



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

Experiment -7

Student Name: Dhruv

Branch: B.E. CSE

Semester: 6th

Subject Name: Foundation of Cloud IOT Edge ML

UID:22BCS10279

Section/Group:639-B

Date of Performance:13/03/25

Subject Code: 22CSP-367

1. **Aim:** Deploy and manage an edge computing environment using Terraform for running real-time IoT workloads close to data sources.
2. **Objective:** To deploy and manage an edge computing environment using Terraform for running real-time IoT workloads close to data sources.
3. **Prerequisites:** Terraform, AWS CLI, Cloud Platform - AWS S3

4. Procedure:

- i. **Log in to AWS:**
Go to the AWS Management Console and log in with your credentials.
- ii. **Open Security Credentials from profile:**
Here, create an access key
- iii. **Copy Access key and secret access key:**
Download .csv file and copy from there.
- iv. **Download AWS CLI – Command Line Interface:**
Select 64-bit Windows version and download it. Then install it.
- v. **Download terraform by HashiCorp:**
Select AMD64 and download binary file for windows. Then extract it.
- vi. **Open Command prompt – type aws configure:**
Enter copied access key and secret access key. Else none.
- vii. **Copy terraform file path and paste in environment variable:**
Copy the path and paste it in PATH in Environment Variables.
- viii. **Check by opening cmd – terraform -v:**
Verify using this command – terraform -v
- ix. **Open Project file in VS code :**
Type 'main.tf' and insert the code from <https://github.com/terraform-aws-modules/terraform-aws-s3-bucket#>
- x. **Make a index.html file and upload it to AWS bucket**
- xi. **Copy the project path and open command prompt:**
Change directory to project folder.
- xii. **Type terraform init:** Initialize Terraform
- xiii. **Type terraform plan:** Plan Deployment
- xiv. **Type terraform apply -auto-approve:** Deploy Infrastructure
- xv. **Open the AWS website endpoint link on browser.**
- xvi. **Type terraform destroy -auto-approve:** Destroy Infrastructure



DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

Discover. Learn. Empower.

5. Screenshot:

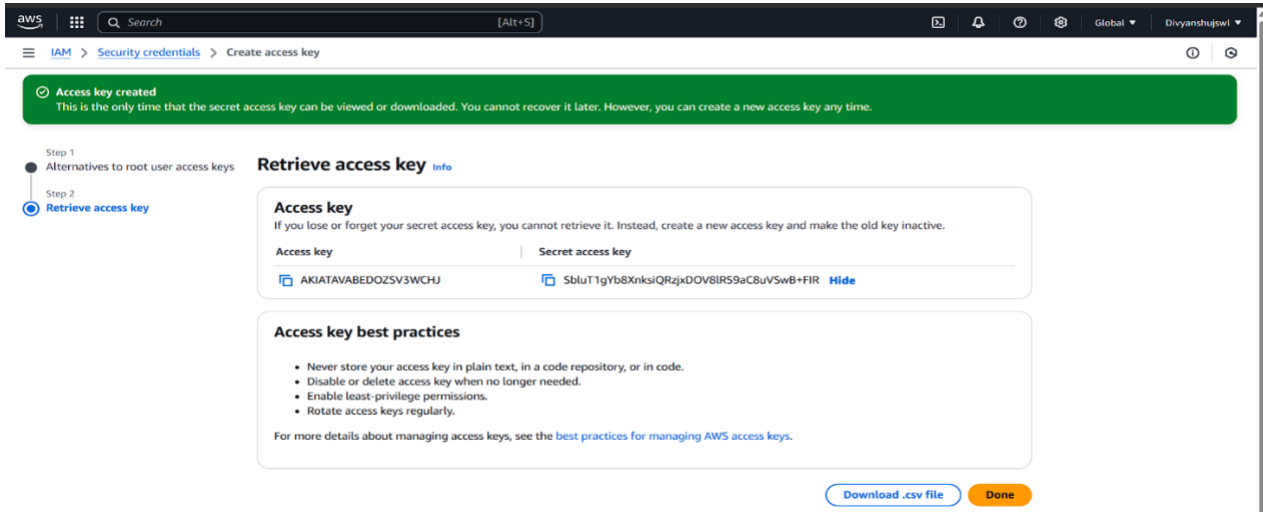


Fig. 1 Retrieve Access key

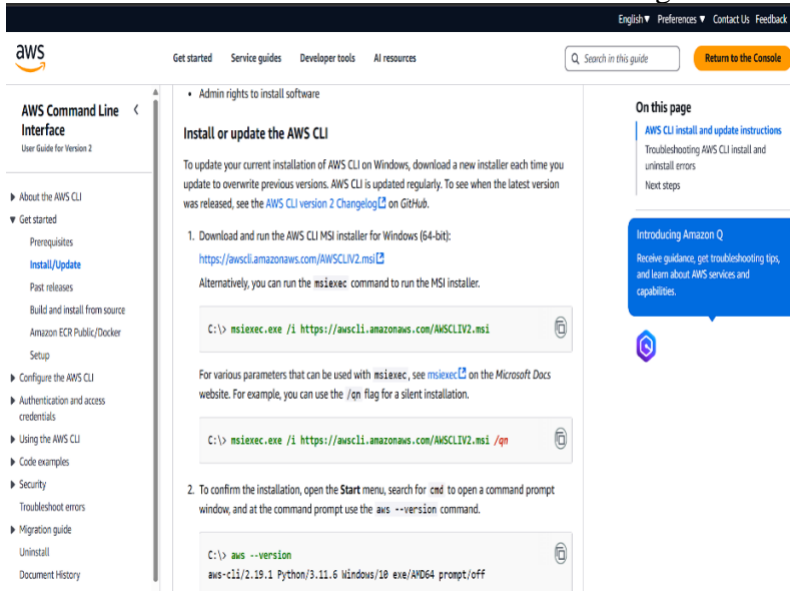


Fig. 2 Install AWS CLI

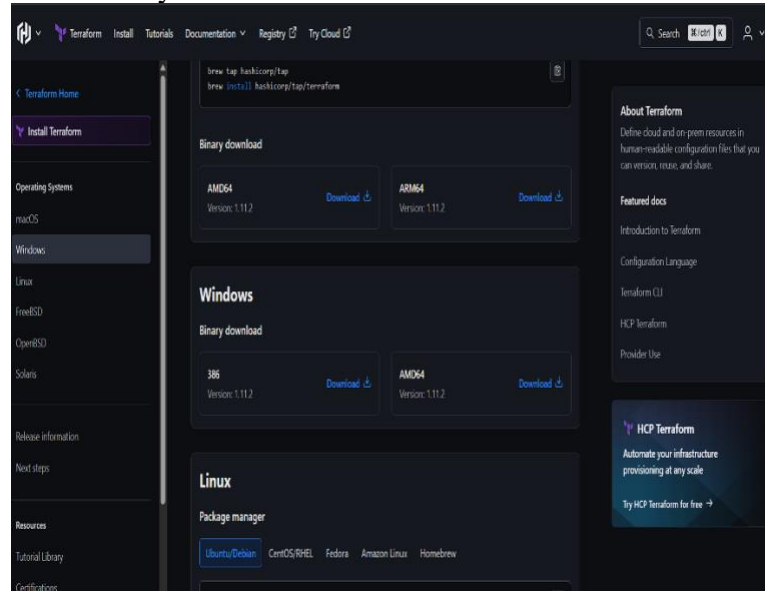


Fig. 3 Install Terraform

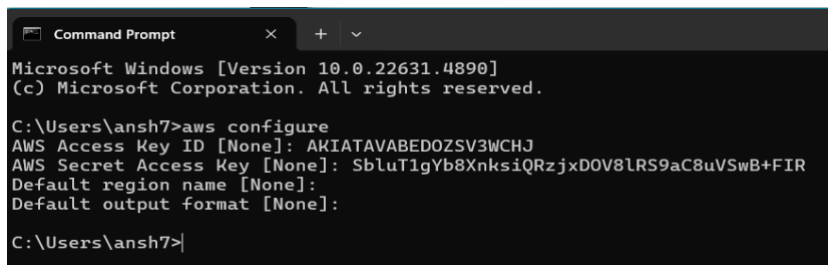


Fig. 4 Configure AWS

```

Command Prompt
Microsoft Windows [Version 10.0.22631.4890]
(c) Microsoft Corporation. All rights reserved.

C:\Users\ansh7>cd C:\Users\ansh7\Downloads\s3\terraform-aws-s3-bucket

C:\Users\ansh7\Downloads\s3\terraform-aws-s3-bucket>"C:\terraform\terraform.exe" init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching ">= 5.83.0"...
- Installing hashicorp/aws v5.91.0...
- Installed hashicorp/aws v5.91.0 (signed by HashiCorp)
Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.

C:\Users\ansh7\Downloads\s3\terraform-aws-s3-bucket>

```

Fig. 5 type 'terraform init'

```

C:\Users\ansh7\Downloads\s3\New>"C:\terraform\terraform.exe" plan

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
+ create

Terraform will perform the following actions:

# aws_s3_bucket.my_bucket will be created
+ resource "aws_s3_bucket" "my_bucket" {
+   acceleration_status = (known after apply)
+   acl                 = (known after apply)
+   arn                 = (known after apply)
+   bucket              = "divyanshu-s3-bucket-123"
+   bucket_domain_name = (known after apply)
+   bucket_prefix       = (known after apply)
+   bucket_regional_domain_name = (known after apply)
+   force_destroy       = false
+   hosted_zone_id      = (known after apply)
+   id                  = (known after apply)
+   object_lock_enabled = (known after apply)
+   policy              = (known after apply)
+   region              = (known after apply)
+   request_payer       = (known after apply)
+   tags_all            = (known after apply)
+   website_domain      = (known after apply)
+   website_endpoint    = (known after apply)
}

```

Fig. 6 type 'terraform plan'

```

Command Prompt

+ block_public_acls = true
+ block_public_policy = true
+ bucket            = (known after apply)
+ id                = (known after apply)
+ ignore_public_acls = true
+ restrict_public_buckets = true
}

# aws_s3_bucket_versioning.versioning_example will be created
+ resource "aws_s3_bucket_versioning" "versioning_example" {
+   bucket = (known after apply)
+   id     = (known after apply)

+   versioning_configuration {
+     mfa_delete = (known after apply)
+     status     = "Enabled"
+   }
}

Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:
+ bucket_name = "divyanshu-s3-bucket-123"
aws_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Creation complete after 3s [id=divyanshu-s3-bucket-123]
aws_s3_bucket_public_access_block.public_access: Creating...
aws_s3_bucket_versioning.versioning_example: Creating...
aws_s3_bucket_acl.my_bucket_acl: Creating...
aws_s3_bucket_public_access_block.public_access: Creation complete after 1s [id=divyanshu-s3-bucket-123]
aws_s3_bucket_versioning.versioning_example: Creation complete after 2s [id=divyanshu-s3-bucket-123]

```

Fig. 7 type 'terraform -auto-approve'

```
Plan: 4 to add, 0 to change, 0 to destroy.

Changes to Outputs:
  + website_url = (known after apply)
aws_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Creation complete after 5s [id=divyanshu-s3-website-bucket]
aws_s3_bucket_public_access_block.public_access: Creating...
aws_s3_bucket_policy.public_read: Creating...
aws_s3_bucket_website_configuration.website: Creating...
aws_s3_bucket_public_access_block.public_access: Creation complete after 1s [id=divyanshu-s3-website-bucket]
aws_s3_bucket_website_configuration.website: Creation complete after 1s [id=divyanshu-s3-website-bucket]
aws_s3_bucket_policy.public_read: Creation complete after 1s [id=divyanshu-s3-website-bucket]

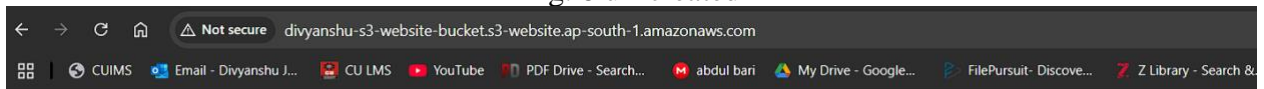
Apply complete! Resources: 4 added, 0 changed, 0 destroyed.

Outputs:

website_url = "divyanshu-s3-website-bucket.s3-website.ap-south-1.amazonaws.com"

C:\Users\ansh7\Downloads\s3\New>
```

Fig. 8 url created



Terraform S3 Bucket for Static Website Deployment

Project Overview

In this project, we use Terraform to provision an S3 bucket on AWS to host a static website. This S3 bucket will serve as the origin for our website content.

How to Use

1. Ensure you have Terraform installed on your system. If not, download and install it from <https://developer.hashicorp.com/terraform/downloads>.
2. Clone this repository to your local machine.
3. Navigate to the directory containing the Terraform configuration files.
4. Run `terraform init` to initialize the working directory.
5. Run `terraform plan` to see the execution plan.
6. Run `terraform apply` to apply the changes and create the S3 bucket.

Terraform Configuration

Below is a simplified version of the Terraform configuration (`main.tf`) used to create the S3 bucket:

```
resource "aws_s3_bucket" "cc-bucket" {
  bucket = var.bucketname
}
```

Fig. 9 online website

6. Output

1. Define Terraform configuration (`main.tf`) to set up edge computing instances.
2. Deploy with Terraform (`init`, `plan`, `apply`).
3. Automate management with Terraform modules.
4. Monitor and scale edge nodes dynamically.

7. Learning Outcomes:

1. Understand Edge Computing: Its benefits in reducing latency and improving real-time IoT processing.
2. Learn Terraform Basics: How to define infrastructure using code and automate deployment.
3. Implement Cloud Infrastructure: Deploy virtual machines and configure security settings.
4. Use Terraform Commands: `init`, `plan`, and `apply` to set up infrastructure.