



# VPC Endpoints

K

Kehinde Abiuwa

The screenshot shows the AWS VPC Endpoint management interface. At the top, there's a search bar with the placeholder "Find endpoints by attribute or tag" and a "Clear filters" button. Below the search bar are four filter dropdowns: "Name" (selected), "VPC endpoint ID", "Endpoint type", and "Status". A single entry is listed: "NextWork VPC Endpoint" with VPC endpoint ID "vpce-09249fcf4c3285dc1", Endpoint type "Gateway", and Status "Available".

Below the table, a detailed view for the endpoint "vpce-09249fcf4c3285dc1 / NextWork VPC Endpoint" is shown. The "Details" tab is selected, followed by "Route tables", "Policy", and "Tags".

**Details**

<b>Endpoint ID</b> vpce-09249fcf4c3285dc1	<b>Status</b> Available	<b>Creation time</b> Tuesday 12 August 2025 at 17:21:26 BST	<b>Endpoint type</b> Gateway
<b>VPC ID</b> vpc-09a29786a3d9c269c (NextWork-vpc)	<b>Status message</b> -	<b>Service name</b> com.amazonaws.eu-north-1.s3	<b>Private DNS names enabled</b> No
<b>Service region</b> eu-north-1			

# Introducing Today's Project!

## What is Amazon VPC?

Amazon VPC is a secure, isolated virtual network within AWS that lets you control networking resources such as IP ranges, subnets, and routing so you can run AWS resources in a customizable and private environment.

## How I used Amazon VPC in this project

I used Amazon VPC to create a vpc, and then created an EC2 instance inside the VPC, Then I created a S3 bucket and uploaded some files into the bucket. I then went back into my VPC and went to Endpoints and then created a VPC endpoint for my S3 bucket and updated my route table so that traffic from my EC2 instance to my S3 bucket travels over the private AWS network instead of the public internet.

## One thing I didn't expect in this project was...

Nothing really. It was an interesting project overall



K

**Kehinde Abiuwa**  
NextWork Student

[nextwork.org](http://nextwork.org)

---

This project took me...

I spent around 40 minutes on this project

# In the first part of my project...

## Step 1 - Architecture set up

In this step I am going to Create a VPC from scratch, Launch an EC2 instance, which I'll connect to using EC2 Instance Connect later and Set up an S3 bucket.

## Step 2 - Connect to EC2 instance

In this step I am going to connect to my EC2 instance.

## Step 3 - Set up access keys

In this step I am going to create an IAM role that enables my EC2 instance to access my S3 buckets

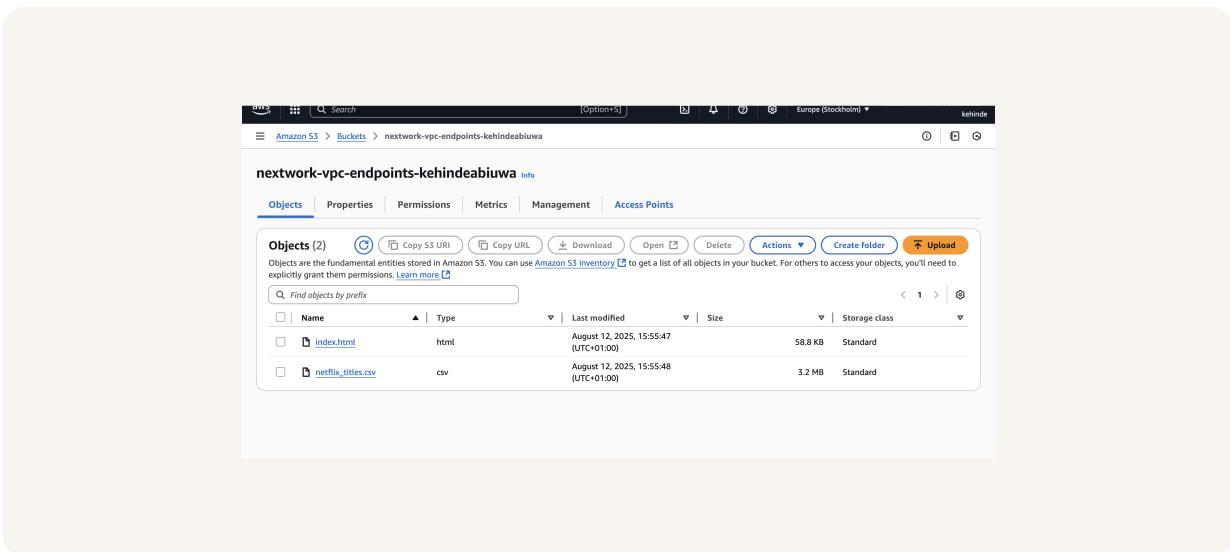
## Step 4 - Interact with S3 bucket

In this step I am going to get my EC2 instance to access my S3 bucket.

# Architecture set up

I started my project by launching a VPC and then launching and instance inside my VPC

I also set up an S3 bucket and then uploaded two files ( index.html & netflix\_titles.csv) into it



# Access keys

## Credentials

To set up my EC2 instance to interact with my AWS environment, I configured the following four things: Access Key ID – A unique identifier associated with my AWS credentials. Secret Access Key – The confidential key paired with the access key ID to securely sign requests. IAM Role or User Permissions – The specific permissions allowing the EC2 instance or user to access AWS resources. AWS CLI or SDK Configuration – The setup of AWS command-line tools or software development kits on the instance to use the credentials for API calls.

Access keys are a pair of security credentials, consisting of an access key ID and a secret access key, that allow you to authenticate and make programmatic requests to AWS services through tools like the AWS CLI

Secret access keys are confidential credentials used in AWS to securely sign programmatic requests to AWS services. They work together with the access key ID to authenticate and authorize API calls, ensuring that only authorized users or applications can access AWS resources. Secret access keys should be kept private and never shared or exposed publicly, as they grant access to your AWS account's resources.

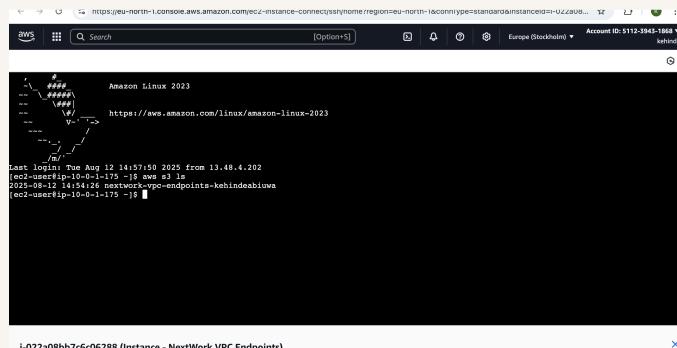
## Best practice

Although I'm using access keys in this project, a best practice alternative is to use IAM roles with attached policies, which provide temporary security credentials to AWS resources without needing to store long-term keys.

# Connecting to my S3 bucket

The command I ran was 'aws s3 ls'. Thhis command is used to list all S3 buckets in my AWS account, and if I specify a bucket name, it lists the objects inside that bucket.

The terminal responded with a list of all my S3 buckets. This indicated that the access keys I set up are working correctly

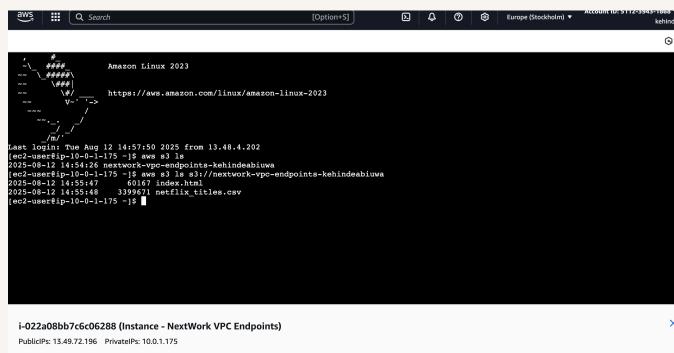


A screenshot of a terminal window on an Amazon Linux 2023 instance. The window shows the output of the 'aws s3 ls' command. The output includes:

```
Last login: Tue Aug 15 14:57:10 2023 from 13.48.4.202
[ec2-user@ip-10-0-1-175 ~]$ aws s3 ls
2023-08-15 14:54:26 nextwork-vpc-endpoints-kehindeabiuwa
[ec2-user@ip-10-0-1-175 ~]$
```

# Connecting to my S3 bucket

I also tested the command 'aws s3 ls s3://nextwork-vpc-endpoints-kehindeabiuwa'. Which returned a list of all the files inside my S3 bucket



The screenshot shows a terminal window titled 'Amazon Linux 2023' with the URL <https://www.amazon.com/linux/amazon-linux-2023>. The terminal displays the following command and its output:

```
Last login: Tue Aug 12 14:57:50 2025 from 13.49.4.202
[ec2-user@ip-10-0-1-175 ~]$ aws s3 ls
2025-08-12 14:57:50    60167 index.html
2025-08-12 14:57:50    339981 netflix_titles.csv
[ec2-user@ip-10-0-1-175 ~]$
```

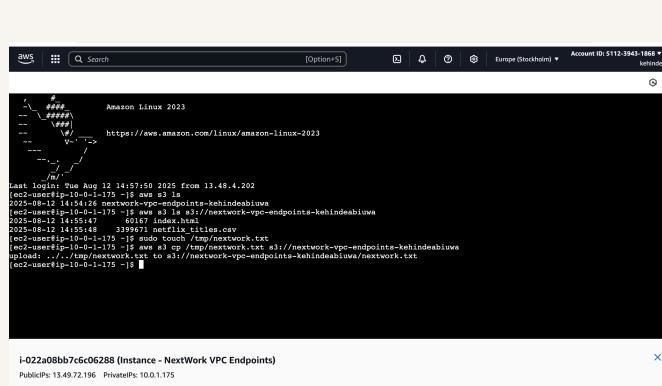
Below the terminal window, the status bar indicates the instance ID (i-022a08bb7c6c06288), public IP (13.49.72.196), and private IP (10.0.1.175).

# Uploading objects to S3

To upload a new file to my bucket, I first ran the command 'sudo touch /tmp/nextwork.txt'. This command creates an empty file called nextwork.txt in a temporary folder on my EC2 instance

The second command I ran was 'aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-endpoints-kehindeabiuwa'. This command will upload the just created nextwork.txt file into my S3 bucket

The third command I ran was 'aws s3 ls s3://nextwork-vpc-endpoints-kehindeabiuwa'. Which validated that the nextwork.txt file has been successfully uploaded into my S3 bucket



A screenshot of a terminal window titled 'Amazon Linux 2023' running on an AWS CloudWatch instance. The terminal shows the following command being run:

```
(ec2-user@ip-10-0-1-175 ~]$ aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-endpoints-kehindeabiuwa
```

Below the command, the terminal displays the output of the 'ls' command in the S3 bucket:

```
2025-08-12 14:55:47      60167 index.html
2025-08-12 14:55:47      1024000 variables.csv
(ec2-user@ip-10-0-1-175 ~]$ aws s3 ls s3://nextwork-vpc-endpoints-kehindeabiuwa
2025-08-12 14:55:47      60167 index.html
2025-08-12 14:55:47      1024000 variables.csv
(ec2-user@ip-10-0-1-175 ~]$ sudo touch /tmp/nextwork.txt
(ec2-user@ip-10-0-1-175 ~]$ aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-endpoints-kehindeabiuwa
upload: /tmp/nextwork.txt to s3://nextwork-vpc-endpoints-kehindeabiuwa/nextwork.txt
(ec2-user@ip-10-0-1-175 ~]$
```

At the bottom of the terminal window, there is a status bar with the text 'i-022a08bb76c06288 (Instance - NextWork VPC Endpoints)' and 'PublicIPs: 13.49.72.196 PrivateIPs: 10.0.1.175'.

# In the second part of my project...

## Step 5 - Set up a Gateway

In this step I am going to setup a VPC endpoint so that my EC2 instance and my S3 bucket can interact without using a public network.

## Step 6 - Bucket policies

In this step I am going to Limit my S3 bucket access's to only traffic from my endpoint. I am doing this to validate my endpoint connection

## Step 7 - Update route tables

In this step I am going to test my VPC endpoint set up & Troubleshoot a connectivity issue. I want to know if my EC2 instance can still access my S3 bucket now that I have limited access to my S3 bucket

## Step 8 - Validate endpoint connection

In this step, I am going to test my VPC endpoint setup and restrict my VPC's access to my AWS environment. The purpose is to validate that my EC2 instance can communicate with my S3 bucket over the private network, ensuring that the connection works as intended and does not rely on the public internet.

# Setting up a Gateway

I set up an S3 Gateway, which is a type of endpoint used specifically for Amazon S3 and DynamoDB (DynamoDB is an AWS database service).

## What are endpoints?

An endpoint is a service in AWS that allows private connections between a VPC and other AWS services without needing the traffic to go over the internet but rather through a private network.

The screenshot shows the AWS VPC Endpoints console. At the top, there is a search bar with the placeholder 'Find endpoints by attribute or tag' and a filter button labeled 'Clear filters'. Below the search bar, there is a table header with columns: 'Name', 'VPC endpoint ID', 'Endpoint type', and 'Status'. A single row is visible in the table, showing 'NextWork VPC Endpoint' with the ID 'vpce-09249fcf4c3285dc1', 'Gateway' as the endpoint type, and 'Available' as the status. At the bottom of the table, there is a link to 'vpce-09249fcf4c3285dc1 / NextWork VPC Endpoint'. Below this link, there are tabs for 'Details', 'Route tables', 'Policy', and 'Tags', with 'Details' being the active tab. The 'Details' tab displays several configuration options:

Endpoint ID	Status	Creation time	Endpoint type
vpce-09249fcf4c3285dc1	Available	Tuesday 12 August 2025 at 17:21:26 BST	Gateway
VPC ID	Status message	Service name	Private DNS names enabled
vpc-09a29786a3d9c269c (NextWork-vpc)	-	com.amazonaws.eu-north-1.s3	No
Service region			
eu-north-1			

# Bucket policies

A bucket policy is a JSON-based access control policy attached directly to an S3 bucket that defines who can access the bucket and what actions they are allowed to perform. It lets you specify permissions for users, roles, or AWS services, controlling access to the bucket and its contents at a granular level. Bucket policies are used to manage security and ensure that only authorized entities can read, write, or manage objects in the bucket.

My bucket policy will deny all S3 actions from any principal unless the request originates from the specified VPC endpoint (my nextwork VPC with ID vpce-09249fcf4c3285dc1). This ensures that only traffic coming through that private VPC endpoint can access my bucket nextwork-vpc-endpoints-kehindeabiwa, effectively blocking all other public or unauthorized access attempts.



The screenshot shows the AWS S3 Bucket Policy editor for the bucket `nextwork-vpc-endpoints-kehindeabiuwa`. The policy document is displayed in JSON format:

```
1  {
2    "Version": "2012-10-17",
3    "Statement": [
4      {
5        "Effect": "Deny",
6        "Principal": "*",
7        "Action": "s3:*",
8        "Resource": [
9          "arn:aws:s3:::nextwork-vpc-endpoints-kehindeabiuwa",
10         "arn:aws:s3:::nextwork-vpc-endpoints-kehindeabiuwa/*"
11       ],
12      "Condition": {
13        "StringNotEquals": {
14          "aws:sourceVpce": "vpce-09249fcf4c3285dc1"
15        }
16      }
17    ]
18  }
19 }
```

The right side of the interface shows the policy editor controls:

- Edit statement** button
- Select a statement** link
- Add new statement** button

A note at the bottom states: "The bucket policy, when in effect, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts." A link to "Learn more" is provided.

## Bucket policies

Right after saving my bucket policy, my S3 bucket page showed 'denied access' warnings. This was because the bucket policy I applied restricts access to only requests coming through the specified VPC endpoint. Since this current access was likely from outside that VPC endpoint such as from the AWS Management Console over the public internet, it triggered the "denied access" warnings. The policy is working as intended by blocking any access that doesn't come from the approved private network.

I also had to update my route table because if my route table doesn't have a route directing traffic destined for S3 to my VPC endpoint, then traffic from my EC2 instance will try to reach the S3 bucket over the public internet instead of through the secure, private connection. This defeats the purpose of using the VPC endpoint for private communication.



The screenshot shows the AWS S3 console with the following details:

- Breadcrumbs:** Amazon S3 > Buckets > nextwork-vpc-endpoints-kehindeabiuwa
- Tab Selection:** Permissions (highlighted in blue)
- Permissions Overview:** A section titled "Permissions overview" with a note about access findings and a link to "View analyzer for eu-north-1".
- Block public access (bucket settings):** A section with a red box around the error message: "You don't have permission to view the Block public access (bucket settings) configuration. You need s3:GetAccountPublicAccessBlock to view the Block public access (bucket settings) configuration. Learn more about Identity and access management in Amazon S3." It also includes a "Diagnose with Amazon Q" button and a "► API response" link.
- Bucket policy:** A section with a note about bucket policies and links to "Edit" and "Delete" buttons.

# Route table updates

To update my route table, I went to the VPC console, selected my VPC endpoint, and then associated the route table linked to my VPC so that traffic from my EC2 instance is routed through the VPC endpoint to access S3 privately.

After updating my public subnet's route table, my terminal could return the list of files in my s3 bucket

The screenshot shows the AWS VPC Subnets page. On the left, there's a sidebar with navigation links like VPC dashboard, EC2 Global View, Filter by VPC, Virtual private cloud (Your VPCs, Subnets, Route tables, Internet gateways, Egress-only Internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections), Security (Network ACLs, Security groups), and a bottom link for Help & Support. The main content area has tabs for Subnets (1/4) and Info. A search bar at the top says "Find subnets by attribute or tag". Below it is a table with columns: Name, Subnet ID, State, and VPC. One row is selected: "NextWork-subnet-public1-eu-north-1a" with Subnet ID "subnet-0bf6c919388569303", State "Available", and VPC "vpc-09a29786a3d9c269c | Next...". To the right of the table are "Actions" (dropdown with "Create subnet") and "Create subnet" buttons. Below the table, a section titled "Route table: rtb-019007ed3ed24ed29 / NextWork-rtb-public" shows a table of routes. It has columns: Destination, Target, and a dropdown for Target. Three routes are listed: "10.0.0.0/16" with Target "local", "0.0.0.0/0" with Target "igw-0215c8d193cc3fde3", and "pl-c5aa4fba" with Target "vpce-09249rf4c3285dc1". There are also "Edit route table association" and "Edit" buttons.

# Endpoint policies

An endpoint policy is a JSON-based resource policy attached to a VPC endpoint that controls which AWS services and resources can be accessed through that endpoint. It allows you to define fine-grained permissions, such as restricting access to specific S3 buckets or DynamoDB tables, and ensures that traffic through the endpoint follows your security and compliance requirements.

I updated my endpoint's policy by changing the line "Effect": "Allow" to "Effect": "Deny"... I could see the effect of this right away, because then I wasnt able to access my S3 bucket from my EC2 instance



The screenshot shows the AWS VPC Endpoints console. On the left, there's a sidebar with links like Route tables, Internet gateways, Egress-only Internet gateways, DHCP option sets, Elastic IPs, Managed prefix lists, NAT gateways, Peering connections, Security (Network ACLs, Security groups), and PrivateLink and Lattice (Getting started, Endpoints, Endpoint services, Service networks, Lattice services, Resource configurations, Resource gateways). The main area shows a success message: "Successfully updated policy vpce-09249fcf4c3285dc1". Below it is a table titled "Endpoints (1/1)" with one item: "NextWork VPC Endpoint" (vpce-09249fcf4c3285dc1) which is a "Gateway" and "Available". At the bottom, there's a detailed view for "vpce-09249fcf4c3285dc1 / NextWork VPC Endpoint" showing a JSON policy document:

```
1  {
2     "Version": "2008-10-17",
3     "Statement": [
4         {
5             "Effect": "Deny",
6             "Principal": "*",
7             "Action": "*",
8             "Resource": "*"
9         }
10    ]
11 }
```



[nextwork.org](https://nextwork.org)

# The place to learn & showcase your skills

Check out [nextwork.org](https://nextwork.org) for more projects

