Playtime with NACL’s and Security Groups

A little brief on Main topics:

1. VPC (Virtual Private Cloud):

A **VPC** is a virtual network dedicated to your AWS account. It allows you to launch AWS resources in a virtual network that you define. This virtual network is logically isolated from other virtual networks in the AWS cloud. You have complete control over your VPC, including selecting your own IP address range, creating subnets, and configuring network gateways and security settings.

2. Security Groups:

Security Groups act as virtual firewalls to control incoming and outgoing traffic for your EC2 instances and other resources. They are stateful, meaning they track the origin of incoming traffic and automatically allow return traffic. Security Groups are associated with instances and can be used to define rules for allowing or blocking traffic based on protocols, ports, and source IP addresses.

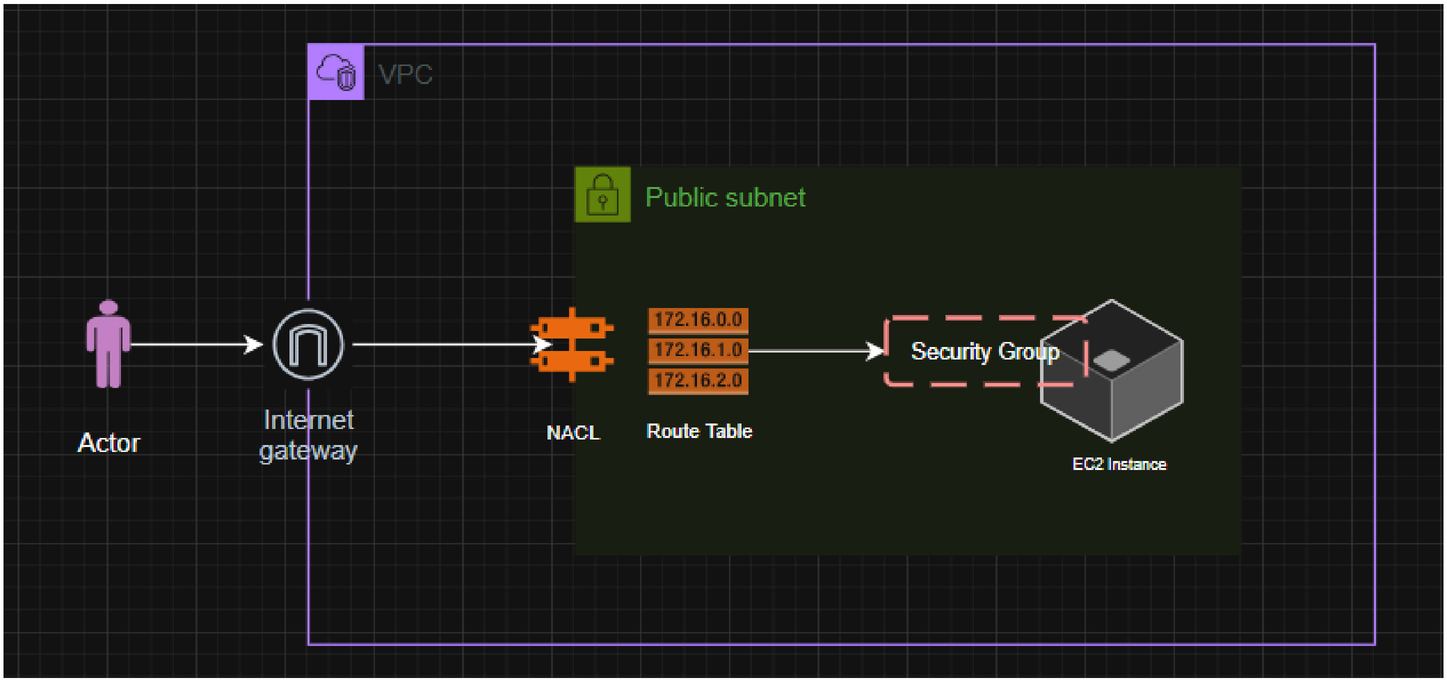
3. Network ACLs:

NACLs are an additional layer of security that controls traffic at the subnet level. Unlike Security Groups, NACLs are stateless, meaning they do not track the origin of the traffic. Therefore, you must explicitly define rules for both incoming and outgoing traffic. NACLs provide a way to enforce network traffic rules across entire subnets, offering a more granular level of control compared to Security Groups.

Lets get into the practical implementation:

I created a project just to understand how the SG’s and NACL’s work in real-time.

For this project, I want to create an EC2-instance based on Ubuntu and host a simple python server. For this instance I want to create a new Security group with just allowing SSH port by default and then only allow inbound traffic to port 8001.



Implementation:

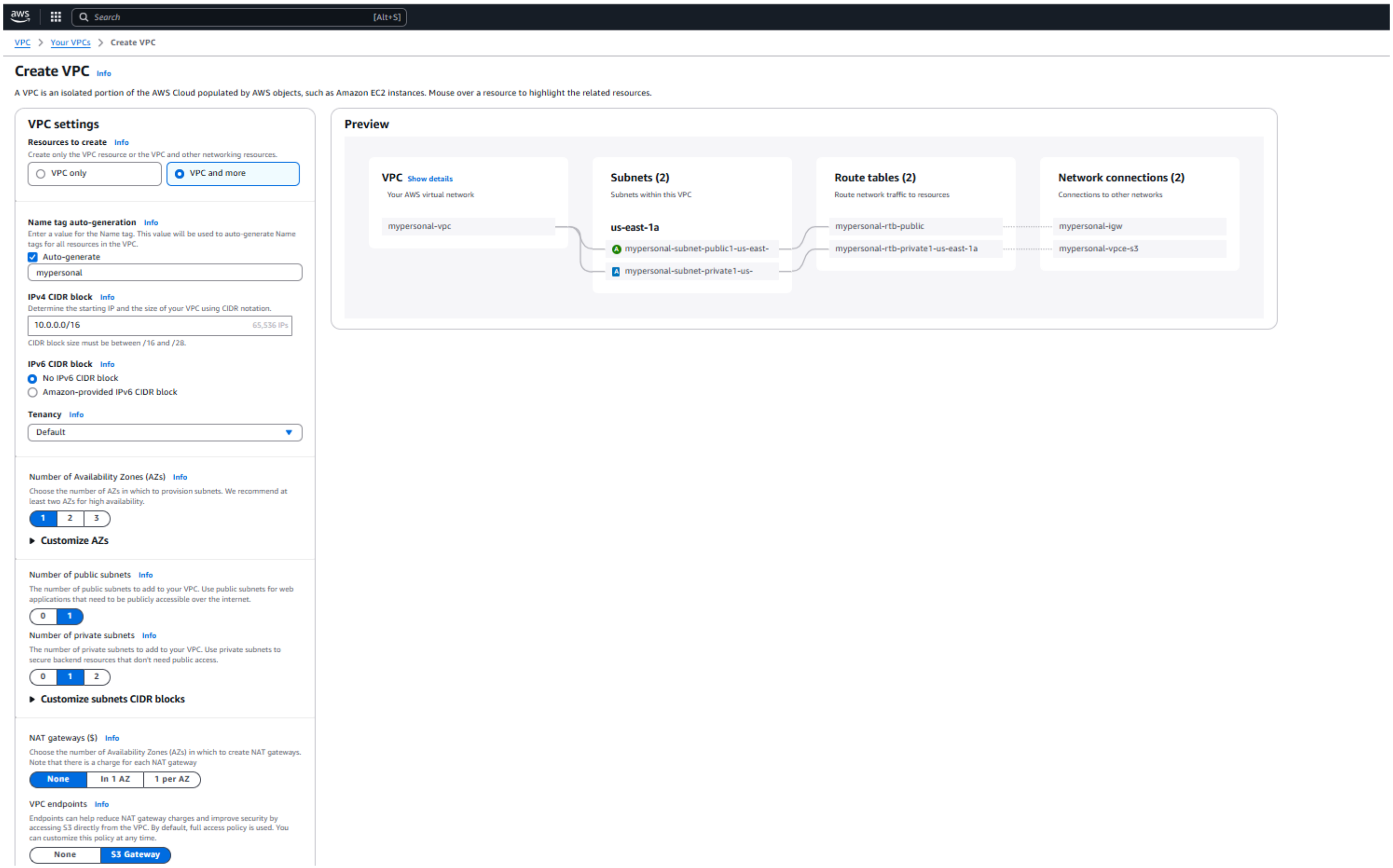
1. Create a VPC:

Lets create a VPC, just so to have a complete control of what we are creating.

Steps:

i. Search VPC in the AWS Search window.
ii. Create VPC
iii. Select VPC and more
iv. Name: mypersonal
v. IPV4 CIDR: 10.0.0.0/16 which creates 65,536 Ips [we can also make it custom like /24 for just 256 Ips – totally up to the usecase.]
vi. I just wanted a single public and private subnets.
vii. Click create.

We can now check the preview, the VPC creates 2 subnets – 1public and 1private. Route tables for each subnet, Internet gateway for the public subnet and VPC endpoint for the private subnet.



VPC > Your VPCs > Create VPC

Create VPC Info

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances. Mouse over a resource to highlight the related resources.

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 auto-generation Info
Enter a value for the Name tag. This value will be used to auto-generate Name tags for all resources in the VPC.
☑ Auto-generate
mypersonal

IPv4 CIDR block Info
Determine the starting IP and the size of your VPC using CIDR notation.
10.0.0.0/16 65,536 IPs
CIDR block size must be between /16 and /28.

IPv6 CIDR block Info
● No IPv6 CIDR block
○ Amazon-provided IPv6 CIDR block

Tenancy Info
Default ▼

Number of Availability Zones (AZs) Info
Choose the number of AZs in which to provision subnets. We recommend at least two AZs for high availability.
[✓ 1] 2 3

▶ Customize AZs

Number of public subnets Info
The number of public subnets to add to your VPC. Use public subnets for web applications that need to be publicly accessible over the internet.
0 [✓ 1]

Number of private subnets Info
The number of private subnets to add to your VPC. Use private subnets to secure backend resources that don't need public access.
0 [✓ 1] 2

▶ Customize subnets CIDR blocks

NAT gateways ($) Info
Choose the number of Availability Zones (AZs) in which to create NAT gateways. Note that there is a charge for each NAT gateway
[✓ None] In 1 AZ 1 per AZ

VPC endpoints Info
Endpoints can help reduce NAT gateway charges and improve security by accessing S3 directly from the VPC. By default, full access policy is used. You can customize this policy at any time.
None [✓ S3 Gateway]

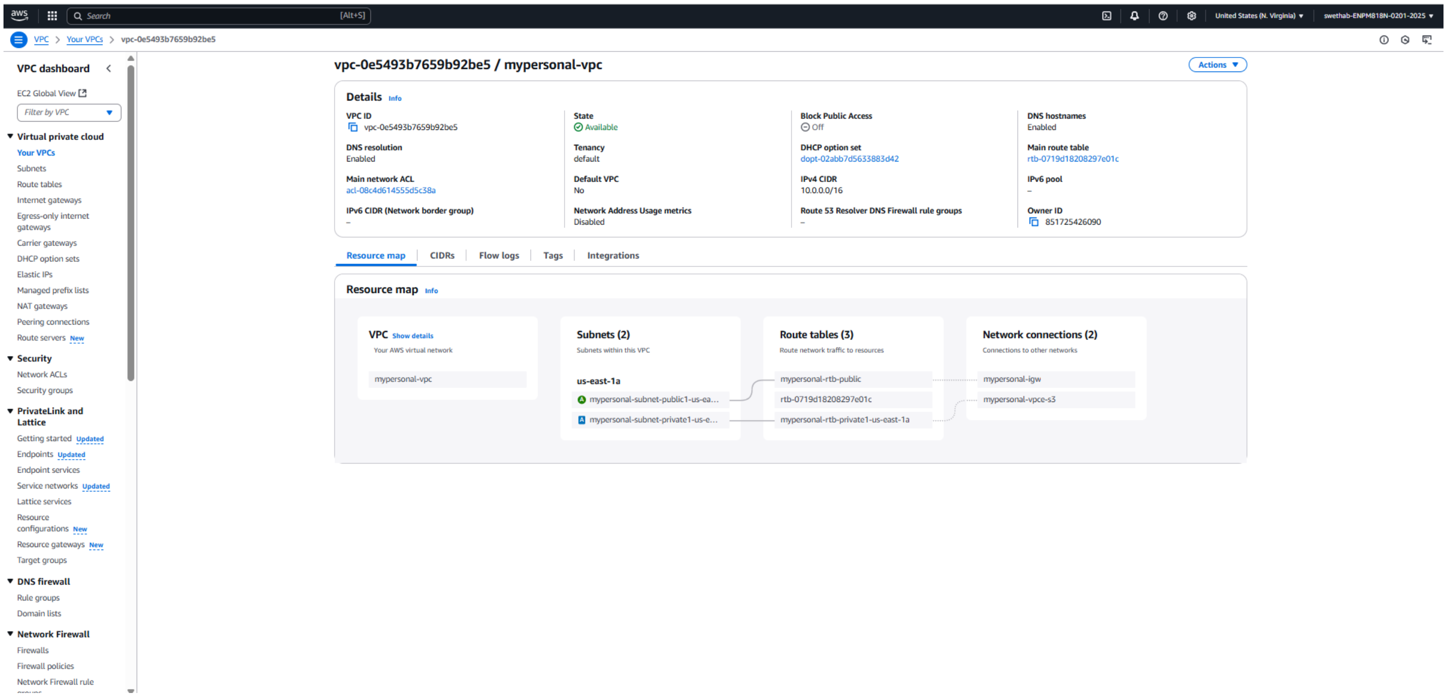
Preview

VPC Show details
Your AWS virtual network
mypersonal-vpc

Subnets (2)
Subnets within this VPC
us-east-1a
mypersonal-subnet-public1-us-east-
mypersonal-subnet-private1-us-

Route tables (2)
Route network traffic to resources
mypersonal-rtb-public
mypersonal-rtb-private1-us-east-1a

Network connections (2)
Connections to other networks
mypersonal-igw
mypersonal-vpce-s3



vpc-0e5493b7659b92be5 / mypersonal-vpc

Details

VPC ID: vpc-0e5493b7659b92be5
State: Available
Block Public Access: Off
DNS hostnames: Enabled
DNS resolution: Enabled
Tenancy: default
DHCP option set: dopt-02abb7d5633883d42
Main route table: rtb-0719d18208297e01c
Main network ACL: acl-08c4d614555d5c38a
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: 851725426090

Resource map

VPC (Your AWS virtual network): mypersonal-vpc

Subnets (2) (Subnets within this VPC): us-east-1a: mypersonal-subnet-public1-us-ea…, mypersonal-subnet-private1-us-e…

Route tables (3) (Route network traffic to resources): mypersonal-rtb-public, rtb-0719d18208297e01c, mypersonal-rtb-private1-us-east-1a

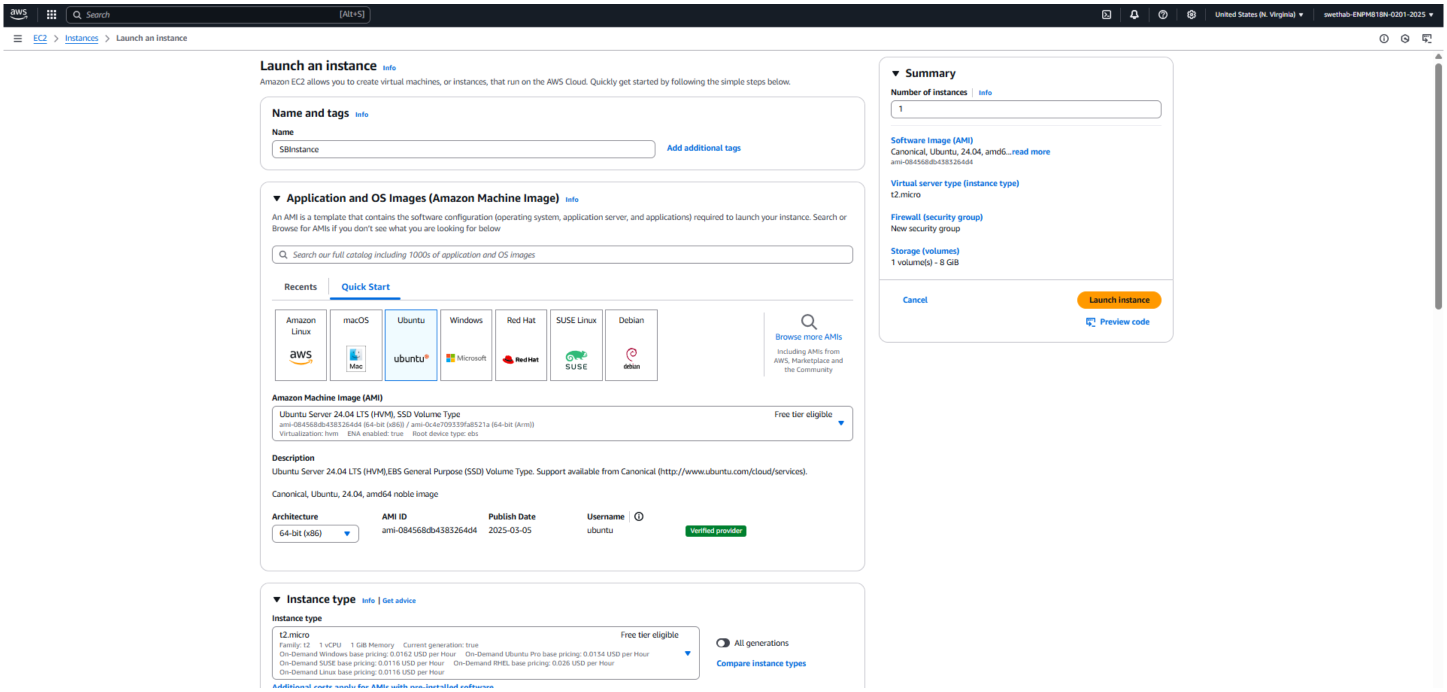
Network connections (2) (Connections to other networks): mypersonal-igw, mypersonal-vpce-s3

Once the VPC is created, lets proceed to create an EC2 instance.

2. Create EC2 instance:

Steps:

i. Name: SBInstance
ii. AMI: Ubuntu
iii. Instance type: t2.micro
iv. Network settings: Select the VPC we just created (mypersonal)
v. By default, the VPC will select the private subnet which is a good practice- in real world we would want our instance to be in the private subnet itself, But for this project lets just keep our instance in the **public instance**.
vi. Enable Assign Public IP.
vii. Create.



Launch an instance

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

Name: SBInstance

Application and OS Images (Amazon Machine Image)

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

Amazon Machine Image (AMI): Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-084568db4383264d4 (64-bit (x86)) / ami-0c4e709339fa8521a (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description: Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Canonical, Ubuntu, 24.04, amd64 noble image

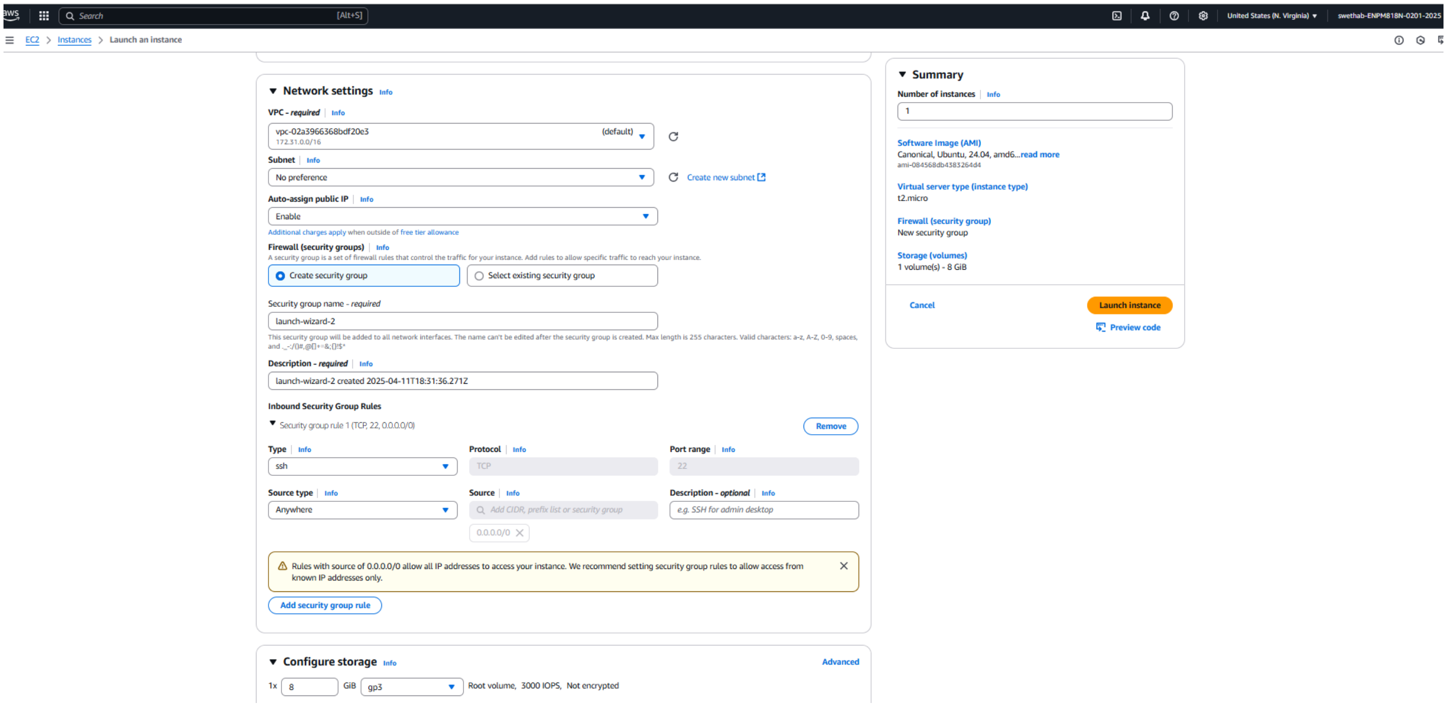
Architecture: 64-bit (x86)
AMI ID: ami-084568db4383264d4
Publish Date: 2025-03-05
Username: ubuntu
Verified provider

Instance type

Instance type: t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand Windows base pricing: 0.0162 USD per Hour On-Demand Ubuntu Pro base pricing: 0.0134 USD per Hour
On-Demand SUSE base pricing: 0.0116 USD per Hour On-Demand RHEL base pricing: 0.026 USD per Hour
On-Demand Linux base pricing: 0.0116 USD per Hour
All generations

Summary

Number of instances: 1
Software Image (AMI): Canonical, Ubuntu, 24.04, amd6...read more
ami-084568db4383264d4
Virtual server type (instance type): t2.micro
Firewall (security group): New security group
Storage (volumes): 1 volume(s) - 8 GiB


aws
Search [Alt+S]
United States (N. Virginia) ▾
swethab-ENPM818N-0201-2025
EC2 > Instances > Launch an instance

▼ Network settings Info

VPC - *required* | Info
vpc-02a3966368bdf20e3 (default)
172.31.0.0/16

Subnet | Info
No preference
Create new subnet

Auto-assign public IP | Info
Enable
Additional charges apply when outside of free tier allowance

Firewall (security groups) | Info
A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.
● Create security group ○ Select existing security group

Security group name - *required*
launch-wizard-2
This security group will be added to all network interfaces. The name can't be edited after the security group is created. Max length is 255 characters. Valid characters: a-z, A-Z, 0-9, spaces, and ._-:/()#,@[]+=&;{}!$*

Description - *required* | Info
launch-wizard-2 created 2025-04-11T18:31:36.271Z

Inbound Security Group Rules
▼ Security group rule 1 (TCP, 22, 0.0.0.0/0) Remove

Type | Info: ssh
Protocol | Info: TCP
Port range | Info: 22
Source type | Info: Anywhere
Source | Info: Add CIDR, prefix list or security group; 0.0.0.0/0
Description - *optional* | Info: e.g. SSH for admin desktop

⚠ Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.

Add security group rule

▼ Configure storage Info Advanced
1x 8 GiB gp3 Root volume, 3000 IOPS, Not encrypted

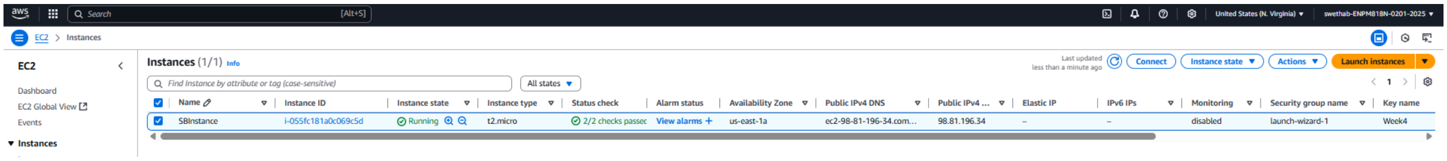
▼ Summary
Number of instances | Info
1

Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-084568db4383264d4

Virtual server type (instance type)
t2.micro

Firewall (security group)
New security group

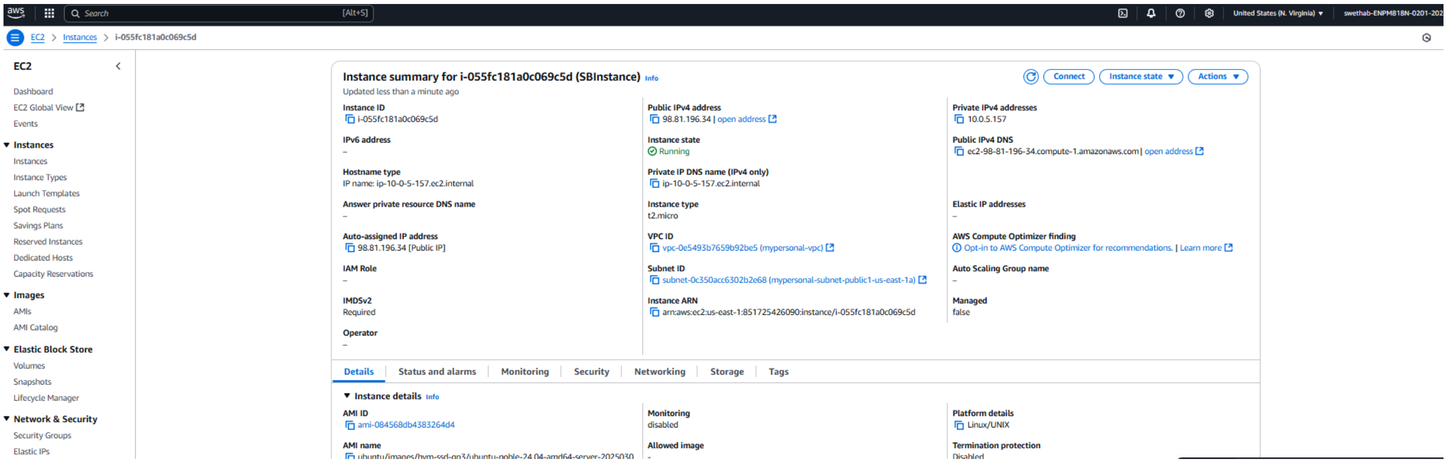
Storage (volumes)
1 volume(s) - 8 GiB



Instances (1/1) Info

Find Instance by attribute or tag (case-sensitive)

☑	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs	Monitoring	Security group name	Key name
☑	SBInstance	i-055fc181a0c069c5d	✓ Running	t2.micro	✓ 2/2 checks passec	View alarms +	us-east-1a	ec2-98-81-196-34.com...	98.81.196.34	–	–	disabled	launch-wizard-1	Week4



Instance summary for i-055fc181a0c069c5d (SBInstance) Info
Updated less than a minute ago

Instance ID
i-055fc181a0c069c5d

IPv6 address
–

Hostname type
IP name: ip-10-0-5-157.ec2.internal

Answer private resource DNS name
–

Auto-assigned IP address
98.81.196.34 [Public IP]

IAM Role
–

IMDSv2
Required

Operator
–

Public IPv4 address
98.81.196.34 | open address

Instance state
✓ Running

Private IP DNS name (IPv4 only)
ip-10-0-5-157.ec2.internal

Instance type
t2.micro

VPC ID
vpc-0e5493b7659b92be5 (mypersonal-vpc)

Subnet ID
subnet-0c350acc6302b2e68 (mypersonal-subnet-public1-us-east-1a)

Instance ARN
arn:aws:ec2:us-east-1:851725426090:instance/i-055fc181a0c069c5d

Private IPv4 addresses
10.0.5.157

Public IPv4 DNS
ec2-98-81-196-34.compute-1.amazonaws.com | open address

Elastic IP addresses
–

AWS Compute Optimizer finding
ⓘ Opt-in to AWS Compute Optimizer for recommendations. | Learn more

Auto Scaling Group name
–

Managed
false

▼ Instance details Info

AMI ID
ami-084568db4383264d4

AMI name
ubuntu/images/hvm-ssd-gp3/ubuntu-noble-24.04-amd64-server-2025030

Monitoring
disabled

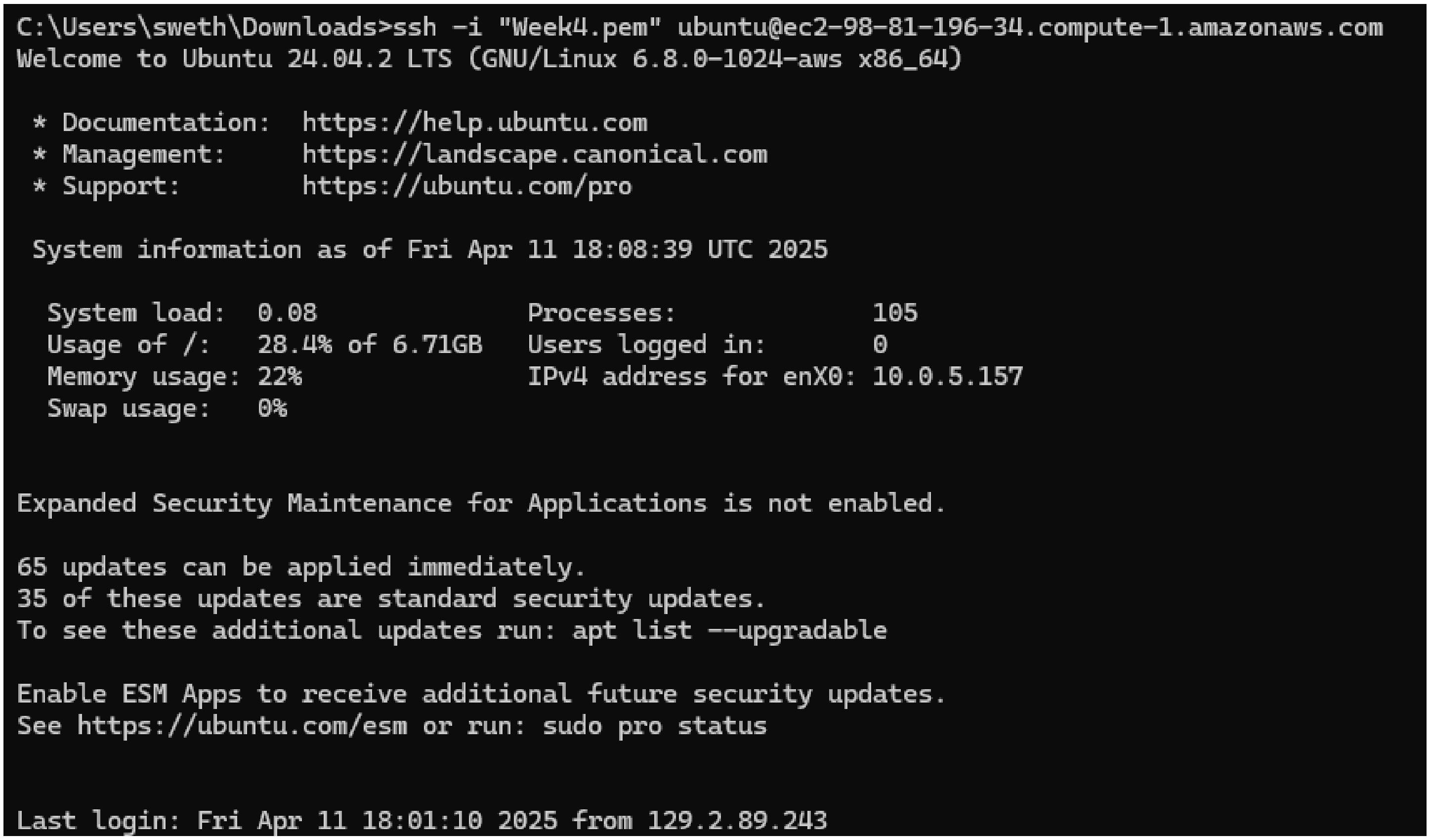
Allowed image
-

Platform details
Linux/UNIX

Termination protection
Disabled

Now, once the instance is created, Lets connect to the instance using SSH.

```
ssh -i "Week4.pem" ubuntu@ec2-98-81-196-34.compute-1.amazonaws.com
```



```
C:\Users\sweth\Downloads>ssh -i "Week4.pem" ubuntu@ec2-98-81-196-34.compute-1.amazonaws.com
Welcome to Ubuntu 24.04.2 LTS (GNU/Linux 6.8.0-1024-aws x86_64)

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

 System information as of Fri Apr 11 18:08:39 UTC 2025

  System load:  0.08               Processes:             105
  Usage of /:   28.4% of 6.71GB    Users logged in:       0
  Memory usage: 22%                IPv4 address for enX0: 10.0.5.157
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

65 updates can be applied immediately.
35 of these updates are standard security updates.
To see these additional updates run: apt list --upgradable

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


Last login: Fri Apr 11 18:01:10 2025 from 129.2.89.243
```

Update the packages and ensure that python3 is installed;

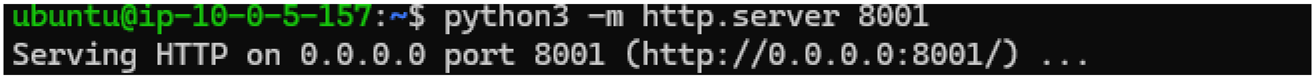
Sudo apt update -y

Check the version of python3.

Python3 –version

Lets begin the python3 server on port 8001:

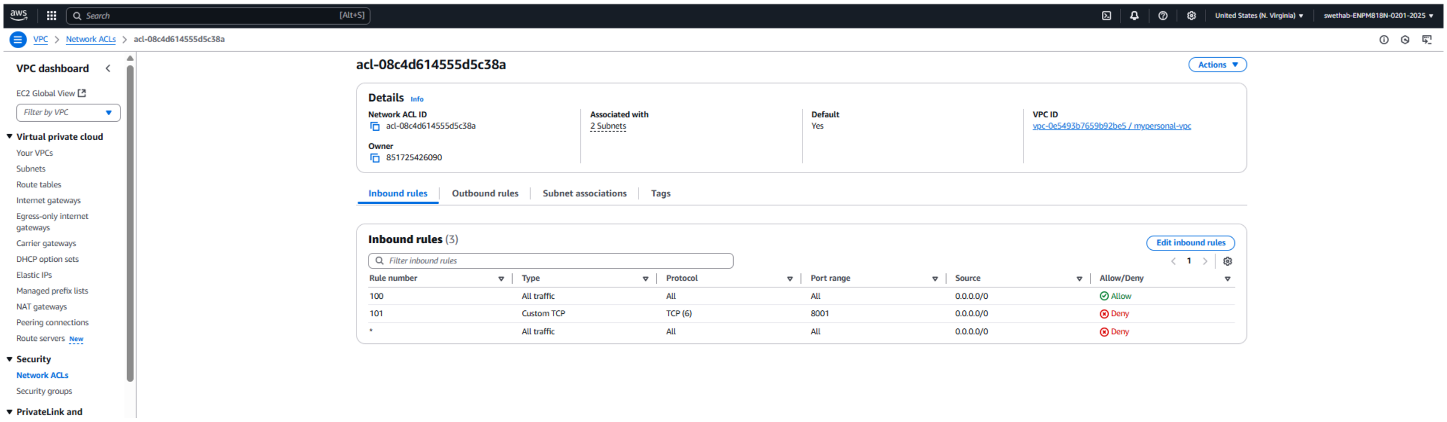
Python3 -m http.server 8001



```
ubuntu@ip-10-0-5-157:~$ python3 -m http.server 8001
Serving HTTP on 0.0.0.0 port 8001 (http://0.0.0.0:8001/) ...
```

By default, if we try to access the IP of the instance, it says bad request

To change this lets play with NACL’s and SG’s



acl-08c4d614555d5c38a

Details

Network ACL ID: acl-08c4d614555d5c38a

Associated with: 2 Subnets

Default: Yes

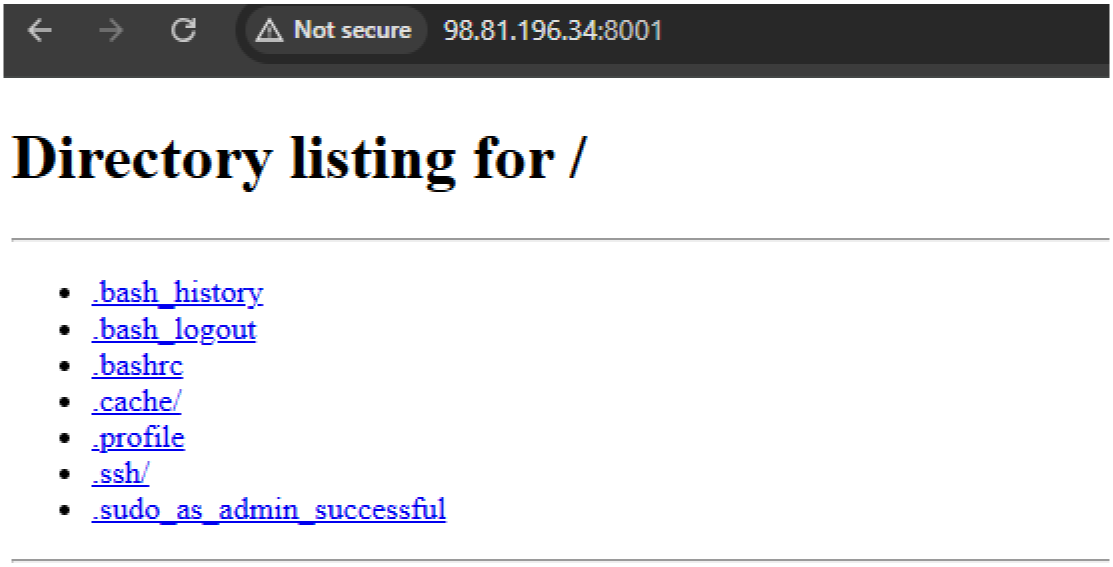
VPC ID: vpc-0e5493b7659b92be5 / mypersonal-vpc

Owner: 851725426090

Inbound rules (3)

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	All traffic	All	All	0.0.0.0/0	Allow
101	Custom TCP	TCP (6)	8001	0.0.0.0/0	Deny
*	All traffic	All	All	0.0.0.0/0	Deny

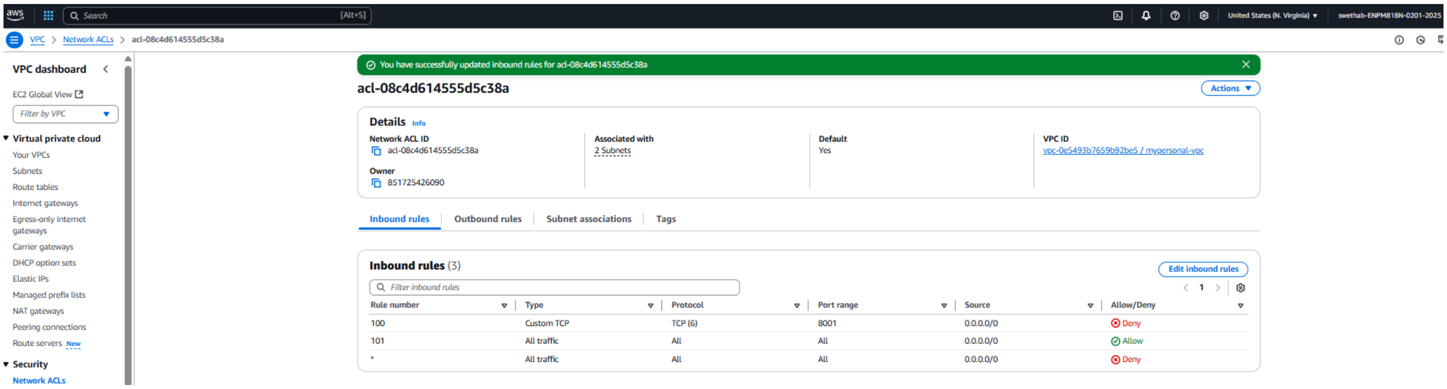
I edited the Inbound rules of NACL and if we look at it correctly, I have enabled all the inbound traffic and then denied traffic from 8001, and tried to access the server,



Directory listing for /

- .bash_history
- .bash_logout
- .bashrc
- .cache/
- .profile
- .ssh/
- .sudo_as_admin_successful

Now, I tweaked the rules and tried to access the server.



You have successfully updated inbound rules for acl-08c4d614555d5c38a

acl-08c4d614555d5c38a

Details Info

Network ACL ID: acl-08c4d614555d5c38a

Associated with: 2 Subnets

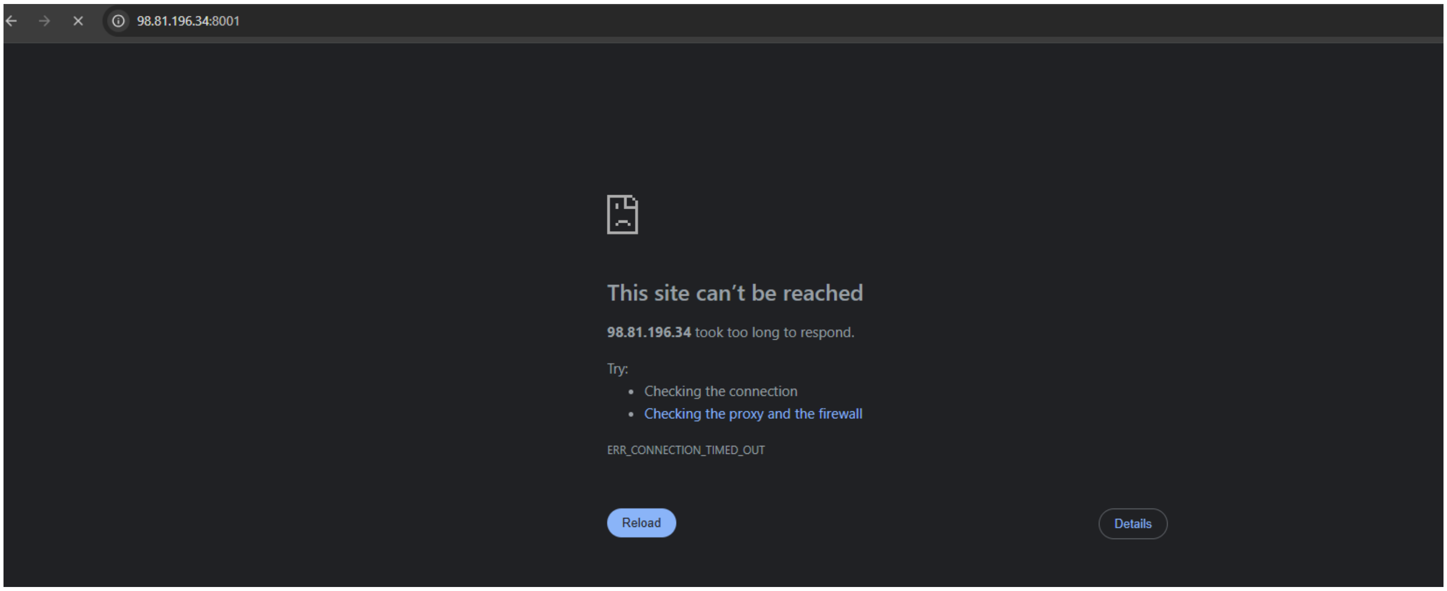
Default: Yes

VPC ID: vpc-0e5493b7659b92be5 / mypersonal-vpc

Owner: 851725426090

Inbound rules (3)

Rule number	Type	Protocol	Port range	Source	Allow/Deny
100	Custom TCP	TCP (6)	8001	0.0.0.0/0	⊗ Deny
101	All traffic	All	All	0.0.0.0/0	✓ Allow
*	All traffic	All	All	0.0.0.0/0	⊗ Deny



This site can't be reached

98.81.196.34 took too long to respond.

Try:

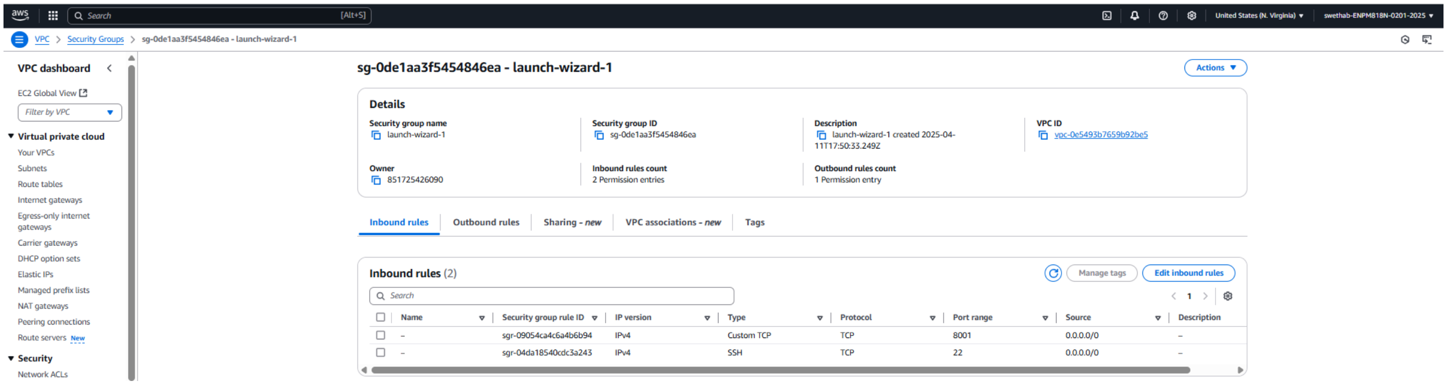
- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_TIMED_OUT

The server is taking time to respond.

The reason is that the NACL inbound rules are executed based on the order. So, in the first case, all the traffic is allowed and then port 8001 is denied since the first rule says all the traffic is allowed, meaning – even if the second rule says to deny traffic from 8001 is overruled.

Since we changed the order now, port 8001 has been denied and hence we cannot access the Python server.



sg-0de1aa3f5454846ea - launch-wizard-1

Details

Security group name: launch-wizard-1

Security group ID: sg-0de1aa3f5454846ea

Description: launch-wizard-1 created 2025-04-11T17:50:33.249Z

VPC ID: vpc-0e5493b7659b92be5

Owner: 851725426090

Inbound rules count: 2 Permission entries

Outbound rules count: 1 Permission entry

Inbound rules (2)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
–	sgr-09054ca4c6a4b6b94	IPv4	Custom TCP	TCP	8001	0.0.0.0/0	–
–	sgr-04da18540cdc3a243	IPv4	SSH	TCP	22	0.0.0.0/0	–

Same is the case with the Security groups as well.

It’s good practice for you to play with SG’s and NACL’s to get a better understanding of each of them.

Key Differences and Use Cases

- **Security Groups** are ideal for controlling traffic to specific instances or groups of instances. They are stateful, which simplifies rule management for bidirectional traffic.
- **NACLs** are best used for enforcing network policies at the subnet level, providing a more comprehensive security layer for entire segments of your network.

As a good practice, please delete all the used Instances and VPC’s to avoid costs.

Hope you have understood it clearly.