Report: Implementation of VPC Architecture in AWS

Name: Chaithanya M

Date: July 13, 2025

Topic: Implementation of High Availability VPC Architecture in AWS

1. Introduction

A Virtual Private Cloud (VPC) is a customizable, logically isolated network within the AWS Cloud. The architecture shown in the diagram is a high-availability, fault-tolerant design spanning across two Availability Zones (AZs), incorporating public and private subnets, NAT gateways, Application Load Balancer (ALB), Auto Scaling, and Security Groups. This VPC design ensures secure, scalable, resilient hosting for applications, backend services.

2. Objective

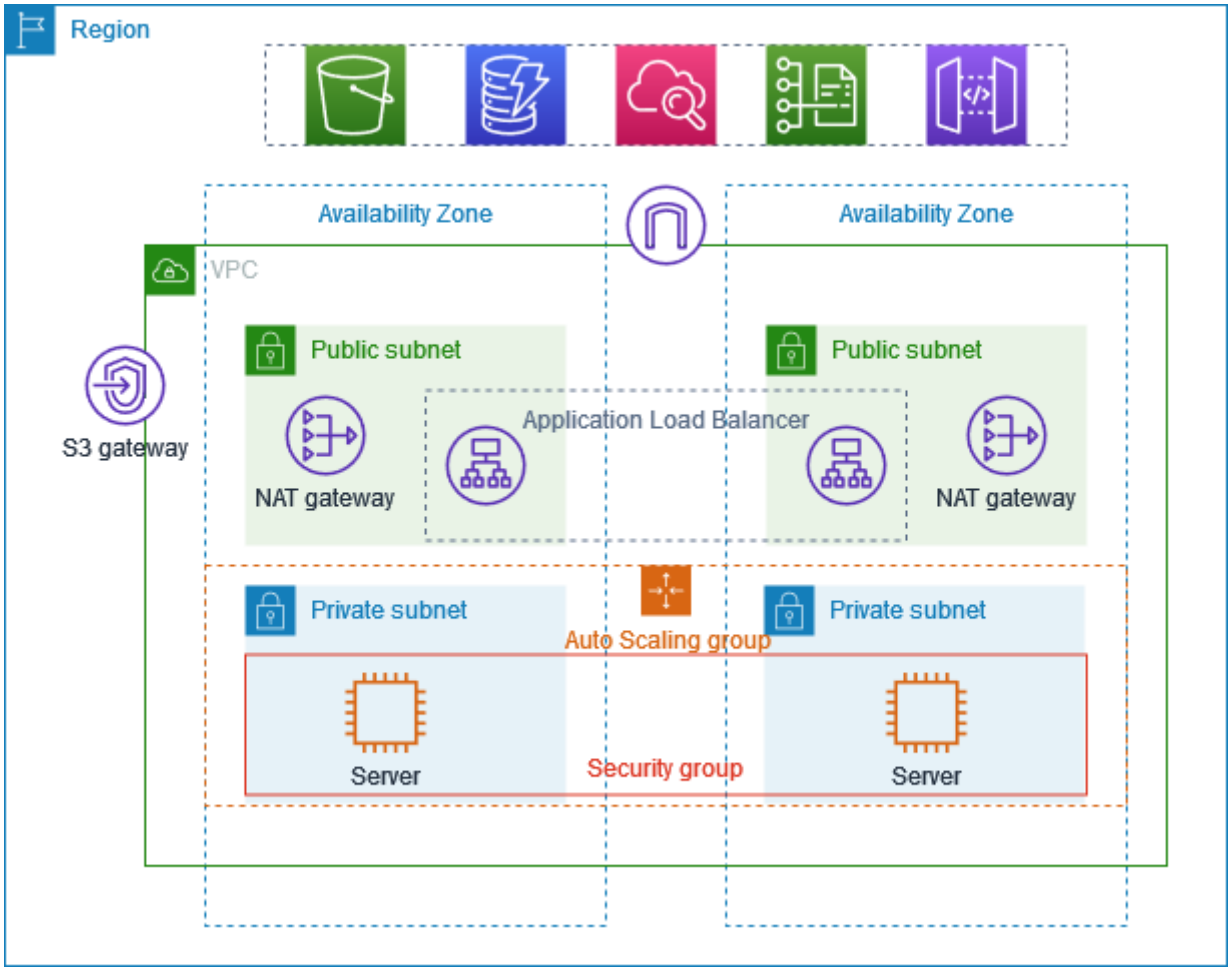
The key objectives of this VPC implementation are:

- To design a multi-AZ high-availability architecture for business-critical applications.
- To implement network segmentation using public and private subnets.
- To enable secure internet access for instances in private subnets via NAT gateways.
- To distribute incoming traffic through an Application Load Balancer (ALB).
- To ensure automatic scaling of application servers using an Auto Scaling Group.
- To define granular security controls using Security Groups.
- To allow S3 access using an S3 Gateway without exposing the instances to the internet.

3. Architecture Overview

Component	Description
VPC	The main container that holds all networking components.
Availability Zones	Two separate zones used to ensure high availability and fault tolerance.
Public Subnets	Host NAT Gateways and Load Balancer; accessible from the internet.

Component	Description
Private Subnets	Host EC2 Instances that run applications, not directly exposed to internet.
NAT Gateways	Provide internet access for private subnet instances to download updates.
Application Load Balancer (ALB)	Distributes incoming traffic across EC2 instances in different AZs.
Auto Scaling Group	Automatically adjusts the number of EC2 instances based on demand.
Security Groups	Firewall rules attached to instances to control traffic at the instance level.
S3 Gateway Endpoint	Enables private subnet instances to access S3 without internet gateway.



4. VPC Design Overview

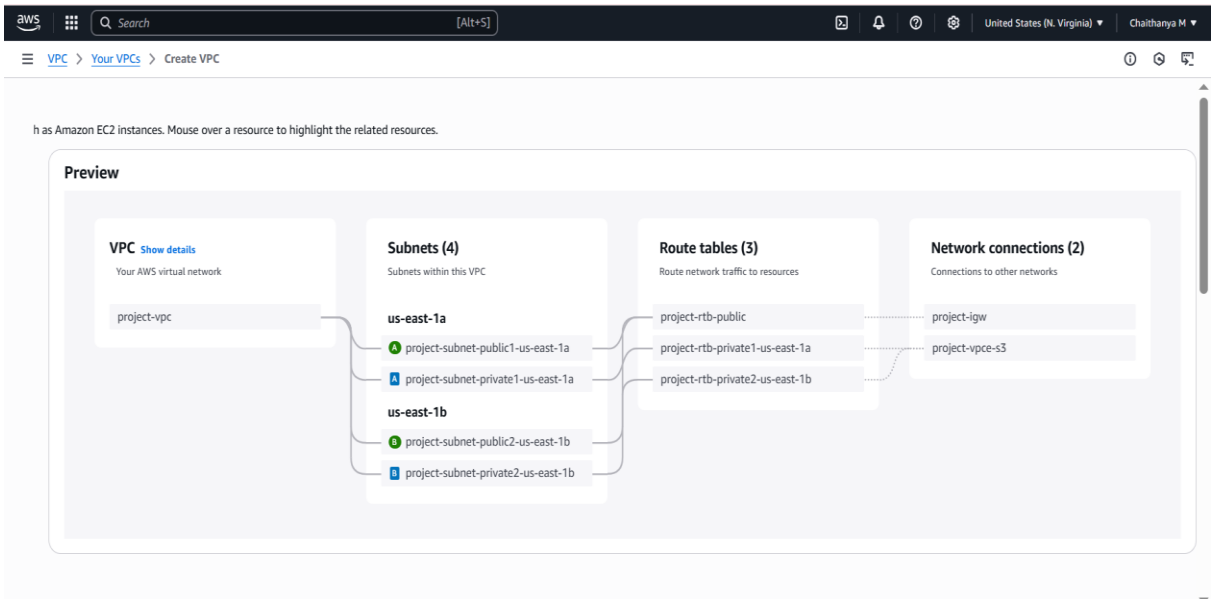
As shown in the architecture diagram and AWS screenshots:

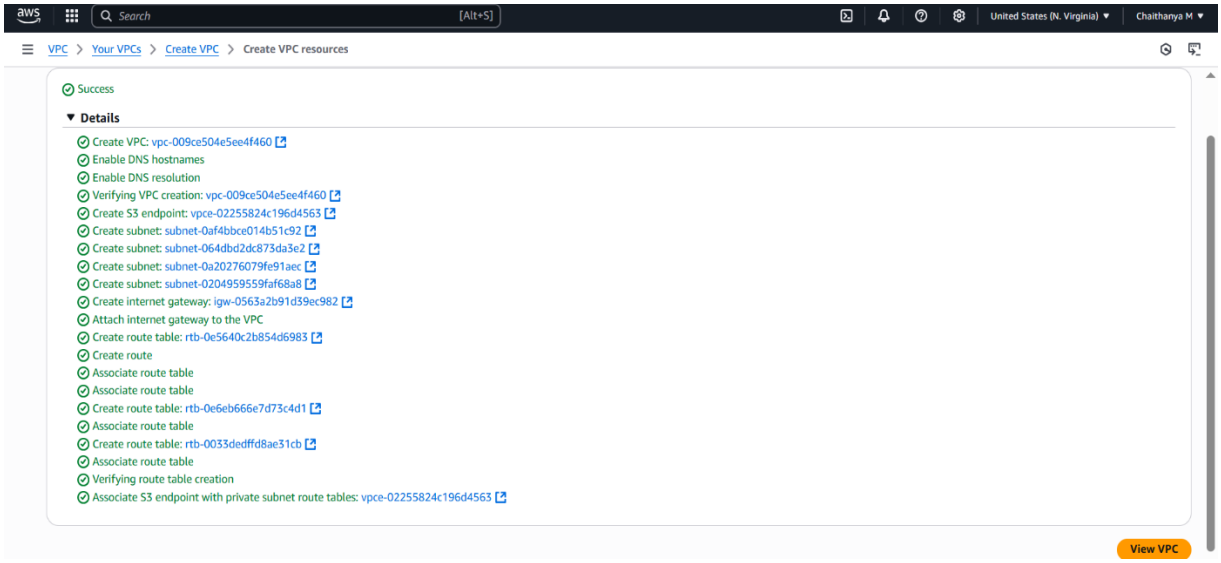
- **Region**: us-east-1
- **VPC Name**: project-vpc
- **Subnets**:
  - 2 Public Subnets (us-east-1a, us-east-1b)
  - 2 Private Subnets (us-east-1a, us-east-1b)
- **Gateways**:
  - Internet Gateway (igw-...) attached to public subnets.
  - NAT Gateways for private subnets to access the internet.
- **Route Tables**:
  - One public and two private route tables configured and associated accordingly.
- **Endpoints**:
  - S3 VPC Endpoint (vpce-s3) created and associated with private subnets.
- **Security Components**:
  - Security Group allows TCP (8000) and SSH (22) from all sources for demonstration.
  - Network ACL configured to allow inbound Custom TCP (8000) traffic.

5. Implementation

Step 1: Create VPC

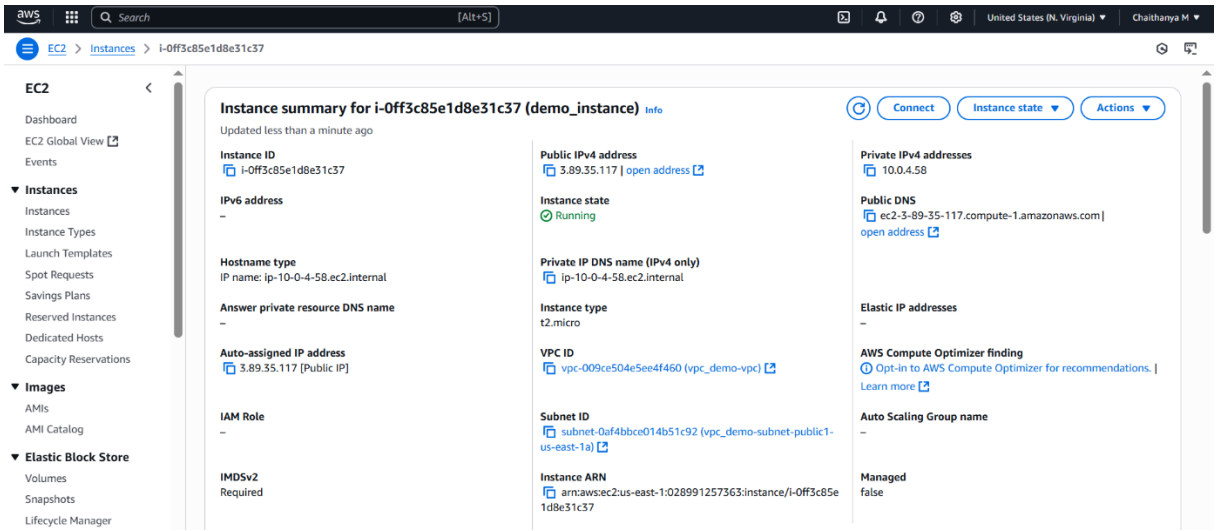
- Go to**:** VPC → Your VPCs → Create VPC
- Click Create VPC





Step 2: Launch EC2 Instance

- Go to EC2 → Launch Instance
- AMI: Ubuntu
- Instance Type**:** t2.micro
- Subnet**:** project-subnet-public1-us-east-1a
- Auto-assign Public IP**:** Enable
- Security Group**:**
    - Allow:
        - TCP 22 (SSH) from My IP
        - TCP 8000 from 0.0.0.0/0



Step 3: Run Python Web Server on EC2

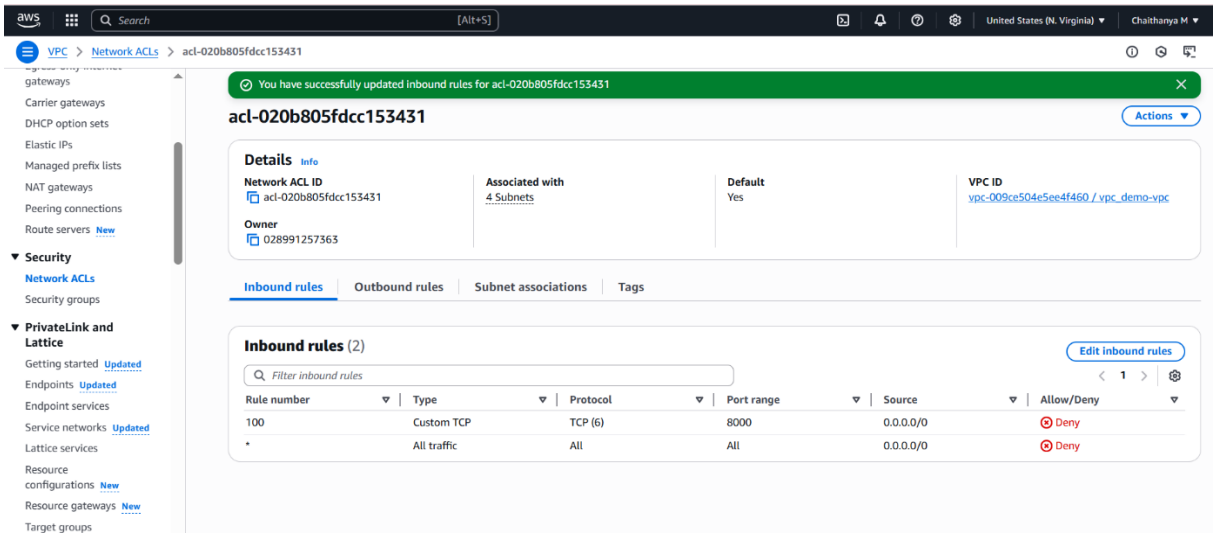
- SSH into the EC2:
  $ ssh -i key.pem ubuntu@3.89.35.117

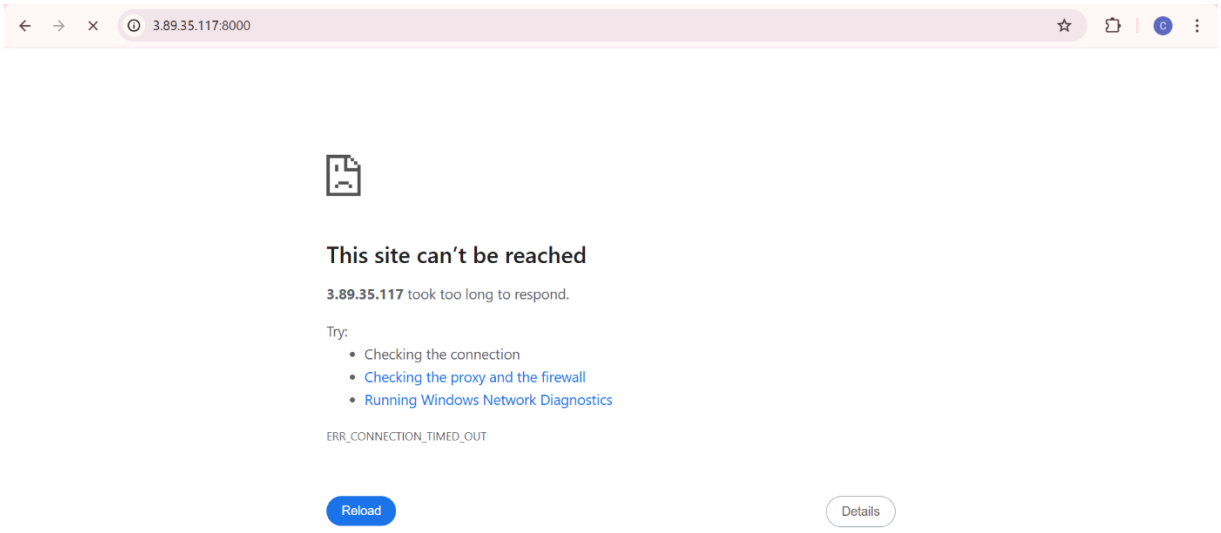
- Run HTTP server:
  $ python3 -m http.server 8000

- Test in browser:
  http://3.89.35.117:8000

Step 4: Impact of Denying Inbound Rules in NACL

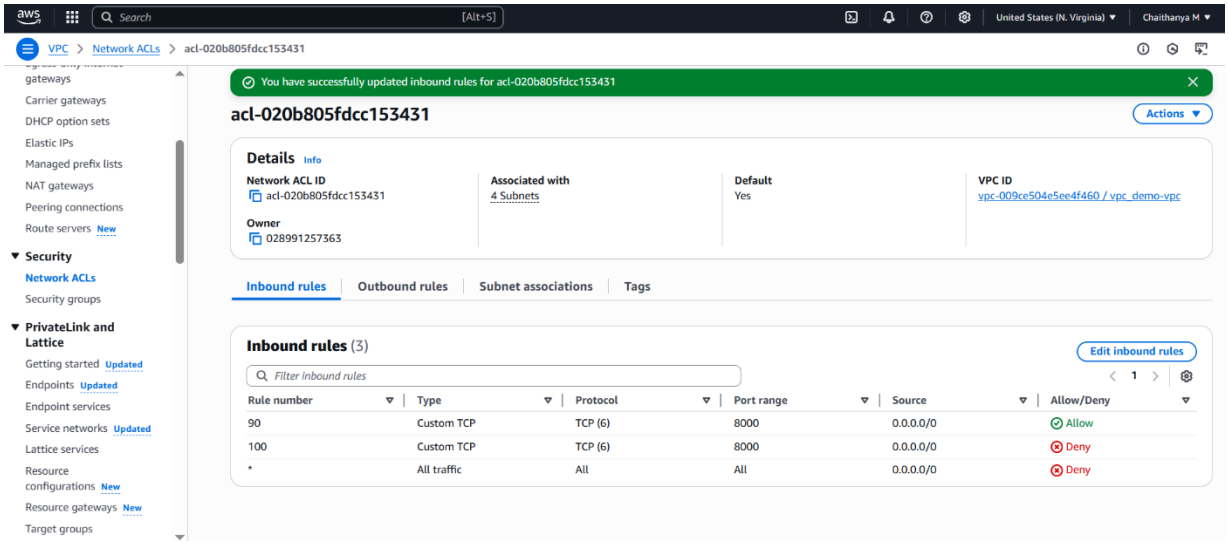
If both inbound rules in the Network ACL are set to "Deny," incoming traffic on specified ports (e.g., 8000 or 22) is completely blocked. This prevents access to hosted web applications and SSH connectivity. Despite a running EC2 instance and open security group, the NACL rules override access.





Step 5: Rule Number Priority

In a Network ACL, rule numbers determine the evaluation order — rules with lower numbers are evaluated first**,** and the first matching rule is applied. If an inbound rule with a lower number explicitly allow traffic on ports, that traffic is blocked allowed**,** even if a later rule allows it.



10. Conclusion

The AWS VPC implementation outlined in this report successfully demonstrates how to design and deploy a logically isolated network architecture in the cloud. Through the step-by-step configuration of VPC components including subnets, gateways, route tables, security groups, and network ACLs the setup provides a secure and flexible environment for hosting applications.

Key takeaways from this implementation include:

- **Granular Network Control**: The use of public and private subnets ensures that only intended components are exposed to the internet, reducing attack surfaces.
- **Security Layering**: Combining security groups (instance-level firewalls) and NACLs (subnet-level firewalls) offers layered protection. Understanding the priority of NACL rule numbers is critical to avoid misconfigurations that can block essential traffic.
- **Traffic Flow Design**: The deployment of a NAT Gateway allows instances in private subnets to securely access the internet without being exposed themselves.
- **Functional Testing**: Hosting a web server on an EC2 instance in a public subnet validated that the routing, gateway, and firewall rules are configured correctly.

This implementation also highlights the importance of testing each layer of the network from instance reachability to access control lists to ensure overall functionality and security.