

Placement Empowerment Program

Cloud Computing and DevOps Centre

Set a private network in cloud – Create a VPC with subnets for your instances. Configure routing for internal communication between subnets

Name : Monisha JR

Department: CSE

Introduction

A Virtual Private Cloud (VPC) is a secure and isolated portion of a cloud provider's infrastructure where you can deploy your resources in a controlled environment. Setting up a VPC involves creating subnets, configuring routing, and implementing security measures to manage traffic and access. This setup is essential for applications that require secure internal communication while being accessible to external networks when necessary.

Objectives

1. **Create a VPC:** Establish a private network in the cloud that suits your application requirements.
2. **Configure Subnets:** Design and implement subnets within the VPC for different types of instances (e.g., public and private).
3. **Set Up Routing:** Configure routing tables to enable internal communication between subnets and external access as required.
4. **Implement Security:** Use security groups and network ACLs to control inbound and outbound traffic to your instances.
5. **Ensure High Availability:** Distribute resources across multiple Availability Zones to enhance resilience.

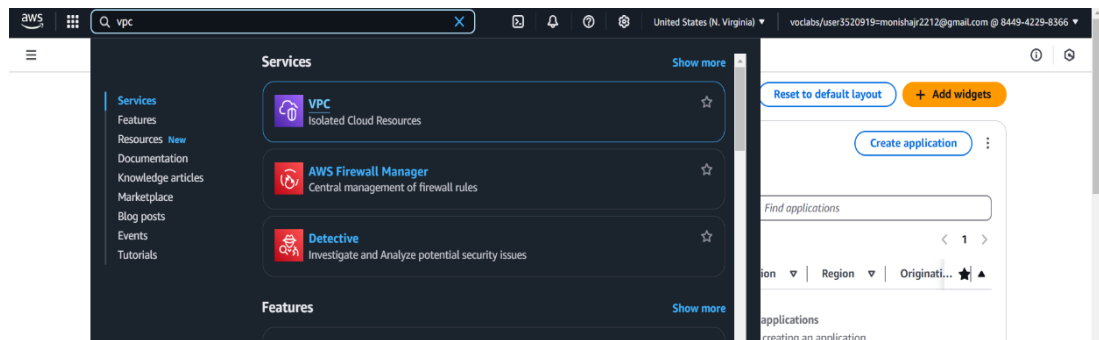
Importance

- **Security:** A VPC allows you to maintain a secure environment, isolating your resources from public internet exposure while enabling controlled access.
- **Customization:** You can tailor the network architecture to meet specific needs, such as private IP addressing and subnet segmentation.
- **Cost Efficiency:** Efficiently using cloud resources helps in managing costs associated with data transfer and resource allocation.
- **Scalability:** Easily scale your infrastructure to accommodate growing workloads without compromising security or performance.
- **Control:** Gain complete control over the networking environment, including IP address ranges, routing, and access controls.

Step-by-Step Overview

Step 1:

1. Go to [AWS Management Console](#).
2. Enter your username and password to log in



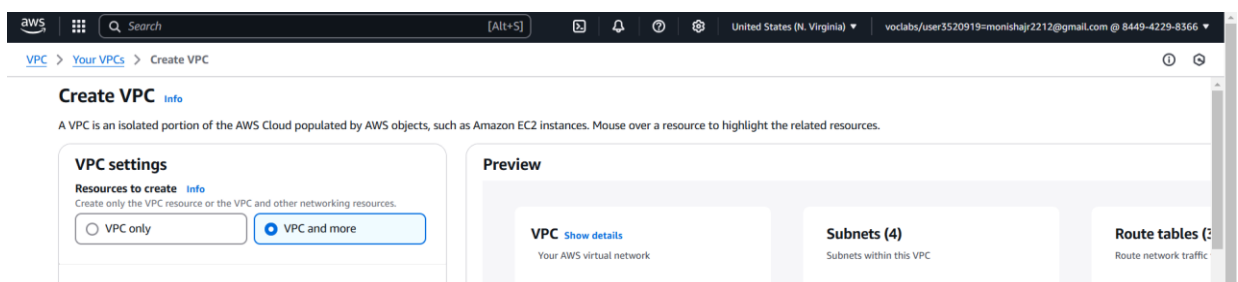
Step 2:

Navigate to the VPC Dashboard

- In the Services menu, select "VPC" to access the VPC Dashboard.
-

Create a VPC

- Click on "Your VPCs" in the left menu, then click "Create VPC."
- Specify the following:
 - **Name tag:** A name for your VPC.
 - **IPv4 CIDR block:** E.g., 10.0.0.0/16 (this gives you 65,536 IP addresses).
 - **IPv6 CIDR block:** (Optional).
 - **Tenancy:** Default is usually sufficient.
- Click "Create."



VPC > Your VPCs > Create VPC

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.

vpc-1

IPv4 CIDR block [Info](#)
☒ IPv4 CIDR manual input
☐ IPAM-allocated IPv4 CIDR block

IPv4 CIDR
10.0.0.0/24
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
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	
Q Name	Q vpc-1	Remove tag

[Add tag](#)

aws Search [Alt+S] United States (N. Virginia) voclabs/user3520919=monishajr2212@gmail.com @ 8449-4229-8366

VPC > Your VPCs > vpc-01f857c112b9f4a29

VPC dashboard [EC2 Global View](#)
[Filter by VPC](#)

Virtual private cloud
Your VPCs
Subnets
Route tables
Internet gateways
Egress-only internet gateways
Carrier gateways
DHCP option sets
Elastic IPs

You successfully created vpc-01f857c112b9f4a29 / myvpc

vpc-01f857c112b9f4a29 / myvpc [Actions](#)

Details Info			
VPC ID vpc-01f857c112b9f4a29	State Available	Block Public Access <input type="radio"/> Off	DNS hostnames Disabled
DNS resolution Enabled	Tenancy default	DHCP option set dopt-014d0b9cd157b2d5e	Main route table rtb-02a62949c377000d0
Main network ACL acl-0a5d8bcb27d85a077	Default VPC No	IPv4 CIDR 10.0.0.0/16	IPv6 pool -
IPv6 CIDR (Network border group) -	Network Address Usage metrics Disabled	Route 53 Resolver DNS Firewall rule groups -	Owner ID 844942298366

Step 3: Create Subnets

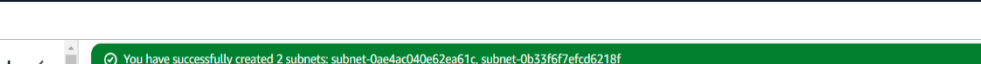
You need at least two private subnets for internal communication:

- 1. Go to Subnets → Click Create Subnet.**
- 2. Select the VPC (MyPrivateVPC) you created earlier.**
- 3. Create two subnets:**

Subnet 1 (Private-Subnet-A)

IPv4 CIDR: 10.0.1.0/24

IPv4 CIDR: 10.0.2.0/24



The screenshot shows the AWS VPC console interface. At the top, there's a navigation bar with the AWS logo, a search bar, and various utility icons. Below this, the 'VPC dashboard' is visible on the left sidebar. The main content area displays a green success message: 'You have successfully created 2 subnets: subnet-0ae4ac040e62ea61c, subnet-0b33f67efcd6218f'. Below this, the 'Subnets (2)' section is active, showing a table of the created subnets. The table has columns for Name, Subnet ID, State, VPC, Block Public..., and IPv4 CIDR. Two subnets are listed: 'mysubnet1' with ID 'subnet-0ae4ac040e62ea61c' and 'mysubnet2' with ID 'subnet-0b33f67efcd6218f'. Both are in the 'Available' state and associated with the 'vpc-01f857c112b9f4a29' VPC. The 'Block Public...' checkbox is turned off for both. The IPv4 CIDR addresses are '10.0.1.0/2' for mysubnet1 and '10.0.2.0/2' for mysubnet2.

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
mysubnet1	subnet-0ae4ac040e62ea61c	Available	vpc-01f857c112b9f4a29 myvpc	Off	10.0.1.0/2
mysubnet2	subnet-0b33f67efcd6218f	Available	vpc-01f857c112b9f4a29 myvpc	Off	10.0.2.0/2

Step 4:

Configure Route Tables for Internal Communication

1. Go to Route Tables → Click Create Route Table.
2. Name it (e.g., PrivateRouteTable).
3. Select MyPrivateVPC.
4. Click Create.

The screenshot shows the 'Create route table' page in the AWS Management Console. The breadcrumb navigation is 'VPC > Route tables > Create route table'. The page title is 'Create route table' with an 'Info' link. A description states: '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.
Input field: route1

VPC
The VPC to use for this route table.
Dropdown menu: vpc-01f857c112b9f4a29 (myvpc)

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
Input field: Name

Value - optional
Input field: route1
Buttons: Remove, Add new tag

Buttons: Cancel, Create route table

The screenshot shows the 'Route table details' page in the AWS Management Console. The breadcrumb navigation is 'VPC > Route tables > rtb-0f5bf8f56389d0f21'. A green success message at the top states: 'Route table rtb-0f5bf8f56389d0f21 | route1 was created successfully.'

rtb-0f5bf8f56389d0f21 / route1

Details

Route table ID rtb-0f5bf8f56389d0f21	Main No	Explicit subnet associations -	Edge associations -
VPC vpc-01f857c112b9f4a29 myvpc	Owner ID 844942298366		

Buttons: Routes, Subnet associations, Edge associations, Route propagation, Tags

Step 5:

Associate the subnets:

Go to Subnet Associations → Click Edit subnet associations.

Select Private-Subnet-A and Private-Subnet-B.

Click Save associations.

The screenshot shows the AWS VPC console interface. The left sidebar contains navigation links for VPC, Route tables, and various network components. The main content area is titled 'rtb-0f5bf8f56389d0f21' and has tabs for Routes, Subnet associations, Edge associations, Route propagation, and Tags. The 'Subnet associations' tab is active, showing 'Explicit subnet associations (0)' and 'Subnets without explicit associations (2)'. The subnets listed are mysubnet1 and mysubnet2.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR
mysubnet1	subnet-0ae4ac040e62ea61c	10.0.1.0/24	-
mysubnet2	subnet-0b33f67efcd6218f	10.0.2.0/24	-

The screenshot shows the 'Edit subnet associations' page in the AWS VPC console. The page title is 'Edit subnet associations' with a subtitle 'Change which subnets are associated with this route table.' The 'Available subnets (2/2)' section shows a table with columns for Name, Subnet ID, IPv4 CIDR, IPv6 CIDR, and Route table ID. The subnets listed are mysubnet1 and mysubnet2. The 'Selected subnets' section shows the same two subnets selected.

Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID
mysubnet1	subnet-0ae4ac040e62ea61c	10.0.1.0/24	-	Main (rtb-02a62949c377000d0)
mysubnet2	subnet-0b33f67efcd6218f	10.0.2.0/24	-	Main (rtb-02a62949c377000d0)

The screenshot shows the 'Route tables' page in the AWS VPC console. A green success message at the top states: 'You have successfully updated subnet associations for rtb-0f5bf8f56389d0f21 / route1.' The route table details are shown, including the Route table ID, VPC, Main status, Explicit subnet associations, and Edge associations. The 'Routes' tab is active, showing a single route with destination 10.0.0.0/16, target local, status Active, and propagated No.

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No

Step 6:

Default route: 10.0.0.0/16 → local (Automatically added).

rtb-0f5bf8f56389d0f21 / route1 Actions ▼

Details info
Route table ID
rtb-0f5bf8f56389d0f21
VPC
vpc-01f857c112b9f4a29 | myvpc

Main
No
Owner ID
844942298366

Explicit subnet associations
2 subnets

Edge associations
–

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

Routes (1) Both ▼ Edit routes

Destination ▼	Target ▼	Status ▼	Propagated ▼
10.0.0.0/16	local	Active	No

Step 7:

Launch Instances in Private Subnets

1. Go to EC2 Dashboard → Launch Instance.
2. Select an AMI (Amazon Linux, Ubuntu, etc.).
3. Choose an Instance Type (e.g., t2.micro).
4. Under Network settings:

Select MyPrivateVPC.

Select Private Subnet-A or Private-Subnet-B.

Disable Auto-assign Public IP (to keep it private).

EC2 > Instances > Launch an instance

Launch an instance [Info](#)

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

Name and tags [Info](#)

Name

[Add additional tags](#)

▼ Application and OS Images (Amazon Machine Image) [Info](#)

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

Recents

My AMIs

Quick Start

Amazon Linux

macOS

Ubuntu

Windows

Red Hat

SUSE Linux

Debian

[Browse more AMIs](#)
 Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI
 ami-0c614dee691cbbf37 (64-bit (x86), uefi-preferred) / ami-0b29c89c15cfb8a6d (64-bit (Arm), uefi)
 Virtualization: hvm ENA enabled: true Root device type: ebs

Free tier eligible

Description

Number of instances: 1

Software I: Amazon Lin ami-0c614de

Virtual ser: t2.micro

Firewall (s): default

Storage (v): 1 volume(s)

Free (or t: insta publ stora banc

Cancel

Step 8:

Enable Internal Communication

Instances inside the private subnets can communicate without an internet gateway.

If instances need internet access (for updates, etc.), configure a NAT Gateway in a Public Subnet.

Use Security Groups to allow inbound traffic only from internal sources (e.g., allow SSH from 10.0.0.0/16).

Step 9 :

Now, your private network is set up, and instances inside can communicate securely! Let me know if you need extra configurations like VPN, Bastion Host, or NAT setup.

Outcome :

After following these steps, you will have:

- A VPC that is isolated from other networks.
- One or more subnets for your instances, with at least one public subnet that can communicate with the Internet.
- Proper routing configured for internal communication between subnets.

