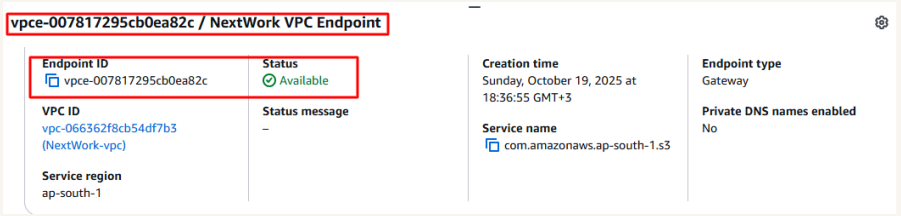
nextwork.org

VPC Endpoints

ABDUL REHMAN



Introducing Today's Project!

What is Amazon VPC?

Amazon VPC is a Virtual Private Cloud, which is a logically isolated section of the AWS cloud where you can launch and manage resources in your own virtual network. It is useful because it gives you full control over networking, including IP address ranges, subnets, route tables, and security settings, allowing you to securely isolate resources, manage traffic, and connect with other networks or VPCs as needed.

How I used Amazon VPC in this project

In today's project, I used Amazon VPC to create a secure, isolated network for my EC2 instance, configure subnets and route tables, and set up a VPC endpoint for private access to S3. This allowed me to control network traffic, enhance security, and ensure my instance could interact with AWS services without using the public internet.

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

One thing I didn't expect in this project was how much control and flexibility VPC endpoints give over private access to AWS services. Even small changes in route tables or endpoint policies immediately affected connectivity, which highlighted the importance of careful configuration and understanding of network security in the cloud.

This project took me...

This project took me approximately 2 hours to complete. Most of the time was spent setting up the VPC, configuring the EC2 instance and S3 bucket, creating the VPC endpoint, and testing access and endpoint policies to ensure secure and private connectivity.

In the first part of my project...

Step 1 - Architecture set up

In this step, I will create a new VPC, launch an EC2 instance, and set up an S3 bucket because I need a secure network environment and storage resource to test how my EC2 instance can access S3 using a VPC endpoint without going through the public internet.

Step 2 - Connect to EC2 instance

In this step, I will connect to my EC2 instance and access my S3 bucket through the public internet because I want to test the default behavior of how EC2 communicates with S3 before setting up a private VPC endpoint connection.

Step 3 - Set up access keys

In this step, I will give my EC2 instance access to my AWS environment because I want it to interact securely with services like S3 using a VPC endpoint, without relying on public internet access or long-term access keys.

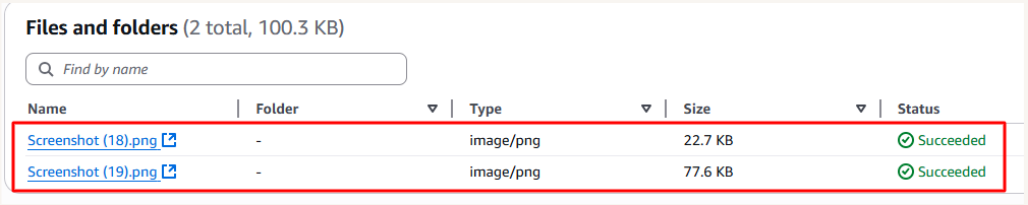
Step 4 - Interact with S3 bucket

In this step, I will return to my EC2 instance and access my S3 bucket because I want to test whether the instance can now reach S3 privately through the VPC endpoint instead of using the public internet.

Architecture set up

I started my project by launching a new VPC with a public subnet and an EC2 instance. This setup provided the foundation needed to later test private connectivity between my VPC and S3 using a VPC endpoint.

In this step, I set up an S3 bucket and uploaded two files to it. This allowed me to have data stored in S3 that I could later access and test through my VPC endpoint connection from the EC2 instance.



Access keys

Credentials

To set up my EC2 instance to interact with my AWS environment, I configured the VPC, subnets, security groups, and the S3 bucket. These configurations ensured that my instance could securely communicate with AWS services and manage data in S3 without needing public internet access.

Access keys are a combination of an Access Key ID and a Secret Access Key that act like a username and password for programmatic access to your AWS account. They allow services, applications, or users to interact with AWS resources through the AWS CLI or SDKs without needing to log in through the console.

The secret access key is like the password that pairs with your access key ID (your username). You need both to access AWS services. Secret is a key word here - anyone who has it can access your AWS account, so we need to keep this away from anyone else!

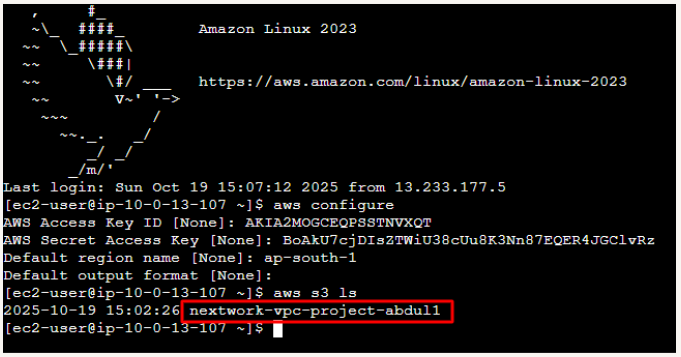
Best practice

The best practice alternative to using access keys is to assign an IAM role to your EC2 instance. This allows the instance to securely access AWS services without storing or managing access keys manually, reducing security risks and simplifying credential management.

Connecting to my S3 bucket

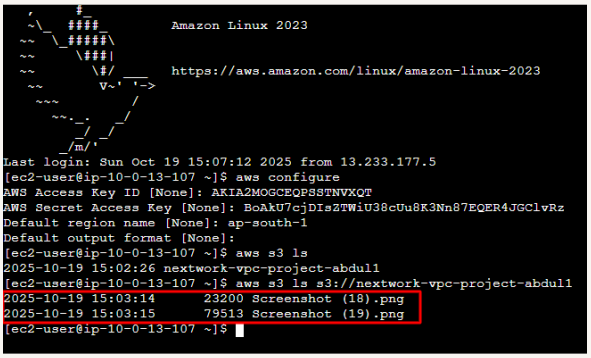
The command I ran was aws s3 ls. This command is used to list all the S3 buckets available in my AWS account, allowing me to confirm that my EC2 instance can successfully communicate with the S3 service over the public internet.

The terminal responded with the list of S3 buckets. This indicated that the EC2 instance had proper permissions and could successfully access the S3 buckets, confirming that the network and IAM configurations were set up correctly.



Connecting to my S3 bucket

I also tested the command aws s3 ls s3://nextwork-vpc-project-abdul1, which returned the two files I had uploaded to the bucket. This confirmed that my EC2 instance could access the specific S3 bucket and view its contents, verifying that the VPC endpoint and bucket permissions were correctly configured.



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 named nextwork.txt in the /tmp directory on my EC2 instance, which I could then upload to the S3 bucket for testing access and permissions.

The second command I ran was"aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-project-abdul1". This command will upload the local file nextwork.txt from my EC2 instance to the specified S3 bucket, making it accessible for storage and management within S3.

The third command I ran was "aws s3 ls s3://nextwork-vpc-project-abdul1", which validated that the file I uploaded from my EC2 instance was successfully stored in the S3 bucket. This confirmed that the instance could write to the bucket and that the VPC endpoint and permissions were correctly configured.

```
Last login: Sun Oct 19 15:07:12 2025 from 13.233.177.5
[ec2-user@ip-10-0-13-107 ~]$ aws configure
AWS Access Key ID [None]: AKIA2MOGCEQPSSTNVXQT
AWS Secret Access Key [None]: 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
Default region name [None]: ap-south-1
Default output format [None]:
[ec2-user@ip-10-0-13-107 ~]$ aws s3 ls
2025-10-19 15:02:26 nextwork-vpc-project-abdul1
[ec2-user@ip-10-0-13-107 ~]$ aws s3 ls s3://nextwork-vpc-project-abdul1
2025-10-19 15:03:14      23200 Screenshot (18).png
2025-10-19 15:03:15      79513 Screenshot (19).png
[ec2-user@ip-10-0-13-107 ~]$ sudo touch /tmp/nextwork.txt
[ec2-user@ip-10-0-13-107 ~]$ aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-project-abdul1
upload: ../../tmp/nextwork.txt to s3://nextwork-vpc-project-abdul1/nextwork.txt
[ec2-user@ip-10-0-13-107 ~]$ aws s3 ls s3://nextwork-vpc-project-abdul1
2025-10-19 15:03:14      23200 Screenshot (18).png
2025-10-19 15:03:15      79513 Screenshot (19).png
2025-10-19 15:25:28          0 nextwork.txt
[ec2-user@ip-10-0-13-107 ~]$
```

In the second part of my project...

Step 5 - Set up a Gateway

In this step, I will set up a VPC endpoint to enable direct communication between my VPC and S3 because I want to allow my EC2 instance to access the S3 bucket privately, without using the public internet, improving security, performance, and control over data traffic.

Step 6 - Bucket policies

In this step, I will configure the S3 bucket policy to allow access only from my VPC endpoint because I want to ensure that all traffic to the bucket comes securely from within my VPC, preventing public internet access and enhancing security and compliance.

Step 7 - Update route tables

In this step, I will test my VPC endpoint setup and troubleshoot any connectivity issues because I want to ensure that my EC2 instance can access the S3 bucket privately and securely, and that all traffic is correctly routed through the VPC endpoint without relying on the public internet.

Step 8 - Validate endpoint conection

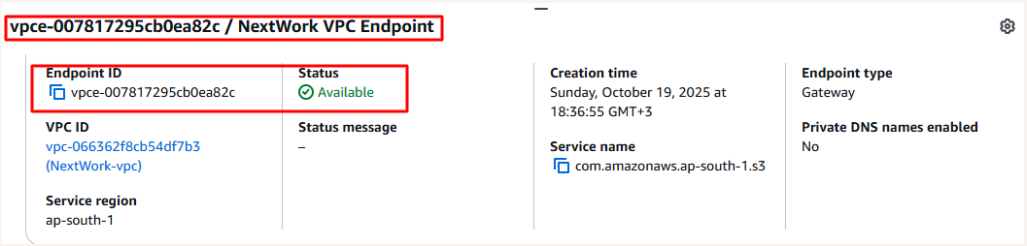
In this step, I will test the VPC endpoint setup again and apply restrictions to my VPC’s access to AWS services because I want to ensure that the EC2 instance can still access S3 privately while limiting unnecessary or public access, enhancing security and control over my environment.

Setting up a Gateway

I set up an S3 Gateway, A Gateway is a type of endpoint used specifically for Amazon S3 and DynamoDB (DynamoDB is an AWS database service). Gateways work by simply adding a route to your VPC route table that directs traffic bound for S3 or DynamoDB to head straight for the Gateway instead of the internet.

What are endpoints?

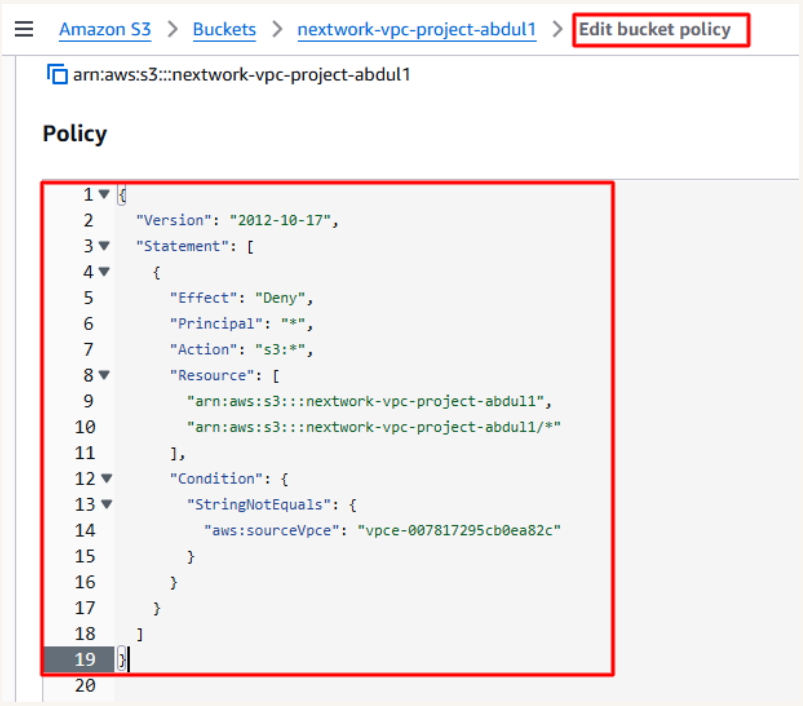
An endpoint in AWS is a service that allows private connections between your VPC and other AWS services without needing the traffic to go over the internet.



Bucket policies

A bucket policy is a type of IAM policy designed for setting access permissions to an S3 bucket. Using bucket policies, you get to decide who can access the bucket and what actions they can perform with it.

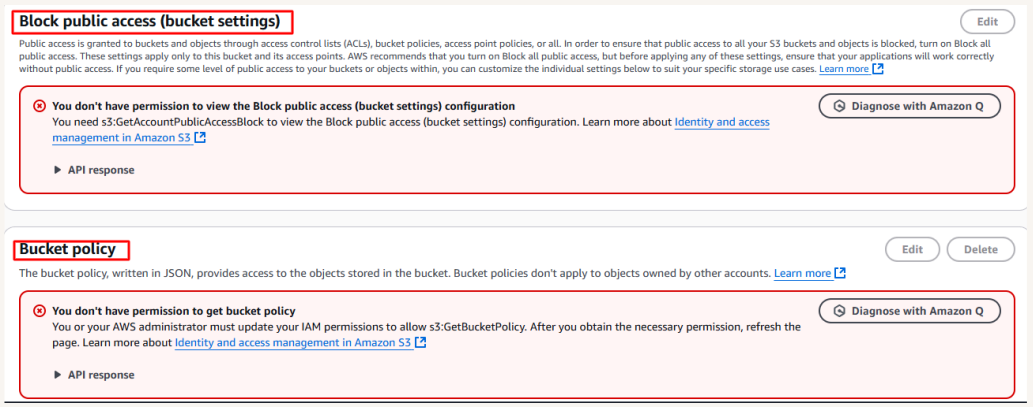
My bucket policy will denies all actions (s3:*) on my S3 bucket and its objects to everyone (Principal: "*")... unless the access is from the VPC endpoint with the ID defined in aws:sourceVpce. In other words, only traffic coming from my VPC endpoint can get any access to my S3 bucket!



Bucket policies

Right after saving my bucket policy, my S3 bucket page showed 'denied access' warnings. This was because my policy denies all actions unless they come from my VPC endpoint. This means any attempt to access my bucket from other sources, including the AWS Management Console, is blocked!

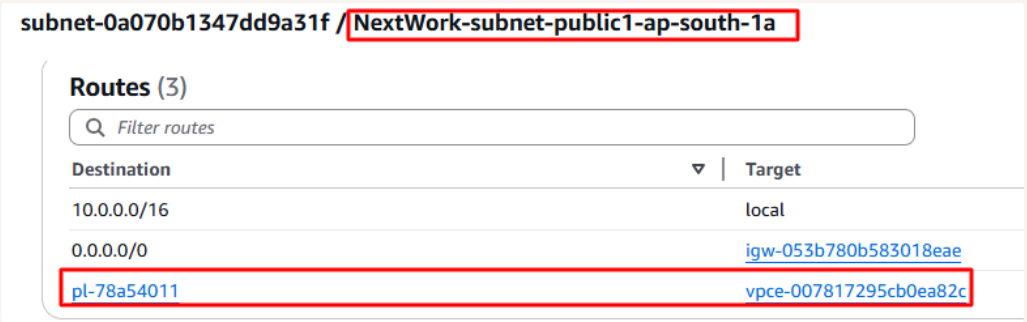
I also had to update my route table because the VPC endpoint needed a specific route to direct traffic destined for S3 through the private endpoint. Without this route, the EC2 instance wouldn’t know to use the endpoint, and traffic might still try to go over the public internet.


Block public access (bucket settings)
Edit
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more
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
Diagnose with Amazon Q
API response
Bucket policy
Edit
Delete
The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more
You don't have permission to get bucket policy
You or your AWS administrator must update your IAM permissions to allow s3:GetBucketPolicy. After you obtain the necessary permission, refresh the page. Learn more about Identity and access management in Amazon S3
Diagnose with Amazon Q
API response

Route table updates

To update my route table, I added a new route with the destination set to the S3 service's prefix list and the target set to my VPC endpoint. This ensured that all traffic from my VPC to S3 would be routed privately through the endpoint instead of going over the public internet.

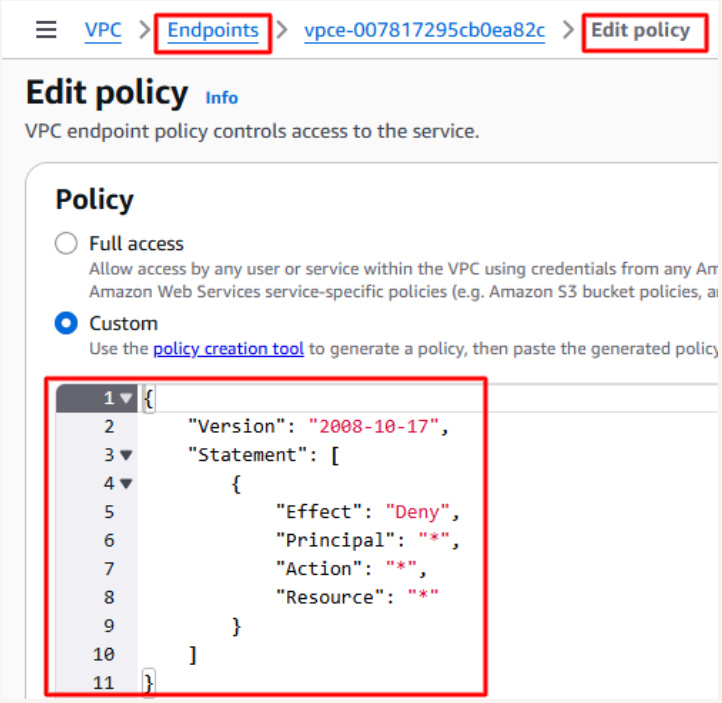
After updating my public subnet's route table, my terminal responded with the names of the files stored in my S3 bucket. This confirmed that the EC2 instance was successfully accessing the bucket through the VPC endpoint, validating that the private routing and permissions were correctly configured.



Endpoint policies

An endpoint policy is a JSON-based policy attached to a VPC endpoint that controls what actions and resources the endpoint can access. It works similarly to an IAM policy but specifically governs traffic coming through the VPC endpoint, allowing you to restrict which S3 buckets or services can be accessed and what operations (like read or write) are allowed. In short, an endpoint policy secures and limits access for private connections between your VPC and AWS services.

I updated my endpoint's policy by setting "Effect": "Deny" for certain actions or resources. I could see the effect of this right away because my EC2 instance was immediately blocked from performing the denied operations on the S3 bucket, confirming that the endpoint policy was actively controlling access as intended.


VPC
Endpoints
vpce-007817295cb0ea82c
Edit policy
Edit policy
Info
VPC endpoint policy controls access to the service.
Policy
Full access
Allow access by any user or service within the VPC using credentials from any Am
Amazon Web Services service-specific policies (e.g. Amazon S3 bucket policies, a
Custom
Use the policy creation tool to generate a policy, then paste the generated policy
{
"Version": "2008-10-17",
"Statement": [
{
"Effect": "Deny",
"Principal": "*",
"Action": "*",
"Resource": "*"
}
]
}

The place to learn & showcase your skills

Check out nextwork.org for more projects