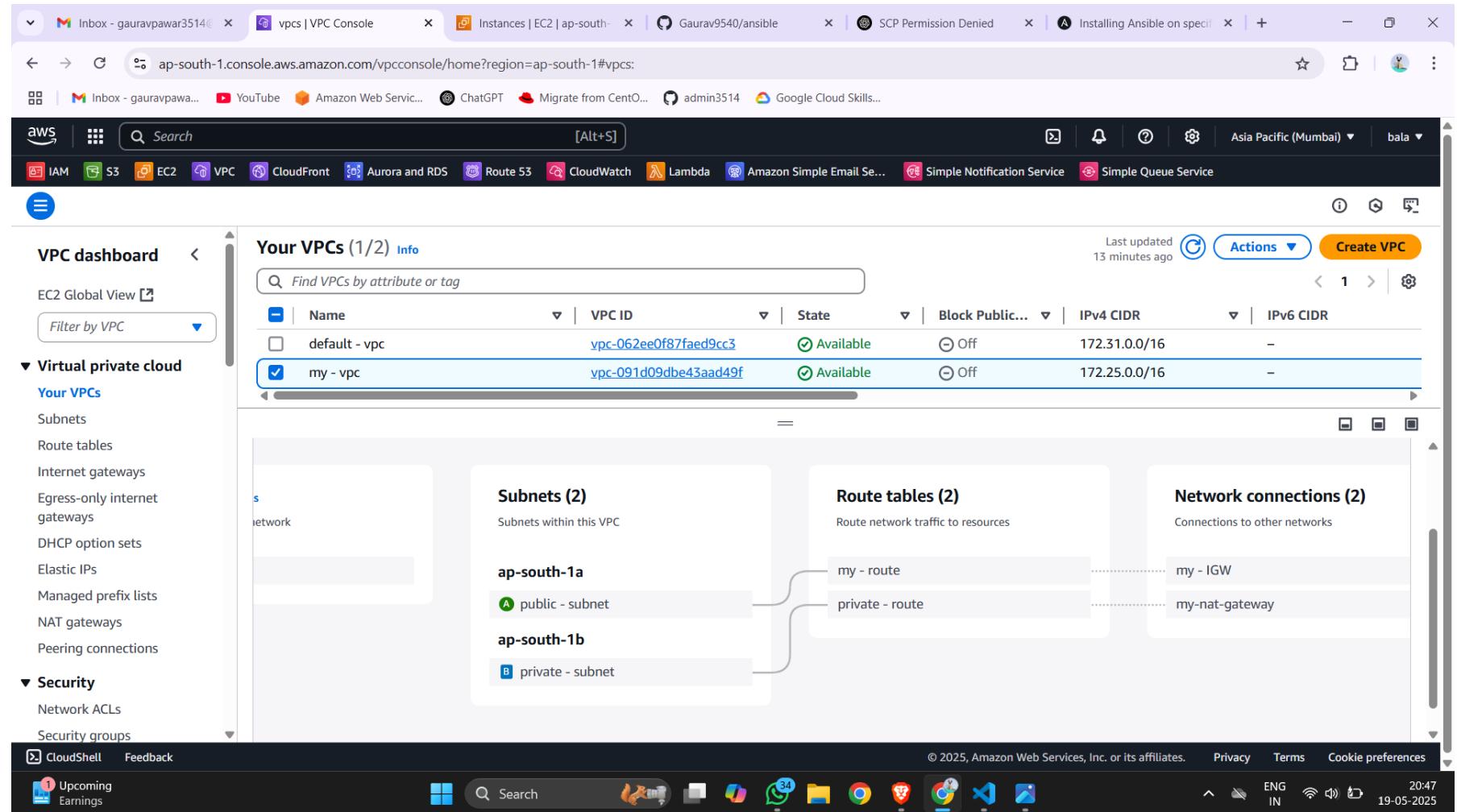
Name	VPC ID	State	Block Public Access	IPv4 CIDR	IPv6 CIDR
default - vpc	vpc-062ee0f87faed9cc3	Available	Off	172.31.0.0/16	-
my - vpc	vpc-091d09dbe43aad49f	Available	Off	172.25.0.0/16	-

Below the table, the details for the selected VPC ('my - vpc') are shown:

Details

VPC ID vpc-091d09dbe43aad49f	State Available	Block Public Access Off	DNS hostnames Disabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-0ccb82e00bb6e94b1	Main route table rtb-02d297bad60a419a6
Main network ACL acl-0461e234ec6bc4ced	Default VPC No	IPv4 CIDR 172.25.0.0/16	IPv6 pool -
IPv6 CIDR (Network border group)	Network Address Usage metrics	Route 53 Resolver DNS Firewall rule	Owner ID

At the bottom, there are links for 'CloudShell', 'Feedback', and a weather notification for 'Heavy rain Tomorrow'. The footer includes copyright information, privacy terms, cookie preferences, and system status indicators (ENG IN, 20:47, 19-05-2025).



3. Subnet Creation in a Custom VPC:

In the custom VPC my-vpc, two subnets were created to support a secure and segregated network architecture:

- **Public Subnet:**

A public subnet (public - subnet) with IPv4 CIDR block 172.25.0.0/20 was created in Availability Zone ap-south-1a. This subnet is designed to host resources that require internet access, such as a Bastion Host or NAT Gateway.

- **Private Subnet:**

A private subnet (private - subnet) with IPv4 CIDR block 172.25.16.0/20 was created in the same VPC. This subnet is intended for resources that should not be directly accessible from the internet, such as backend servers or databases.

These subnets are key to enabling secure and organized routing within the VPC, with the public subnet facilitating controlled external access and the private subnet maintaining internal-only connectivity.

Inbox - gauravpawar3514 ✎ subnets | VPC Console Instances | EC2 | ap-south-1 SCP Permission Denied Installing Ansible on specific

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

Inbox - gauravpawar3514 YouTube Amazon Web Service... ChatGPT Migrate from CentO... admin3514 Google Cloud Skills...

aws Search [Alt+S]

IAM S3 EC2 VPC CloudFront Aurora and RDS Route 53 CloudWatch Lambda Amazon Simple Email Service Simple Notification Service Simple Queue Service

VPC dashboard Subnets (1/5) Info Last updated 14 minutes ago Actions Create subnet

Find subnets by attribute or tag

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
public - subnet	subnet-0d895e1136839f612	Available	vpc-091d09dbe43aad49f my - vpc	Off	172.25.0.0/20
private - subnet	subnet-04b6bb3131ba144c5	Available	vpc-091d09dbe43aad49f my - vpc	Off	172.25.16.0/20

Subnet-0d895e1136839f612 / public - subnet

Details Flow logs Route table Network ACL CIDR reservations Sharing Tags

Details

Subnet ID	Subnet ARN	State	Block Public Access
subnet-0d895e1136839f612	arn:aws:ec2:ap-south-1:593793034916:subnet/subnet-0d895e1136839f612	Available	Off
IPv4 CIDR	IPv6 CIDR	IPv6 CIDR association ID	-
172.25.0.0/20	-	-	-
Availability Zone	Available IPv4 addresses	Network border group	VPC
ap-south-1a	4090	ap-south-1	vpc-091d09dbe43aad49f my - vpc
Route table	Availability Zone ID	Default subnet	Auto-assign public IPv4 address
rtb-02d207bad60a419a-6 my - route	aps1-az1	No	No

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Upcoming Earnings Search ENG IN 20:47 19-05-2025

4. Route Table Configuration in VPC

To enable proper routing and traffic management between public and private subnets in the my-vpc, two route tables were configured:

- **Public Route Table:**

This route table is associated with the **public subnet**.

- **Destination:** 0.0.0.0/0

- **Target:** Internet Gateway (IGW)

This configuration allows all outbound traffic from the public subnet to reach the internet via the Internet Gateway. It is typically used for resources like Bastion Hosts or NAT Gateways that require direct internet access.

- **Private Route Table:**

This route table is associated with the **private subnet**.

- **Destination:** 0.0.0.0/0

- **Target:** NAT Gateway

This allows resources in the private subnet (such as EC2 instances) to access the internet **for updates or external API calls**, while still keeping them **inaccessible from the internet**, thereby ensuring higher security.

Inbox - gauravpawar3514@gmail.com | RouteTables | VPC Console | Instances | EC2 | ap-south-1 | Gaurav9540/ansible | SCP Permission Denied | Installing Ansible on specific instance | - | ○ | X

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

Inbox - gauravpawar3514@gmail.com | YouTube | Amazon Web Services | ChatGPT | Migrate from CentOS | admin3514 | Google Cloud Skills...

aws | Search [Alt+S]

IAM | S3 | EC2 | VPC | CloudFront | Aurora and RDS | Route 53 | CloudWatch | Lambda | Amazon Simple Email Service | Simple Notification Service | Simple Queue Service

VPC dashboard < | Route tables (1/3) | Create route table

Last updated 10 minutes ago | Actions | Filter by VPC

Find route tables by attribute or tag

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
default - route	rtb-01158db7f0cdad2c0	-	-	Yes	vpc-062ee0f87faed9cc3 del
my - route	rtb-02d297bad60a419a6	subnet-0d895e1136839f...	-	Yes	vpc-091d09dbe43aad49f m
private - route	rtb-03b597748d43da2c7	subnet-04b6bb3131ba14...	-	No	vpc-091d09dbe43aad49f m

rtb-02d297bad60a419a6 / my - route

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

Both | Edit routes

Filter routes

Destination	Target	Status	Propagated
0.0.0.0/0	igw-081895b261874e08f	Active	No
172.25.0.0/16	local	Active	No

CloudShell | Feedback | © 2025, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

Upcoming Earnings | 1 | ENG IN | 20:47 | 19-05-2025

Inbox - gauravpawar3514@gmail.com | RouteTables | VPC Console | Instances | EC2 | ap-south-1 | Gaurav9540/ansible | SCP Permission Denied | Installing Ansible on specific | - | X

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

Inbox - gauravpawar3514@gmail.com YouTube Amazon Web Services ChatGPT Migrate from CentOS admin3514 Google Cloud Skills...

aws Search [Alt+S]

IAM S3 EC2 VPC CloudFront Aurora and RDS Route 53 CloudWatch Lambda Amazon Simple Email Service Simple Notification Service Simple Queue Service

VPC dashboard < Route tables (1/3) Info Last updated 10 minutes ago Actions Create route table

Find route tables by attribute or tag

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
default - route	rtb-01158db7f0cdad2c0	-	-	Yes	vpc-062ee0f87faed9cc3 default
my - route	rtb-02d297bad60a419a6	subnet-0d895e1136839f...	-	Yes	vpc-091d09dbe43aad49f my
private - route	rtb-03b597748d43da2c7	subnet-04b6bb3131ba14...	-	No	vpc-091d09dbe43aad49f my

rtb-03b597748d43da2c7 / private - route

Details Routes Subnet associations Edge associations Route propagation Tags

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	nat-0b393eeccdfeb45a8	Active	No
172.25.0.0/16	local	Active	No

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Upcoming Earnings ENG IN 20:47 19-05-2025

5. Internet Gateway (IGW) Configuration

As part of the VPC setup, an **Internet Gateway** was created and attached to the custom VPC (my-vpc) to enable internet access for resources in the public subnet.

- Purpose: The Internet Gateway allows instances in the **public subnet** to **send and receive traffic to and from the internet**. It is a **highly available, horizontally scaled, and redundant VPC component** that acts as the bridge between the VPC and the internet.
- Attachment: The IGW was **attached to my-vpc** and associated with the **public route table**.
- Route Table Entry:

In the **public route table**, a route was added with:

Destination: 0.0.0.0/0

Target: The attached **Internet Gateway**

This route enables all outbound internet traffic from instances in the public subnet to go through the IGW.

The Internet Gateway is essential for services like **bastion hosts, public-facing web servers**, or any resources that require **direct access to and from the internet**.

Inbox - gauravpawar3514@gmail.com | igws | VPC Console | Instances | EC2 | ap-south-1 | Gaurav9540/ansible | SCP Permission Denied | Installing Ansible on specific host | + | - | ☰ | X

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

M Inbox - gauravpawar3514@gmail.com YouTube Amazon Web Servic... ChatGPT Migrate from CentO... admin3514 Google Cloud Skills...

aws Search [Alt+S]

IAM S3 EC2 VPC CloudFront Aurora and RDS Route 53 CloudWatch Lambda Amazon Simple Email Service Simple Notification Service Simple Queue Service Asia Pacific (Mumbai) bala

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 Managed prefix lists NAT gateways Peering connections

Internet gateways (1/2) Info

Find internet gateways by attribute or tag

Name	Internet gateway ID	State	VPC ID	Owner
default-IGW	igw-0ea00b7cf14c786b0	Attached	vpc-062ee0f87faed9cc3 default-vpc	593793034916
my - IGW	igw-081895b261874e08f	Attached	vpc-091d09dbe43aad49f my-vpc	593793034916

Actions Create internet gateway

igw-081895b261874e08f / my - IGW

Details Tags

Details

Internet gateway ID igw-081895b261874e08f	State Attached	VPC ID vpc-091d09dbe43aad49f my-vpc	Owner 593793034916
--	-------------------	--	-----------------------

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Upcoming Earnings 20:48 ENG IN 19-05-2025

6. Elastic IP (EIP) Address Configuration

Elastic IP provides a static, public IPv4 address for dynamic cloud computing, allowing consistent access to an EC2 instance even after restarts.

The screenshot shows the AWS VPC dashboard with the 'Elastic IP addresses' section selected. The left sidebar shows navigation options like 'Your VPCs', 'Subnets', 'Route tables', etc. The main area displays a table of allocated elastic IP addresses:

Name	Allocated IPv4 address	Type	Allocation ID	Reverse DNS record
-	13.126.144.38	Public IP	eipalloc-05dccc03b09ffd6ec	-

Below the table, the IP address **13.126.144.38** is highlighted. A summary card provides detailed information about this specific IP:

Summary			
Allocated IPv4 address 13.126.144.38	Type Public IP	Allocation ID eipalloc-05dccc03b09ffd6ec	Reverse DNS record -
Association ID eipassoc-0057a2f9b393a540a	Scope VPC	Associated instance ID -	Private IP address 172.25.13.94

7. NAT Gateway Configuration

A NAT Gateway is deployed in a public subnet with an Elastic IP to enable private subnet instances to access the internet securely, without allowing inbound traffic.

This involves allocating an Elastic IP, selecting a public subnet for deployment, and updating route tables for private subnets to forward internet-bound traffic through the NAT Gateway.

The screenshot shows the AWS VPC dashboard with the 'NAT gateways' section selected. A single NAT gateway named 'my-nat-gateway' is listed. The details pane shows its configuration, including a primary public IPv4 address of 13.126.144.38 and a primary private IPv4 address of 172.25.13.94. The NAT gateway ID is nat-0b393eeccdfb45a8, and it is in a Public connectivity type with an Available state.

Name	NAT gateway ID	Connectivity...	State	Primary public I...	Primary private
my-nat-gateway	nat-0b393eeccdfb45a8	Public	Available	13.126.144.38	172.25.13.94

Details

NAT gateway ID nat-0b393eeccdfb45a8	Connectivity type Public	State Available	State message -
NAT gateway ARN arn:aws:ec2:ap-south-1:593793034916:natgateway/nat-0b393eeccdfb45a8	Primary public IPv4 address 13.126.144.38	Primary private IPv4 address 172.25.13.94	Primary network interface ID eni-04f0a54e4acb38527
VPC	Subnet subnet-0d895e1136839f612 / public - subnet	Created Monday, May 19, 2025 at 20:37:18 GMT+5:30	Deleted -

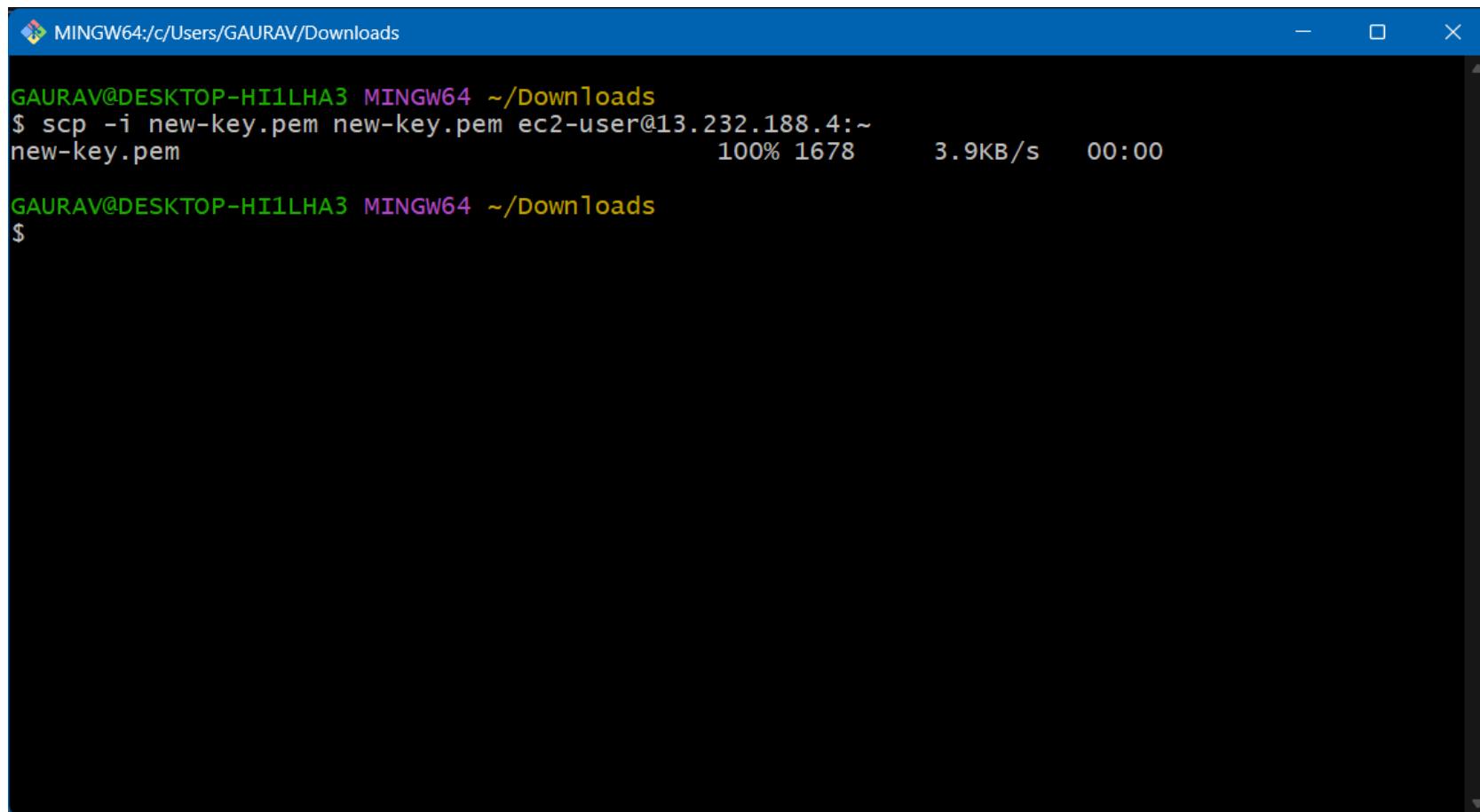
8. Transferring SSH Key to Jump Server for Private Instance Access

To securely access a private EC2 instance, the SSH private key is copied from the local machine to the jump server (bastion host) using the `scp` command. This allows the jump server to act as an intermediary for SSH connections into the private instance via key-based authentication.

The key transfer can be done with:

```
scp -i <local-private-key.pem> <key-to-transfer.pem> ubuntu@<jump-server-ip>:/home/ubuntu/
```

This command authenticates using the local key and securely transfers the private key to the jump server's home directory.



A screenshot of a terminal window titled "MINGW64:/c/Users/GAURAV/Downloads". The window shows a command being run: "scp -i new-key.pem new-key.pem ec2-user@13.232.188.4:~". The progress bar indicates 100% completion with a file size of 1678 and a transfer rate of 3.9KB/s, taking 00:00. Below the command, the prompt "\$" is visible.

9. SSH Access to Jump Server from Local Machine

To initiate a secure connection to the jump server (bastion host), SSH is used from the local machine with the appropriate private key. This allows access to internal network resources by forwarding traffic through the jump server. It's essential to ensure the private key permissions are correctly set (typically chmod 600) to prevent unauthorized access.

Here's the SSH command to connect to a jump server (bastion host) using a private key:

```
ssh -i <local-private-key.pem> ubuntu@<jump-server-ip>
```

The screenshot shows a Windows terminal window with a dark theme. The command `ssh -i new-key.pem ec2-user@172.25.17.128` is entered, followed by a warning about host fingerprint verification. The user types "yes" to proceed. The terminal then displays the Amazon Linux 2023 logo and URL. Finally, a ping command is run to 8.8.8.8, showing three successful packets transmitted with a round-trip time of approximately 2ms. The session ends with a control-C (^C).

```
ec2-user@ip-172-25-8-191:~ % ssh -i new-key.pem ec2-user@172.25.17.128
The authenticity of host '172.25.17.128 (172.25.17.128)' can't be established.
ED25519 key fingerprint is SHA256:LG/ybf0jD7GUH4vwft+FPDZfLobicVZqjJQMyqBNC8Mk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.25.17.128' (ED25519) to the list of known hosts.

[ec2-user@ip-172-25-8-191 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=114 time=2.71 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=114 time=2.31 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=114 time=2.49 ms
^C
--- 8.8.8.8 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 2.310/2.501/2.708/0.162 ms
[ec2-user@ip-172-25-8-191 ~]$ ls
[ec2-user@ip-172-25-8-191 ~]$ ls
```

System tray icons include a weather icon (Thunderstorm), a search bar, file explorer, messaging, and browser icons. The status bar at the bottom right shows the date (19-05-2025), time (20:44), and language (ENG IN).

10. SSH into a Private EC2 Instance via a Jump Server (Bastion Host)

To securely connect to a private EC2 instance (not publicly accessible), you can route your SSH session through a public-facing bastion host (jump server). This setup allows indirect access to the private instance without exposing it to the internet.

Here's how to do it using SSH agent forwarding or a ProxyJump:

```
ssh -i /path/to/private_key.pem -J user@bastion host user@private_instance_ip
```

11. Verifying Internet Connectivity on a Private Instance Using ping

After successfully SSH into the private instance through the jump server, it's important to confirm that the instance has internet access—especially if it's within a private subnet. This ensures that it can reach external services, download updates, or connect to APIs.

- ✓ Step to Check Internet Connectivity Once you're logged into the private instance:

ping 8.8.8.8 or ping www.google.com

The screenshot shows a Windows terminal window titled "ec2-user@ip-172-25-17-128~". The terminal output is as follows:

```
[ec2-user@ip-172-25-8-191 ~]$ ssh -i new-key.pem ec2-user@172.25.17.128
The authenticity of host '172.25.17.128 (172.25.17.128)' can't be established.
ED25519 key fingerprint is SHA256:7FTthhcxJ1A5ETuNRl6ZV9ZeFNA8xXYzSQL8fPu04Ag.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '172.25.17.128' (ED25519) to the list of known hosts.
@@@@@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####
@      WARNING: UNPROTECTED PRIVATE KEY FILE!      @
@@@@@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####@@@#####
Permissions 0444 for 'new-key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
Load key "new-key.pem": bad permissions
ec2-user@172.25.17.128: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
[ec2-user@ip-172-25-8-191 ~]$ ls -l
total 4
-r--r--r--. 1 ec2-user ec2-user 1678 May 19 15:15 new-key.pem
[ec2-user@ip-172-25-8-191 ~]$ chmod 400 new-key.pem
[ec2-user@ip-172-25-8-191 ~]$ ls -l
total 4
-r-----. 1 ec2-user ec2-user 1678 May 19 15:15 new-key.pem
[ec2-user@ip-172-25-8-191 ~]$ ssh -i new-key.pem ec2-user@172.25.17.128
#
~\_\_ #####\          Amazon Linux 2023
~~ \_\#\#\#\_
~~ \#\#\#
~~   '#/ ___ https://aws.amazon.com/linux/amazon-linux-2023
~~   V~' '->
~~   /
~~ .-.
~/ -/
~/m'
[ec2-user@ip-172-25-17-128 ~]$ ls
[ec2-user@ip-172-25-17-128 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
64 bytes from 8.8.8.8: icmp_seq=1 ttl=113 time=2.90 ms
64 bytes from 8.8.8.8: icmp_seq=2 ttl=113 time=2.17 ms
64 bytes from 8.8.8.8: icmp_seq=3 ttl=113 time=2.39 ms
64 bytes from 8.8.8.8: icmp_seq=4 ttl=113 time=2.45 ms
64 bytes from 8.8.8.8: icmp_seq=5 ttl=113 time=2.54 ms
```

The terminal window is part of a Windows desktop environment, as evidenced by the taskbar icons at the bottom, including a weather icon for "Heavy rain Tomorrow", a search bar, and various application icons like File Explorer, Microsoft Edge, and Visual Studio Code.

12. Performing Installations and Updates on a Private Instance After Verifying Internet Access

Once internet connectivity has been confirmed on the private instance using the ping command, you can proceed with various system operations that require external access. These include installing software packages, updating system dependencies, and downloading necessary files or configurations from the internet.

```
ec2-user@ip-172-25-17-128:~ ~ + ~ - X
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
Preparing : 1/1
Installing : apr-1.7.5-1.amzn2023.0.4.x86_64 1/12
Installing : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64 2/12
Installing : apr-util-1.6.3-1.amzn2023.0.1.x86_64 3/12
Installing : mailcap-2.1.49-3.amzn2023.0.3.noarch 4/12
Installing : httpd-tools-2.4.62-1.amzn2023.x86_64 5/12
Installing : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 6/12
Running scriptlet: httpd-filesystem-2.4.62-1.amzn2023.noarch 7/12
Installing : httpd-filesystem-2.4.62-1.amzn2023.noarch 7/12
Installing : httpd-core-2.4.62-1.amzn2023.x86_64 8/12
Installing : mod_http2-2.0.27-1.amzn2023.0.3.x86_64 9/12
Installing : mod_lua-2.4.62-1.amzn2023.x86_64 10/12
Installing : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch 11/12
Installing : httpd-2.4.62-1.amzn2023.x86_64 12/12
Running scriptlet: httpd-2.4.62-1.amzn2023.x86_64 12/12
Verifying : apr-1.7.5-1.amzn2023.0.4.x86_64 1/12
Verifying : apr-util-1.6.3-1.amzn2023.0.1.x86_64 2/12
Verifying : apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64 3/12
Verifying : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch 4/12
Verifying : httpd-2.4.62-1.amzn2023.x86_64 5/12
Verifying : httpd-core-2.4.62-1.amzn2023.x86_64 6/12
Verifying : httpd-filesystem-2.4.62-1.amzn2023.noarch 7/12
Verifying : mod_http2-2.0.27-1.amzn2023.0.3.x86_64 8/12
Verifying : mod_lua-2.4.62-1.amzn2023.x86_64 9/12
Verifying : libbrotli-1.0.9-4.amzn2023.0.2.x86_64 10/12
Verifying : mailcap-2.1.49-3.amzn2023.0.3.noarch 11/12
Verifying : httpd-2.4.62-1.amzn2023.x86_64 12/12
Verifying : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch 12/12
Installed:
apr-1.7.5-1.amzn2023.0.4.x86_64           apr-util-1.6.3-1.amzn2023.0.1.x86_64           apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64
generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch   httpd-2.4.62-1.amzn2023.x86_64           httpd-core-2.4.62-1.amzn2023.x86_64
httpd-filesystem-2.4.62-1.amzn2023.noarch       httpd-tools-2.4.62-1.amzn2023.x86_64           libbrotli-1.0.9-4.amzn2023.0.2.x86_64
mailcap-2.1.49-3.amzn2023.0.3.noarch         mod_http2-2.0.27-1.amzn2023.0.3.x86_64           mod_lua-2.4.62-1.amzn2023.x86_64
Complete!
[ec2-user@ip-172-25-17-128 ~]$ |
```



Heavy rain Tomorrow

Search ENG IN 20:46 19-05-2025