Negin Hezarjaribi

INTERNATIONAL UNIVERSITY OF APPLIED SCIENCES

Course Name: Cloud Programming

Course Code: DLBSEPCP01_E

Deployment of a simple webpage on AWS

18th March 2025

Abstract

This project deploys a secure and scalable, globally optimized static website on AWS (Amazon Web Services). The solution is aligned with cloud best practices, including utilizing Amazon S3 for storage, Amazon CloudFront CDN, and Origin Access Identity OAI for security. Using Terraform, an Infrastructure as Code (IaC) tool, we automate the deployment process, providing efficiency and repeatability.

Concept and Solution

Some use cases may require a reliable hosting solution for static websites with high availability, security, and fast content delivery. We build a cloud-based solution instead of a traditional web hosting solution using AWS services that is cost-effective. There are three main components in the architecture:

Amazon S3 (Simple Storage Service): Stores the website files (HTML, CSS) while blocking public access to enhance security.

Amazon CloudFront (CDN): Distributes website content globally, reducing latency and improving user experience. It also secures the site by restricting direct access to S3.

Origin Access Identity (OAI): Grants exclusive read permissions to CloudFront, ensuring that content is only accessible through the CDN and not directly from S3.

This approach enhances website security, minimizes latency, and ensures scalability to handle varying traffic loads.

Technical Approach

The implementation follows a structured deployment process using Terraform:

Setting Up AWS Credentials: Configure an IAM user with necessary permissions for Terraform automation.

Configuring AWS CLI and Installing Terraform: Install required tools and set up authentication for resource management.

Creating and Configuring the S3 Bucket: Secure the bucket by disabling public access and setting up policies that restrict access to CloudFront's OAI.

Deploying CloudFront as a CDN: Configure the distribution with caching policies, custom error responses, and SSL certificates for secure communication.

Defining Infrastructure as Code (IaC) with Terraform: Automate resource provisioning using Terraform scripts, making the deployment process repeatable and scalable.

Validating and Deploying: Execute terraform init, terraform plan, and terraform apply to deploy the resources, ensuring all configurations are correctly applied.

Retrieving Deployment Outputs: Extract the CloudFront URL from Terraform to access the hosted static website.

Conclusion

This project successfully implements a robust, secure, and efficient static website hosting solution using AWS services and Terraform. By automating the infrastructure setup, we ensure a scalable and easily maintainable deployment. The use of CloudFront and OAI significantly enhances security while improving content delivery performance. This solution is a suitable way to host your static websites in a cost-effective and easy to manage way in cloud, while practicing the best practices of cloud computing.