

Create a Virtual Private Cloud (VPC) on AWS

Project Overview

Built a custom Virtual Private Cloud (VPC) on AWS with subnets and internet connectivity. Completed both the AWS Console method and the AWS CLI method using CloudShell to demonstrate different infrastructure deployment approaches.

Duration: ~30 minutes | **Difficulty:** Easy | **Region:** eu-west-3 (Paris)

What I Built

A complete VPC infrastructure including:

- Custom VPC with IPv4 CIDR block (10.0.0.0/16)
 - Public subnet in Availability Zone eu-west-3a
 - Internet Gateway for internet connectivity
 - Auto-assigned public IP addresses for resources
-

Technologies & Concepts

AWS Services:

- Amazon VPC (Virtual Private Cloud)
- AWS CloudShell
- AWS CLI

Key Concepts:

- VPCs and network isolation
 - Subnets (public vs private)
 - CIDR blocks and IP addressing
 - Internet Gateways
 - Availability Zones
-

Why VPCs Matter

Without VPCs, every AWS resource would exist in one giant open space with no privacy or organization. VPCs provide:

- **Privacy** - Isolate your resources from others
 - **Control** - Manage how resources communicate
 - **Security** - Set up access rules and restrictions
 - **Organization** - Group resources logically
-

Method 1: AWS Console (GUI)

Step 1: Create the VPC

Navigated to the VPC console to create a custom VPC.

Your VPCs															
VPCs		VPC encryption controls													
Your VPCs (2) Info															
<input type="text"/> Find VPCs by attribute or tag															
<input type="checkbox"/>	Name	VPC ID	State	Encryption c...	Encryption control ...	Block Public...	IPv4 CIDR	Last updated 16 minutes ago							
<input type="checkbox"/>	Prod	vpc-02881f2680610f7ba	Available	-	-	Off	192.168.0.0/16	Actions Create VPC							
<input type="checkbox"/>	-	vpc-0f420f6cd010b8079	Available	-	-	Off	172.31.0.0/16								

VPC Console showing existing default VPC

Configuration:

- **Name:** nextwork-vpc
- **IPv4 CIDR:** 10.0.0.0/16 (65,536 possible IP addresses)
- **Region:** eu-west-3 (Paris)
- **VPC only** (not VPC and more)

What's a CIDR block? It defines your IP address range. The **/16** means the first 16 bits are fixed (10.0), giving you 65,536 addresses (2^{16}).

Create VPC [Info](#)

A VPC is an isolated portion of the AWS Cloud populated by AWS objects, such as Amazon EC2 instances.

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

IPv4 CIDR block [Info](#)
 IPv4 CIDR manual input IPAM-allocated IPv4 CIDR block
10.0.0.0/16
CIDR block size must be between /16 and /28.

IPv6 CIDR block [Info](#)
 No IPv6 CIDR block IPAM-allocated IPv6 CIDR block Amazon-provided IPv6 CIDR block IPv6 CIDR owned by me

Tenancy [Info](#)
Default

VPC encryption control (\$) [Info](#)
Monitor mode provides visibility into encryption status without blocking traffic. Enforce mode prevents unencrypted traffic. [Additional charges apply](#)

None Monitor mode See which resources in your VPC are unencrypted but allow the creation of unencrypted resources. Enforce mode Requires all resources, except exclusions, in your VPC to be encryption-capable and blocks creation of unencrypted resources.

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**

Name nextwork-vpc Remove tag Add tag

You can add 49 more tags

Creating VPC with CIDR block 10.0.0.0/16

Step 2: Create a Public Subnet

Create a subnet within the VPC - like creating a neighborhood in a city.

Subnets (3) Info						
<input type="checkbox"/>	Name	Subnet ID	State	VPC	Action	IPv4 CIDR
<input type="checkbox"/>	-	subnet-01effc93001e5d065	Available	vpc-0f420f6cd010b8079	<input type="checkbox"/> Off	172.31.16.0/20
<input type="checkbox"/>	-	subnet-0c991a5c3cfa91208	Available	vpc-0f420f6cd010b8079	<input type="checkbox"/> Off	172.31.0.0/20
<input type="checkbox"/>	-	subnet-0c7e6aa16f23d7b76	Available	vpc-0f420f6cd010b8079	<input type="checkbox"/> Off	172.31.32.0/20

Subnet console showing 3 default subnets

Configuration:

- **VPC:** nextwork-vpc (vpc-017c5f1e2d59c67c1)
- **Name:** public-1
- **Availability Zone:** eu-west-3a (Europe Paris - first AZ)
- **IPv4 CIDR:** 10.0.0.0/24 (256 IP addresses)

Create subnet [Info](#)

VPC

VPC ID
Create subnets in this VPC.
[vpc-017c5f1e2d59c67c1 \(nextwork-vpc\)](#)

Associated VPC CIDRs

IPv4 CIDRs
10.0.0.0/16

Selecting the VPC for the subnet

Subnet settings
Specify the CIDR blocks and Availability Zone for the subnet.

Subnet 1 of 1

Subnet name
Create a tag with a key of 'Name' and a value that you specify.

The name can be up to 256 characters long.

Availability Zone [Info](#)
Choose the zone in which your subnet will reside, or let Amazon choose one for you.
[Europe \(Paris\) / euw3-az1 \(eu-west-3a\)](#)

IPv4 VPC CIDR block [Info](#)
Choose the VPC's IPv4 CIDR block for the subnet. The subnet's IPv4 CIDR must lie within this block.

IPv4 subnet CIDR block
 256 IPs

Tags - optional

Key	Value - optional
<input type="text" value="Name"/>	<input type="text" value="public-1"/> Remove

[Add new tag](#)
You can add 49 more tags.
[Remove](#)

[Add new subnet](#)

[AWS Command Line Interface command](#)

Configuring subnet settings with AZ and CIDR block

Why "Public"? This subnet will connect to the internet. Private subnets stay isolated for internal resources like databases.

You have successfully created 1 subnet: subnet-04a16915a9af09edb

Last updated 4 minutes ago Actions Create subnet

Subnets (1) Info

Find subnets by attribute or tag

Subnet ID : subnet-04a16915a9af09edb Clear filters

Name	Subnet ID	State	VPC	Block Public...	IPv4 CIDR
public-1	subnet-04a16915a9af09edb	Available	vpc-017c5f1e2d59c67c1 next...	Off	10.0.0.0/24

Subnet successfully created with 251 available IPs (AWS reserves 5)

Step 3: Enable Auto-Assign Public IP

Modified the subnet to automatically assign public IP addresses to resources launched in it.

Subnets (1/1) Info

Find subnets by attribute or tag

Subnet ID : subnet-04a16915a9af09edb Clear filters

Name	Subnet ID	State	VPC
public-1	subnet-04a16915a9af09edb	Available	vpc-017c5f1e2d59c67c1 nex...

Last updated 5 minutes ago Actions Create subnet

View details

Create flow log

Edit subnet settings

Edit IPv6 CIDRs

Edit network ACL association

Edit route table association

Edit CIDR reservations

Share subnet

Manage tags

Delete subnet

Actions menu - Edit subnet settings

Edit subnet settings

Subnet

Subnet ID: subnet-04a16915a9af09edb Name: public-1

Auto-assign IP settings

Enable AWS to automatically assign a public IPv4 or IPv6 address to a new primary network interface for an instance in this subnet.

Enable auto-assign public IPv4 address Info

Enable auto-assign customer-owned IPv4 address Info
Option disabled because no customer owned pools found.

Resource-based name (RBN) settings

Specify the hostname type for EC2 instances in this subnet and optional RBN DNS query settings.

Enable resource name DNS A record on launch Info

Enable resource name DNS AAAA record on launch Info

Hostname type Info

Resource name
 IP name

DNS64 settings

Enable DNS64 to allow IPv6-only services in Amazon VPC to communicate with IPv4-only services and networks.

Enable DNS64 Info

Activate Windows

Enabling auto-assign public IPv4 address

Why this matters: Without this, EC2 instances launched in this subnet won't get public IPs automatically, preventing internet access.

Step 4: Create Internet Gateway

Created an Internet Gateway - the bridge connecting the VPC to the internet.

Configuration:

- **Name:** nextwork-IG

Create internet gateway Info

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

nextwork-IG

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.

Key	Value - optional
<input type="text" value="Name"/>	<input type="text" value="nextwork-IG"/> X

Add new tag
You can add 49 more tags.

Cancel Create internet gateway

Creating the Internet Gateway

Step 5: Attach Internet Gateway to VPC

Connected the Internet Gateway to the VPC to enable internet connectivity.

igw-0f222ea65e72ec61f / nextwork-IG

Details <small>Info</small>		Actions					
Internet gateway ID igw-0f222ea65e72ec61f	State Detached	VPC ID -	Owner 056481036163				
Tags (1)		Manage tags < 1 > ⚙️					
<table border="1"> <thead> <tr> <th>Key</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>Name</td> <td>nextwork-IG</td> </tr> </tbody> </table>				Key	Value	Name	nextwork-IG
Key	Value						
Name	nextwork-IG						

Selecting nextwork-vpc for attachment

Attach to VPC (igw-0f222ea65e72ec61f) Info

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.

X

AWS Command Line Interface command

Cancel Attach internet gateway

Confirming the attachment

What does attaching do? It enables resources with public IPs in the VPC to access the internet and be accessible from the internet.

The screenshot shows the AWS VPC Internet Gateway details page for 'igw-0f222ea65e72ec61f / nextwork-IG'. It displays the following information:

- Internet gateway ID:** igw-0f222ea65e72ec61f
- State:** Attached
- VPC ID:** vpc-017c5f1e2d59c67c1 | nextwork-vpc
- Owner:** 056481036163

Tags (1)

Key	Value
Name	nextwork-IG

Actions ▾

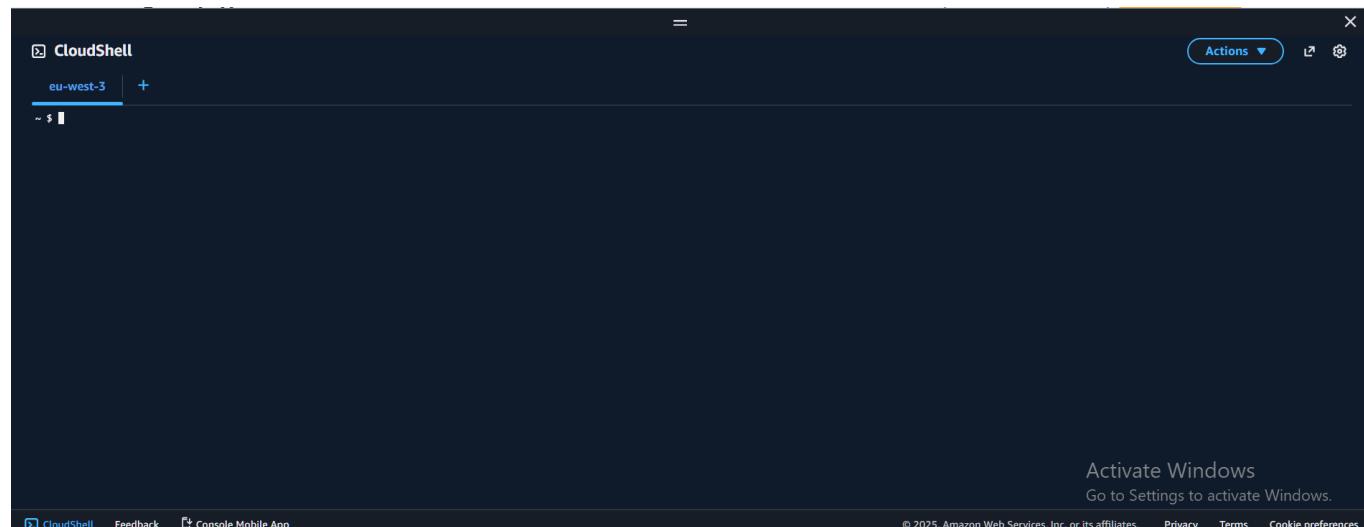
Internet Gateway successfully attached - State: Attached

💎 Secret Mission: AWS CloudShell & CLI

Instead of clicking through the console, I recreated the entire VPC infrastructure using AWS CLI commands in CloudShell!

Step 6: Open AWS CloudShell

Accessed the browser-based terminal by clicking the icon in the AWS Console top navigation bar.



AWS CloudShell - Pre-configured terminal with AWS CLI ready

What is CloudShell? A browser-based command-line interface with AWS CLI pre-installed. No setup or configuration needed!

Step 7: Create VPC with CLI

Used a single command to create the VPC:

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --tag-specifications
'ResourceType=vpc,Tags=[{Key=Name,Value=NextWork-CLI-VPC}]'
```

Output received:

```
{  
  "VpcId": "vpc-037197e4fc836955d",  
  "State": "pending",  
  "CidrBlock": "10.0.0.0/16",  
  "IsDefault": false  
}
```

Key Info:

- VPC ID: `vpc-037197e4fc836955d` ← Save this for next commands!
 - State: pending → available in seconds
 - 65,536 IP addresses available
-

Step 8: Create Subnet with CLI

Created the subnet using the VPC ID:

```
aws ec2 create-subnet \  
  --vpc-id vpc-037197e4fc836955d \  
  --cidr-block 10.0.0.0/24 \  
  --availability-zone eu-west-3a \  
  --tag-specifications 'ResourceType=subnet,Tags=[{Key=Name,Value=Public-CLI-1}]'
```

Output received:

```
{  
  "SubnetId": "subnet-0d91f172da7765724",  
  "State": "available",  
  "VpcId": "vpc-037197e4fc836955d",  
  "CidrBlock": "10.0.0.0/24",  
  "AvailableIpAddressCount": 251,  
  "MapPublicIpOnLaunch": false  
}
```

Key Info:

- Subnet ID: `subnet-0d91f172da7765724` ← Save for next command!
 - 251 usable IPs (AWS reserves 5 for internal use)
 - Public IP auto-assign is OFF - need to enable it
-

Step 9: Enable Auto-Assign Public IP

Modified the subnet attribute:

```
aws ec2 modify-subnet-attribute \
--subnet-id subnet-0d91f172da7765724 \
--map-public-ip-on-launch
```

Result: No output = Success!

The subnet now automatically assigns public IPs, matching our console configuration.

Step 10: Create Internet Gateway with CLI

Created the internet gateway:

```
aws ec2 create-internet-gateway \
--tag-specifications 'ResourceType=internet-gateway,Tags=
[{"Key":Name,Value=NextWork-CLI-IG}]'
```

Output received:

```
{
  "InternetGatewayId": "igw-062ca4ceebe4d8ab2",
  "Attachments": [],
  "Tags": [{"Key": "Name", "Value": "NextWork-CLI-IG"}]
}
```

Key Info:

- Internet Gateway ID: **igw-062ca4ceebe4d8ab2** ← Save for next step!
 - Attachments empty = not connected yet
-

Step 11: Attach Internet Gateway to VPC

Connected the gateway to the VPC:

```
aws ec2 attach-internet-gateway \
--internet-gateway-id igw-062ca4ceebe4d8ab2 \
--vpc-id vpc-037197e4fc836955d
```

Result: No output = Success!

The Internet Gateway is now attached and functional!

Key Learnings

Technical Skills Gained

- **VPCs isolate resources** - Essential for security and organization
- **CIDR notation** - /16 = 65,536 IPs, /24 = 256 IPs. Lower number = more addresses
- **Subnets organize VPCs** - Like neighborhoods in a city
- **Internet Gateways enable internet** - Must be attached to VPC to work
- **Availability Zones** - Each subnet exists in one AZ for redundancy
- **CLI is significantly faster** - 6 commands vs. many console clicks
- **JSON output structure** - IDs from one command become inputs for the next

Console vs CLI Comparison

Aspect	Console (GUI)	CLI (CloudShell)
Speed	Slower, many clicks	Fast, single commands
Learning Curve	Visual, beginner-friendly	Requires command knowledge
Automation	Manual, repetitive	Scriptable, repeatable
Documentation	Screenshots needed	Commands are self-documenting
Error Messages	Pop-ups	Detailed JSON responses
Reproducibility	Must click again	Re-run saved commands

My Verdict: CLI is definitely faster and more efficient once you learn the commands! Perfect for automation and Infrastructure as Code (IaC).

Best Practices Learned

- Choose nearby regions** - Better performance for users
- Enable auto-assign public IPs** - On public subnets only
- Tag all resources** - Makes identification and cost tracking easier
- Save resource IDs** - Essential when using CLI
- Delete unused resources** - Avoid unexpected charges
- Use CloudShell** - No need to install AWS CLI locally

Cleanup Process

Deleted all resources to avoid charges:

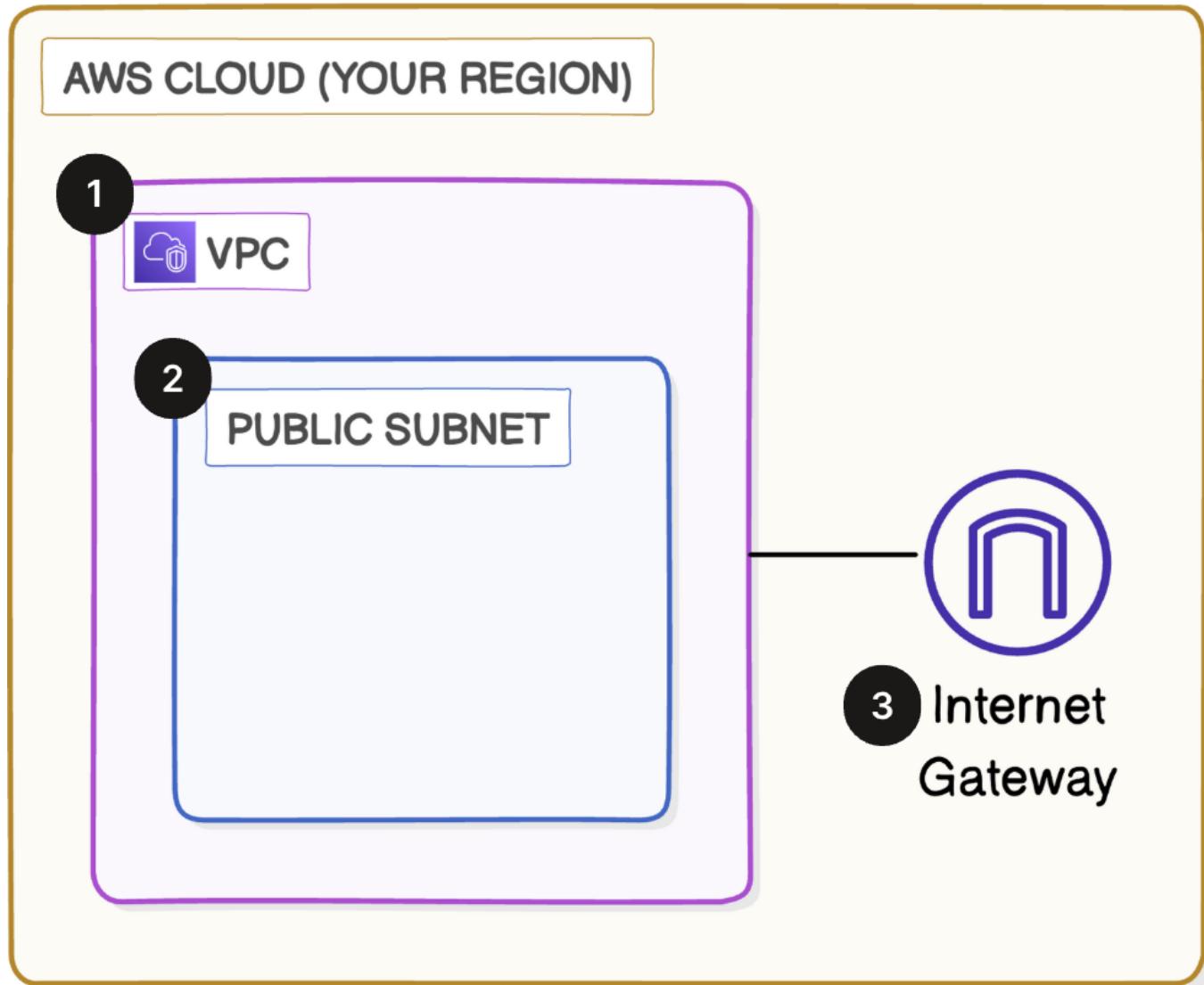
Console Method:

1. Navigate to VPC Dashboard
2. Select nextwork-vpc
3. Actions → Delete VPC
4. Subnet and Internet Gateway deleted automatically

CLI Method: Same process for NextWork-CLI-VPC

Important: When you delete a VPC, AWS automatically removes associated subnets and internet gateways!

Architecture Diagram



What's Next?

This is **Part 1** of the NextWork VPC series! The foundation is built, now it's time to complete the networking setup.

Next Project: VPC Traffic Flow and Security

In the next project, I'll configure:

- **Route Tables** - Direct traffic to the internet gateway
- **Security Groups** - Firewall rules for EC2 instances
- **Network ACLs** - Subnet-level security controls

Conclusion

Successfully created a complete VPC infrastructure using both AWS Console and AWS CLI! The CLI approach proved significantly faster and is essential for automation and Infrastructure as Code practices.

Project Completed: December 2025

Author: YOUHAD AYOUB

Part of: NextWork 7 Day DevOps Challenge - Day 2

This project demonstrates foundational AWS networking skills essential for cloud architecture and DevOps practices.