

# Experiment No 08

**Aim:**

To design and deploy a secure Virtual Private Cloud (VPC) on Amazon Web Services (AWS) that provides isolated networking resources for hosting and managing IoT services.

**Tools Required:**

- AWS Management Console / AWS CLI
- AWS VPC Service (Virtual Private Cloud)
- Subnets (Public & Private)
- Route Tables, Internet Gateway (IGW), NAT Gateway
- Security Groups & Network ACLs
- AWS IoT Core
- EC2 Instance for IoT device simulation

**Theory:**

A Virtual Private Cloud (VPC) on AWS creates a logically isolated section of the AWS cloud where you can launch AWS resources in a virtual network that you define. VPCs offer deep control over networking, including selection of IP address ranges, creation of subnets, and configuration of route tables and network gateways.

**Importance for IoT:**

IoT services handle sensitive device data and demand secure, robust infrastructure. Deploying these services inside a VPC safeguards them through network-level isolation and robust access control. Within a VPC, public and private subnets can separate network-sensitive resources (e.g., databases) from those needing public internet access. Routing and firewalls—via route tables, security groups, and network ACLs—determine how and with whom resources communicate.

- **Internet Gateway (IGW):** Attach to a public subnet for outbound and inbound internet access.
- **NAT Gateway:** Allows resources in private subnets to access the internet securely, without exposing them to inbound connections.
- **Security Groups & NACLs:** These virtual firewalls rigorously control traffic into and out of your AWS resources.

- **AWS IoT Core Integration:** VPC provides the backbone for secure, scalable IoT data collection and processing, as IoT devices and cloud endpoints communicate over regulated, encrypted paths.

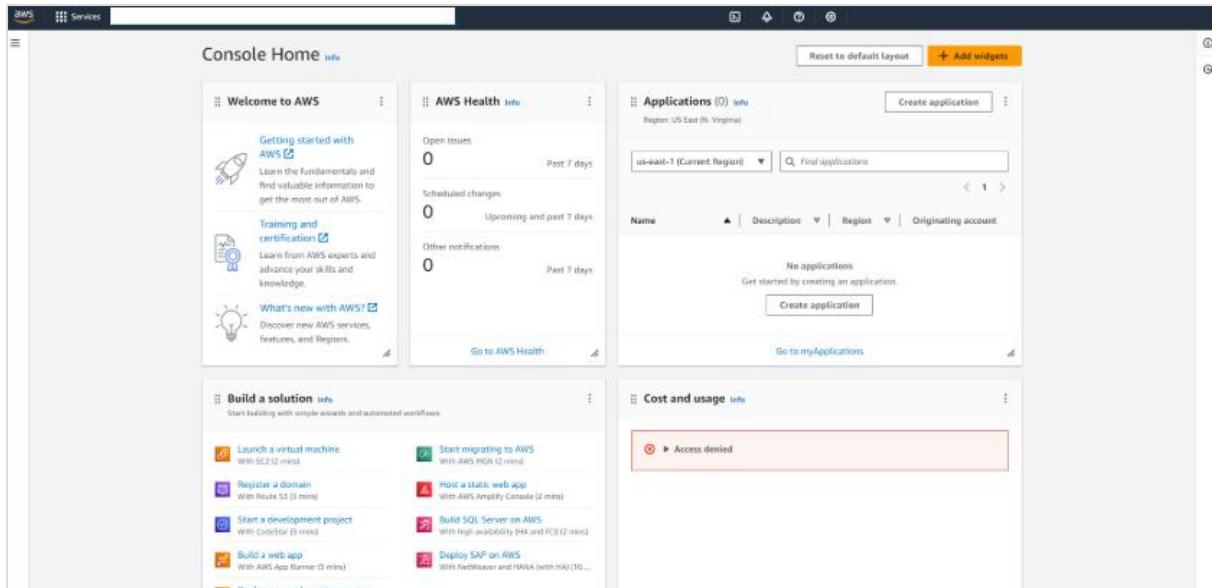
#### Key Benefits:

- Network segmentation and isolation for security
- Flexible routing and controlled access
- Compliance with security standards for sensitive workloads
- Easy scaling and reliable connectivity for diverse IoT scenarios

#### Procedure:

##### 1. Login to AWS Console:

- Sign in to your AWS account and access the AWS Management Console.



##### 2. Create a VPC:

- Go to the VPC Dashboard, click "Create VPC", and select the preset to include one public and one private subnet.
- Assign an appropriate IPv4 CIDR block (e.g., 10.0.0.0/16).

The screenshot shows the AWS VPC Dashboard. On the left, there's a sidebar with sections for Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only internet gateways, Carrier gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, Route servers), Security (Network ACLs, Security groups), and PrivateLink and Lattice. The main area displays 'Resources by Region' for N. Virginia, including VPCs, NAT Gateways, Subnets, VPC Peering Connections, Route Tables, Network ACLs, Internet Gateways, Security Groups, Egress-only Internet Gateways, Customer Gateways, and Virtual Private Gateways. A 'Create VPC' button is at the top. On the right, there are boxes for Service Health (refresh resources), Settings (Block Public Access, Zones, Console Experiments), Additional Information (VPC Documentation, All VPC Resources, Forums, Report an Issue), and AWS Network Manager (provides tools and features to help manage and monitor your network on AWS). The top right shows account information: Account ID: 5901-8379-5833, United States (N. Virginia), and a specific user's email.

This screenshot shows the 'Create VPC' wizard. It has two tabs: 'VPC only' (selected) and 'VPC and more'. Under 'VPC settings', it shows 'Name tag auto-generation' (auto-generated as 'project'), 'IPv4 CIDR block' (set to '10.0.0.0/16'), 'IPv6 CIDR block' (set to 'No IPv6 CIDR block'), and 'Tenancy' (set to 'Default'). The 'Preview' section shows the VPC structure: 'VPC Show details' (project-vpc), 'Subnets (4)' (us-east-1a, us-east-1b, project-subnet-public1-us-east-1a, project-subnet-private1-us-east-1a), and 'Route tables (3)' (project-rtb-public, project-rtb-private1-us-east-1a, project-rtb-private2-us-east-1b).

### 3. Configure Subnets:

- Create a public subnet (e.g., 10.0.1.0/24) for exposing the IoT gateway or EC2 instance.
- Create a private subnet (e.g., 10.0.2.0/24) for backend processing services.

This screenshot continues the 'Create VPC' wizard. It shows 'Customize AZs' with options for 'Number of public subnets' (0, 2, 4 selected), 'Number of private subnets' (0, 2, 4 selected), and 'NAT gateways (5)' (None, In 1 AZ, 1 per AZ selected). It also shows 'Customize subnets CIDR blocks' and 'VPC endpoints' (None, S3 Gateway selected). The 'Preview' section is identical to the previous one, showing the VPC structure with 4 subnets and 3 route tables.

**Create VPC workflow**

**Success**

**Details**

- Create VPC: vpc-0874482465d323eb5
- Enable DNS hostnames
- Enable DNS resolution
- Verifying VPC creation: vpc-0874482465d323eb5
- Create S3 endpoint: vpc-0cd6c7bdeb0f2d166
- Create subnet: subnet-0e0475985f90fc081
- Create subnet: subnet-0aebd34a8cfcbfb93
- Create subnet: subnet-0fe47411badcc4111
- Create subnet: subnet-04a11c3419744b521
- Create internet gateway: igw-0b881c8ee00fa59ea
- Attach internet gateway to the VPC
- Create route table: rtb-0deadb4d33adcab7e
- Create route
- Associate route table
- Associate route table
- Create route table: rtb-0d968238128bc2ff3
- Associate route table
- Create route table: rtb-0eed3c6a232d9934b6
- Associate route table
- Verifying route table creation
- Associate S3 endpoint with private subnet route tables: vpc-0cd6c7bdeb0f2d166

**VPC dashboard**

**vpc-0874482465d323eb5 / project-vpc**

**Details**

VPC ID: vpc-0874482465d323eb5	State: Available	Block Public Access: Off	DNS hostnames: Enabled
DNS resolution: Enabled	Tenancy: default	DHCP option set: dopt-003765be6939a293d	Main route table: rtb-0b2633ab92d4fe690
Main network ACL: ad-09d0b31a277561c7	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: Failed to load rule groups	Owner ID: 590183795833

**Resource map**

**Actions**

#### 4. Attach Internet Gateway:

- Create and attach an IGW to the VPC.
- Attach the IGW to the public subnet for internet connectivity.

**Create internet gateway**

An internet gateway is a virtual router that connects a VPC to the internet. To create a new internet gateway specify the name for the gateway below.

**Internet gateway settings**

Name: my-internet-gateway

**Tags - optional**

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.

Add new tag

Cancel Create internet gateway

#### 5. Set Up Route Tables:

- Configure the route table for the public subnet to allow traffic through the IGW.

- Associate the private subnet with a NAT Gateway (optional) for secure outbound internet access.

The screenshot shows the AWS VPC Route tables page. The left sidebar includes sections for EC2 Global View, Virtual private cloud (Your VPCs, Subnets, Route tables), Security (Network ACLs, Security groups), and PrivateLink and Lattice. The main content area displays a table titled "Route tables (7) Info" with columns for Name, Route table ID, Explicit subnet associations, Edge associations, Main, VPC, Owner ID, and Last updated. The table lists various route tables, including "project-rtb-private2-us-east-1b" and "Public Route Table".

## 6. Configure Security Groups:

- Define security group rules to allow necessary traffic (e.g., MQTT ports 1883/8883 for IoT, HTTPS port 443 for cloud APIs).

The screenshot shows the AWS Create security group page. It has sections for Basic details (Security group name: MyWebServerGroup, Description: Allows SSH access to developers, VPC info: vpc-04af7533e95b6593), Inbound rules (empty), Outbound rules (Type: All traffic, Protocol: All, Port range: All, Destination: Anywhere, Description: optional), and Tags - optional (empty). A note at the bottom states: "Rules with destination of 0.0.0.0 / -/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses."

## 7. Launch EC2 Instance:

- Deploy an EC2 instance within the public subnet for IoT gateway simulation.
- Configure the instance to communicate securely with AWS IoT Core.

**Resources**

You are using the following Amazon EC2 resources in the United States (N. Virginia) Region:

Instances (running)	1	Auto Scaling Groups	0 API Error	Capacity Reservations	0
Dedicated Hosts	0	Elastic IPs	0	Instances	1
Key pairs	1	Load balancers	0 API Error	Placement groups	0
Security groups	4	Snapshots	0	Volumes	1

**Launch instance**

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

**Service health**

An error occurred

An error occurred retrieving service health information

**Zones**

Zone name	Zone ID
us-east-1a	use1-az4
us-east-1b	use1-az6
us-east-1c	use1-az1

**Account attributes**

**Default VPC**

vpc-04af7533e9a5b6593

**Settings**

Data protection and security  
Allowed AMIs  
Zones  
EC2 Serial Console  
Default credit specification  
EC2 console preferences

**Explore AWS**

Enable Best Price-Performance with AWS Graviton2

AWS Graviton2 powered EC2 instances enable up to 40% better price performance for a broad spectrum of cloud workloads. [Learn more](#)

Amazon GuardDuty Malware Protection

GuardDuty now provides agentless malware detection in Amazon EC2 & EC2 container workloads. [Learn more](#)

Save up to 90% on EC2 with Spot Instances

Optimize price-performance by combining EC2 purchase options in a single EC2 ASG. [Learn more](#)

## 8. Integrate with AWS IoT Core:

- Register a virtual IoT device in AWS IoT Core.
- Generate security certificates and attach the correct policies.
- Configure the EC2 gateway or a simulated device to publish and subscribe to IoT messages as per requirements.

**AWS IoT**

Monitor

Connect

Connect one device

Connect many devices

Domain configurations

Test

Device Advisor

MQTT test client

Device Location

Query connectivity status

Manage

All devices

Greengrass devices

LPWAN devices

Network analyzer

Coverage [New](#)

Gateways

**AWS IoT Core for LoRaWAN**

Connect and manage LoRaWAN gateways and devices with AWS cloud

Setup a private LoRaWAN network by connecting your own devices and gateways with no LoRaWAN Network Server setup required.

**How it works**

**Get started with AWS IoT Core for LoRaWAN**

Register your private LoRaWAN gateways and devices. [Learn more](#)

**Pricing - US East (N. Virginia)**

[Learn More](#)

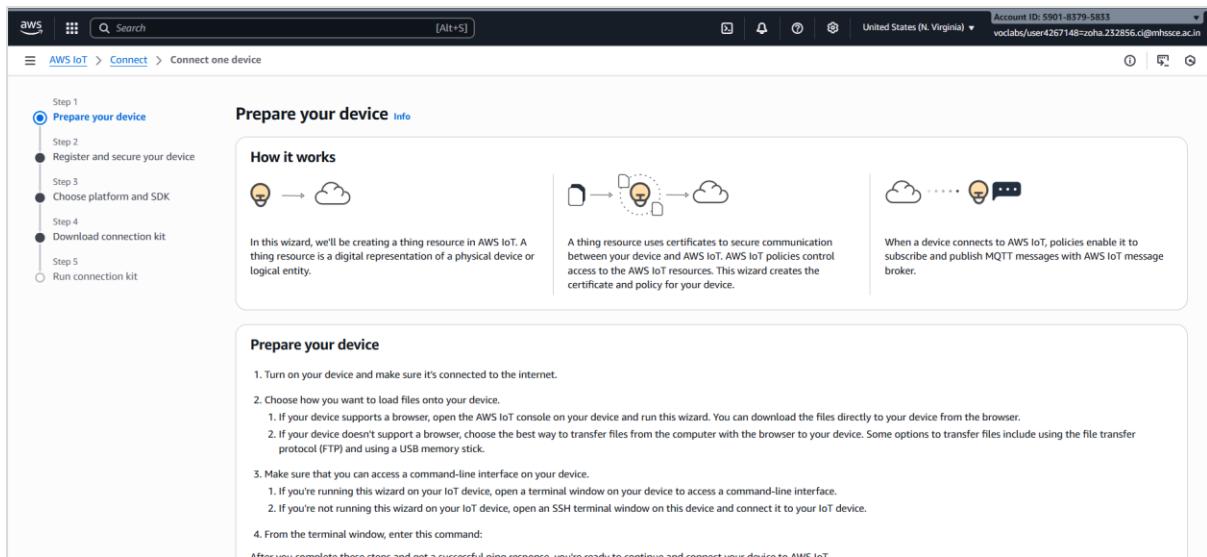
**Getting started**

[AWS IoT Core for LoRaWAN Workshop](#)

[LoRaWAN Feature page](#)

## 9. Test & Validate Communication:

- Use the AWS IoT MQTT test client to verify secure communication between your EC2 instance (or IoT simulator) and AWS IoT Core endpoints via your VPC.



## Result:

A secure VPC was successfully deployed and configured with public and private subnets, IGW, route tables, and robust security groups. The implemented architecture ensured that IoT devices and backend services could communicate securely and efficiently, with strict control over network exposure.

## Conclusion:

This experiment demonstrated the deployment of an AWS VPC tailored for IoT scenarios, emphasizing security, isolation, and flexibility. Utilizing subnet segmentation, routing, and security configurations, the experiment established a scalable cloud network that met the needs of modern IoT services while mitigating risks associated with public network exposure. A properly designed VPC infrastructure is fundamental to establishing secure, reliable, and manageable cloud environments for real-world IoT applications.