

# VPC Traffic Flow and Security

---

## Project Overview

This project demonstrates advanced AWS networking by configuring traffic flow and implementing multi-layered security controls in a Virtual Private Cloud. I successfully set up route tables to direct internet-bound traffic, created security groups for resource-level protection, and deployed Network ACLs for subnet-level security - building a production-ready network infrastructure with defense in depth.

**Project Duration:** Approximately 90 minutes

**Difficulty Level:** Easy

**AWS Region Used:** eu-west-3 (Paris)

---

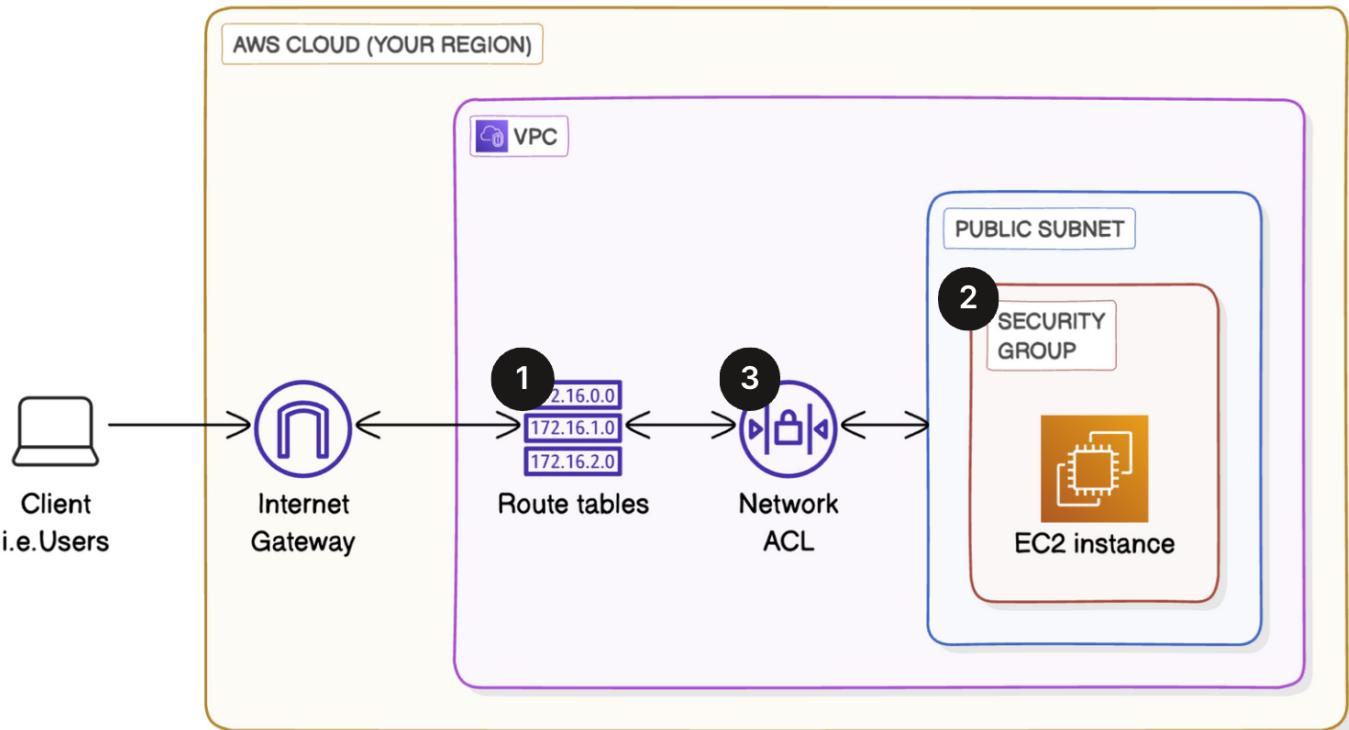
## Table of Contents

- [What I Built](#)
  - [Technologies & Concepts](#)
  - [Step-by-Step Implementation](#)
    - [Step 1: Create VPC and Subnet](#)
    - [Step 2: Create Route Tables](#)
    - [Step 3: Create Security Group](#)
    - [Step 4: Create Network ACL](#)
  - [Understanding Traffic Flow](#)
  - [Cleanup](#)
  - [Conclusion](#)
- 

## What I Built

A complete VPC networking infrastructure with layered security:

- **VPC:** NextWork VPC (10.0.0.0/16) - Private network space
- **Public Subnet:** Public 1 (10.0.1.0/24) - For internet-accessible resources
- **Internet Gateway:** NextWork IG - Connection to the internet
- **Custom Route Table:** Routes internet traffic (0.0.0.0/0) to Internet Gateway
- **Security Group:** NextWork Security Group - Resource-level firewall with HTTP access
- **Custom Network ACL:** NextWork Network ACL - Subnet-level firewall allowing all traffic



This architecture demonstrates the three critical layers of AWS networking:

- Connectivity Layer:** Route tables directing traffic flow
- Resource Security Layer:** Security groups protecting individual resources
- Subnet Security Layer:** Network ACLs guarding entire subnets

## Technologies & Concepts

### AWS Services Used

- **Amazon VPC** - Virtual Private Cloud for isolated networking
- **Subnets** - Network subdivisions within the VPC
- **Internet Gateway** - Enables internet connectivity
- **Route Tables** - Define traffic routing rules
- **Security Groups** - Stateful firewalls at the resource level
- **Network ACLs** - Stateless firewalls at the subnet level

### Key Networking Concepts Learned

#### 1. Route Tables

- GPS systems for network traffic
- Define where packets should go based on destination
- Each subnet must be associated with exactly one route table
- Default route (0.0.0.0/0) sends all internet traffic to Internet Gateway

#### 2. Security Groups

- Virtual firewalls for individual resources (EC2 instances)
- **Stateful:** Return traffic is automatically allowed

- Applied at the resource level
- Rules specify: Protocol, Port, and Source/Destination
- By default, allow all outbound traffic

### 3. Network ACLs

- Traffic cops at subnet entry/exit points
- **Stateless:** Must explicitly allow both inbound and outbound traffic
- Applied at the subnet level
- Rules are evaluated in numbered order (100, 200, etc.)
- Asterisk (\*) rule acts as catch-all deny

### 4. Defense in Depth

- Multiple security layers protect resources
- Traffic must pass through: Network ACL → Subnet → Security Group → Resource
- If any layer blocks traffic, the request fails
- Provides redundancy and granular control

### 5. Public vs Private Subnets

- **Public subnet requirements:**
  - Route to Internet Gateway (0.0.0.0/0 → IGW)
  - Resources have public IP addresses
  - Network ACL allows traffic
- **Private subnets:** No route to Internet Gateway

## Step-by-Step Implementation

### Step 1: Create VPC and Subnet

#### What I did:

I started by creating the foundational networking components - a VPC and a public subnet.

#### Create the VPC

#### VPC Configuration:

##### Your VPCs

[VPCs](#) | [VPC encryption controls](#)

Your VPCs (2) [Info](#)

Your VPCs							Last updated	<a href="#">Actions</a>	<a href="#">Create VPC</a>
							5 minutes ago		
Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...	IPv4 CIDR			
-	vpc-0f420f6cd010b8079	Available	-	-	Off	172.31.0.0/16	<	1	>   <a href="#">⚙️</a>
NextWork VPC	vpc-072dff9a0c503e946	Available	-	-	Off	10.0.0.0/16			

- **Name:** NextWork VPC
- **VPC ID:** vpc-072dff9a0c503e946
- **IPv4 CIDR:** 10.0.0.0/16

- **State:** Available

### Why this matters:

The VPC is the container for all networking resources. The 10.0.0.0/16 CIDR block provides 65,536 IP addresses, giving plenty of room for growth.

### Create the Public Subnet

#### Subnet Configuration:

You have successfully created 1 subnet: subnet-02cbe07bc0fdcc970

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
Public 1	<a href="#">subnet-02cbe07bc0fdcc970</a>	Available	<a href="#">vpc-072dff9a0c503e946</a>   Next...	Off	10.0.1.0/24

- **Name:** Public 1
- **Subnet ID:** subnet-02cbe07bc0fdcc970
- **VPC:** vpc-072dff9a0c503e946 (NextWork VPC)
- **Availability Zone:** eu-west-3a
- **IPv4 CIDR:** 10.0.1.0/24
- **State:** Available
- **Available IP addresses:** 251 (out of 256 total)

#### Key Point:

Even though it's named "Public 1", this subnet isn't actually public yet! It needs a route to an Internet Gateway to become truly public.

### Step 2: Create Route Tables

#### What I did:

I configured route tables to direct traffic from my subnet to the internet via the Internet Gateway.

#### Understanding Route Tables

**Route tables are like GPS systems** - they tell network traffic where to go based on the destination IP address.

Name	Route table ID	Explicit subnet associ...	Edge associations	Main	VPC	Ow
-	<a href="#">rtb-0681ccd4912886688</a>	-	-	Yes	<a href="#">vpc-0f420f6cd010b8079</a>	05€
-	<a href="#">rtb-0eee6b8856347b8bf</a>	-	-	Yes	<a href="#">vpc-072dff9a0c503e946</a>   Next...	05€

I can see two route tables in my account:

- **rtb-0681ccd4912886688**: Default VPC's main route table
- **rtb-0eee6b8856347b8bf**: NextWork VPC's main route table (this is the one I'm working with)

## Examining the Default Route Table

### Default VPC Route Table:

The screenshot shows the AWS Route Tables page. It lists two route tables:

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
-	rtb-0681ccd4912886688	-	-	Yes	vpc-0f420f6cd010b8079
-	rtb-0eee6b8856347b8bf	-	-	Yes	vpc-072dff9a0c503e946   Next...

Details for rtb-0681ccd4912886688:

**Routes (2)**

Destination	Target	Status	Propagated	Route Origin
0.0.0.0/0	igw-0cdbe7fb9a117405b	Active	No	Create Route
172.31.0.0/16	local	Active	No	Create Route Table

The default VPC already has internet connectivity configured:

- **Route 1**: 0.0.0.0/0 → igw-0cdbe7fb9a117405b (Internet Gateway)
- **Route 2**: 172.31.0.0/16 → local (internal VPC traffic)

### What this means:

- Any traffic destined for the internet (0.0.0.0/0) goes to the Internet Gateway
- Traffic within the VPC (172.31.0.0/16) stays local

## Examining NextWork VPC's Route Table

### NextWork VPC Route Table (Before):

The screenshot shows the AWS Route Tables page for the NextWork VPC. It lists one route table:

Name	Route table ID	Explicit subnet associations	Edge associations	Main	VPC
-	rtb-0681ccd4912886688	-	-	Yes	vpc-0f420f6cd010b8079
<b>rtb-0eee6b8856347b8bf</b>	<b>rtb-0eee6b8856347b8bf</b>	<b>-</b>	<b>-</b>	<b>Yes</b>	<b>vpc-072dff9a0c503e946</b>

Details for rtb-0eee6b8856347b8bf:

**Routes (1)**

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	Create Route Table

A context menu is open over the route table, showing options: View details, Set main route table, Edit subnet associations, Edit edge associations, Edit route propagation, Edit routes, Manage tags, and Delete route table.

Initially, my route table only has one route:

- **Route:** 10.0.0.0/16 → local

**Problem:** There's no route to the Internet Gateway, so traffic can't reach the internet!

## Adding the Internet Gateway Route

### Creating the Internet Gateway:

First, I created and attached an Internet Gateway to my VPC:

Name	Internet gateway ID	State	VPC ID
-	igw-0cdbe7fb9a117405b	Attached	vpc-0f420f6cd010b8079
NextWork IG	igw-05408184cd4328c04	Detached	-

- **Name:** NextWork IG
- **Internet Gateway ID:** igw-05408184cd4328c04
- **State:** Attached
- **VPC:** vpc-072dff9a0c503e946

### Attach to VPC (igw-05408184cd4328c04)

VPC  
Attach an internet gateway to a VPC to enable the VPC to communicate with the internet. Specify the VPC to attach below.

Available VPCs  
Attach the internet gateway to this VPC.

vpc-072dff9a0c503e946

AWS Command Line Interface command

Cancel Attach internet gateway

## Editing Routes:

### Edit routes

Destination	Target	Status	Propagated	Route Origin
10.0.0.0/16	local	Active	No	CreateRouteTable
0.0.0.0/0	Internet Gateway igw-05408184cd4328c04	-	No	CreateRoute

Cancel Preview Save changes

I added a new route to direct internet-bound traffic to the Internet Gateway:

- **Destination:** 0.0.0.0/0 (all IP addresses)
- **Target:** Internet Gateway (igw-05408184cd4328c04)

## Why 0.0.0.0/0?

This is called a "default route" - it matches ANY IP address that doesn't match a more specific route. When traffic doesn't match the local route (10.0.0.0/16), it takes this route to the internet.

## Route Evaluation:

Routes are evaluated from most specific to least specific:

1. First, check if destination is in 10.0.0.0/16 → send to local
2. If not, check 0.0.0.0/0 → send to Internet Gateway

## Associate Route Table with Subnet

### Edit Subnet Associations:

#### Edit subnet associations

Change which subnets are associated with this route table.

Available subnets (1/1)					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR	Route table ID	
<input checked="" type="checkbox"/> Public 1	subnet-02cbe07bc0fdcc970	10.0.1.0/24	-	Main (rtb-0eee6b8856347b8bf)	

Selected subnets	
subnet-02cbe07bc0fdcc970 / Public 1	

[Cancel](#) [Save associations](#)

I associated my Public 1 subnet with the route table:

- **Selected subnet:** subnet-02cbe07bc0fdcc970 / Public 1
- **Route table:** Main (rtb-0eee6b8856347b8bf)

### Result:

Details <a href="#">Info</a>		Main	Explicit subnet associations	Edge associations
Route table ID	rtb-0eee6b8856347b8bf	<input checked="" type="checkbox"/> Yes	subnet-02cbe07bc0fdcc970 / Public 1	-
VPC	vpc-072dff9a0c503e946   NextWork VPC	<input checked="" type="checkbox"/> Owner ID	056481036163	

[Routes](#) [Subnet associations](#) [Edge associations](#) [Route propagation](#) [Tags](#)

Explicit subnet associations (1)					
<a href="#">Edit subnet associations</a>					
Find subnet association					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR		
Public 1	subnet-02cbe07bc0fdcc970	10.0.1.0/24	-		

Subnets without explicit associations (0)					
<a href="#">Edit subnet associations</a>					
The following subnets have not been explicitly associated with any route tables and are therefore associated with the main route table:					
Find subnet association					
Name	Subnet ID	IPv4 CIDR	IPv6 CIDR		
No subnets without explicit associations					
All your subnets are associated with a route table.					

[Activate Windows](#)

The route table now shows:

- **Explicit subnet associations:** subnet-02cbe07bc0fdcc970 / Public 1
- **VPC:** vpc-072dff9a0c503e946 | NextWork VPC
- **Main:** Yes

**Success!** My subnet now has a path to the internet through the route table!

## Step 3: Create Security Group

### What I did:

I created a security group to control inbound and outbound traffic at the resource level (for EC2 instances).

## What Are Security Groups?

If VPCs are cities and subnets are neighborhoods, **security groups are security checkpoints at each building** (resource). They check who's coming in and going out.

### Key Characteristics:

- Work at the **resource level** (EC2 instances, databases, etc.)
- **Stateful**: If you allow inbound traffic, the response is automatically allowed
- By default, allow all outbound traffic
- Rules specify: Type, Protocol, Port, and Source/Destination

## Existing Security Groups

Security Groups (7) <a href="#">Info</a>		<a href="#">Actions</a>		<a href="#">Export security groups to CSV</a>	<a href="#">Create security group</a>	
<input type="checkbox"/>	Name	<input type="checkbox"/>	Security group ID	Security group name	VPC ID	Description
<input type="checkbox"/>	-	<a href="#">sg-09cb81ea7324312c7</a>		default	<a href="#">vpc-072dff9a0c503e946</a>	default VPC security group
<input type="checkbox"/>	-	<a href="#">sg-0c3b2d34f7dc9a33e</a>		launch-wizard-3	<a href="#">vpc-0f420f6cd010b8079</a>	launch-wizard-3 created 2025-12-18T20:31
<input type="checkbox"/>	-	<a href="#">sg-0d42e0f383e26665c</a>		demo-sg-load-balancer	<a href="#">vpc-0f420f6cd010b8079</a>	Allows HTTP into ALB
<input type="checkbox"/>	-	<a href="#">sg-0dee702a1d81ef340</a>		default	<a href="#">vpc-0f420f6cd010b8079</a>	default VPC security group
<input type="checkbox"/>	-	<a href="#">sg-008b0d34f1e244865</a>		launch-wizard-4	<a href="#">vpc-0f420f6cd010b8079</a>	launch-wizard-4 created 2025-12-18T20:40
<input type="checkbox"/>	-	<a href="#">sg-0e2d62cf0c3ca3bd0</a>		launch-wizard-1	<a href="#">vpc-0f420f6cd010b8079</a>	launch-wizard-1 created 2025-06-07T19:16
<input type="checkbox"/>	-	<a href="#">sg-0cda41e4b9b8daf79</a>		launch-wizard-2	<a href="#">vpc-0f420f6cd010b8079</a>	launch-wizard-2 created 2025-07-03T20:08

I can see several existing security groups:

- **sg-09cb81ea7324312c7**: default (for NextWork VPC)
- **sg-0c3b2d34f7dc9a33e**: launch-wizard-3
- **sg-0d42e0f383e26665c**: demo-sg-load-balancer
- **sg-0dee702a1d81ef340**: default (for default VPC)
- And more...

## Why are there existing security groups?

AWS automatically creates a default security group for every VPC. This allows internal communication between resources in the same VPC.

## Creating NextWork Security Group

### Basic Configuration:

**Basic details**

**Security group name** Info  
NextWork Security Group  
Name cannot be edited after creation.

**Description** Info  
A Security Group for NextWork VPC

**VPC Info**  
vpc-072dff9a0c503e946 (NextWork VPC)

**Inbound rules** Info

Type	Protocol	Port range	Source	Description - optional
HTTP	TCP	80	Anywhere... ▾	0.0.0.0/0 X

**Add rule**

**Outbound rules** Info

Type	Protocol	Port range	Destination	Description - optional
All traffic	All	All	Custom ▾	0.0.0.0/0 X

**Add rule**

**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.  
No tags associated with the resource.

**Add new tag**  
You can add up to 50 more tags

Activate Windows  
Cancel Create security group  
Go to Settings to activate automatically

- Security group name:** NextWork Security Group
- Security group ID:** sg-0616d4f236f97b58d
- Description:** A Security Group for NextWork VPC
- VPC ID:** vpc-072dff9a0c503e946
- Inbound rules count:** 1 Permission entry
- Outbound rules count:** 1 Permission entry

## Inbound Rules:

**sg-0616d4f236f97b58d - NextWork Security Group** Actions ▾

**Details**

Security group name NextWork Security Group	Security group ID sg-0616d4f236f97b58d	Description A Security Group for NextWork VPC	VPC ID vpc-072dff9a0c503e946
Owner 056481036163	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

**Inbound rules** | Outbound rules | Sharing | VPC associations | Tags

**Inbound rules (1)**

Name	Security group rule ID	IP version	Type	Protocol	Port range	Source
-	sgr-04355ce78f6ad0d89	IPv4	HTTP	TCP	80	0.0.0.0/0

I configured one inbound rule:

- Type:** HTTP
- Protocol:** TCP (automatically filled)
- Port:** 80 (automatically filled)
- Source:** Anywhere-IPv4 (0.0.0.0/0)

## ⚠ Yellow Warning Banner:

### sg-0616d4f236f97b58d - NextWork Security Group

[Actions ▾](#)

**Details**

Security group name NextWork Security Group	Security group ID sg-0616d4f236f97b58d	Description A Security Group for NextWork VPC	VPC ID <a href="#">vpc-072dff9a0c503e946</a>
Owner 056481036163	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

[Inbound rules](#) | [Outbound rules](#) | [Sharing](#) | [VPC associations](#) | [Tags](#)

**Inbound rules (1)**

Name	Security group rule ID	Type	Protocol	Port range	Source	
-	sgr-04355ce78f6ad0d89	IPv4	HTTP	TCP	80	0.0.0.0/0

AWS warns me that allowing 0.0.0.0/0 means ANY IP address can access my resource on port 80. This is necessary for public websites but risky for sensitive resources.

**Best practice:** For production, restrict access to known IP addresses when possible.

## Outbound Rules:

### sg-0616d4f236f97b58d - NextWork Security Group

[Actions ▾](#)

**Details**

Security group name NextWork Security Group	Security group ID sg-0616d4f236f97b58d	Description A Security Group for NextWork VPC	VPC ID <a href="#">vpc-072dff9a0c503e946</a>
Owner 056481036163	Inbound rules count 1 Permission entry	Outbound rules count 1 Permission entry	

[Inbound rules](#) | [Outbound rules](#) | [Sharing](#) | [VPC associations](#) | [Tags](#)

**Outbound rules (1)**

Name	Security group rule ID	Type	Protocol	Port range	Destination
-	sgr-040477f9da1c92932	IPv4	All traffic	All	0.0.0.0/0

The default outbound rule allows all traffic:

- **Type:** All traffic
- **Protocol:** All
- **Port:** All
- **Destination:** Custom (0.0.0.0/0)

## Why allow all outbound by default?

Resources need to communicate with various services (databases, APIs, software updates). AWS trusts that if you launched a resource, you want it to communicate outward.

## ⚠ Another Warning:

**sg-0616d4f236f97b58d - NextWork Security Group**

[Actions ▾](#)

<b>Details</b>	
Security group name <a href="#">NextWork Security Group</a>	Security group ID <a href="#">sg-0616d4f236f97b58d</a>
Description <a href="#">A Security Group for NextWork VPC</a>	VPC ID <a href="#">vpc-072dff9a0c503e946</a>
Owner <a href="#">056481036163</a>	Inbound rules count 1 Permission entry
	Outbound rules count 1 Permission entry

Inbound rules | **Outbound rules** | Sharing | VPC associations | Tags

**Outbound rules (1)**

[Manage tags](#) | [Edit outbound rules](#)

Name	Security group rule ID	IP version	Type	Protocol	Port range	Destination
-	sgr-040477f9da1c92932	IPv4	All traffic	All	All	0.0.0.0/0

AWS recommends being more restrictive with outbound rules for better security.

## Step 4: Create Network ACL

### What I did:

I created a Network ACL (NACL) to add an additional layer of security at the subnet level.

### What Are Network ACLs?

Think of **Network ACLs** as **traffic cops stationed at every entry and exit point of a subnet**, checking each data packet against a table of rules.

### Key Characteristics:

- Work at the **subnet level** (affect all resources in the subnet)
- **Stateless**: Must explicitly allow both inbound AND outbound traffic
- Rules are numbered and evaluated in order (100, 200, 300, etc.)
- Asterisk (\*) rule is the catch-all that denies everything not explicitly allowed

### Understanding Data Packets

#### What are data packets?

When you browse a website, your request is broken into tiny pieces called **data packets**. Each packet contains:

- Part of the data being sent
- Source IP address
- Destination IP address
- Protocol information

Network ACLs inspect these packets at the subnet boundary.

### Default Network ACL

Network ACLs (1/2) <a href="#">Info</a>						
<input type="text"/> Find Network ACLs by attribute or tag					<a href="#">Actions</a> <a href="#">Create network ACL</a>	
Name	Network ACL ID	Associated with	Default	VPC ID	Inbound rules count	
<input checked="" type="checkbox"/> -	acl-0623e4cc6dc629090	subnet-02cbe07bc0fdcc970 / Public 1	Yes	vpc-072dff9a0c503e946 / NextWork VPC	2 Inbound rules	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Details</a>
<input type="checkbox"/> -	acl-0afdc761399094d7	3 Subnets	Yes	vpc-0f420f6cd010b8079	2 Inbound rules	<a href="#">Edit</a> <a href="#">Delete</a> <a href="#">Details</a>

I can see two existing Network ACLs:

- **acl-0623e4cc6dc629090**: Default NACL for NextWork VPC (associated with Public 1)
- **acl-0afdc761399094d7**: Default NACL for default VPC (3 subnets)

### Why do default NACLS exist?

AWS creates a default NACL for every VPC that allows all inbound and outbound traffic. This way, your subnet works immediately without configuration.

### Examining Default NACL Rules

#### Inbound Rules:

acl-0623e4cc6dc629090										
<a href="#">Details</a>		<a href="#">Inbound rules</a>		<a href="#">Outbound rules</a>		<a href="#">Subnet associations</a>				
<b>Inbound rules (2)</b>										
<input type="text"/> Filter inbound rules										
Rule number	Type	Protocol	Port range	Source	Allow/Deny					
100	All traffic	All	All	0.0.0.0/0	<a href="#">Allow</a>	<a href="#">Edit inbound rules</a>				
*	All traffic	All	All	0.0.0.0/0	<a href="#">Deny</a>	<a href="#">Activate Windows</a> Go to Settings to activate Windows.				

Default NACL has two rules:

- **Rule 100**: Type: All traffic, Protocol: All, Port: All, Source: 0.0.0.0/0, **Allow**
- **Asterisk (\*)**: Type: All traffic, Protocol: All, Port: All, Source: 0.0.0.0/0, **Deny**

#### How rule evaluation works:

1. Traffic arrives at subnet
2. Check Rule 100 → Matches! → ALLOW (stop checking)
3. If Rule 100 didn't match, check \* rule → DENY

#### Outbound Rules:

acl-0623e4cc6dc629090										
<a href="#">Details</a>		<a href="#">Inbound rules</a>		<a href="#">Outbound rules</a>		<a href="#">Subnet associations</a>				
<b>Outbound rules (2)</b>										
<input type="text"/> Filter outbound rules										
Rule number	Type	Protocol	Port range	Destination	Allow/Deny					
100	All traffic	All	All	0.0.0.0/0	<a href="#">Allow</a>	<a href="#">Edit outbound rules</a>				
*	All traffic	All	All	0.0.0.0/0	<a href="#">Deny</a>	<a href="#">Activate Windows</a> Go to Settings to activate Windows.				

Same pattern:

- **Rule 100**: Destination: 0.0.0.0/0, **Allow**

- **Asterisk (\*)**: Destination: 0.0.0.0/0, **Deny**

**Result:** The default NACL allows ALL traffic in and out.

## Creating Custom Network ACL

### Why create a custom NACL?

The project description suggests recreating the default NACL configuration manually to understand how it works. In production, you'd customize these rules based on security requirements.

### Create Network ACL:

**Create network ACL** Info

A network ACL is an optional layer of security that acts as a firewall for controlling traffic in and out of a subnet.

**Network ACL settings**

**Name - optional**  
Creates a tag with a key of 'Name' and a value that you specify.

**VPC**  
VPC to use for this network ACL.

**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
<input type="text" value="Name"/>	<input type="text" value="NextWork NetWork ACL"/> <span style="border: 1px solid #ccc; padding: 2px;">X</span> <span style="border: 1px solid #ccc; padding: 2px;">Remove tag</span>

**Add tag**  
You can add 49 more tags

Cancel Create network ACL

- **Name:** NextWork Network ACL
- **VPC:** vpc-072dff9a0c503e946 (NextWork VPC)
- **Tag:** Key=Name, Value=NextWork Network ACL

### Success Message:

You successfully created acl-05fa3a7f59516d111 / NextWork NetWork ACL.

Network ACLs (1/3) <small>Info</small>					
<input type="text" value="Find Network ACLs by attribute or tag"/> <span style="border: 1px solid #ccc; padding: 2px;">Actions</span> <span style="border: 1px solid orange; color: white; padding: 2px 10px;">Create network ACL</span>					
Name	Network ACL ID	Associated with	Default	VPC ID	Inbound rules count
-	acl-0623e4cc6dc629090	subnet-02cbe07bc0fdcc970 / Public 1	Yes	vpc-072dff9a0c503e946 / NextWork VPC	2 Inbound rules
-	acl-0afdc761399094d7	3 Subnets	Yes	vpc-0f420f6cd010b8079	2 Inbound rules
<input checked="" type="checkbox"/> NextWork NetWork ...	acl-05fa3a7f59516d111	-	No	vpc-072dff9a0c503e946 / NextWork VPC	1 Inbound rule

acl-05fa3a7f59516d111 / NextWork NetWork ACL					
<span style="border: 1px solid #ccc; padding: 2px;">Details</span> <span style="border: 1px solid #ccc; padding: 2px;">Inbound rules</span> <span style="border: 1px solid #ccc; padding: 2px;">Outbound rules</span> <span style="border: 1px solid #ccc; padding: 2px;">Subnet associations</span> <span style="border: 1px solid #ccc; padding: 2px;">Tags</span>					
<b>Inbound rules (1)</b>					
<input type="text" value="Filter inbound rules"/> <span style="border: 1px solid #ccc; padding: 2px;">Edit inbound rules</span>					
Rule number	Type	Protocol	Port range	Source	Allow/Deny
*	All traffic	All	All	0.0.0.0/0	<span style="color: red;">Deny</span>

"You successfully created acl-05fa3a7f59516d111 / NextWork NetWork ACL."

## Custom NACL Overview

Network ACLs (1/3) <a href="#">Info</a>					
	Name	Network ACL ID	Associated with	Default	VPC ID
<input type="checkbox"/>	-	acl-0623e4cc6dc629090	subnet-02cbe07bc0fdcc970 / Public 1	Yes	vpc-072dff9a0c503e946 / NextWork VPC
<input type="checkbox"/>	-	acl-0afdc761399094d7	3 Subnets	Yes	vpc-0f420f6cd010b8079
<input checked="" type="checkbox"/>	NextWork NetWork ...	acl-05fa3a7f59516d111	-	No	vpc-072dff9a0c503e946 / NextWork VPC

acl-05fa3a7f59516d111 / NextWork NetWork ACL						
<a href="#">Details</a> <a href="#">Inbound rules</a> <a href="#">Outbound rules</a> <a href="#">Subnet associations</a> <a href="#">Tags</a>						
<b>Inbound rules (1)</b>						
<input type="button" value="Edit inbound rules"/>	<input type="button" value="Filter inbound rules"/>	<input type="button" value="1"/>	<input type="button" value="Activate Windows"/>			
Rule number	Type	Protocol	Port range	Source	Allow/Deny	
*	All traffic	All	All	0.0.0.0/0	<input checked="" type="radio"/> Deny	<input type="button" value="Remove"/>

Now I have three Network ACLs:

- **acl-0623e4cc6dc629090**: Default (associated with Public 1)
- **acl-0afdc761399094d7**: Default (3 subnets)
- **acl-05fa3a7f59516d111**: NextWork NetWork ACL (not associated yet)

## Adding Inbound Rules

### Edit Inbound Rules:

Edit inbound rules [Info](#)  
Inbound rules control the incoming traffic that's allowed to reach the VPC.

Rule number <a href="#">Info</a>	Type <a href="#">Info</a>	Protocol <a href="#">Info</a>	Port range <a href="#">Info</a>	Source <a href="#">Info</a>	Allow/Deny <a href="#">Info</a>
100	All traffic	All	All	0.0.0.0/0	Allow
<input type="button" value="Add new rule"/>	<input type="button" value="Sort by rule number"/>				<input type="button" value="Remove"/>
					<input type="button" value="Cancel"/> <input type="button" value="Preview changes"/> <input type="button" value="Save changes"/>

I added Rule 100 to allow all traffic:

- **Rule number**: 100
- **Type**: All traffic
- **Protocol**: All (automatically filled)
- **Port range**: All (automatically filled)
- **Source**: 0.0.0.0/0
- **Allow/Deny**: Allow

### Why Rule Number 100?

Starting at 100 gives you room to add rules before it (e.g., Rule 50, Rule 75) if you need to block specific traffic later. Lower numbers are evaluated first.

### Final Inbound Rules:

acl-05fa3a7f59516d111 / NextWork NetWork ACL

Details | **Inbound rules** | Outbound rules | Subnet associations | Tags

**Inbound rules (2)**

Inbound rules (2)						<a href="#">Edit inbound rules</a>
<input type="text"/> Filter inbound rules						<a href="#">Activate Windows</a>
Rule number	Type	Protocol	Port range	Source	Allow/Deny	
100	All traffic	All	All	0.0.0.0/0	<input checked="" type="radio"/> Allow	<a href="#">Remove</a>
*	All traffic	All	All	0.0.0.0/0	<input type="radio"/> Deny	<a href="#">Go to Settings to activate Windows.</a>

- **Rule 100:** All traffic from 0.0.0.0/0 → **Allow**
- **Asterisk (\*):** All traffic from 0.0.0.0/0 → **Deny**

## Adding Outbound Rules

### Edit Outbound Rules:

[Edit outbound rules](#) Info  
Outbound rules control the outgoing traffic that's allowed to leave the VPC.

Rule number	Type	Protocol	Port range	Destination	Allow/Deny	<a href="#">Remove</a>
100	All traffic	All	All	0.0.0.0/0	Allow	<a href="#">Remove</a>
*	All traffic	All	All	0.0.0.0/0	Deny	<a href="#">Remove</a>

[Add new rule](#) [Sort by rule number](#)

[Cancel](#) [Preview changes](#) [Save changes](#)

I added Rule 100 to allow all outbound traffic:

- **Rule number:** 100
- **Type:** All traffic
- **Destination:** 0.0.0.0/0
- **Allow/Deny:** Allow

### Final Outbound Rules:

acl-05fa3a7f59516d111 / NextWork NetWork ACL

Details | Inbound rules | **Outbound rules** | Subnet associations | Tags

**Outbound rules (2)**

Outbound rules (2)						<a href="#">Edit outbound rules</a>
<input type="text"/> Filter outbound rules						<a href="#">Activate Windows</a>
Rule number	Type	Protocol	Port range	Destination	Allow/Deny	
100	All traffic	All	All	0.0.0.0/0	<input checked="" type="radio"/> Allow	<a href="#">Remove</a>
*	All traffic	All	All	0.0.0.0/0	<input type="radio"/> Deny	<a href="#">Go to Settings to activate Windows.</a>

- **Rule 100:** All traffic to 0.0.0.0/0 → **Allow**
- **Asterisk (\*):** All traffic to 0.0.0.0/0 → **Deny**

## Associating NACL with Subnet

**Important:** A Network ACL won't do anything until it's associated with a subnet!

### Subnet Associations (Before):

The screenshot shows the 'Subnet associations' tab of the Network ACL details page. A search bar at the top left contains the placeholder 'Filter subnet associations'. Below it is a table header with columns: Name, Subnet ID, Associated with, Availability Zone, IPv4 CIDR, and IPv6 CIDR. A message in the center of the table area states 'No subnets in this region are associated with this network ACL.' On the right side, there is a button labeled 'Edit subnet associations' and a status message: 'Activate Windows' with the subtext 'Go to Settings to activate Windows.'

"No subnets in this region are associated with this network ACL."

### Edit Subnet Associations:

I associated my custom NACL with Public 1 subnet.

The screenshot shows the 'Edit subnet associations' dialog. It has two main sections: 'Available subnets (1/1)' and 'Selected subnets'. The 'Available subnets' section lists one subnet: 'Public 1' (subnet-02cbe07bc0fdcc970, acl-0623e4cc6dc629090, euw3-az1 (eu-west-3a), 10.0.1.0/24). The 'Selected subnets' section contains one item: 'subnet-02cbe07bc0fdcc970 / Public 1'. At the bottom right are 'Cancel' and 'Save changes' buttons.

"You have successfully updated subnet associations for acl-05fa3a7f59516d111 / NextWork NetWork ACL."

### Final Network ACL List:

The screenshot shows the 'Network ACLs (1/3)' list. It includes a success message: 'You have successfully updated subnet associations for acl-05fa3a7f59516d111 / NextWork NetWork ACL.' The list table has columns: Name, Network ACL ID, Associated with, Default, VPC ID, and Inbound rules count. The table shows three rows: a default NACL (acl-0623e4cc6dc629090), a subnet-associated NACL (acl-0afdc761399094d7), and the custom NACL (acl-05fa3a7f59516d111) which is selected and associated with the Public 1 subnet.

The screenshot shows the 'Details' tab of the Network ACL details page for 'acl-05fa3a7f59516d111'. It displays the following details: Network ACL ID (acl-05fa3a7f59516d111), Associated with (subnet-02cbe07bc0fdcc970 / Public 1), Owner (056481036163), Default (No), and VPC ID (vpc-072dff9a0c503e946 / NextWork VPC).

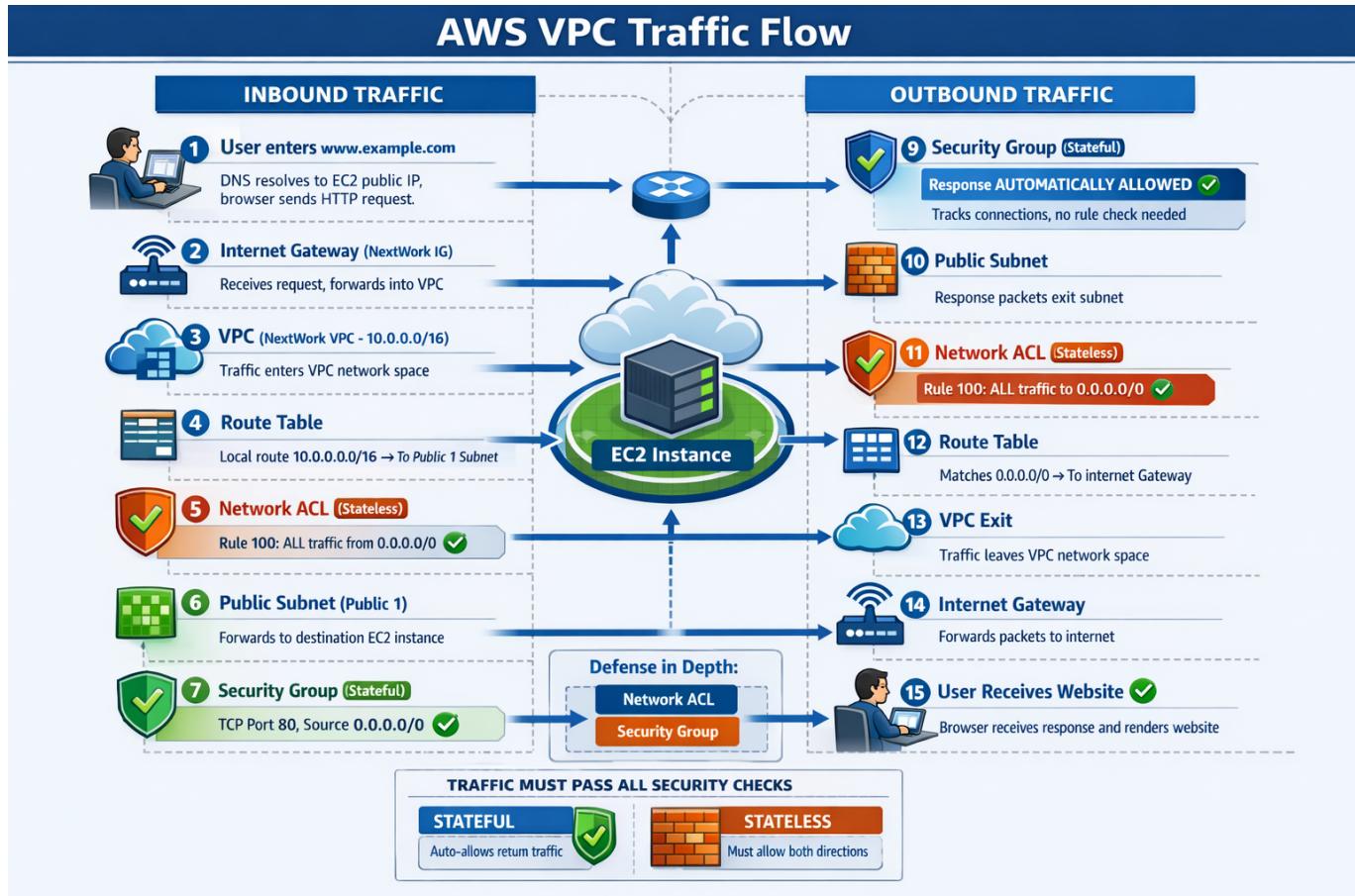
Now showing:

- **acl-0623e4cc6dc629090**: Associated with nothing (replaced by custom NACL)
- **acl-0afdc761399094d7**: Associated with 3 Subnets
- **acl-05fa3a7f59516d111**: Associated with **subnet-02cbe07bc0fdcc970 / Public 1**

**Key Point:** Only ONE Network ACL can be associated with a subnet at a time. When I associated my custom NACL, it replaced the default NACL for Public 1.

**Subnet now protected by custom Network ACL! 🎉**

## Understanding Traffic Flow



## Cleanup

### Why cleanup matters:

AWS charges for some resources (NAT Gateways, EC2 instances). While VPCs, subnets, and route tables are free, cleanup helps you avoid accidental charges, stay organized, and maintain good cloud hygiene.

### Deletion Order

Delete in reverse order of dependencies:

- Delete Custom Network ACL** - Disassociate from subnet, then delete
- Delete Security Group** - Ensure no instances use it, then delete
- Delete VPC** - Auto-deletes subnets, route tables, Internet Gateway attachments, and NACLs

### Steps:

- Go to VPC Console
- Select NextWork VPC → Actions → Delete VPC
- Confirm with VPC ID
- Verify deletion of subnets, route tables, Internet gateways, NACLs, and security groups

## Conclusion

This project provided hands-on experience with AWS networking fundamentals by building a complete VPC infrastructure with multi-layered security.

---

## What's Next?

This project is **Part 2** of a 9-part VPC series. The journey continues

---

**Project Completed:** December 2025

**Author:** YOUHAD AYOUB

**Region:** eu-west-3 (Paris)

**NextWork Challenge:** AWS Beginners Challenge - Project 4

---

*This project was completed as part of the NextWork AWS Beginners Challenge. Special thanks to the NextWork community for their comprehensive project structure and excellent educational materials. The hands-on approach of configuring route tables, security groups, and Network ACLs provided invaluable experience in understanding AWS networking at a deep level.*