

Pulkit

500125835

Lab Exercise 5—Provisioning an S3 Bucket on AWS

Exercise Steps:

Step 1: Create a New Directory:

Create a new directory to store your Terraform configuration:

```
mkdir Terraform-S3-Demo  
cd Terraform-S3-Demo
```

Step 2: Create the Terraform Configuration File (main.tf):

Create a file named main.tf with the following content:

```
terraform {  
    required_providers {  
        aws = {  
            source = "hashicorp/aws"  
            version = "5.31.0"  
        }  
    }  
}  
  
provider "aws" {  
    region    = "us-east-1" # Replace with your preferred region  
    access_key = "your IAM access key" # Replace with your Access Key  
    secret_key = "your secret access key" # Replace with your Secret Key  
}
```

This file sets up the Terraform AWS provider.

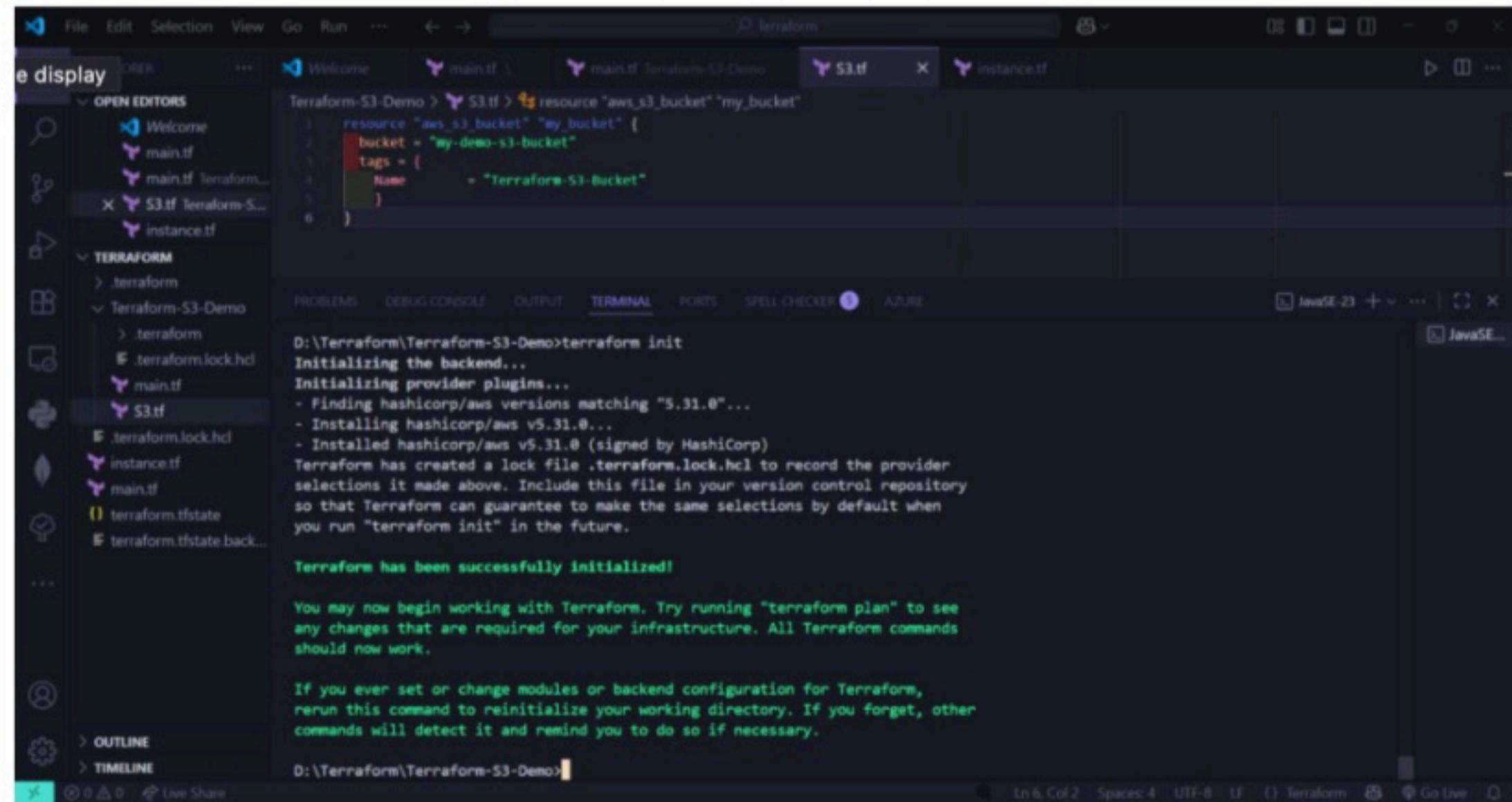
Create another file named s3.tf with the following content:

```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name      = "Terraform-S3-Bucket"
  }
}
```

This file provisions an S3 bucket with a unique name using a random string suffix.

Step 4: Initialize Terraform:

Run the following command to initialize your Terraform working directory:



```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name      = "Terraform-S3-Bucket"
  }
}

D:\Terraform\Terraform-S3-Demo>terraform init
Initializing the backend...
Initializing provider plugins...
- Finding hashicorp/aws versions matching "5.31.0"...
- Installing hashicorp/aws v5.31.0...
- Installed hashicorp/aws v5.31.0 (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.

D:\Terraform\Terraform-S3-Demo>
```

terraform init

Step 5: Review the Plan:

Preview the changes Terraform will make:

terraform plan

Review the output to ensure it meets your expectations.

```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "my-demo-s3-bucket"
  tags = {
    Name = "Terraform-S3-Bucket"
  }
}

Plan: 1 to add, 0 to change, 0 to destroy.

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
```

Step 6: Apply the Changes:

Create the resources:

```
terraform apply
```

When prompted, type yes to confirm.

```
+ logging (known after apply)
+ object_lock_configuration (known after apply)
+ replication_configuration (known after apply)
+ server_side_encryption_configuration (known after apply)
+ versioning (known after apply)
+ website (known after apply)

Plan: 1 to add, 0 to change, 0 to destroy.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Still creating... [00m10s elapsed]
aws_s3_bucket.my_bucket: Still creating... [00m20s elapsed]
aws_s3_bucket.my_bucket: Creation complete after 27s [id=my-demo-s3-bucketttttt]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
```

Step 7: Verify Resources:

The screenshot shows the AWS S3 General purpose buckets page. On the left, there's a sidebar with options like General purpose buckets, Directory buckets, Table buckets, Vector buckets, Access Grants, and Storage Lens. The main area displays a table of buckets with columns for Name, AWS Region, and Creation date. There are three buckets listed:

Name	AWS Region	Creation date
my-demo-s3-bucket	US East (N. Virginia) us-east-1	September 3, 2025, 00:31:30 (UTC+05:30)
static-website-hosting45	US East (N. Virginia) us-east-1	March 14, 2025, 10:45:49 (UTC+05:30)
uni-books	US East (N. Virginia) us-east-1	May 3, 2025, 14:12:30 (UTC+05:30)

On the right, there are two cards: 'Account snapshot' and 'External access summary - new'. The bottom of the screen includes standard AWS navigation links like CloudShell, Feedback, and cookie preferences.

1. Log in to your AWS Management Console.
2. Navigate to the **S3** dashboard.
3. Verify that the S3 bucket has been created with the specified configuration.

Step 8: Cleanup Resources:

The screenshot shows a code editor with a Terraform configuration file open. The file contains the following code:

```
resource "aws_s3_bucket" "my_bucket" {
  server_side_encryption_configuration {
    rule {
      bucket_key_enabled = false
      apply_server_side_encryption_by_default {
        sse_algorithm = "AES256"
      }
    }
  }
  versioning {
    enabled = false
    mfa_delete = false
  }
}
```

Below the code, a confirmation message from Terraform asks if the user really wants to destroy all resources. The user has entered 'yes' and the command is being run in the terminal tab of the code editor.

To remove the resources created, run the following command:

```
terraform destroy
```

When prompted, type yes to confirm.
