

Network Mastery with AWS VPC

Author: Oluwaseun Osunsola

Environment: AWS

Project Link: <https://github.com/Oluwaseunoa/DevOps-Projects/tree/main>

Project Overview

In this session, we explore the core concepts of Amazon Web Services (AWS), focusing specifically on Virtual Private Clouds (VPCs). Our objective is to understand the fundamental components of VPC infrastructure, including subnets, gateways, and routing tables. Through practical demonstrations and interactive exercises, we navigate the AWS Management Console to deploy and manage these critical components effectively.

Before proceeding with setting up VPCs, ensure a solid understanding of cloud networking basics.

Project Goals

- Understand the fundamentals of Virtual Private Cloud (VPC) and its components.
- Gain hands-on experience in setting up and configuring VPC, subnets, Internet Gateway, NAT Gateway, and VPC peering connections.
- Learn how to enable internet connectivity securely within a VPC.
- Implement outbound internet access through the NAT Gateway.
- Establish direct communication between VPCs using VPC peering.

Learning Outcomes

- Acquired knowledge about VPC and its essential components, such as subnets, gateways, and route tables.
- Developed skills in creating and configuring VPC resources using the AWS Management Console.
- Learned how to set up routing tables to enable internet connectivity and outbound internet access securely.
- Gained understanding of VPC peering and its significance in connecting multiple VPCs within the same or different regions.
- Enhanced understanding of network security principles and best practices for cloud environments.

Key Concepts

What is VPC, Subnets, Internet Gateway, and NAT Gateway?

Imagine building a virtual space for the company GatoGrowFast.com so that its computers can communicate securely. That's what a **Virtual Private Cloud (VPC)** is all about—a private room in the cloud for GatoGrowFast.com's use.

Example: Think of GatoGrowFast.com's office building with different departments like HR, Finance, and IT, each with its own area and access rules. In a VPC, GatoGrowFast.com creates different sections, called **subnets**, for different parts of the business.

To connect its office to the internet, GatoGrowFast.com uses a router to control data flow. In a VPC, this is achieved with an **Internet Gateway**, allowing secure communication with the internet.

A **NAT (Network Address Translation) Gateway** acts as a secret agent between GatoGrowFast.com's computers and the internet. When a computer in the virtual office communicates with the internet, the NAT Gateway translates the message and hides the sender's identity, similar to sending a letter without a return address. This keeps GatoGrowFast.com's computers safe and anonymous online.

Note: A **router** directs data packets between networks, acting like a traffic cop for the internet. Data is broken into packets, and the router uses **routing tables**—like maps of the internet—to determine the best path for these packets based on destination IP addresses.

What is an IP Address?

An **IP address** is like a phone number for a computer, a unique set of numbers that helps devices find and communicate with each other on a network, like the internet.

Types of IP Addresses:

- **Public IP Address:** Like a home address, it's unique and allows other computers on the internet to find your device. Assigned by an Internet Service Provider (ISP), it enables global communication. Public IPs can be **dynamic** (changing periodically) or **static** (constant, used for servers or consistent connectivity).
- **Private IP Address:** Like an internal extension number in an office, used for communication within a specific network (e.g., home Wi-Fi or office network). Assigned by a router or DHCP server, private IPs are not routable over the internet and can be reused across different private networks without conflict.

IP Address Versions:

- **IPv4:** The most common type, a 32-bit numeric address written in decimal format (e.g., 192.168.0.1). Each octet ranges from 0 to 255, divided into classes A, B, and C for host addressing.
- **IPv6:** Designed to replace IPv4 due to address exhaustion, a 128-bit hexadecimal address (e.g., 2001:0db8:85a3:0000:0000:8a2e:0370:7334), offering a vast address space.

What is CIDR?

CIDR (Classless Inter-Domain Routing) simplifies talking about groups of IP addresses. Instead of listing each address, CIDR uses a shorthand, like saying "all houses on Main Street" instead of naming each house.

Example: For the IP address 192.168.1.0, CIDR notation like 192.168.1.0/24 refers to all addresses from 192.168.1.0 to 192.168.1.255.

Calculating Available IP Addresses in a CIDR Block: Use the formula: $2^{(32 - \text{CIDR notation})} - 2$ (subtracting 2 for the network and broadcast addresses).

Example: For 192.168.1.0/24:

- $2^{(32 - 24)} - 2 = 2^8 - 2 = 256 - 2 = 254$ available IP addresses.

What is a Gateway?

Gateways are like doorways between networks, enabling data to travel between a local network and others, like the internet. They act as a traffic cop, directing data to its destination.

Example: In a city with neighborhoods, to visit a friend in Neighborhood B from Neighborhood A, you pass through a gateway connecting the two. Similarly, a network gateway connects your local network to the internet.

What is a Route Table?

A **route table** is like a map guiding data around a network. It lists destinations and paths (routes) to reach them. Devices consult the route table to determine where to send data packets.

Example: To send data to a website, a computer checks its route table to find the gateway to the internet. The router then forwards the data to the next stop, ensuring it reaches its destination.

Connection Between Gateway and Route Table

- **Gateways:** Devices (e.g., routers, firewalls) serving as entry/exit points between networks with different IP ranges. They receive packets and determine the next destination based on routing information.
- **Route Tables:** Maintained by networking devices, they list destination networks and the next hop (gateway) to reach them. Devices use route tables to find the best path for packets.
- **Connection:** When a device sends data outside its local network, it checks the route table to identify the gateway. The packet is forwarded to the gateway, which continues routing until the packet reaches its destination.

Difference Between Internet Gateway and NAT Gateway

- **Internet Gateway:** A door to the internet for a subnet, allowing resources (e.g., EC2 instances) to send and receive internet traffic. It enables two-way communication.
- **NAT Gateway:** A one-way street for subnet traffic, allowing resources to access the internet without allowing incoming internet traffic, enhancing security.

What is VPC Peering?

VPC peering connects two virtual offices (VPCs) in the cloud for direct communication, like neighboring offices sharing files without a middleman. By default, EC2 instances in different VPCs cannot communicate. VPC peering establishes a direct network connection between VPCs.

Why VPC Peering?

It enables different parts of a cloud network (e.g., development and marketing VPCs) to share data securely and efficiently.

Key Points:

- VPCs require peering, VPN, or AWS Direct Connect for connectivity.
- Subnets within the same VPC communicate by default via AWS-configured route tables.
- EC2 instances in the same or different subnets within a VPC communicate if security group rules and route tables allow.
- **VPC Peering Basics:**

- Allows direct communication using private IP addresses.
- Supports same or different regions and AWS accounts.
- CIDR blocks must not overlap.
- Requires proper Security Group and Network Access Control List (NACL) configurations.
- No transitive traffic (traffic cannot flow through a peered VPC to another VPC).
- Route tables must include routes for the peer VPC's CIDR block.
- Limits exist on the number of peering connections and route entries.

What is a VPC Endpoint?

A **VPC endpoint** is a secure, dedicated tunnel between a VPC and an AWS service (e.g., S3), bypassing the public internet. It ensures private, safe access to resources.

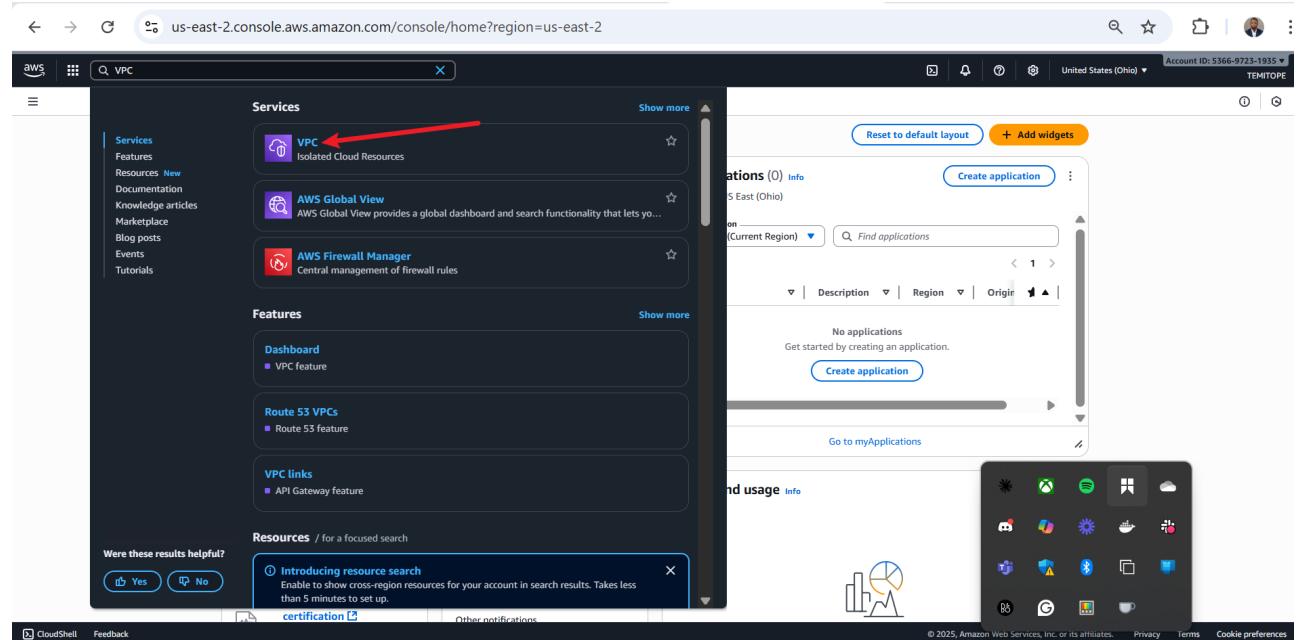
Problem Context: Backing up data from an EC2 instance to an S3 bucket typically goes over the internet, risking sensitive data exposure. A VPC endpoint creates a private connection, keeping data secure.

Note: An **EC2 instance** is a virtual server in AWS for running applications, offering scalable computing power for hosting websites, software, or data processing.

Practical Steps

Part 1: Setting Up a Virtual Private Cloud (VPC)

1. Navigate to the search bar, enter 'VPC', and click on the VPC service.



2. On the VPC dashboard, click **Create VPC**.

The screenshot shows the AWS VPC dashboard. At the top center, there is a blue button labeled "Create VPC" with a yellow oval around it and a red arrow pointing to it. Below this button is another blue button labeled "Launch EC2 Instances". The dashboard displays various resources by region, including VPCs, Subnets, Route Tables, Internet Gateways, and NAT Gateways, all currently set to Ohio. On the right side, there are sections for "Service Health", "Settings", and "Additional Information". The bottom of the screen shows the AWS navigation bar with CloudShell, Feedback, and other links.

3. Select **VPC only**, set the IPv4 CIDR block (e.g., 10.0.0.0/16), and click **Create VPC**.

The screenshot shows the "Create VPC" configuration page. In the "VPC settings" section, the "Resources to create" dropdown is set to "VPC only" (indicated by a red arrow). Below this, the "Name tag - optional" field contains "my-vpc-01". Under "IPv4 CIDR block", the input field shows "10.0.0.0/16" (also indicated by a red box). In the bottom right corner of the page, there is a large orange button labeled "Create VPC" with a red arrow pointing to it. Other buttons visible include "Cancel", "Preview code", and "Add tag".

4. VPC successfully created.

The screenshot shows the AWS VPC console interface. A green success message at the top states "You successfully created vpc-059fcaa09dbb14e12". Below this, the VPC details are listed:

VPC ID vpc-059fcaa09dbb14e12	State Available	Block Public Access Off	DNS hostnames Disabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-09149d5972dcad56b	Main route table rtb-01b8c1967af12ebbo
Main network ACL acl-0dd802252bcd3adcb	Default VPC	IPv4 CIDR 10.0.0.0/16	IPv6 pool -
IPv6 CIDR -		Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -

Below the details, there are tabs for Resource map, CIDRs, Flow logs, Tags, and Integrations. The Resource map tab is selected, showing a summary of resources:

- VPC: Your AWS virtual network (vpc-059fcaa09dbb14e12)
- Subnets (0): Subnets within this VPC
- Route tables (1): Route network traffic to resources (rtb-01b8c1967af12ebbo)

At the bottom right, there is a "Show all details" link and a small icon bar with various AWS services.

5. **Note:** If a CIDR block size error occurs, ensure the block size is between /16 and /28.

IPv4 CIDR

10.0.0.0/8

⚠ CIDR block size must be between /16 and /28.

CIDR block size must be between /16 and /28.

IPv4 CIDR

10.0.0.0/16

CIDR block size must be between /16 and /28.

Part 2: Configuring Subnets within the VPC

1. From the created VPC, click **Subnets** to create a subnet.

The screenshot shows the AWS VPC console interface. In the left sidebar, under 'Virtual private cloud', the 'Your VPCs' section is expanded, and the 'Subnets' link is highlighted with a red box. At the top right, there is a 'Actions' dropdown menu with a red arrow pointing to the 'Create VPC' button. The main content area displays a table titled 'Your VPCs (1/2) Info' with two rows of VPC information. The first row is selected and has a blue border. The second row is 'vpc-0946dc2d5bf4cbe'. The table columns include Name, VPC ID, State, Block Public Access, IPv4 CIDR, IPv6 CIDR, DHCP option set, and Main route table.

2. On the subnet dashboard, click **Create subnet**.

The screenshot shows the AWS VPC console interface. In the left sidebar, under 'Virtual private cloud', the 'Your VPCs' section is expanded, and the 'Subnets' link is highlighted with a red box. At the top right, there is a 'Actions' dropdown menu with a red arrow pointing to the 'Create subnet' button. The main content area displays a table titled 'Subnets (3) Info' with three rows of subnet information. All subnets are listed as 'Available'. The table columns include Subnet ID, State, VPC, Block Public Access, IPv4 CIDR, IPv6 CIDR, and IPv6 CIDR. Below the table, a section titled 'Select a subnet' is visible.

3. Select the VPC created in Part 1.

The screenshot shows the 'Create subnet' wizard in the AWS VPC console. The first step, 'Select a VPC', is displayed. A dropdown menu lists two VPCs: 'vpc-059fcaa09dbb14e12' (selected) and 'vpc-0946gddc2d5bf4cbe'. The selected VPC has its CIDR range '10.0.0.0/16' highlighted with a red box. The status '(default)' is shown next to the second VPC. Below the dropdown, a note says 'Select a VPC first to create new subnets.' At the bottom right are 'Cancel' and 'Create subnet' buttons.

4. Enter subnet name (e.g., **my-public-subnet-1**), set availability zone, specify IPv4 CIDR block (e.g., 10.0.1.0/24), and click **Add new subnet**.

The screenshot shows the 'Subnet Settings' step of the 'Create subnet' wizard. The 'Subnet 1 of 1' section is visible. The 'Subnet name' field contains 'my-public-subnet-1', which is highlighted with a red box. The 'Availability Zone' dropdown shows 'United States (Ohio) / use2-ad1 (us-east-2a)', also highlighted with a red box. The 'IPv4 VPC CIDR block' dropdown shows '10.0.0.0/16'. The 'IPv4 subnet CIDR block' dropdown shows '10.0.6.0/24', which is highlighted with a red box. In the 'Tags - optional' section, a single tag 'Name: my-public-subnet-1' is listed. A red arrow points from the 'Add new subnet' button at the bottom left to the 'Create subnet' button at the bottom right.

5. Enter subnet name (e.g., **my-private-subnet-1**), set availability zone, specify IPv4 CIDR block (e.g., 10.0.2.0/24), and click **Create subnet**.

Subnet 2 of 2

Subnet name
Create a tag with the key of 'Name' and a value that you specify.
 The name can be up to 256 characters long.

Availability Zone Info
Choose the zone in which your subnet will reside, or let Amazon choose one for you.

IPv4 VPC CIDR block Info
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

IPv4 subnet CIDR block
 256 IPs
◀ ▶ ▲ ▼

Tags - optional

Key Value - optional

You can add 49 more tags.

6. View the architecture.



7. Subnets **my-public-subnet-1** and **my-private-subnet-1** successfully created.

The screenshot shows the AWS VPC Subnets page. A green success message at the top states: "You have successfully created 2 subnets: subnet-04e9903eb59663bdb, subnet-0bffd09a345bd2e1f". The main table lists two subnets:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR
my-private-subnet-1	subnet-0bffd09a345bd2e1f	Available	vpc-059fcfa09dbb14e12	Off	10.0.7.0/24	-	-
my-public-subnet-1	subnet-04e9903eb59663bdb	Available	vpc-059fcfa09dbb14e12	Off	10.0.6.0/24	-	-

A red arrow points to the "my-public-subnet-1" row. The left sidebar shows the "Subnets" section under "Virtual private cloud".

Part 3: Creating Internet Gateway and Attaching it to VPC

1. Click **Internet Gateways** to connect the public subnet to the internet.

The screenshot shows the AWS VPC Subnets page. A red arrow points to the "Internet gateways" link in the left sidebar under the "Virtual private cloud" section. The main table lists the same two subnets as before:

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR	IPv6 CIDR
my-private-subnet-1	subnet-0bffd09a345bd2e1f	Available	vpc-059fcfa09dbb14e12	Off	10.0.7.0/24	-	-
my-public-subnet-1	subnet-04e9903eb59663bdb	Available	vpc-059fcfa09dbb14e12	Off	10.0.6.0/24	-	-

The "Select a subnet" dropdown is visible below the table. The left sidebar shows the "Subnets" section under "Virtual private cloud".

2. Click **Create Internet Gateway**.

The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#igws. The left sidebar is open, showing sections like '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, Route servers), 'Security' (Network ACLs, Security groups), and 'PrivateLink and Lattice' (Getting started, Endpoints, Endpoint services, Service networks). The main area displays 'Internet gateways (1) Info' with a table showing one entry: Name (igw-037e56e68b137c83b), Internet gateway ID (igw-037e56e68b137c83b), State (Attached), VPC ID (vpc-0a946cd2d5bf4cbe), and Owner (536697231935). Below the table is a section titled 'Select an internet gateway above' with a grid of icons representing different AWS services. The top right corner of the Actions dropdown menu has a red arrow pointing to the 'Create internet gateway' button.

3. Name the Internet Gateway and click **Create Internet Gateway**.

The screenshot shows the 'Create internet gateway' wizard with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#CreateInternetGateway. The left sidebar is open, showing the 'Internet gateways' section. The main area has two sections: 'Internet gateway settings' (Name tag: my-internet-gw-1) and 'Tags - optional' (Key: Name, Value: optional, my-internet-gw-1). At the bottom right is a large orange 'Create internet gateway' button with a red arrow pointing to it. The bottom of the screen shows the standard AWS footer with links for CloudShell, Feedback, and various service icons.

4. Internet Gateway created, note it is detached.

The screenshot shows the AWS VPC console with the URL <https://us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#InternetGateway:id=igw-05cba0c2e0b35021e>. The page displays a message: "The following internet gateway was created: igw-05cba0c2e0b35021e - my-internet-gw-1. You can now attach to a VPC to enable the VPC to communicate with the internet." Below this, the Internet gateway details are shown: ID (igw-05cba0c2e0b35021e), Name (my-internet-gw-1), State (Detached, highlighted with a red box), VPC ID (-), Owner (536697231935), and Tags (my-internet-gw-1). The Actions menu is visible on the right.

5. Click Actions and Attach to VPC.

The screenshot shows the same AWS VPC console page as above. A red arrow points to the "Actions" button in the top right corner of the main content area. The "Attach to VPC" option is visible in the dropdown menu.

6. Select the VPC and click Attach Internet Gateway.

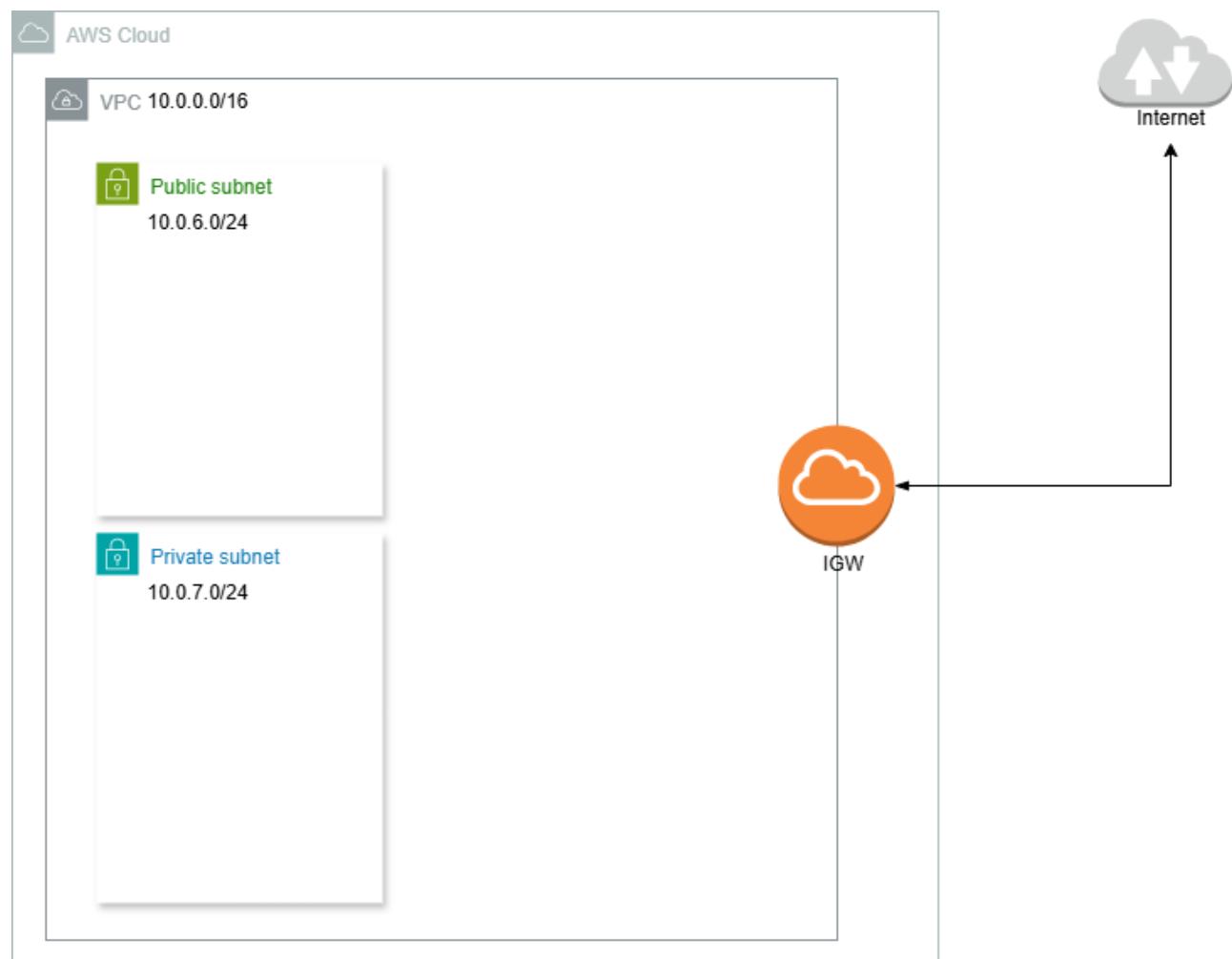
The screenshot shows the "Attach to VPC" dialog box. It includes fields for selecting a VPC (with "vpc-059fcaa09dbb14e2" selected) and an "AWS Command Line Interface command". A red arrow points to the "Attach internet gateway" button at the bottom right of the dialog.

7. Internet Gateway successfully attached to VPC.

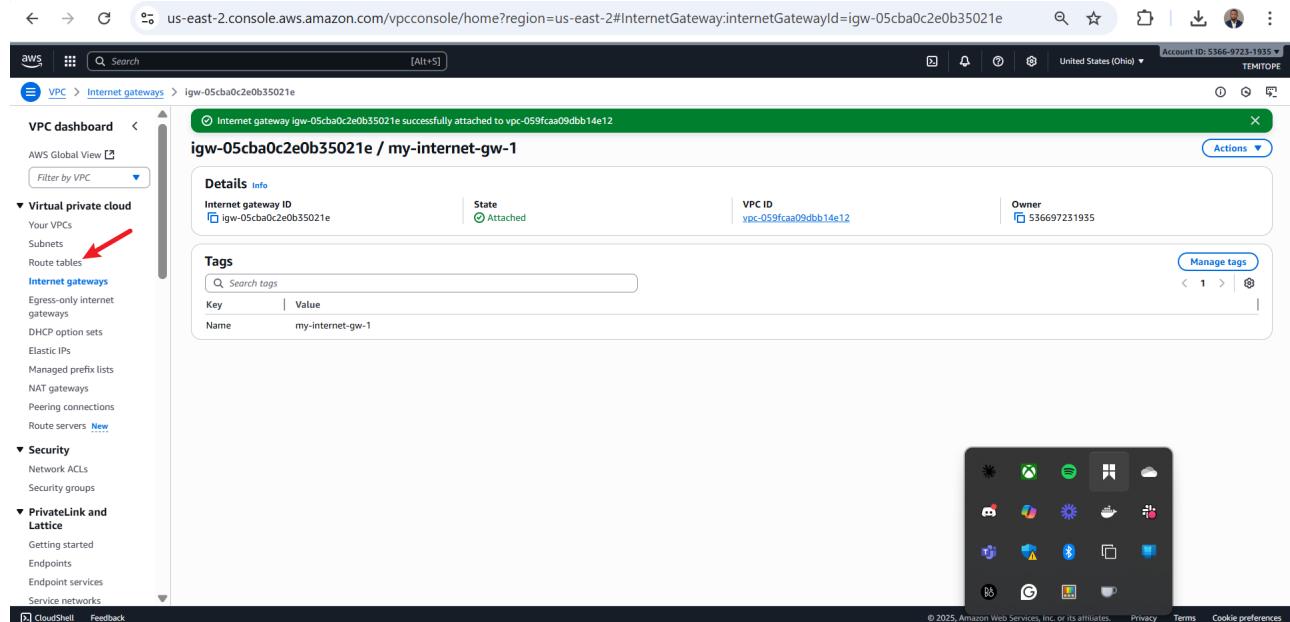
The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#InternetGateway:id=igw-05cba0c2e0b35021e. The main content area displays an Internet gateway named "igw-05cba0c2e0b35021e / my-internet-gw-1". A green banner at the top indicates that the gateway has been successfully attached to a VPC. The "State" field is highlighted with a red box and shows "Attached". The "VPC ID" is listed as "vpc-059fcaa09dbb14e12". The "Owner" is listed as "536697231935". On the left sidebar, the "Internet gateways" section is selected. A small screenshot of a Windows desktop environment is visible in the bottom right corner.

Part 4: Enabling Internet Connectivity with the Internet Gateway by Setting Up Routing Tables

1. View the current architecture.

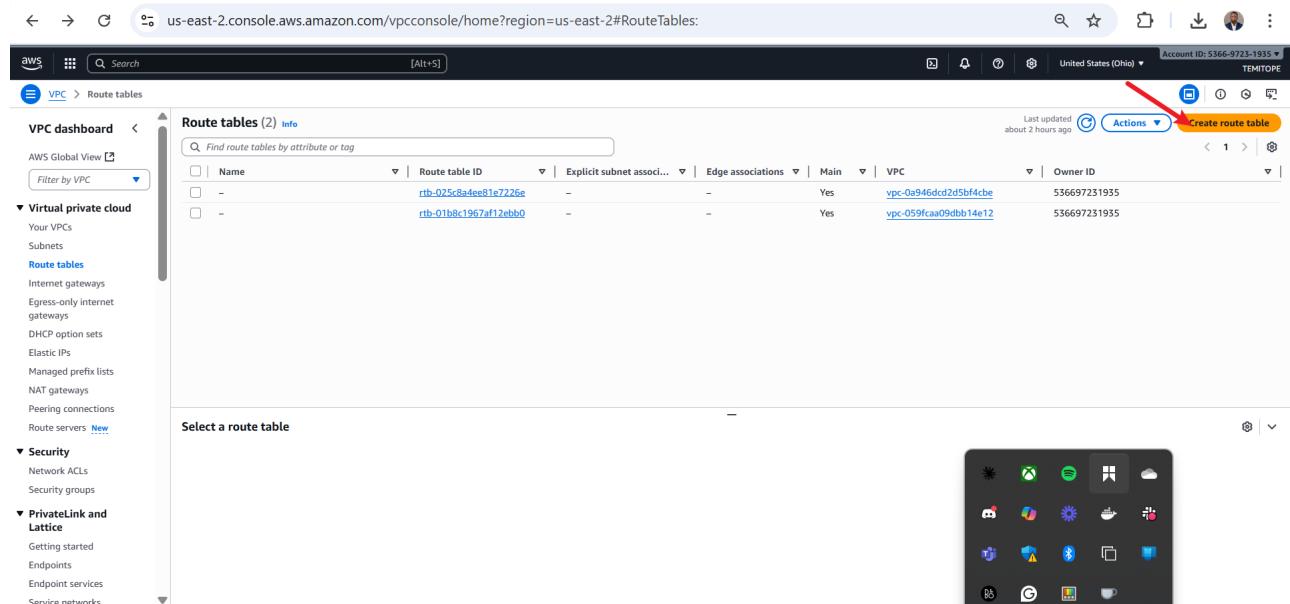


2. Click Route Tables.



The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpc/home?region=us-east-2#InternetGateway:internetGatewayId=igw-05cba0c2e0b35021e. The main content area displays information about an Internet gateway named 'igw-05cba0c2e0b35021e / my-internet-gw-1'. It includes details like Internet gateway ID (igw-05cba0c2e0b35021e), State (Attached), VPC ID (vpc-059fcaa09dbb14e12), and Owner (536697231935). Below this, there's a 'Tags' section with one tag: Name = my-internet-gw-1. The left sidebar has a tree view with 'Route tables' highlighted and a red arrow pointing to it. Other options in the sidebar include 'Virtual private cloud' (Your VPCs, Subnets, Route tables), 'Security' (Network ACLs, Security groups), and 'PrivateLink and Lattice' (Getting started, Endpoints, Endpoint services, Service networks).

3. Click Create route table.



The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpc/home?region=us-east-2#RouteTables. The main content area shows a table titled 'Route tables (2) Info' with two entries. The columns include Name, Route table ID, Explicit subnet associ..., Edge associations, Main, VPC, and Owner ID. The first entry is 'rtb-025c8adeeb1e7226e' associated with 'vpc-0a946cd2d5bf4cbe' and owner '536697231935'. The second entry is 'rtb-01b8c1967af12eb0' associated with 'vpc-059fcaa09dbb14e12' and owner '536697231935'. Below the table, there's a section titled 'Select a route table'. The left sidebar is identical to the previous screenshot, showing 'Route tables' selected in the 'Virtual private cloud' section.

4. Name the route table, select the VPC, and click **Create route table**.

aws [Alt+S]

VPC > Route tables > Create route table

Create route table Info

A route table specifies how packets are forwarded between the subnets within your VPC, the internet, and your VPN connection.

Route table settings

Name optional
Create a tag with a key of 'Name' and a value that you specify.

VPC
The VPC to use for this route table.

Tags
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 49 more tags.

5. Click **Subnet associations**, then **Edit subnet associations**.

aws [Alt+S]

VPC > Route tables > rtb-0d41bba2ed11c8401

rtb-0d41bba2ed11c8401 / my-route-table-1

Details Info

Route table ID:
Main: No
Owner ID:

Explicit subnet associations: None

Edge associations: None

Routes **Subnet associations** **Edge associations** **Route propagation** **Tags**

Explicit subnet associations (0)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
<small>No subnet associations</small> You do not have any subnet associations.			

Edit subnet associations

Subnets without explicit associations (2)

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
my-public-subnet-1	subnet-0480c4a5151b0c74c	10.0.7.0/24	-
my-public-subnet-1	subnet-0ff5b95c2043b1a6f	10.0.6.0/24	-

Edit subnet associations

6. Select the public subnet and click **Save associations**.

The screenshot shows the 'Edit subnet associations' page for a route table. In the 'Available subnets' section, 'my-public-subnet-1' is selected. In the 'Selected subnets' section, 'subnet-04e9903eb59663bdb / my-public-subnet-1' is listed. A red arrow points to the 'Save associations' button at the bottom right.

7. Under **Routes** tab, click **Edit routes**.

The screenshot shows the 'rtb-0d41bba2ed11c8401 / my-route-table-1' details page. The 'Routes' tab is selected, showing one route entry: Destination 10.0.0.0/16, Target local, Status Active, Propagated No, and Route Origin CreateRouteTable. A red arrow points to the 'Edit routes' button at the top right of the routes table.

8. Click **Add route**.

The screenshot shows the 'Edit routes' page for the same route table. The 'Add route' button is highlighted with a red arrow. The table shows one existing route entry: Destination 10.0.0.0/16, Target local, Status Active, Propagated No, and Route Origin CreateRouteTable.

9. Set Destination to **0.0.0.0/0**, Target to the created Internet Gateway, and save changes.

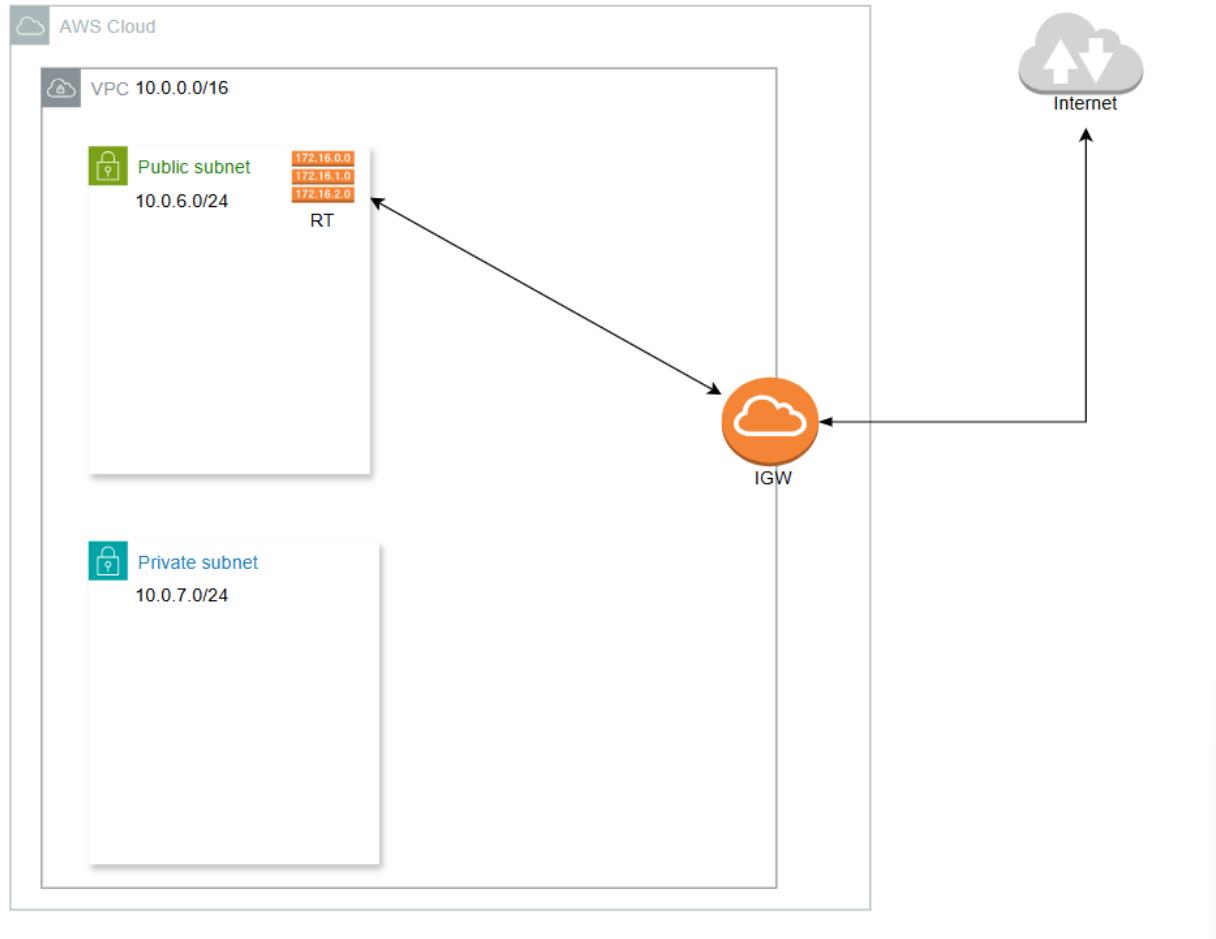
The screenshot shows the 'Edit routes' page for a specific route table. The 'Destination' field is set to '0.0.0.0/0'. The 'Target' dropdown is set to 'Internet Gateway', and the list shows 'igw-05cba0c2e0b35021e' selected. The 'Save changes' button is highlighted with a red arrow.

10. Route table updated successfully.

The screenshot shows the 'Route table details' page for 'rtb-0d41bba2ed11c8401 / my-route-table-1'. It displays the route table ID, VPC, and explicit subnet associations. The 'Routes' tab is active, showing two routes: one for '0.0.0.0/0' pointing to 'igw-05cba0c2e0b35021e' and another for '10.0.0.16' pointing to 'local'. The 'Actions' dropdown is open on the right.

Part 5: Enabling Outbound Internet Access through NAT Gateway

1. View the current VPC architecture.



2. Click NAT Gateways.

Screenshot of the AWS VPC Route Tables page for route table **rtb-0d41bba2ed11c8401 / my-route-table-1**.

Details

- Route table ID: **rtb-0d41bba2ed11c8401**
- Main: **No**
- Owner ID: **536697231935**
- Explicit subnet associations: **subnet-04e9903eb59663bdb / my-public-subnet-1**
- Edge associations: **-**

Routes (2)

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-05cba0c2e0b35021e	Active	No	Create Route
10.0.0.16	local	Active	No	Create Route Table

A red arrow points to the **NAT gateways** link in the left sidebar under the **Route tables** section.

3. Click **Create NAT Gateway**.

The screenshot shows the AWS VPC NAT gateways page. On the left, there's a navigation sidebar with sections like 'Virtual private cloud', 'Security', and 'PrivateLink and Lattice'. The main area is titled 'NAT gateways info' and shows a table with columns: Name, NAT gateway ID, Connectivity..., State, State message, Primary public I..., Primary private I..., Primary network..., and VPC. A search bar at the top says 'Find NAT gateways by attribute or tag'. At the top right, there are 'Actions' and 'Create NAT gateway' buttons. A red arrow points to the 'Create NAT gateway' button.

4. Name the NAT Gateway, select the private subnet, set connectivity type to **Private**, and click **Create NAT Gateway**.

The screenshot shows the 'Create NAT gateway' wizard. The first step is 'NAT gateway settings'. It has fields for 'Name - optional' (with 'my-NAT-gw-1' entered), 'Subnet' (selected as 'subnet-0bffd09a345bd2e1f (my-private-subnet-1)'), and 'Connectivity type' (set to 'Private'). A note below says 'Private NAT gateway traffic can't reach the internet.' The next step, 'Additional settings', is partially visible. At the bottom, there's a 'Tags' section where a tag 'Name: my-NAT-gw-1' is added. The final step is 'Create NAT gateway', which is highlighted with a red arrow.

5. NAT Gateway created successfully.

The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#NatGatewayDetails:natGatewayId=nat-02527773795a97d0f. The main page displays a green success message: "NAT gateway nat-02527773795a97d0f | my-NAT-gw-1 was created successfully." Below this, the "Details" tab for the NAT gateway is selected, showing the following information:

- NAT gateway ID:** nat-02527773795a97d0f
- Connectivity type:** Private
- Primary public IPv4 address:** -
- Subnet:** subnet-0bffd09a345bd2e1f / my-private-subnet-1
- State:** Pending
- Primary private IPv4 address:** 10.0.7.35
- Created:** Wednesday, October 8, 2025 at 17:34:18 GMT+1
- State message:** -
- Primary network interface ID:** eni-0ba2fe8358ad3b490
- Deleted:** -

The "Secondary IPv4 addresses" tab is also visible, showing a search bar and a note: "Secondary IPv4 addresses are not available for this nat gateway." A small Windows taskbar icon is visible in the bottom right corner of the browser window.

6. Select the NAT Gateway, locate the subnet ID in the Details tab, and click it.

The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#NatGateways. The "NAT gateways" table lists one entry:

Name	NAT gateway ID	Connectivity...	State	Primary public I...	Primary private I...	Primary network...	VPC
my-NAT-gw-1	nat-02527773795a97d0f	Private	Available	-	10.0.7.35	eni-0ba2fe8358ad3...	vpc-059fcfaa09dbb14e1

The "Details" tab for the selected NAT gateway (nat-02527773795a97d0f) is shown, displaying the following information:

- NAT gateway ID:** nat-02527773795a97d0f
- Connectivity type:** Private
- Primary public IPv4 address:** -
- Subnet:** subnet-0bffd09a345bd2e1f / **my-private-subnet-1**
- State:** Available
- Primary private IPv4 address:** 10.0.7.35
- Created:** Wednesday, October 8, 2025 at 17:34:18 GMT+1

A red arrow points from the text "Click route table ID(...).ln_the_subnet" to the "my-private-subnet-1" link in the "Subnet" field. A small Windows taskbar icon is visible in the bottom right corner of the browser window.

7. Navigate to the **Route Table** section and click the route table ID.

! [Click route table ID(...).ln_the_subnet

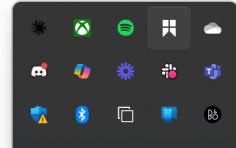
page_navigate_to_Route_Table_section_and_click_one_route_table_id(rtb-...).png)

8. Click **Routes**, then **Edit routes**.

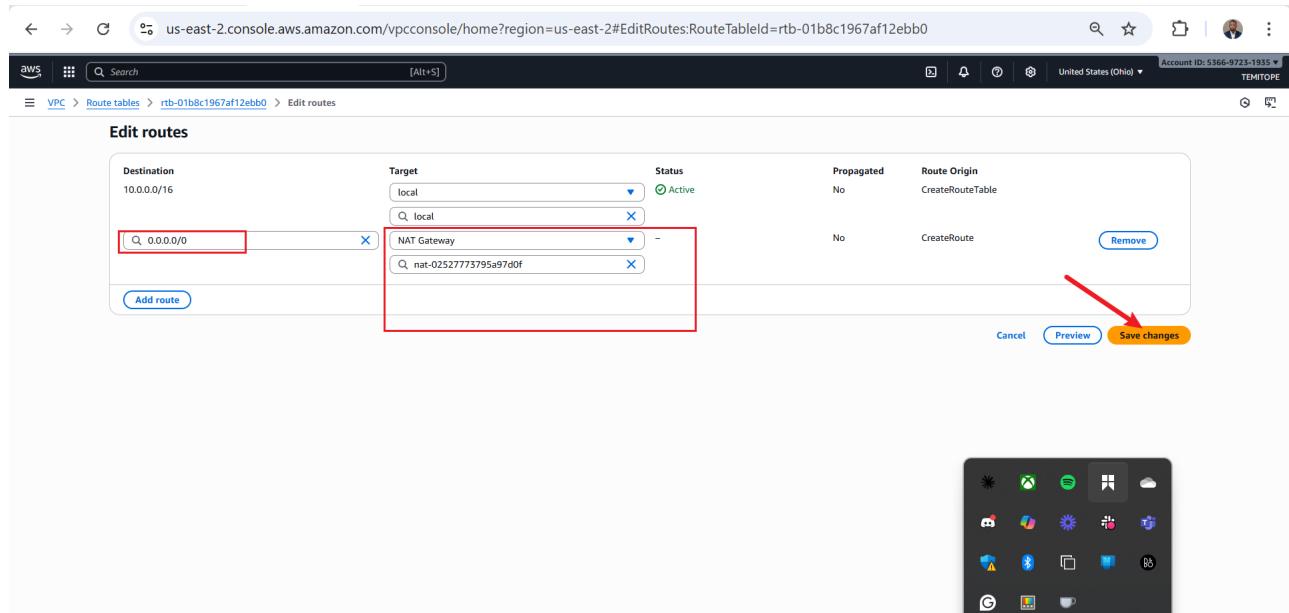
The screenshot shows the AWS VPC Route Tables page. On the left sidebar, under 'Virtual private cloud' > 'Route tables', there is a list of options: AWS Global View, Filter by VPC, Your VPCs, Subnets, Route tables (which is selected), Internet gateways, Egress-only internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, and Route servers. The main content area displays 'Route tables (1/1) Info'. A table shows one route table named 'rtb-01b8c1967af12eb0'. The 'Routes' tab is selected. At the bottom right of this section, there is a blue button labeled 'Edit routes' with a red arrow pointing to it.

9. Click **Add route**.

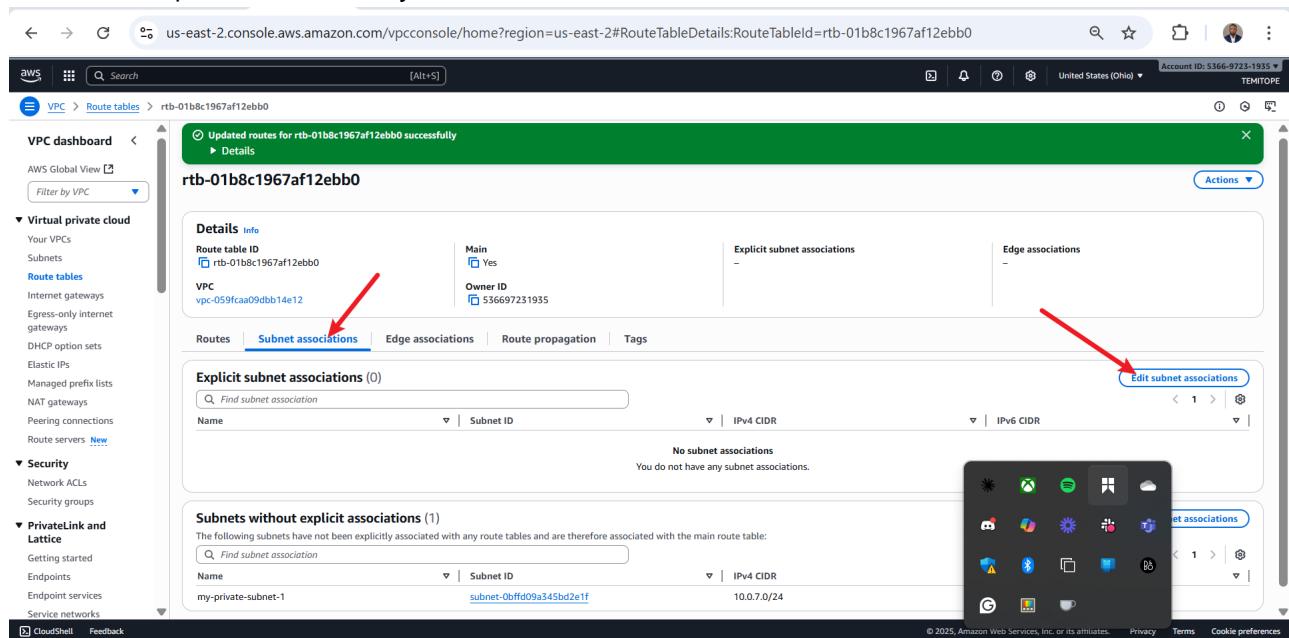
The screenshot shows the 'Edit routes' page. It has a table with columns: Destination, Target, Status, Propagated, and Route Origin. One row is shown with Destination '10.0.0.0/16', Target 'local', Status 'Active', Propagated 'No', and Route Origin 'CreateRouteTable'. At the bottom left of the table, there is a blue button labeled 'Add route' with a red arrow pointing to it. At the bottom right, there are three buttons: 'Cancel', 'Preview', and 'Save changes'.



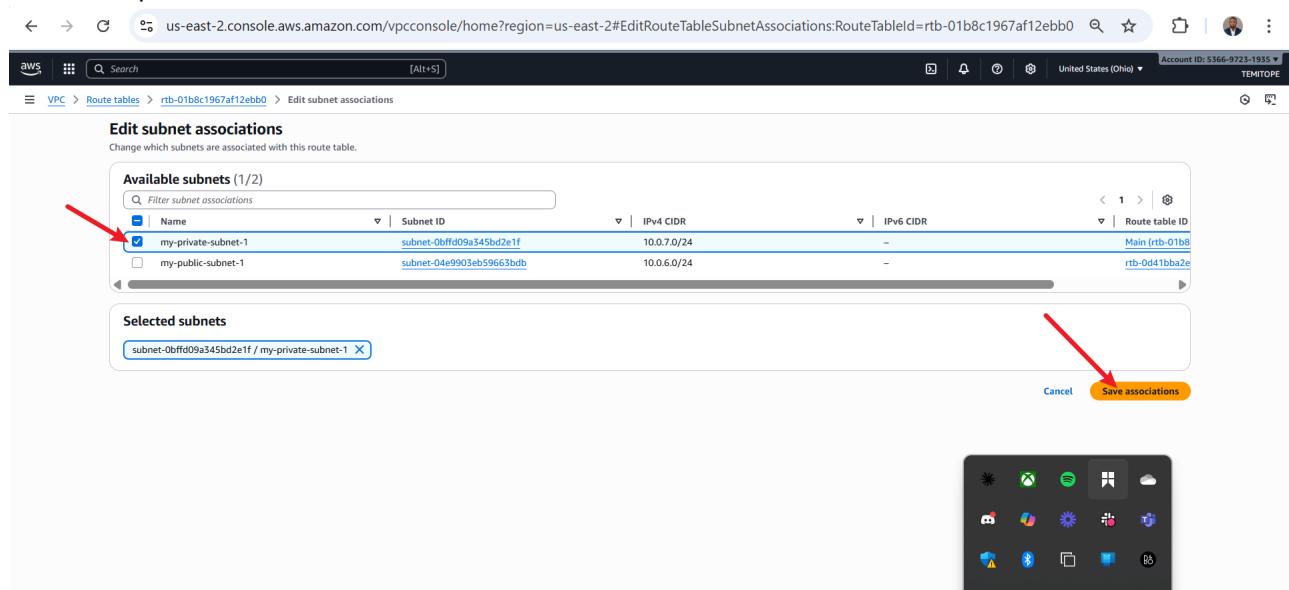
10. Set **Destination** to **0.0.0.0/0**, **Target** to **NAT Gateway**, select the created NAT Gateway, and save changes.



11. Route table updated successfully, click **Subnet associations**, then **Edit subnet associations**.



12. Select the private subnet and click **Save associations**.

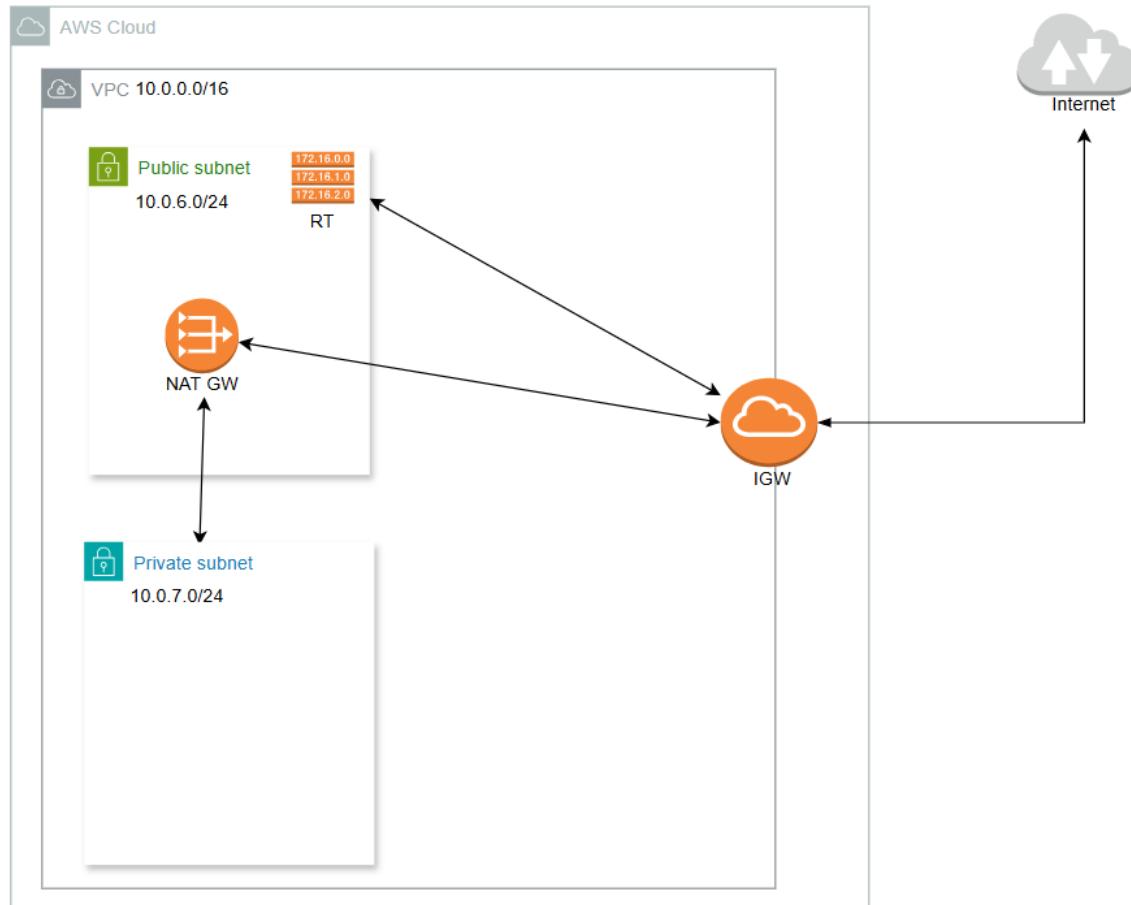


13. Subnet successfully attached to the route table.

The screenshot shows the AWS VPC Route Tables console. A green success message at the top states: "You have successfully updated subnet associations for rtb-01b8c1967af12ebb0." Below this, the route table details for "rtb-01b8c1967af12ebb0" are shown. The "Main" checkbox is selected. The "Owner ID" is listed as "vpc-059fcaa09dbb14e12". Under the "Explicit subnet associations" section, it shows "subnet-0bffd09a345bd2e1f / my-private-subnet-1". The "Edge associations" section is empty. At the bottom, there are tabs for "Routes", "Subnet associations", "Edge associations", "Route propagation", and "Tags". The "Routes" tab is selected, displaying two routes:

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	nat-0252777379a97d0f	Active	No	Create Route
10.0.0.16	local	Active	No	Create Route Table

14. View the current VPC architecture.



15. Create an Ubuntu EC2 instance named Public-EC2.

The screenshot shows the 'Launch an instance' wizard on the AWS EC2 console. The 'Name and tags' section has 'Public-EC2' entered. The 'Application and OS Images (Amazon Machine Image)' section shows the 'Ubuntu Server 24.04 LTS (HVM) SSD Volume Type' selected. The 'Summary' panel indicates 1 instance will be launched. A callout box highlights the 'Free tier' information: 'In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free-tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of snapshots, and 10 GiB of data transfer charge-free tier allowance.'

16. In network settings, click Edit.

The screenshot shows the 'Network settings' section of the 'Launch an instance' wizard. A red arrow points to the 'Edit' button next to the security group dropdown. The dropdown contains 'temskkey1'. The 'Edit' button is highlighted with a blue circle. A callout box highlights the 'Free tier' information: 'In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free-tier AMIs, 750 hours per month of public IPv4 address usage, 30 GiB of snapshots, and 10 GiB of data transfer charge-free tier allowance.'

17. Set VPC to 10.0.0.0/16, subnet to the public subnet, enable **Auto-assign public IP**, and launch instance.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Network settings' section, a VPC (vpc-059fa09dbb14e12) and a subnet (subnet-04e9903eb59663bd) are selected. The 'Auto-assign public IP' checkbox is checked. In the 'Summary' section, it shows 1 instance being launched. The 'Software Image (AMI)' is set to Canonical, Ubuntu, 24.04, amd64. The 'Virtual server type (instance type)' is t2.micro. The 'Launch instance' button is highlighted with a red box and an arrow.

18. Create an Ubuntu EC2 instance named **Private-EC2**.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' section, the name 'Private-EC2' is typed into the 'Name' field. In the 'Application and OS Images (Amazon Machine Image)' section, the 'Quick Start' tab is selected, showing recent AMIs like Amazon Linux, macOS, and Ubuntu. The 'Ubuntu' AMI is selected. In the 'Summary' section, it shows 1 instance being launched. The 'Software Image (AMI)' is set to Canonical, Ubuntu, 24.04, amd64. The 'Virtual server type (instance type)' is t2.micro. The 'Launch instance' button is highlighted with a red box and an arrow.

19. In network settings, click **Edit**.

The screenshot shows the 'Network settings' section of the EC2 instance creation wizard. It includes fields for VPC, subnet, and security group selection. A red arrow highlights the 'Edit' button located next to the 'Create security group' button.

20. Set VPC to 10.0.0.0/16, subnet to the private subnet, and launch instance.

The screenshot shows the 'Network settings' section with the VPC dropdown set to '10.0.0.0/16' and the subnet dropdown set to 'my-private-subnet-1'. A red box highlights these two fields. A red arrow points to the 'Launch instance' button at the bottom right of the summary panel.

21. Both EC2 instances created, click the Public-EC2 ID.

The screenshot shows the AWS EC2 Instances page. On the left sidebar, under the 'Instances' section, there is a list of various EC2-related options like Instance Types, Launch Templates, and Security Groups. The main area displays a table titled 'Instances (2) Info' with columns for Name, Instance ID, Instance state, Instance type, Status check, Alarm status, Availability Zone, Public IPv4 DNS, Public IPv4, and Elastic IP. Two rows are listed: 'Public-EC2' (Instance ID: i-0616ddaa4676914e62) and 'Private-EC2' (Instance ID: i-0d96caf021a83bd8). Both instances are shown as 'Running'. A red arrow points to the Instance ID of 'Public-EC2'.

22. Copy the public IP for remote connection.

The screenshot shows the AWS EC2 Instance summary page for instance 'i-0e182c5c5bd39ce3a' (Public-EC2). The left sidebar is identical to the previous screenshot. The main content area is titled 'Instance summary for i-0e182c5c5bd39ce3a (Public-EC2)' and includes sections for Instance ID, IPv6 address, Hostname type, Answer private resource DNS name, Auto-assigned IP address, IAM Role, IMDSv2, Operator, and more. A red arrow points to the 'Public IPv4 address' field, which is currently set to 3.145.105.141. The right side of the screen shows additional details like Private IP, VPC ID, Subnet ID, Instance ARN, and various AWS services icons.

23. Connect via SSH.

```
MINGW64/c/Users/HP/Downloads

ED25519 key fingerprint is SHA256:xeXdkC7+GbfubqTs09ztyx+ZyiHUi+z9PgbCnR1CPk.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.145.105.141' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Oct 9 16:29:41 UTC 2025

system load: 0.0 Processes: 108
Usage of /: 25.6% of 6.71GB Users logged in: 0
Memory usage: 20% IPv4 address for enx0: 10.0.6.194
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-6-194:~$
```



24. Test outbound traffic with `git clone`.

```
MINGW64/c/Users/HP/Downloads

ubuntu@ip-10-0-6-194:~$ git --version
git version 2.43.0
ubuntu@ip-10-0-6-194:~$ git clone https://github.com/Oluwaseunoa/hng-stage-1-task.git
Cloning into 'hng-stage-1-task'...
remote: Enumerating objects: 27, done.
remote: Counting objects: 100% (27/27), done.
remote: Compressing objects: 100% (18/18), done.
remote: Total 27 (delta 12), reused 20 (delta 8), pack-reused 0 (from 0)
Receiving objects: 100% (27/27), 17.05 KiB | 3.41 MiB/s, done.
Resolving deltas: 100% (12/12), done.
ubuntu@ip-10-0-6-194:~$ ls hng-stage-1-task/
README.md index.js package-lock.json package.json
ubuntu@ip-10-0-6-194:~$
```



25. Copy the public IP of **Private-EC2** for remote connection (note: connection should fail).

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with navigation links like Dashboard, EC2 Global View, Events, Instances, Images, Elastic Block Store, Network & Security, and more. The main area displays an instance summary for 'i-0a63f5b8ff529ad69 (Private-EC2)'. Key details shown include:

- Public IPv4 address:** 3.148.188.140 (highlighted with a red arrow)
- Instance state:** Running
- Private IP DNS name (IPv4 only):** ip-10-0-7-223.us-east-2.compute.internal
- Instance type:** t2.micro
- VPC ID:** vpc-059fcaa09dbb14e12
- Subnet ID:** subnet-0bffd09a345bd2e1f (my-private-subnet-1)
- Instance ARN:** arn:aws:ec2:us-east-2:536697231935:instance/i-0a63f5b8ff529ad69

At the bottom, there are tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The Details tab is selected.

26. Unable to connect to **Private-EC2** over the internet.

The screenshot shows a terminal window with the following text:

```
HPDESKTOP-I9M74R1 MINGW64 ~/Downloads
$ ssh -i "tempkey1.pem" ubuntu@3.148.188.140
ssh: connect to host 3.148.188.140 port 22: Connection timed out

HPDESKTOP-I9M74R1 MINGW64 ~/Downloads
$ ]
```

The terminal is running on a Windows system, as indicated by the taskbar icons at the bottom.

27. Verify NAT Gateway functionality by navigating to **NAT Gateways** and selecting the created NAT Gateway.

NAT gateways (1) info

Name	NAT gateway ID	Connectivity...	State	Primary public I...	Primary private I...	Primary network...
my-NAT-gw-1	nat-02527773795a97d0f	Private	Available	-	10.0.7.35	eni-0ba2fe8358ad3b490...

Select a NAT gateway

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

28. Observe outbound traffic data in **Packet Out to Destination** and **Byte Out to Destination**.

nat-02527773795a97d0f / my-NAT-gw-1

Details

NAT gateway ID nat-02527773795a97d0f	Connectivity type Private	State Available	State message -
NAT gateway ARN arnaws:ec2:us-east-2:536697231935:natgateway/nat-02527773795a97d0f	Primary public IPv4 address -	Primary private IPv4 address 10.0.7.35	Primary network interface ID eni-0ba2fe8358ad3b490
VPC vpc-059fcaa09dbb14e12	Subnet subnet-0bffd09a345bd2e1f / my-private-subnet-1	Created Wednesday, October 8, 2025 at 17:34:18 GMT+1	Deleted -

Monitoring

Packets out to destination (Count)

Packets out to source (Count)

Bytes out to destination (Bytes)

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

29. Create another EC2 instance to test connectivity within the same subnet.

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' step, the name is set to 'Public-EC2'. In the 'Application and OS Images (Amazon Machine Image)' step, the 'ubuntu' AMI is selected. A tooltip on the right provides information about the free tier, stating: 'Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. 750 hours per month of public IPv4 address usage, 30 GiB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet. Data transfer charges are not included as part of the free tier allowance.'

30. Set VPC to 10.0.0.0/16, subnet to the public subnet, and launch instance.

The screenshot shows the AWS EC2 'Launch an instance' wizard in the 'Network settings' step. The VPC dropdown is set to 'vpc-059caa09ddb14e12 10.0.0.0/16'. The subnet dropdown is set to 'subnet-04e9030eb59663db my-public-subnet-1'. A red arrow points to the 'Create security group' button. Another red arrow points to the 'Launch Instance' button at the bottom right. The tooltip from the previous step is still visible on the right.

31. Two EC2 instances now in the public subnet.

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

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Private-EC2	i-0a63f5b8ff529ad69	Stopped	t2.micro	-	View alarms	us-east-2a	-	-	-
Public-EC2-2	i-076053c6f003a065e	Pending	t2.micro	-	View alarms	us-east-2a	-	3.148.164.208	-
Public-EC2	i-0e182c5c5bd39ce3a	Running	t2.micro	2/2 checks passed	View alarms	us-east-2a	-	3.136.233.109	-

Below the table, a detailed view of the selected instance (i-0e182c5c5bd39ce3a) is shown. The instance summary tab is active. It displays the instance ID (i-0e182c5c5bd39ce3a), a Public IPv4 address (3.136.233.109), and a Private IPv4 address (10.0.6.1).

32. Click the new Public-EC2.

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

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP
Private-EC2	i-0a63f5b8ff529ad69	Stopped	t2.micro	-	View alarms	us-east-2a	-	-	-
Public-EC2-2	i-076053c6f003a065e	Running	t2.micro	Initializing	View alarms	us-east-2a	-	3.148.164.208	-
Public-EC2	i-0e182c5c5bd39ce3a	Running	t2.micro	2/2 checks passed	View alarms	us-east-2a	-	3.136.233.109	-

Below the table, a modal window titled "Select an instance" is open, listing the three instances: Private-EC2, Public-EC2-2, and Public-EC2.

33. Click Connect.

aws EC2 > Instances > i-076053c6f003a065e

Instance summary for i-076053c6f003a065e (Public-EC2-2) Info

Updated less than a minute ago

Instance ID	i-076053c6f003a065e	Public IPv4 address	3.148.164.208 open address	Private IP4 addresses	10.0.6.224
IPv6 address	-	Instance state	Running	Public DNS	-
Hostname type	IP name: ip-10-0-6-224.us-east-2.compute.internal	Private IP DNS name (IPv4 only)	ip-10-0-6-224.us-east-2.compute.internal	Elastic IP addresses	-
Answer private resource DNS name	-	Instance type	t2.micro	AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Auto-assigned IP address	3.148.164.208 [Public IP]	VPC ID	vpc-059caa09dbb14e12	Auto Scaling Group name	-
IAM Role	-	Subnet ID	subnet-04e9903eb59663bdb (my-public-subnet-1)	Managed	false
IMDSv2	Required	Instance ARN	arn:aws:ec2:us-east-2:536697231935:instance/i-076053c6f003a065e		
Operator	-				

Details Status and alarms Monitoring Security Networking Storage Tags

Instance details Info

AMI ID: ami-0cfde0ea8edd512d4

Monitoring: disabled

Platform details: Linux/UNIX

© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

34. Select EC2 Instance Connect tab and click Connect.

aws EC2 > Instances > i-076053c6f003a065e > Connect to instance

Connect Info

Connect to an instance using the browser-based client.

EC2 Instance Connect Session Manager SSH client EC2 serial console

Instance ID: i-076053c6f003a065e (Public-EC2-2)

Connection type:

- Connect using a Public IP Connect using a public IPv4 or IPv6 address
- Connect using a Private IP Connect using a private IP address and a VPC endpoint

Public IPv4 address: 3.148.164.208

IPv6 address: -

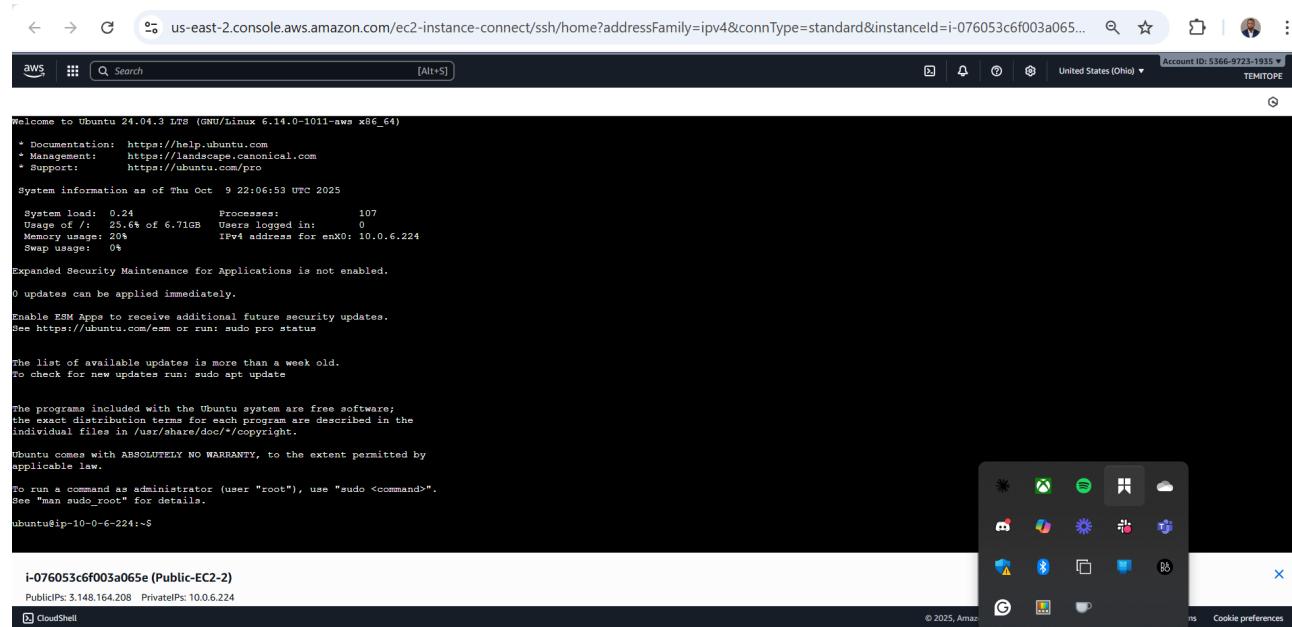
Username: Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ubuntu.

Note: In most cases, the default username, ubuntu, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

Connect Cancel

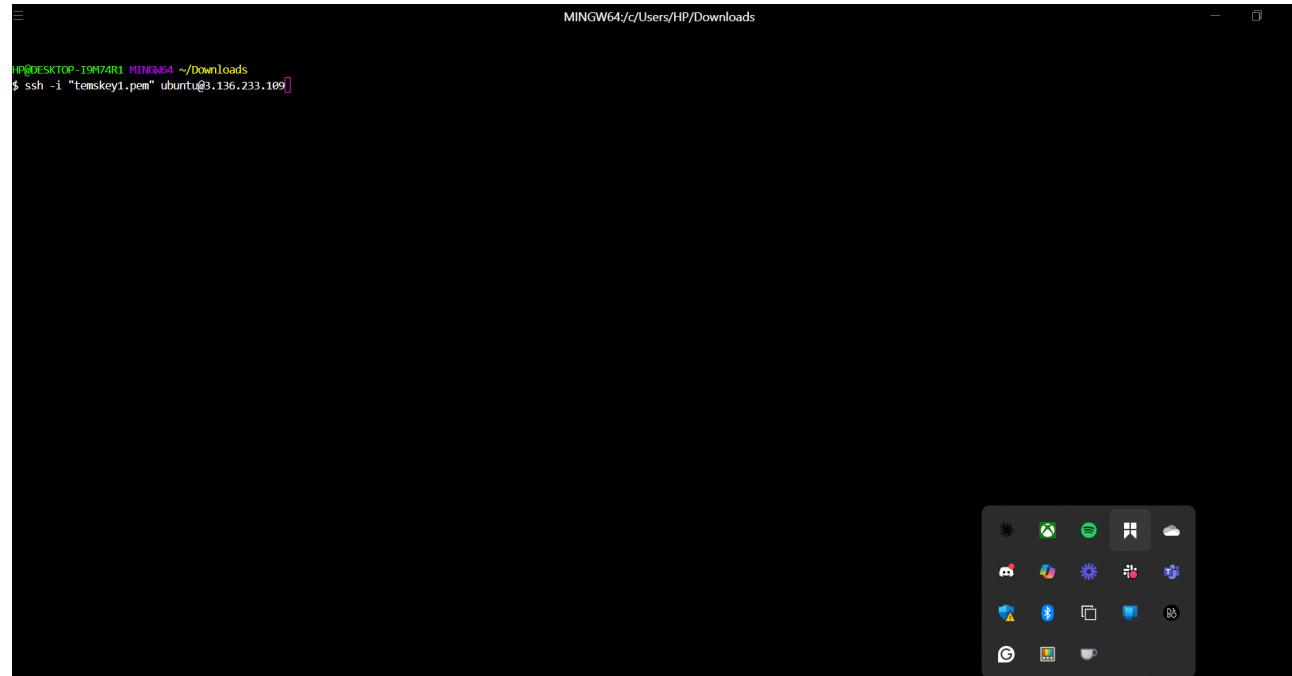
© 2025, Amazon Web Services, Inc. or its affiliates. [Privacy](#) [Terms](#) [Cookie preferences](#)

35. Successfully connected via browser.



The screenshot shows a terminal window on an AWS EC2 instance. The URL in the address bar is `us-east-2.console.aws.amazon.com/ec2-instance-connect/ssh/home?addressFamily=ipv4&connType=standard&instanceId=i-076053c6f003a065...`. The terminal output displays system information for Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64). It includes details like system load (0.24), memory usage (25.6% of 6.71GB), and swap usage (0%). The terminal also shows a message about security maintenance and a note about ESM Apps. At the bottom, it shows the command `ubuntu@ip-10-0-6-224:~$`.

36. Connect to the other EC2 in the public subnet using a terminal.



The screenshot shows a terminal window titled "MINGW64:/c/Users/HP/Downloads". The command entered is `ssh -i "temskey1.pem" ubuntu@3.136.233.109`. The terminal is currently at the prompt `HPDESKTOP-I9N74R1 MINGW4 ~/Downloads`.

37. Successfully connect via terminal.

```
HPDESKTOP-19M74R1 MINGW64 ~/Downloads
$ ssh -i "tempkey1.pem" ubuntu@3.136.233.109
The authenticity of host '3.136.233.109' ('3.136.233.109') can't be established.
ED25519 key fingerprint is SHA256:xeXkC74dbfubqTSQ9zyx+ZyiNUi+z9PGbCNr1cpK.
This host key is known by the following other names/addresses:
  ./ssh/known_hosts:30: 3.145.105.141
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '3.136.233.109' (ED25519) to the list of known hosts.
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Oct 9 22:09:48 UTC 2025

System load: 0.0      Processes:          108
Usage of /: 27.9% of 6.71GB  Users logged in:    0
Memory usage: 21%           IPv4 address for enx0: 10.0.6.194
Swap usage:  0%

Expanded Security Maintenance for Applications is not enabled.

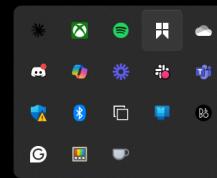
0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

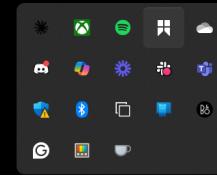
Last login: Thu Oct 9 16:29:42 2025 from 98.97.79.50
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-6-194:~$
```

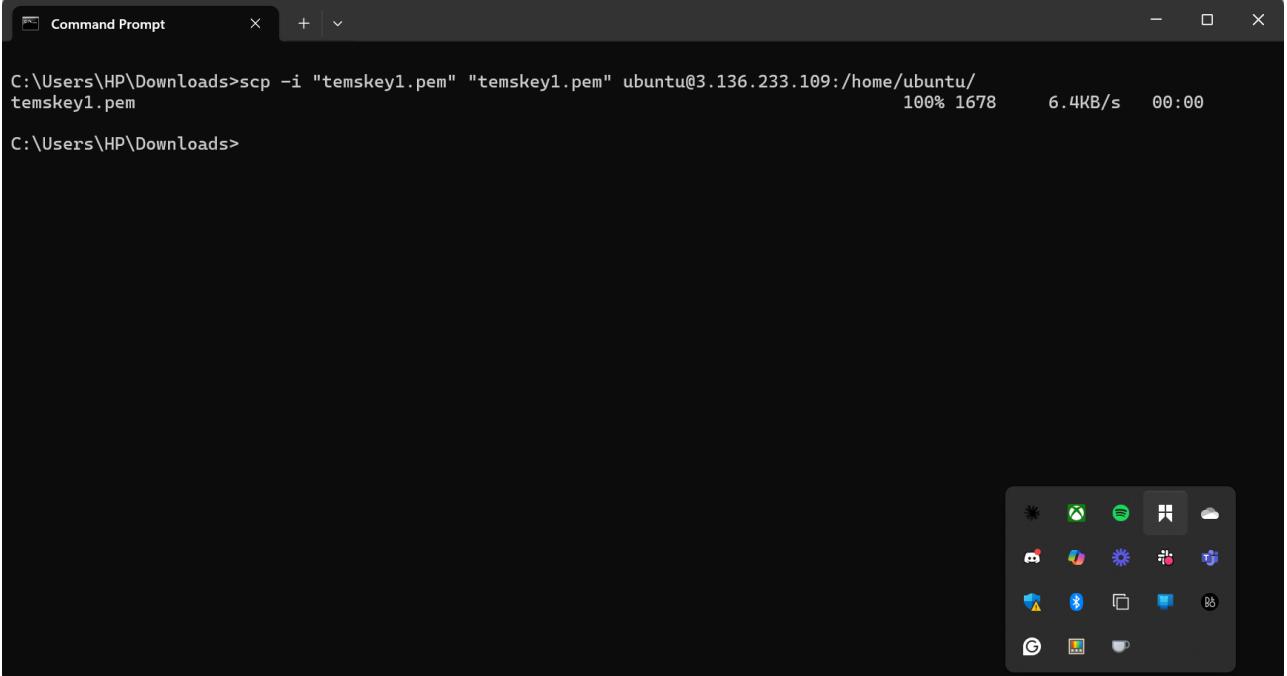


38. Ping the second EC2 to ensure reachability.

```
ubuntu@ip-10-0-6-194:~$ ping 3.148.164.208
```

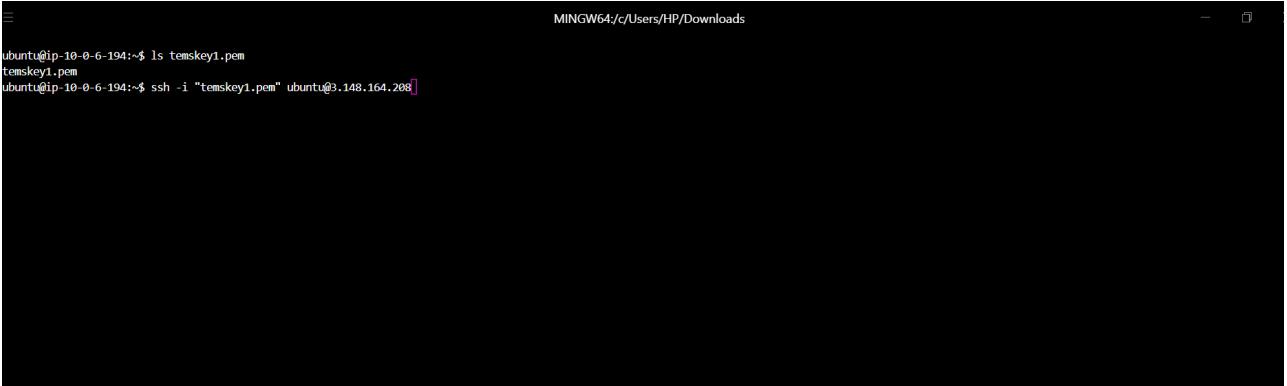


39. Move the key pair to EC2 using SCP to SSH into the second EC2.



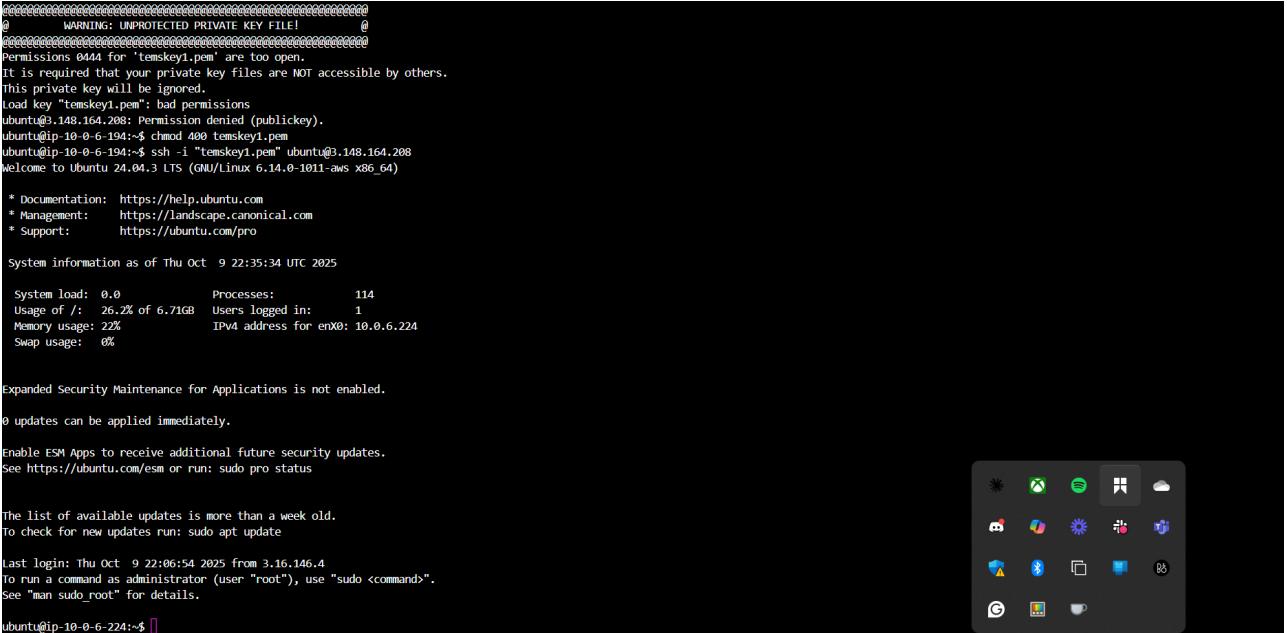
```
C:\Users\HP\Downloads>scp -i "temskey1.pem" "temskey1.pem" ubuntu@3.136.233.109:/home/ubuntu/
temskey1.pem                                         100% 1678      6.4KB/s   00:00
C:\Users\HP\Downloads>
```

40. Confirm key pair copied and SSH into **Public-EC2-2** from **Public-EC2**.



```
ubuntu@ip-10-0-6-194:~$ ls temskey1.pem
temskey1.pem
ubuntu@ip-10-0-6-194:~$ ssh -i "temskey1.pem" ubuntu@3.148.164.208
```

41. Successfully SSH into **Public-EC2-2**.



```
ubuntu@ip-10-0-6-194:~$ ls temskey1.pem
temskey1.pem
ubuntu@ip-10-0-6-194:~$ ssh -i "temskey1.pem" ubuntu@3.148.164.208
Warning: UNPROTECTED PRIVATE KEY FILE!
Permissions 0444 for 'temskey1.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 "temskey1.pem": bad permissions
ubuntu@3.148.164.208: Permission denied (publickey).
ubuntu@ip-10-0-6-194:~$ chmod 400 temskey1.pem
ubuntu@ip-10-0-6-194:~$ ssh -i "temskey1.pem" ubuntu@3.148.164.208
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of Thu Oct  9 22:35:34 UTC 2025

System load: 0.0          Processes:           114
Usage of /: 26.2% of 6.71GB  Users logged in:  1
Memory usage: 22%          IPv4 address for enx0: 10.0.6.224
Swap usage: 0%
```

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See <https://ubuntu.com/esm> or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Oct 9 22:06:54 2025 from 3.16.146.4
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

```
ubuntu@ip-10-0-6-224:~$
```

42. Log out of both instances.

```
ubuntu@ip-10-0-6-194:~$ ssh -i "tmskey1.pem" ubuntu@3.148.164.208
Welcome to Ubuntu 24.04.3 LTS (GNU/Linux 6.14.0-1011-aws x86_64)

 * Documentation: https://help.ubuntu.com
 * Management: https://landscape.canonical.com
 * Support: https://ubuntu.com/pro

System information as of Thu Oct 9 22:35:34 UTC 2025

System load: 0.0          Processes:           114
Usage of /: 26.2% of 6.71GB Users logged in:      1
Memory usage: 22%          IPv4 address for enx0: 10.0.6.224
Swap usage: 0%

Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

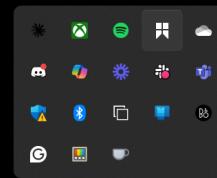
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Thu Oct 9 22:06:54 2025 from 3.16.146.4
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

ubuntu@ip-10-0-6-224:~$ ^C
ubuntu@ip-10-0-6-224:~$ exit
logout
Connection to 3.148.164.208 closed.
ubuntu@ip-10-0-6-194:~$ exit
logout
Connection to 3.136.233.109 closed.

HPDESKTOP-I9H74R1 MINGW64 ~/Downloads
$
```



Part 6: Establishing VPC Peering Connections

1. Create the requester VPC (192.168.0.0/16).

Create VPC [Info](#)

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings [Info](#)

Create only this VPC resource or the VPC and other networking resources:

VPC only VPC and more

Name tag - optional [Info](#)

Creates a tag with a key of 'Name' and a value that you specify.

requester-VPC

IPv4 CIDR block [Info](#)

IPv4 CIDR manual input IPAM-allocated IPv4 CIDR block

IPv4 CIDR: 192.168.0.0/16

CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)

No IPv6 CIDR block IPAM-allocated IPv6 CIDR block Amazon-provided IPv6 CIDR block IPv6 CIDR owned by me

Tenancy [Info](#)

Default

Tags [Info](#)

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
<input type="text" value="Name"/>	<input type="text" value="requester-VPC"/>
Remove tag	
Add tag	
You can add up to 49 more tags	

[Cancel](#) [Preview code](#) **Create VPC**



2. Create the accepter VPC (172.16.0.0/16).

Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

VPC settings

Resources to create: [Info](#)
Create only the VPC resource or the VPC and other networking resources.
 VPC only VPC and more

Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.
accepter-VPC

IPv4 CIDR block: [Info](#)
 IPv4 CIDR manual input IPAM-allocated IPv4 CIDR block
172.16.0.0/16

IPv6 CIDR block: [Info](#)
 No IPv6 CIDR block IPAM-allocated IPv6 CIDR block
 Amazon-provided IPv6 CIDR block
 IPv6 CIDR owned by me

Tenancy: [Info](#)
Default

Tags
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
<input type="text" value="Name"/>	<input type="text" value="accepter-VPC"/>

[Add tag](#) You can add 49 more tags

[Cancel](#) [Preview code](#) **Create VPC**

3. Both VPCs created, navigate to Peering Connections.

Your VPCs (4) Info

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table	Main network ACL	Tenancy	Default VPC
vpc-059faaa09b0a14e12	Available	<input type="radio"/> Off	10.0.0.0/16	-	-	dopt-09149d5972cad...	rb-01bb1967af12eb0	ac-04d802252b0f3ad8	default	No
vpc-0945dc12fb5f4cbe	Available	<input type="radio"/> Off	172.31.0.0/16	-	-	dopt-09149d5972cad...	rb-025ca8a4ee81e7226e	ac-0x38fae99ea75b8	default	Yes
requester-VPC	Available	<input type="radio"/> Off	192.168.0.0/16	-	-	dopt-09149d5972cad...	-	-	default	No
accepter-VPC	Available	<input type="radio"/> Off	172.16.0.0/16	-	-	dopt-09149d5972cad...	-	-	default	No

Actions [Create VPC](#)

Peering connections

Select a VPC above

4. Click Create Peering Connection.

The screenshot shows the AWS VPC console's Peering Connections page. On the left, there's a navigation sidebar with sections like VPC dashboard, Virtual private cloud, Security, and PrivateLink and Latency. The main area is titled "Peering connections" and has a search bar. At the top right, there's a "Actions" dropdown with a "Create peering connection" button highlighted by a red arrow. Below the search bar, there's a table header with columns for Name, Peering connection ID, Status, Requester VPC, Acceptor VPC, Requester CIDRs, Acceptor CIDRs, and Requester C. The table body below says "No peering connection found".

5. Name the peering connection, select requester VPC, choose **My account**, select accepter VPC, and click **Create Peering Connection**.

The screenshot shows the "Create peering connection" wizard. Step 1 is titled "Peer connection settings". It asks for a "Name - optional" and shows "my-first-vpc-peering" in a text input field. Below it, there's a section "Select a local VPC to peer with" with a dropdown for "VPC ID (Requester)" set to "vpc-0777a0c1649ff1ab4 (requester-VPC)". Under "VPC CIDRs for vpc-0777a0c1649ff1ab4 (requester-VPC)", there's a table with one entry: CIDR 192.168.0.0/16 and Status Associated. The next section, "Select another VPC to peer with", includes fields for "Account" (radio buttons for "My account" and "Another account"), "Region" (radio buttons for "This Region (us-east-2)" and "Another Region"), and "VPC ID (Acceptor)" (dropdown set to "vpc-0c059193282cd03e6 (accepter-VPC)"). A table for "VPC CIDRs for vpc-0c059193282cd03e6 (accepter-VPC)" shows one entry: CIDR 172.16.0.0/16 and Status Associated. The bottom of the screen shows standard AWS footer links and a "Create peering connection" button.

6. Scroll down and create the peering connection.

The screenshot shows the continuation of the "Create peering connection" wizard. It includes a "Tags" section where a tag named "my-first-vpc-peering" is being added. The "Create peering connection" button is highlighted with a red arrow at the bottom right. The bottom of the screen shows the standard AWS footer.

7. VPC peering connection created, click Actions and Accept Request.

AWS Global View

pcx-0f6e1f02cfce7b81e / my-first-vpc-peering

Pending acceptance
You can accept or reject this peering connection request using the 'Actions' menu. You have until Friday, October 17, 2025 at 00:07:05 GMT+1 to accept or reject the request, otherwise it expires.

Details <small>Info</small>		Acceptor owner ID <input type="button" value="536697231935"/>	VPC Peering connection ARN <input type="button" value="arn:aws:ec2:us-east-2:536697231935:vpc-2fcf7b81e"/>
Requester owner ID	<input type="button" value="536697231935"/>	Requester VPC <input type="button" value="vpc-0777a0c1649ff1ab4 / requester-VPC"/>	Acceptor CIDRs <input type="button" value="192.168.0.0/16"/>
Peering connection ID	<input type="button" value="pcx-0f6e1f02cfce7b81e"/>	Status <input type="button" value="Pending Acceptance by 536697231935"/>	Requester Region <input type="button" value="Ohio (us-east-2)"/>
Expiration time	<input type="button" value="Friday, October 17, 2025 at 00:07:05 GMT+1"/>	Requester CIDRs <input type="button" value="—"/>	Acceptor Region <input type="button" value="Ohio (us-east-2)"/>

DNS

DNS settings

Requester VPC (Info)
Allow accepter VPC to resolve DNS of hosts in requester VPC to private IP addresses

Acceptor VPC (Info)
Allow requester VPC to resolve DNS of hosts in accepter VPC to private IP addresses

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

8. Click Accept Request in the pop-up.

Accept VPC peering connection request Info

Are you sure you want to accept this VPC peering connection request? (pcx-0f6e1f02cfce7b81e / my-first-vpc-peering)

Requester VPC <input type="button" value="vpc-0777a0c1649ff1ab4 / requester-VPC"/>	Acceptor VPC <input type="button" value="vpc-0c059193282cd03e6 / accepter-VPC"/>	VPC Peering connection ARN <input type="button" value="arn:aws:ec2:us-east-2:536697231935:vpc-peering-connection/pcx-0f6e1f02cfce7b81e"/>
Requester CIDRs <input type="button" value="—"/>	Requester CIDRs <input type="button" value="192.168.0.0/16"/>	Requester Region <input type="button" value="Ohio (us-east-2)"/>
Requester owner ID <input type="button" value="536697231935 (This account)"/>	Acceptor owner ID <input type="button" value="536697231935 (This account)"/>	Acceptor Region <input type="button" value="Ohio (us-east-2)"/>

Cancel **Accept request** Edit DNS settings

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

9. VPC peering connection established, click **VPC**.

The screenshot shows the AWS VPC Peering Connections console. A green banner at the top indicates that a VPC peering connection has been established. The main pane displays the details of the peering connection, including the requester and accepter VPCs, their CIDRs, and regions. The requester VPC is 'requester-VPC' (ID: vpc-0f6e1f02cfce7b81e) and the accepter VPC is 'accepter-VPC' (ID: vpc-0c059193282cd03e6). Both are in the 'Active' state. The requester's CIDR is 192.168.0.0/16 and the accepter's CIDR is 172.16.0.0/16. They are both located in the 'Ohio (us-east-2)' region. The peering connection ARN is arn:aws:ec2:us-east-2:536697231935:vpc-peering-connection/pcx-0f6e1f02cfce7b81e.

10. Click the accepter VPC's main route table ID.

The screenshot shows the AWS VPCs console. The 'Your VPCs' table lists three VPCs: 'accepter-VPC' (ID: vpc-0c059193282cd03e6), 'requester-VPC' (ID: vpc-0f6e1f02cfce7b81e), and another unnamed VPC. The 'accepter-VPC' row is highlighted. The 'Details' tab of the 'requester-VPC' page is shown, displaying its configuration. A red arrow points to the 'rtb-0a7c459e61b1aa146' route table ID.

11. Click the route table ID.

The screenshot shows the AWS VPC console with the URL <https://us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#RouteTables:routeTableId=rtb-0d6e66b374058101b>. The page displays a table of route tables. The first row, which has a red arrow pointing to its 'Route table ID' cell, contains the following data:

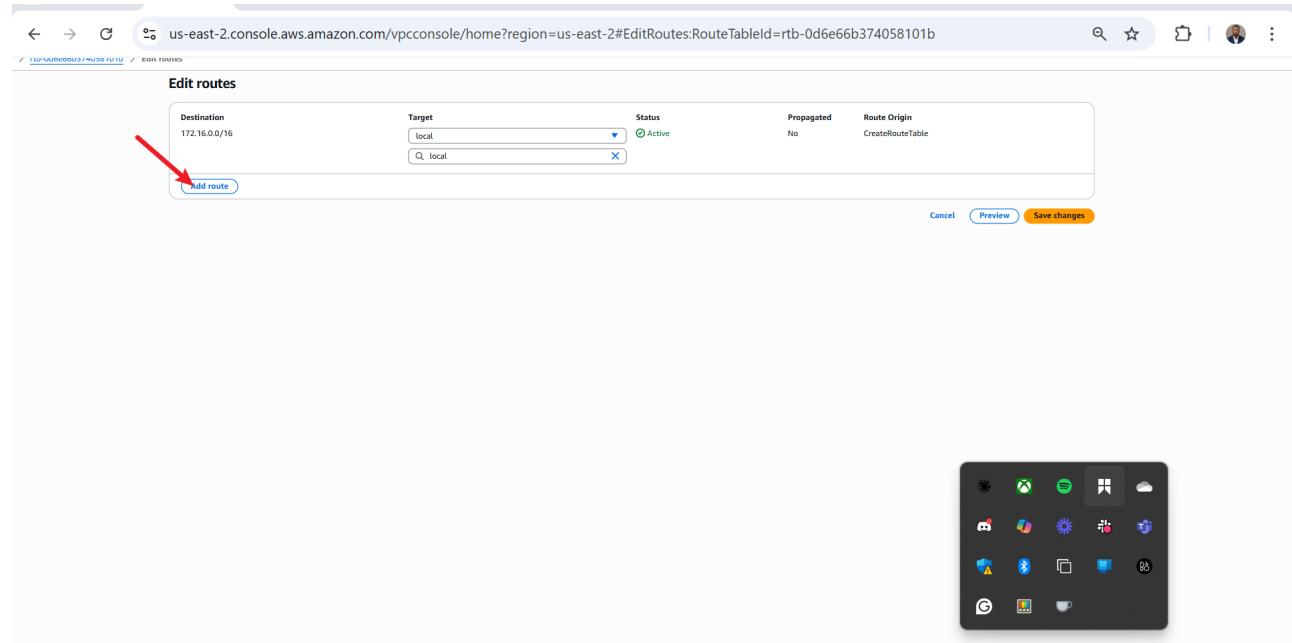
Name	Route table ID	Explicit subnet assoc...	Main	VPC	Owner ID
-	rtb-0d6e66b374058101b	-	Yes	vpc-0c059193282cd03e6 accepter-VPC	536697231935

12. Click **Edit routes**.

The screenshot shows the AWS VPC console with the URL <https://us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#RouteTableDetails:RouteTableId=rtb-0d6e66b374058101b>. The page displays the details of the selected route table, 'rtb-0d6e66b374058101b'. The 'Routes' tab is selected, showing one route entry:

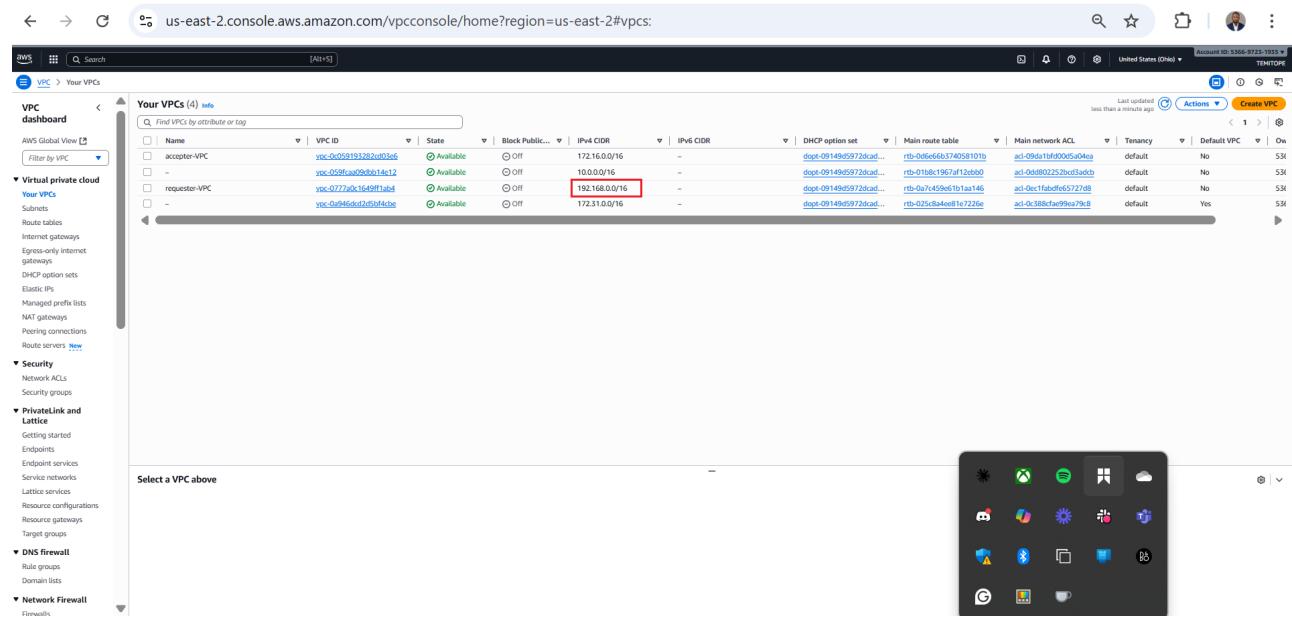
Destination	Target	Status	Propagated
172.16.0.0/16	local	Active	No

13. Click Add route.



The screenshot shows the 'Edit routes' page in the AWS VPC console. A red arrow points to the 'Add route' button at the bottom left of the main form area. The form includes fields for Destination (172.16.0.0/16), Target (local), Status (Active), Propagated (No), and Route Origin (CreateRouteTable). Below the form are 'Cancel', 'Preview', and 'Save changes' buttons.

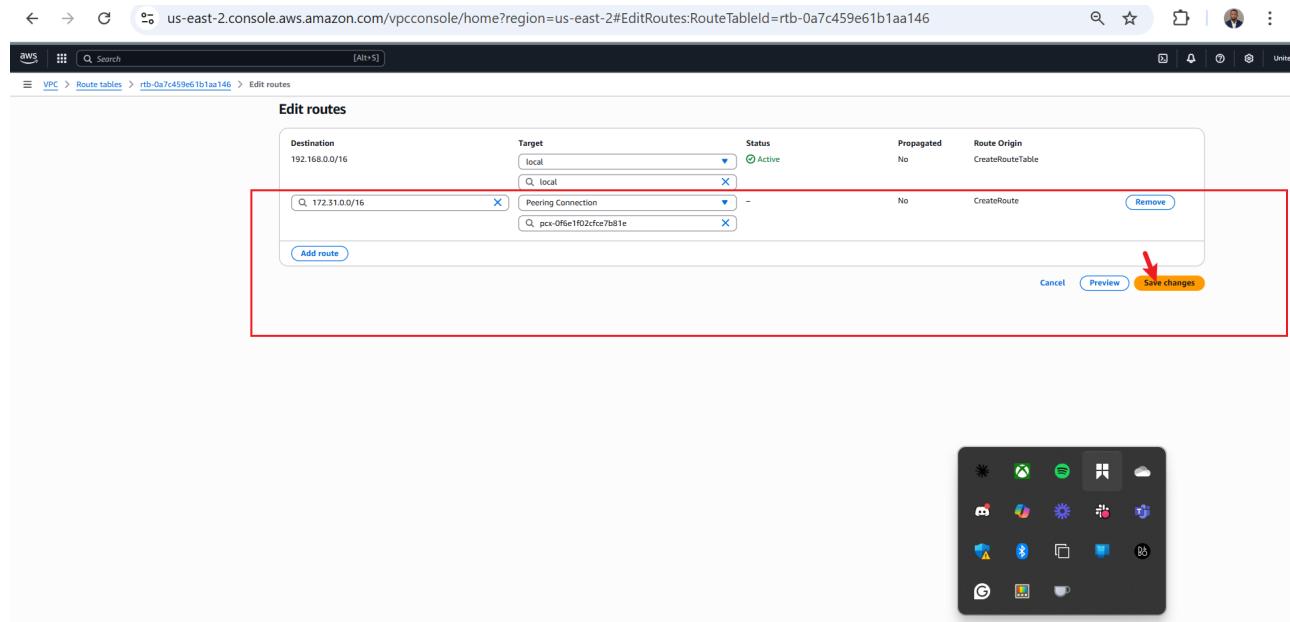
14. Copy the IPv4 CIDR of the requester VPC (192.168.0.0/16).



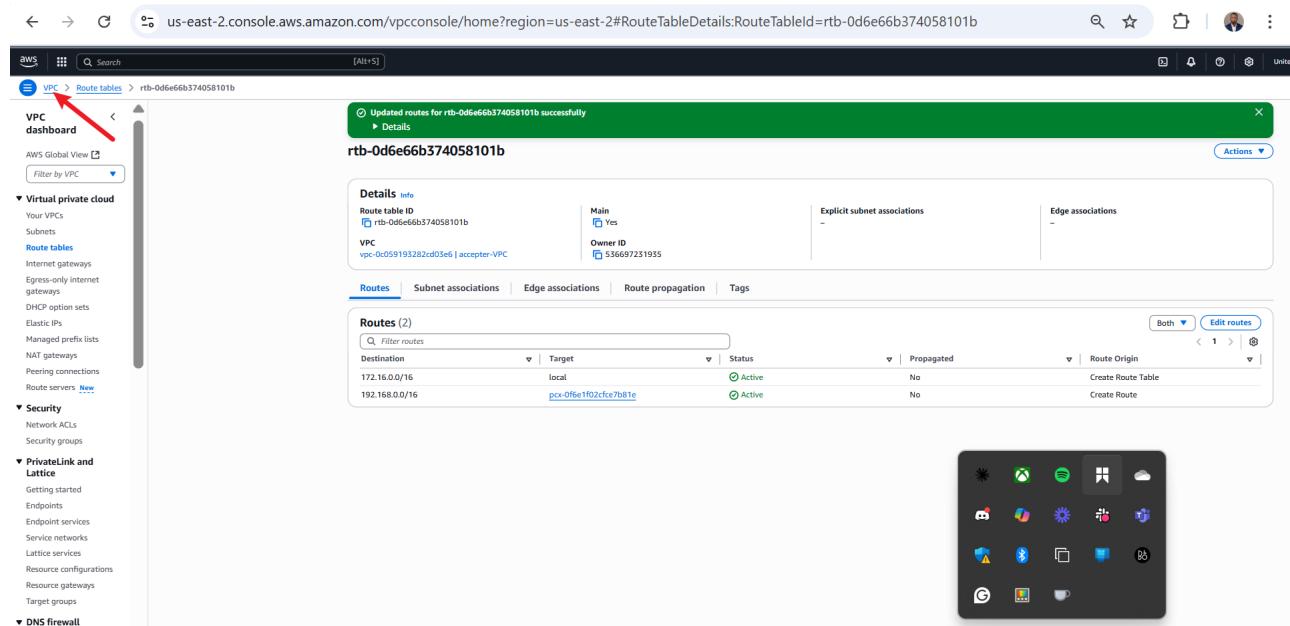
The screenshot shows the 'Your VPCs' section of the AWS VPC dashboard. A red box highlights the IPv4 CIDR '192.168.0.0/16' for the 'requester-VPC'. The table lists four VPCs: 'accepter-VPC', 'requester-VPC', and two unnamed entries. The requester VPC is the second row from the top.

Name	VPC ID	State	IPv4 CIDR	IPv6 CIDR	DHCP option set	Main route table	Main network ACL	Tenancy	Default VPC	On
accepter-VPC	vpc-0c059193282e0f05e6	Available	172.16.0.0/16	-	dopt-09149e5972cad...	rtb-0d6ee66374058101b	acl-09da1ff60005a0ea	default	No	53t
-	vpc-0596aa09eb014e12	Available	10.0.0.0/16	-	dopt-09149e5972cad...	rtb-01bae1967af12e6b0	acl-0eddb022c2b053a6b	default	No	53t
requester-VPC	vpc-0777a0b1549f1fa4	Available	192.168.0.0/16	-	dopt-09149e5972cad...	rtb-0a7c459d510a1aa146	acl-0ec1abde65f727d8	default	No	53t
-	vpc-0a9465cd27bf0f4ce	Available	172.31.0.0/16	-	dopt-09149e5972cad...	rtb-025ca4e081e7226e	acl-0c3bb1ae99b0a79c8	default	Yes	53t

15. Add the requester VPC's CIDR to the **Destination**, set **Target** to the peering connection, and save changes.



16. Route table updated successfully, return to VPC.



17. Copy the IPv4 CIDR of the accepter VPC (172.16.0.0/16) and click the requester VPC's main route table ID.

Your VPCs (1/4) Info

Name	VPC ID	State	Block Public...	IPv4 CIDR	DHCP option set	Main route table	Main network ACL	Ter
accepter-VPC	vpc-0c059193282cd03e6	Available	Off	172.16.0.0/16	dopt-09149d5972dcad...	rtb-0d6e66b574058101b...	ad-09611bfdf0d5a04ea...	def
-	vpc-059ca0c09db114e12	Available	Off	10.0.0.0/16	dopt-09149d5972dcad...	rtb-018c1967af12eb6b...	ad-0dd802252b0d5a04cb...	def
requester-VPC	vpc-0777a0c1649ff1ab4	Available	Off	192.168.0.0/16	dopt-09149d5972dcad...	rtb-0a7c459e61b1aa146	ad-0ec1fabdf657272d8...	def
-	vpc-0394b0c02059f4c08	Available	Off	172.31.0.0/16	dopt-09149d5972dcad...	rtb-0a7c459e61b1aa146	ad-025c8a4ee816722b6...	def

18. Click Edit route.

Details Info

Route table ID	Main	Explicit subnet associations	Edge associations
rtb-0a7c459e61b1aa146	Yes	-	-
VPC	Owner ID	536697231935	-

Actions ▾

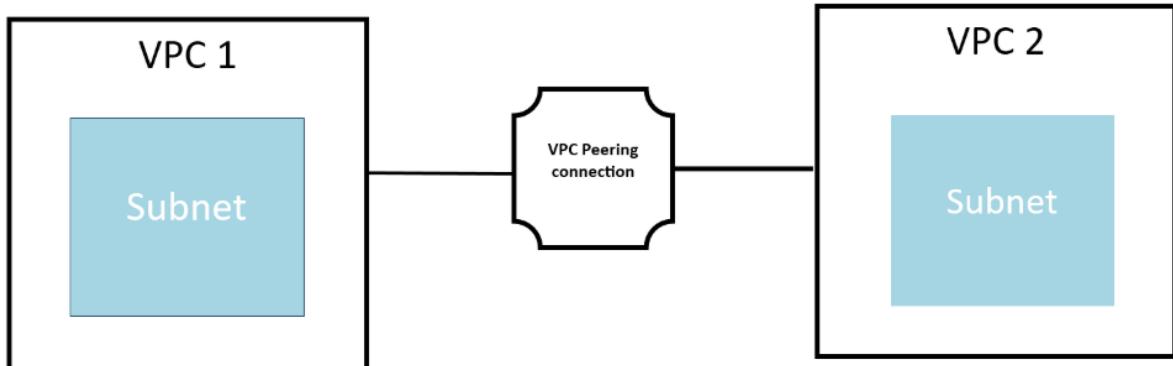
rtb-0a7c459e61b1aa146

Routes (1)	Both ▾	edit routes
Destination Target Status Propagated	192.168.0.0/16 local Active No	< 1 > ⌂

19. Add the accepter VPC's CIDR (172.16.0.0/16) to the **Destination**, set **Target** to the peering connection, select the created peering connection, and save changes.

The screenshot shows the AWS VPC console with the URL us-east-2.console.aws.amazon.com/vpcconsole/home?region=us-east-2#EditRoutes:RouteTableId=rtb-0d6e66b374058101b. The page displays the 'Edit routes' section for a specific route table. A red arrow points to the 'Save changes' button at the bottom right of the form.

20. Peering connection successfully established.



The VPC peering connection has been successfully established. Resources in the accepter VPC (172.16.0.0/16) can now communicate with resources in the requester VPC (192.168.0.0/16), and vice versa, using private IP addresses, ensuring secure and direct inter-VPC communication.

Project Validation

To validate the setup, we performed connectivity tests:

- Public Subnet Connectivity:** Connected to the **Public-EC2** instance via SSH using its public IP and tested outbound internet access by cloning a repository, confirming the Internet Gateway's functionality.
- Private Subnet Isolation:** Attempted to connect to the **Private-EC2** instance using its public IP, which failed as expected, verifying that the private subnet is not directly accessible from the internet.
- NAT Gateway Verification:** Observed outbound traffic metrics (Packet Out to Destination and Byte Out to Destination) in the NAT Gateway dashboard, confirming that the private subnet can access the internet securely.

- **Intra-Subnet Communication:** Created a second EC2 instance ([Public-EC2-2](#)) in the public subnet, established SSH connectivity between the two public EC2 instances, and verified reachability using ping, confirming default intra-subnet communication within the VPC.
- **VPC Peering:** Configured route tables for both VPCs to include each other's CIDR blocks, enabling direct communication between resources in the requester and accepter VPCs.

Project Reflection

- **Achievements:**
 - Successfully completed all project tasks, including setting up a VPC, configuring subnets, attaching an Internet Gateway, enabling outbound access via a NAT Gateway, and establishing VPC peering.
 - Gained hands-on expertise in navigating the AWS Management Console to deploy and manage VPC resources.
 - Demonstrated the ability to configure secure network architectures, ensuring resources in public subnets have internet access while private subnets remain isolated yet capable of outbound connectivity.
 - Established and validated VPC peering, enabling seamless communication between two VPCs.
- **Challenges and Solutions:**
 - Encountered CIDR block size limitations during VPC creation, resolved by adjusting the CIDR block to fall within the recommended range (/16 to /28).
 - Ensured non-overlapping CIDR blocks for VPC peering to avoid conflicts, reinforcing the importance of proper IP address planning.
 - Configured security group rules and route tables meticulously to enable connectivity while maintaining security best practices.
- **Key Learnings:**
 - Developed a deeper understanding of cloud networking concepts, including the roles of subnets, gateways, and route tables in AWS VPCs.
 - Learned the significance of network security measures like NAT Gateways and VPC endpoints for protecting sensitive data.
 - Gained practical experience in troubleshooting network configurations and validating connectivity through real-world tests.

Conclusion

This project provided a comprehensive exploration of AWS VPC fundamentals, from creating and configuring a VPC to enabling secure internet access and inter-VPC communication. The hands-on exercises reinforced theoretical knowledge with practical application, enhancing proficiency in cloud network architecture. The successful setup of public and private subnets, Internet and NAT Gateways, and VPC peering connections demonstrates the ability to design secure and scalable cloud environments. The project also highlighted the importance of network security and proper configuration to prevent unauthorized access while enabling necessary connectivity.

Overall, this mini-project offered valuable insights into cloud networking and practical skills in deploying VPC infrastructure on AWS, equipping participants with the knowledge and confidence to tackle more complex cloud networking challenges in the future.
