Report: AWS Security Using Security Groups and NACLs

Name: Chaithanya M

Date: July 11, 2025

Topic: AWS Security Using Security Groups and NACLs

1. Introduction

Security is one of the foundational pillars of any cloud architecture. AWS provides several built-in services to ensure the security of its resources. Among them, Security Groups and Network Access Control Lists (NACLs) serve as core components of network security.

This project demonstrates how to secure EC2 instances and subnets using Security Groups (stateful firewalls) and NACLs (stateless filters), and shows how they complement each other to build a layered defence model.

2. Objective

- To understand and implement Security Groups for EC2-level access control.
- To configure NACLs for subnet-level network traffic filtering.
- To build a secure VPC environment using best practices.
- To test and validate the effectiveness of layered security in a real AWS environment.
- To follow AWS-recommended least privilege principles for secure infrastructure design.

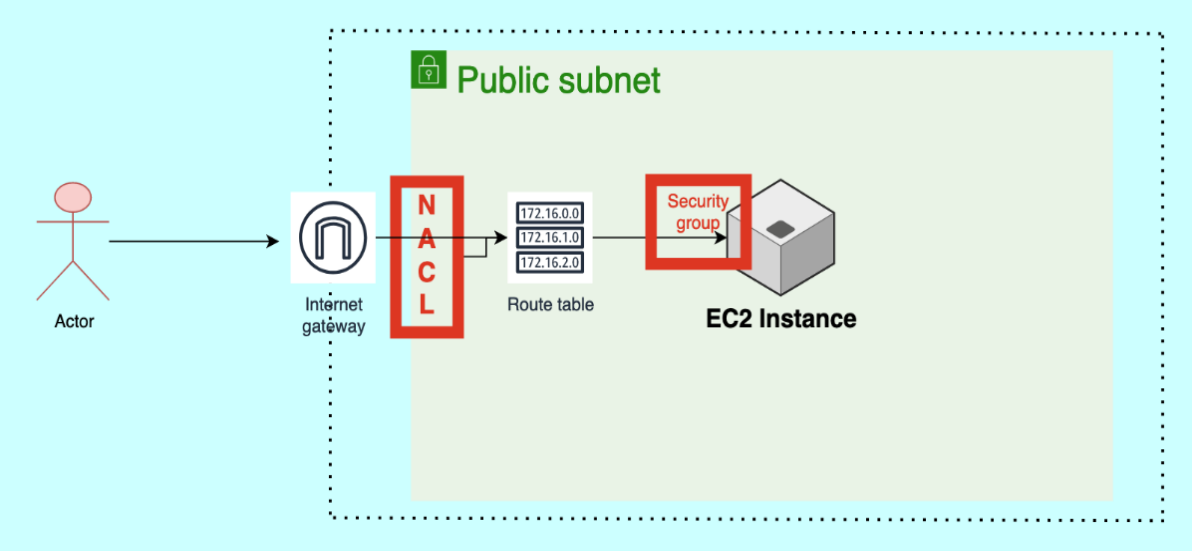
3. Architecture Overview

The architecture demonstrates a secure setup of an EC2 instance in a public subnet within a VPC using layered security controls:

- **Internet Gateway** allows internet access to the VPC.
- **Network ACL (NACL)** acts as a stateless firewall at the subnet level, filtering both inbound and outbound traffic based on IP and port rules.
- **Security Group** functions as a stateful firewall at the instance level, allowing only defined inbound and outbound traffic (e.g., HTTP, SSH).

- The **EC2 instance** is hosted in the public subnet and is protected by both NACL and Security Group for a defense-in-depth model.

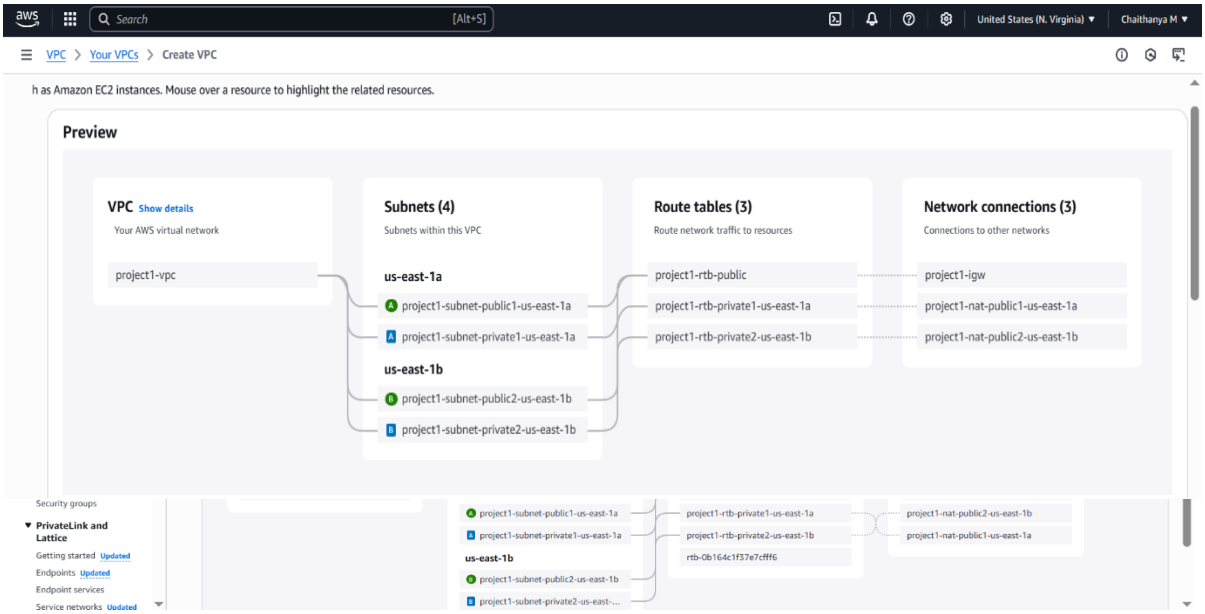
This layered approach ensures that only trusted and explicitly permitted traffic reaches the instance.



4. Implementation

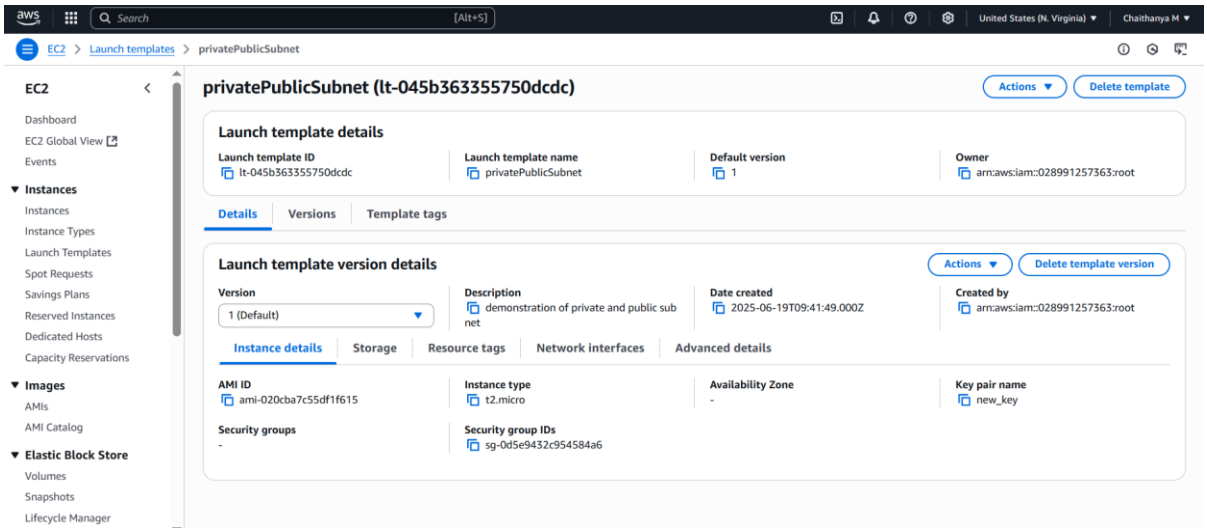
Step 1: VPC and Subnet Setup

- Create a Custom VPC (e.g., 172.16.0.0/16)
- Create a Public Subnet (e.g., 172.16.1.0/24)
- Enable Auto-assign Public IP for the subnet



Step 2: Launch Template Configuration

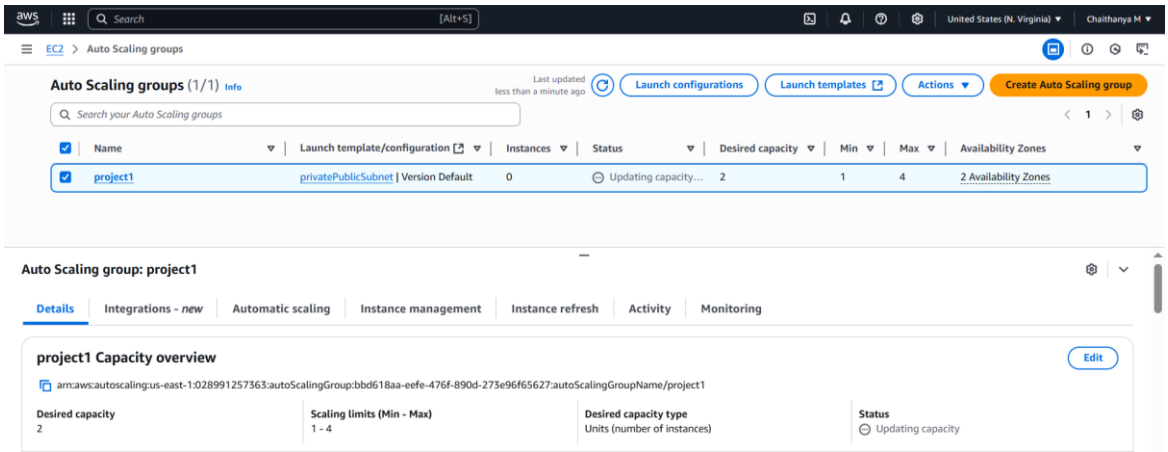
To ensure repeatable, secure, and scalable deployment of EC2 instances, a Launch Template named privatePublicSubnet was created. This template defines the instance configuration used to launch virtual machines into both public and private subnets within a custom VPC.



Step 3: Auto Scaling Group Configuration

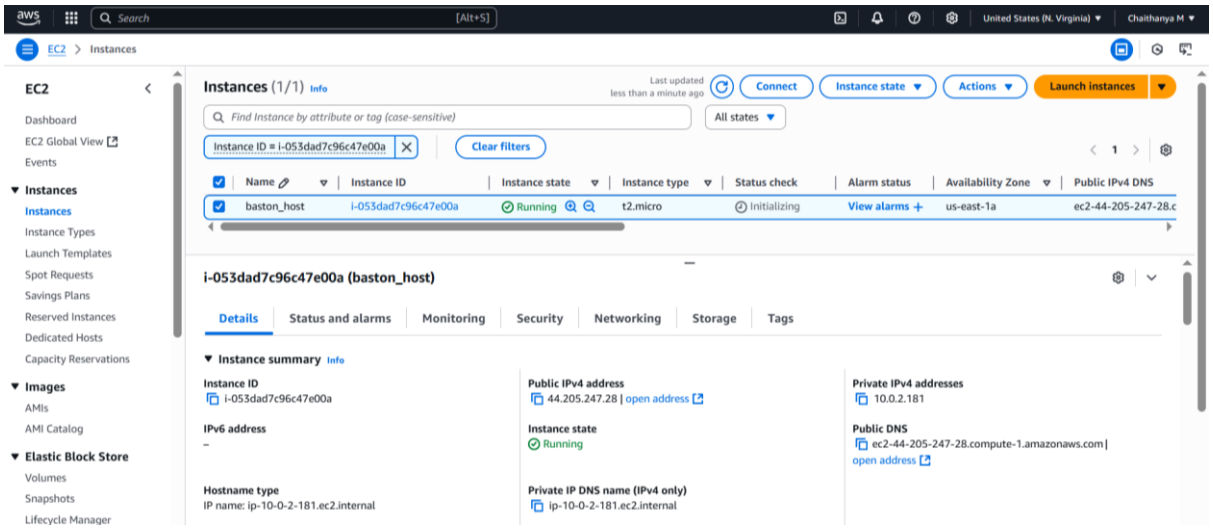
To ensure high availability and scalability of EC2 instances, an Auto Scaling Group (ASG) was configured using the previously defined Launch Template (privatePublicSubnet). This setup automatically adjusts the number of EC2 instances based on workload demand.

- The group maintains at least one running EC2 instance at all times.
- If CPU usage exceeds 50%, it automatically adds an additional instance (up to 2).
- When load decreases, the group scales back to the desired count.



Step 4: Bastion Host Configuration

To securely manage instances in private subnets, a Bastion Host (Jump Server) was deployed in the public subnet of the VPC. This host serves as a controlled access point for SSH into private instances, improving security by avoiding direct public exposure of internal systems.



Step 5: Secure SSH Access via Bastion Host

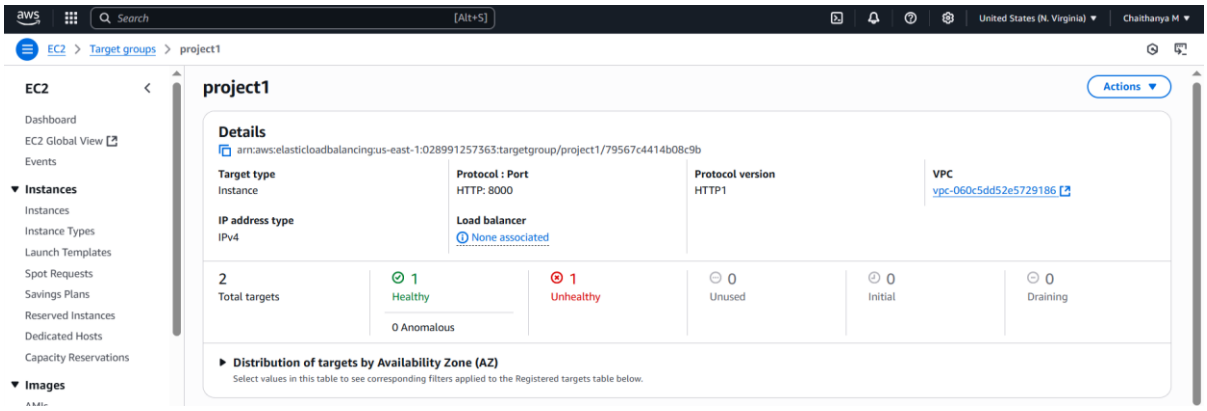
To maintain a secure environment, direct access to private subnet instances was disabled from the internet. Instead, a Bastion Host was used as an intermediary SSH jump box to access internal resources. The screenshot below demonstrates successful remote access to a private instance.

```
ubuntu@ip-10-0-2-181:~$ ls
new_key.pem
ubuntu@ip-10-0-2-181:~$ ssh -i new_key.pem ubuntu@10.0.137.155
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1029-aws x86_64)

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

 System information as of Thu Jun 19 10:07:07 UTC 2025

  System load:  0.0               Processes:             103
  Usage of /:   25.3% of 6.71GB   Users logged in:       0
  Memory usage: 20%               IPv4 address for enX0: 10.0.137.155
  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.
```

Step 6: Target Group Configuration

To support load balancing and health monitoring of backend EC2 instances, a Target Group named project1 was configured under the Elastic Load Balancing (ELB) service. This group enables distributing incoming traffic across multiple instances based on health status and configured port/protocol.

- **1 Healthy** – One instance passed the health check.
- **1 Unhealthy** – The other instance failed the health check.

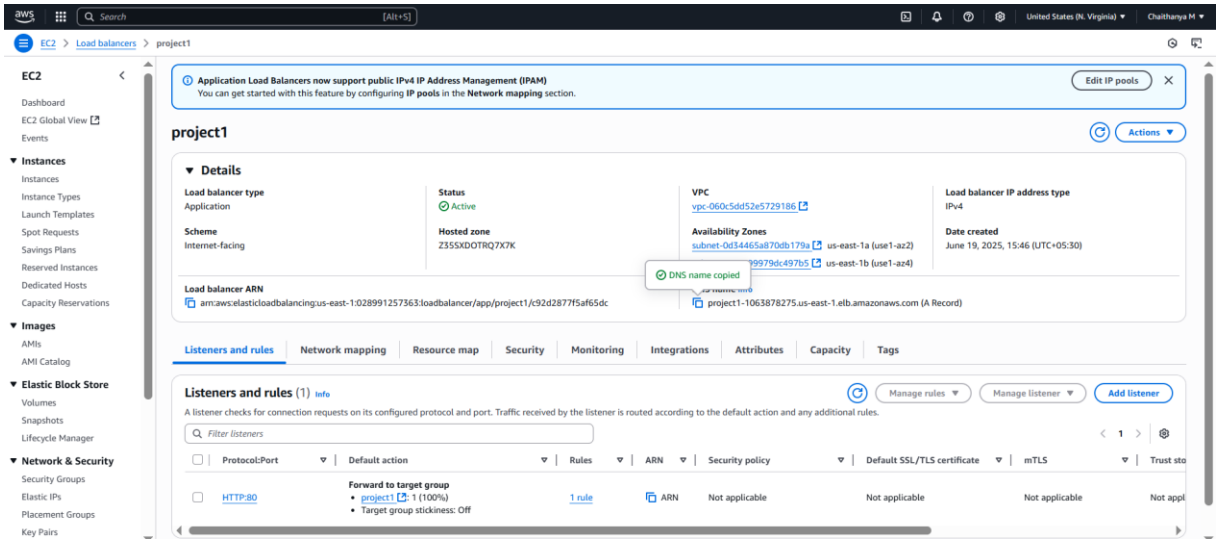


Step 7: Application Load Balancer Configuration

To provide high availability**,** fault tolerance**,** and traffic distribution across multiple EC2 instances, an Application Load Balancer (ALB) named project1 was created. This ALB is configured to route traffic to the previously defined target group (project1).

When you copy and paste the ALB DNS name http://project1-1063878275.us-east-1.elb.amazonaws.com into browser:

- The HTTP listener on port 80 receives the request.
- The ALB forwards the request to one of the healthy EC2 targets in Target Group project1 (on port 8000).
- If at least one target is healthy and has the web server or app running on port 8000, a webpage is displayed.
- If all targets are unhealthy or the app isn't properly set up, the ALB returns a 502 Bad Gateway or similar error.



project to understand public and private subnet in VPC

My first paragraph.

12. Conclusion

In this project, we successfully designed and implemented a secure and scalable cloud infrastructure using key AWS services. The use of Security Groups and Network ACLs (NACLs) provided layered security for both public and private subnets, ensuring fine-grained access control at both the instance and subnet levels.

A Bastion Host was configured in the public subnet to securely access instances in private subnets, eliminating direct internet exposure for sensitive workloads. Using Launch Templates, Auto Scaling Groups, and Target Groups, we deployed and managed EC2 instances efficiently with built-in fault tolerance and load distribution.

An Application Load Balancer (ALB) was used to route incoming traffic to healthy instances based on real-time health checks, ensuring high availability and zero downtime during failures. Health check results and listener configuration ensured that only healthy instances received traffic on the specified application port.

Through this setup, we demonstrated best practices in:

- **Cloud security**
- **Network segmentation**
- **High availability architecture**
- **Centralized access control**

This project not only highlights how to build secure VPC-based solutions on AWS but also forms the foundation for deploying production-grade, resilient applications in the cloud.