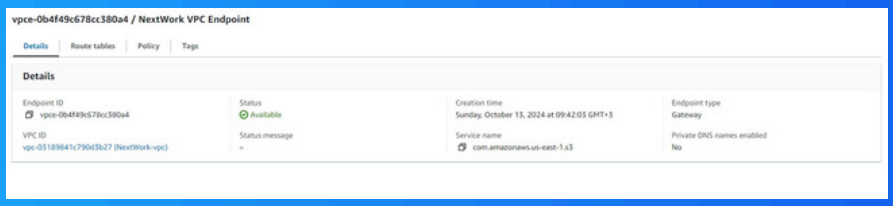
NextWork.org

VPC Endpoints

Hassan Gachoka



Hassan Gachoka
linkedin.com/gachokahassan

Introducing Today's Project!

What is Amazon VPC?

Amazon VPC is a service that creates a private network within AWS, allowing users to customize their network configuration. It enhances security, isolates resources, and enables direct connectivity to AWS services, optimizing cost and performance.

How I used Amazon VPC in this project

In today's project, I used Amazon VPC to create a private network for my EC2 instance, allowing secure communication with an S3 bucket. I set up a VPC endpoint to ensure traffic was routed privately, enhancing security and performance.

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

One thing I didn't expect in this project was the complexity of configuring the endpoint policies. I realized how critical they are for managing access and enhancing security, allowing for granular control over resource access within my VPC.

This project took me...

This project took me about 60 minutes to complete. It involved setting up the VPC, configuring endpoints, adjusting policies, and testing the connections, which required careful attention to detail at each step.

Hassan Gachoka
linkedin.com/gachokahassan

In the first part of my project...

Step 1 Architecture set up

In this step, we're setting up the core components for our project: a VPC, an EC2 instance, and an S3 bucket. These will form the foundation for securely connecting our VPC to S3 using a private endpoint.

Step 2 Connect to EC2 instance

In this step, we're connecting to the EC2 instance to test accessing the S3 bucket through the public internet. This helps us understand the current setup and highlights why creating a VPC endpoint for secure access is important.

Step 3 Set up access keys

In this step, we're creating access keys to give our EC2 instance secure credentials. This enables the instance to authenticate and interact with AWS services like S3, allowing us to perform tasks programmatically.

Step 4 Interact with S3 bucket

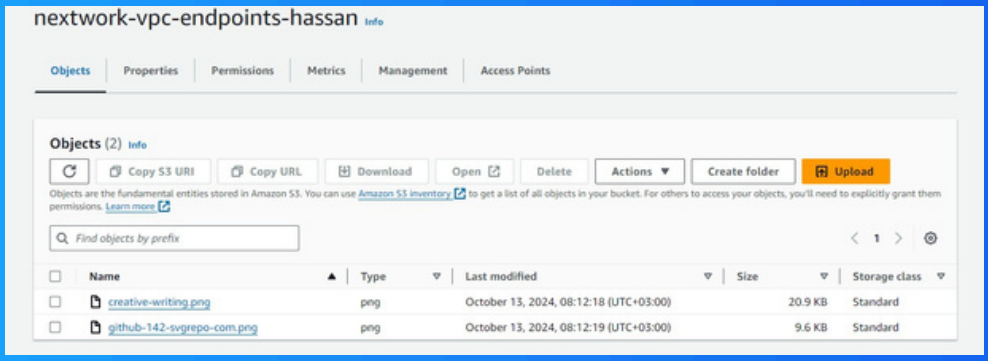
In this step, we're testing the access keys by connecting to our EC2 instance and attempting to access the S3 bucket. This will confirm that our instance has the necessary permissions to interact with S3 and verify that the setup is working correctly

Hassan Gachoka
linkedin.com/gachokahassan

Architecture set up

I started my project by launching a VPC with a public subnet and an EC2 instance within that subnet. This setup allows for internet access and prepares the foundation for further configurations and resource interactions.

I also set up an S3 bucket named nextwork-vpc-endpoints-hassan with default settings. After creating the bucket, I uploaded two files from my local computer to store in the bucket for testing purposes.



Hassan Gachoka
linkedin.com/gachokahassan

Access keys

Credentials

To set up my EC2 instance to interact with my AWS environment, I configured the following four things: the Access Key ID, the Secret Access Key, and the Default region name, and left the Default output format.

Access keys are credentials consisting of an access key ID and a secret access key, used to authenticate programmatic access to AWS services. They allow applications or servers to securely interact with AWS resources.

Secret access keys are the passwords paired with access key IDs, used to authenticate and access AWS services. They must be kept secure, as anyone with the secret access key can gain access to your AWS account and resources.

Best practice

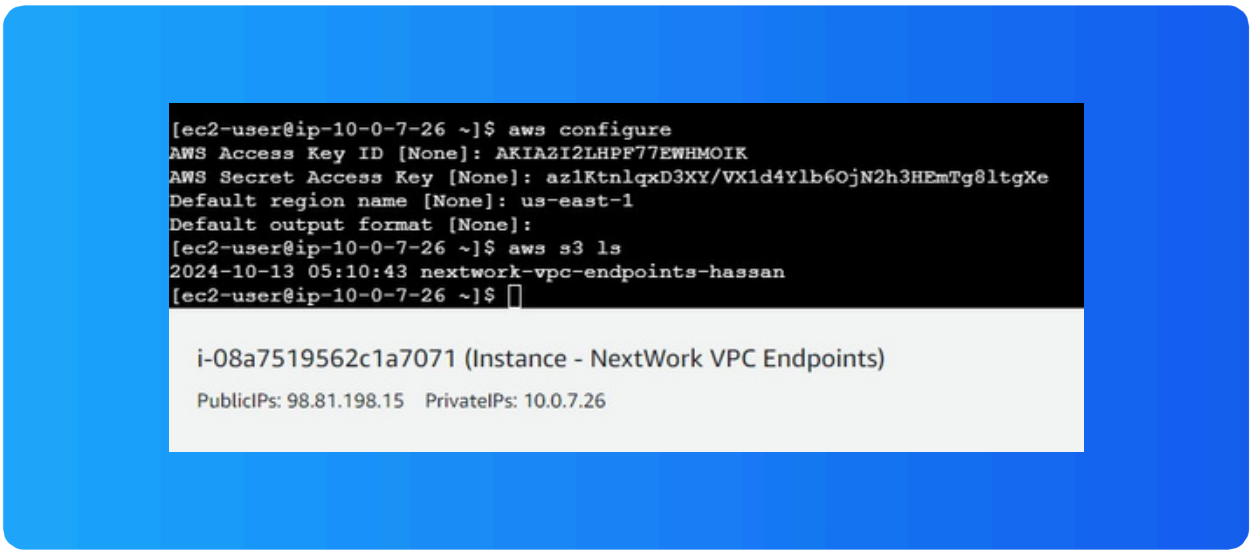
Although I'm using access keys in this project, a best practice alternative is to use IAM roles. Roles allow for temporary credentials and provide a more secure way to grant access to AWS services without hardcoding access keys.

Hassan Gachoka
linkedin.com/gachokahassan

Connecting to my S3 bucket

The command I ran was AWS S3 ls. This command is used to list all the S3 buckets in my AWS account, allowing me to verify access to S3 from the EC2 instance.

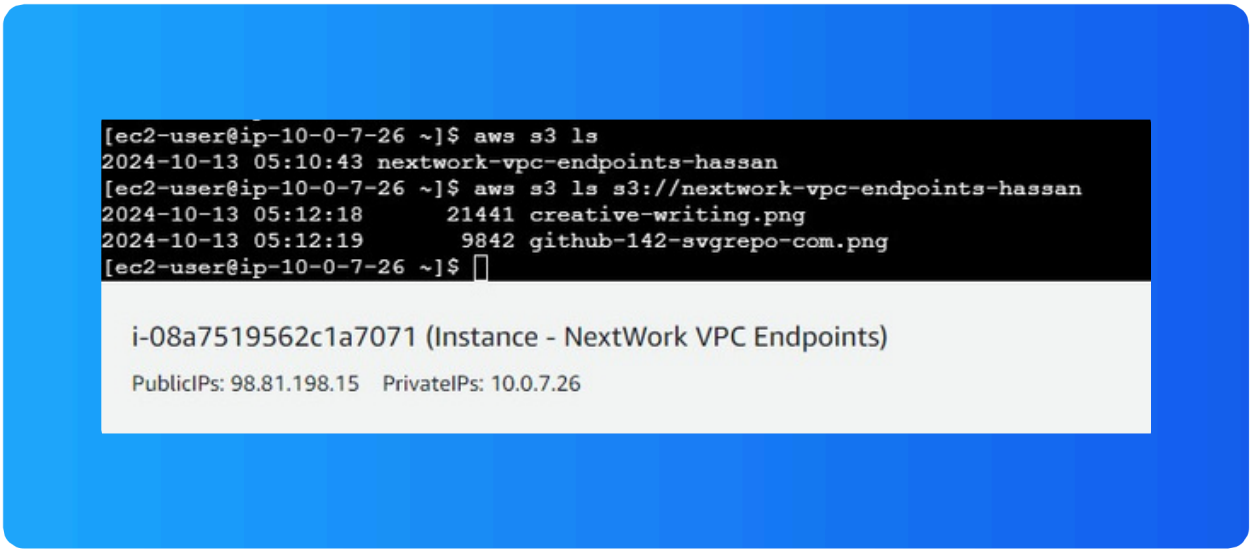
The terminal responded with a list of S3 buckets, including nextwork-vpc-endpoints-hassan. This indicated that the access keys I set up successfully allowed my EC2 instance to access my S3 bucket.



Hassan Gachoka
linkedin.com/gachokahassan

Connecting to my S3 bucket

I also tested the command aws s3 ls s3://nextwork-vpc-endpoints-hassan, which returned a list of objects stored in my S3 bucket. This confirmed that I could access and view the contents of the bucket successfully.



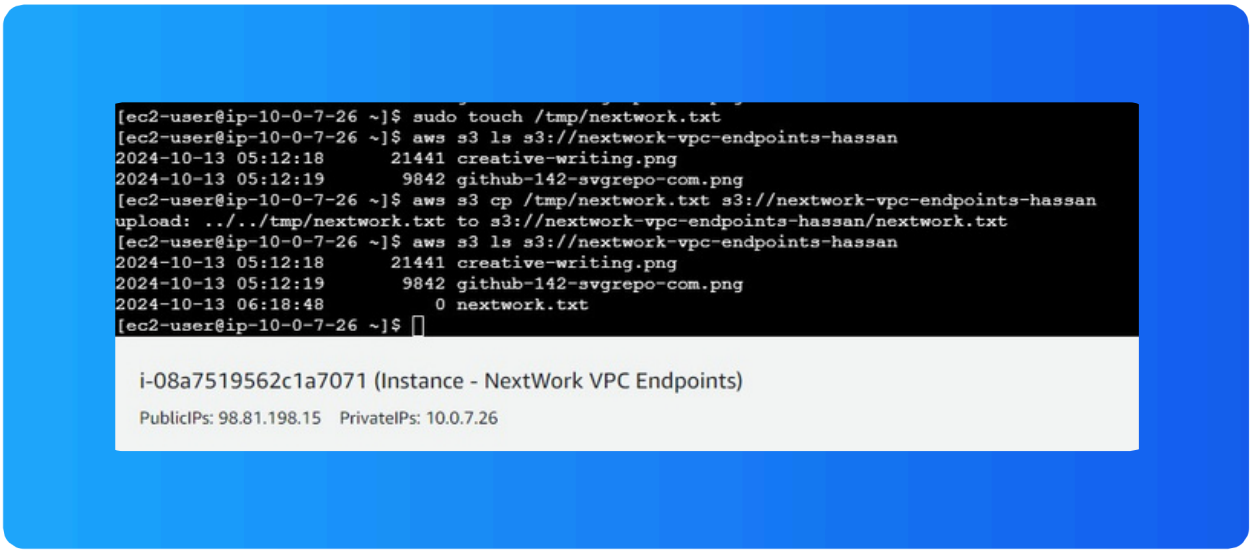
Hassan Gachoka
linkedin.com/gachokahassan

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 text file named nextwork.txt in the /tmp directory of my EC2 instance.

The second command I ran was 'aws s3 cp /tmp/nextwork.txt s3://nextwork-vpc-endpoints-hassan'. This command will upload the nextwork.txt file from my EC2 instance to my specified S3 bucket.

The third command I ran was 'aws s3 ls s3://nextwork-vpc-endpoints-hassan', which validated that the nextwork.txt file was successfully uploaded to my S3 bucket by listing the objects inside it.



In the second part of my project...

Step 5 Set up a Gateway

In this step, we're creating a VPC endpoint to establish a private connection between my VPC and S3. This setup ensures secure communication without exposing data to the public internet, minimizing the risk of attacks and enhancing overall security.

Step 6 Bucket policies

In this step, we're setting a super secure bucket policy to restrict access to our S3 bucket. We'll block all traffic except from our VPC endpoint, ensuring that only the EC2 instance connected to it can access the bucket.

Step 7 Update route tables

In this step, I'm testing whether my EC2 instance can still access the S3 bucket after limiting access to only the VPC endpoint. This confirms if the endpoint is properly set up to secure private connections.

Step 8 Validate endpoint conection

In this step, I'm testing the VPC endpoint setup by having theEC2 instance access the S3 bucket. This final check confirms that the endpoint is set up correctly, ensuring secure access and keeping communication within the AWS network.

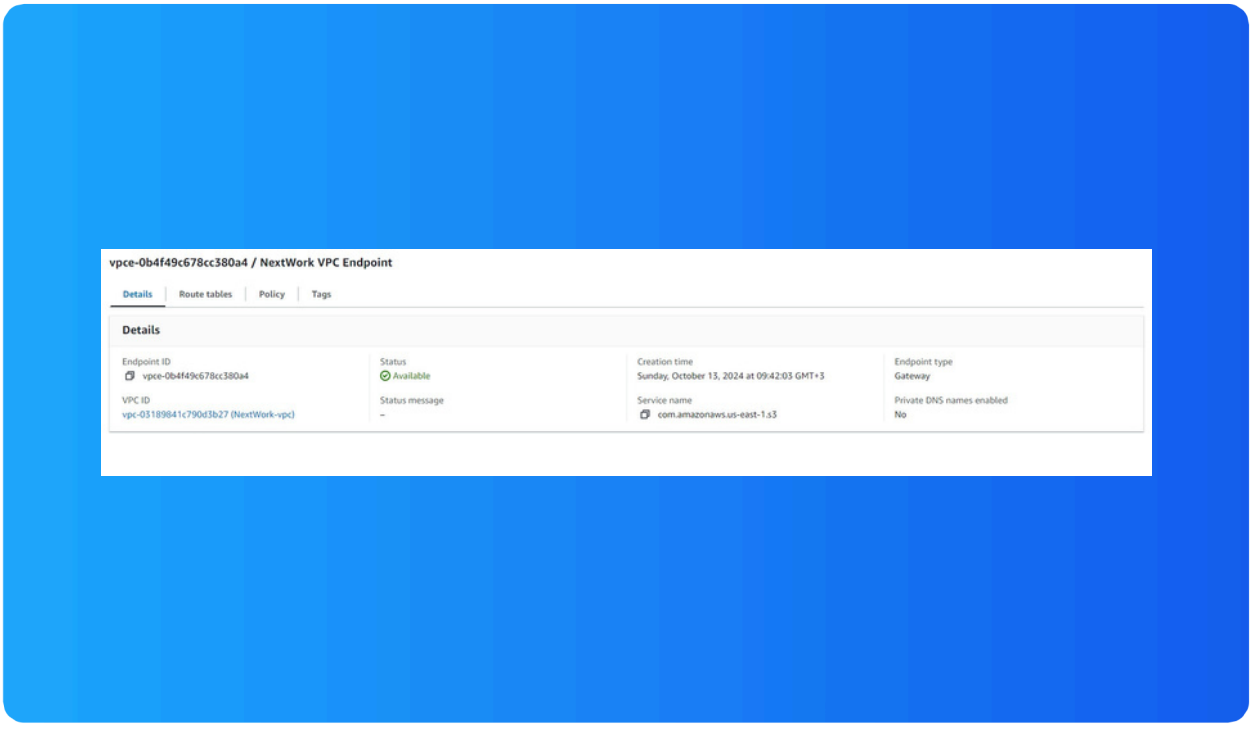
Hassan Gachoka
linkedin.com/gachokahassan

Setting up a Gateway

I set up an S3 Gateway, which is a type of endpoint specifically used for Amazon S3 and DynamoDB. It directs traffic bound for these services through a private route in your VPC, enhancing security by avoiding the public internet.

What are endpoints?

An endpoint is a service in AWS that enables private connections between your VPC and other AWS services, allowing data to flow without traversing the public internet. This enhances security and keeps traffic within the AWS network.



Hassan Gachoka
linkedin.com/gachokahassan

Bucket policies

A bucket policy is an IAM policy that controls access to an S3 bucket, specifying who can access it, what actions they can take, and under what conditions. It allows for detailed management of permissions for bucket secur it y.

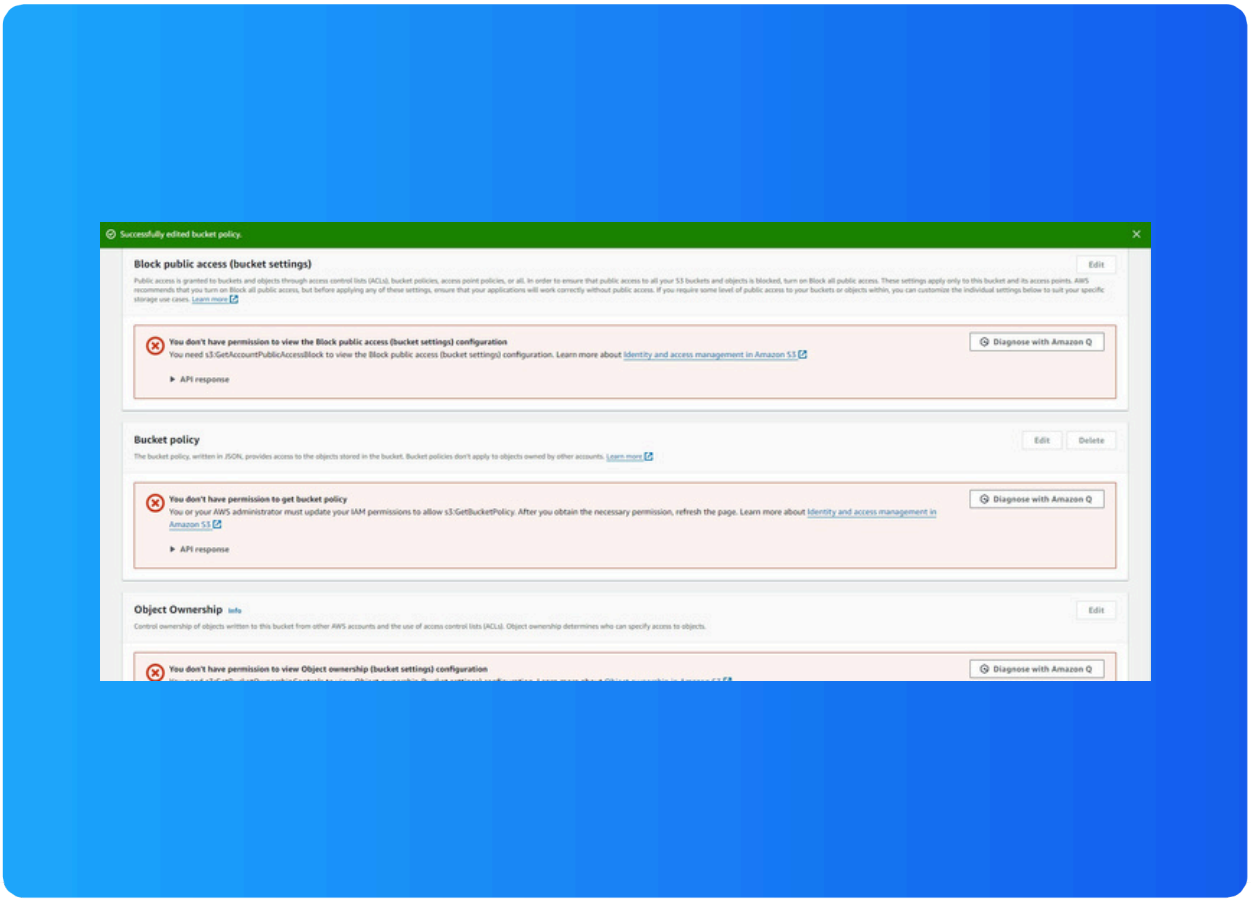
My bucket policy will deny all actions on the S3 bucket to everyone unless the access comes from the specified VPC endpoint. It ensures that only traffic originating from the VPC endpoint can interact with the bucket.

Hassan Gachoka
linkedin.com/gachokahassan

Bucket policies

Right after saving my bucket policy, my S3 bucket page showed "denied access" warnings. This was because the policy denies all actions unless the access originates from the specified VPC endpoint, blocking other sources like the Management Console.

I also had to update my route table because it lacked a route to the VPC endpoint. Without this route, traffic from my EC2 instance would attempt to access the S3 bucket through the public internet, compromising the secure connection.

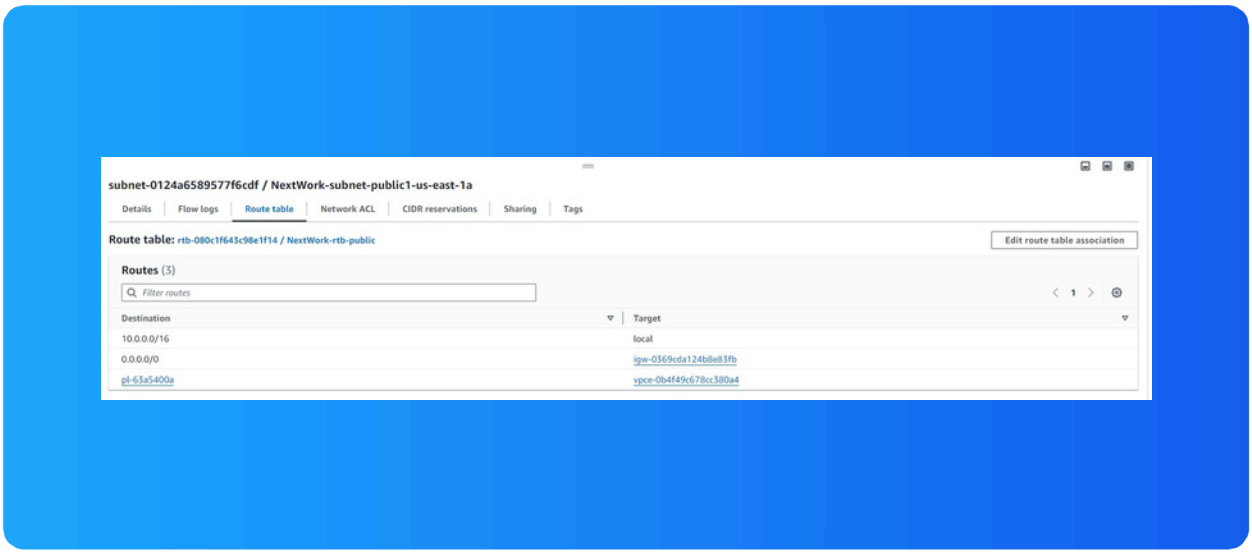


Hassan Gachoka
linkedin.com/gachokahassan

Route table updates

To update my route table, I selected the checkbox next to my VPC endpoint in the Endpoints section, clicked on Route tables, and then managed the route tables to ensure my public subnet had a route directing traffic to the VPC endpoint.

After updating my public subnet's route table, my terminal could return the list of objects in my S3 bucket. This confirmed that the VPC endpoint was set up correctly, allowing private access from the EC2 instance to the S3 bucket.



Hassan Gachoka
linkedin.com/gachokahassan

Endpoint policies

An endpoint policy is a set of permissions that define what actions can be performed through a VPC endpoint. It controls access to specific AWS services and resources, allowing for granular management of allowed and denied requests.

I updated my endpoint's policy by changing "Effect": "Allow" to "Effect": "Deny." I could see the effect of this right away because my EC2 instance was no longer able to access the S3 bucket, confirming the policy change blocked access.