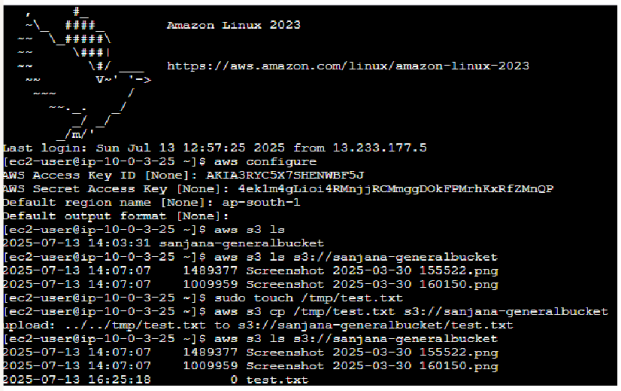
nextwork.org

Access S3 from a VPC

Sanjana Tripathy



Introducing Today's Project!

What is Amazon VPC?

Amazon VPC (Virtual Private Cloud) lets you create a secure, isolated network within AWS where you can launch and manage resources like EC2 instances. It gives full control over IP ranges, subnets, routing, and security.

How I used Amazon VPC in this project

In today’s project, I used Amazon VPC to launch an EC2 instance in a public subnet and connect it securely to an S3 bucket using AWS CLI. The VPC provided isolation, security, and internet access needed for my instance to interact with AWS services.

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

One thing I didn't expect in this project was how enjoyable using the AWS CLI would be. Running commands and seeing instant results made the experience fun and interactive. I'm now excited to explore more CLI capabilities in future projects.

This project took me...

This project took me approximately 2 hours.

Sanjana Tripathy
NextWork Student

In the first part of my project...

Step 1 - Architecture set up

In this step I am going to create VPC from scratch.then I will be launching an EC2 instance in it.

Step 2 - Connect to my EC2 instance

In this step, I launched a new VPC and created an EC2 instance inside it. This setup will help me test how to access Amazon S3 using programmatic access with an access key

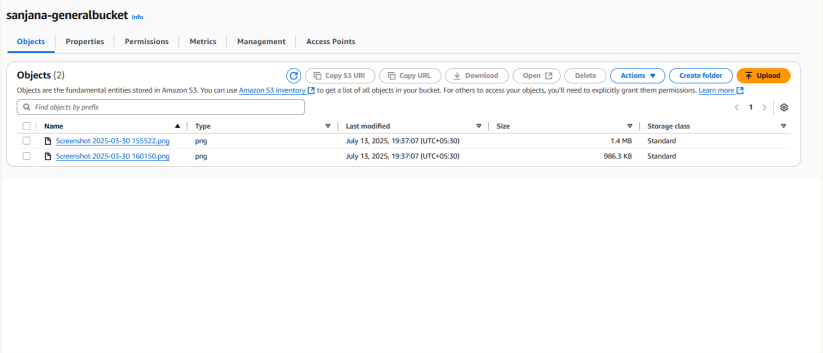
Step 3 - Set up access keys

In this step I am creating access keys for my created EC2 instance, so that my instance can access the AWS environment.

Architecture set up

In this step, I launched a new VPC and created an EC2 instance inside it. This setup will help me test how to access Amazon S3 using programmatic access with an access key

I also set up the storage environment by creating an S3 bucket and uploading two files to it. This setup is essential for testing whether my EC2 instance can successfully access and list objects from the bucket later in the project.

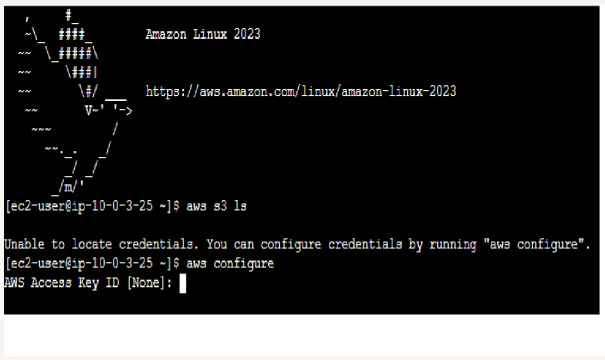


Running CLI commands

AWS CLI is a command-line tool that lets you interact with AWS services using terminal commands. I have access to AWS CLI because I configured it with my IAM user's access key, allowing secure programmatic access to AWS resources.

The first command I ran was "aws s3 ls", which is used to list all accessible S3 buckets under my AWS account. It helps verify that my CLI is correctly configured and that my IAM credentials have the necessary permissions to access S3.

The second command I ran was "aws configure", which sets up my AWS CLI by prompting for credentials like Access Key, Secret Key, region, and output format. This ensures secure and personalized access to AWS services from the command line.



Access keys

Credentials

To set up my EC2 instance to interact with my AWS environment, I configured the AWS CLI using the aws configure command. This stored my access key, secret key, region, and output format, enabling secure communication with AWS services like S3.

Access keys are a combination of an Access Key ID and Secret Access Key that allow programmatic access to your AWS account. They authenticate CLI commands or API requests, ensuring secure, identity-based access to AWS services.

Secret access keys are sensitive credentials used with access key IDs to securely authenticate API requests to AWS services. They must be kept confidential, as anyone with access can potentially control your AWS resources.

Best practice

Although I'm using access keys in this project, a best practice alternative is to create an IAM role with necessary permissions and attach it to the EC2 instance—this enhances security by avoiding hardcoded credentials.

In the second part of my project...

Step 4 - Set up an S3 bucket

In this step, I am going to create an S3 bucket and upload two objects to it. Creating the bucket is essential, as later in the project, my EC2 instance will access this bucket to list and verify the objects it contains.

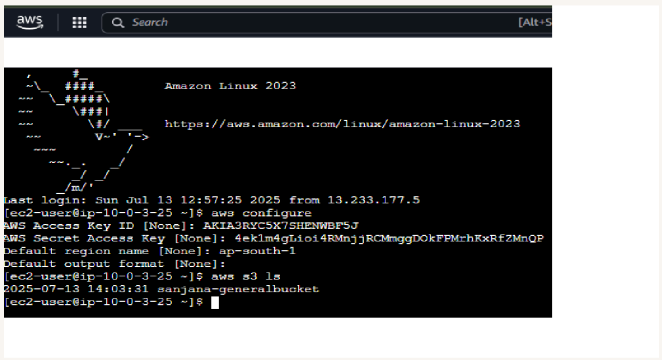
Step 5 - Connecting to my S3 bucket

In this step, I'll enable interaction between my EC2 instance and the S3 bucket. This involves using the AWS CLI on the instance to check if it can successfully access and list the contents of the bucket—verifying connectivity and permissions.

Connecting to my S3 bucket

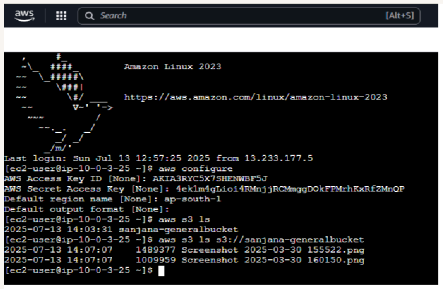
The first command I ran was "aws s3 ls", which is used to list all accessible S3 buckets under my AWS account. It helps verify that my CLI is correctly configured and that my IAM credentials have the necessary permissions to access S3.

When I ran the command aws s3 ls again, the terminal responded with my S3 bucket name. This indicated that the AWS CLI was correctly configured and my EC2 instance successfully connected to the S3 service using the access credentials I provided.



Connecting to my S3 bucket

Another CLI command I ran was aws s3 ls s3://<my-s3-bucket-name>, where I replaced the placeholder with my actual bucket name. This command returned a list of all objects stored in that specific S3 bucket, confirming successful access.

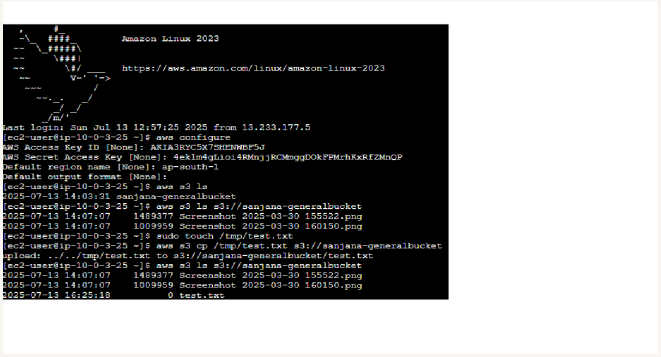


Uploading objects to S3

To upload a new file to my bucket, I first ran the command sudo touch /tmp/test.txt. This command creates an empty file named test.txt in the /tmp directory, preparing it for upload to the S3 bucket in the next step.

The second command I ran was aws s3 cp /tmp/test.txt s3://nextwork-vpc-project-yourname, replacing the placeholder with my actual S3 bucket name. This command uploads the test.txt file from my EC2 instance to the specified S3 bucket.

The third command I ran was aws s3 ls s3://nextwork-vpc-project-yourname, replacing the placeholder with my actual bucket name.This command validated that the file was successfully uploaded by listing the contents (objects)of the specified S3 bucket.



The place to learn & showcase your skills

Check out nextwork.org for more projects