

Lab Exercise 5—Provisioning an S3 Bucket on AWS

Exercise Steps:

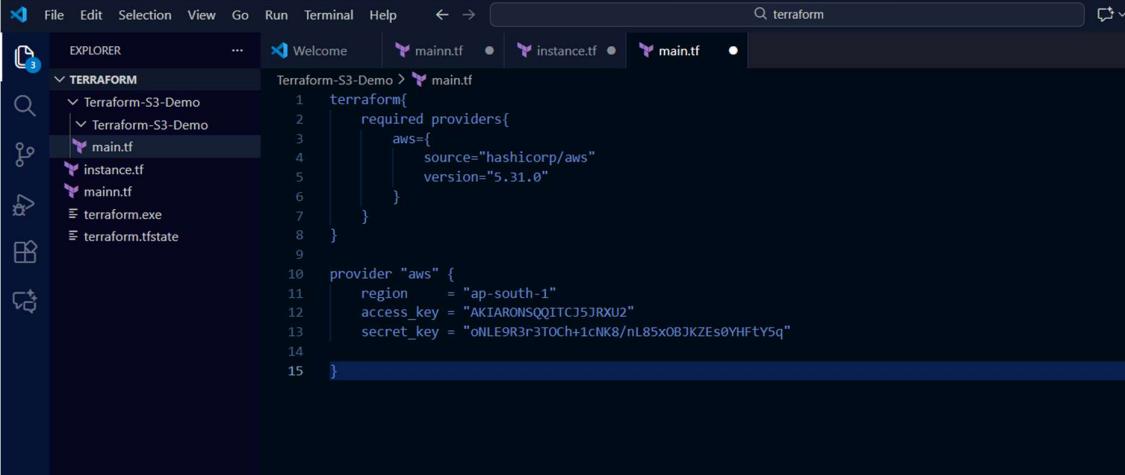
Step 1: Create a New Directory:

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:

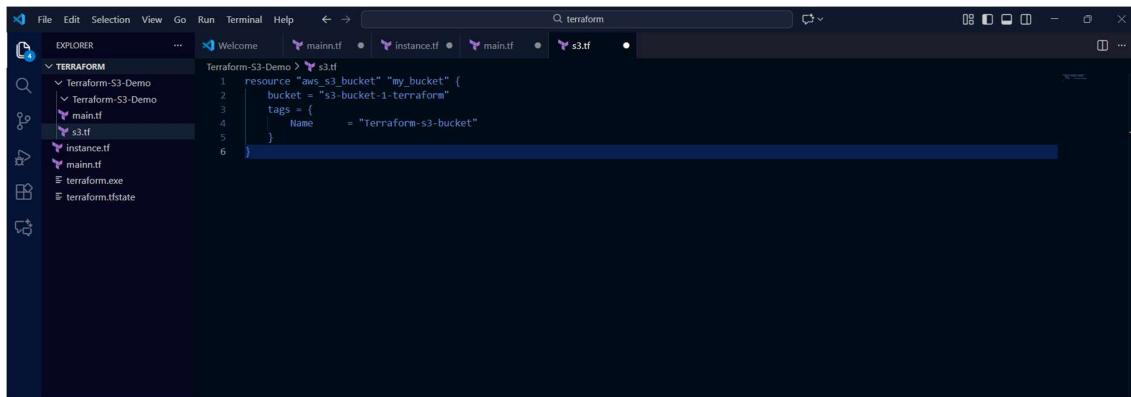


```
File Edit Selection View Go Run Terminal Help ⏪ ⏩ Q: terraform  
EXPLORER TERRAFORM Welcome Terraform-S3-Demo > main.tf  
Terraform-S3-Demo  
└── main.tf  
    └── mainn.tf  
        └── instance.tf  
        └── mainn.tf  
    └── terraform.exe  
    └── terraform.tfstate  
  
Terraform-S3-Demo > main.tf  
1  terraform{  
2      required_providers{  
3          aws={  
4              source="hashicorp/aws"  
5              version="5.31.0"  
6          }  
7      }  
8  }  
9  
10 provider "aws" {  
11     region      = "ap-south-1"  
12     access_key  = "AKIARONSQJTCJ5JRXU2"  
13     secret_key  = "oNLE9R3r3TOCh+1cNK8/nL85x0BJKZE0YHftY5q"  
14 }  
15 }
```

This file sets up the Terraform AWS provider.

Step 3: Create a Terraform Configuration File for the S3 Bucket (s3.tf):

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



```
resource "aws_s3_bucket" "my_bucket" {
  bucket = "s3-bucket-1-terraform"
  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:

```
terraform init
```

Step 5: Review the Plan:

Preview the changes Terraform will make:

```
terraform plan
```

Output:

```
Terminal Help ← → ⓘ terraform-demo | PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraform-demo> terraform plan
aws_instance.my_instance: Refreshing state... [id=0eeefb50d8d38c841e]
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              = "s3-bucket-1-terraform"
    bucket_domain_name = (known after apply)
    bucket_prefix       = (known after apply)
    bucketRegionalDomainName = (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               = {
        "Name" = "Terraform-S3-Bucket"
    }
    tags_all           = {
        "Name" = "Terraform-S3-Bucket"
    }
    website_domain     = (known after apply)
    website_endpoint   = (known after apply)

    cors_rule (known after apply)
    grant (known after apply)
    lifecycle_rule (known after apply)
    logging (known after apply)
}
```

```
Terminal Help ← → ⓘ terraform-demo | PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraform-demo>
aws_instance.my_instance: Refreshing state... [id=0eeefb50d8d38c841e]
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              = "s3-bucket-1-terraform"
    bucket_domain_name = (known after apply)
    bucket_prefix       = (known after apply)
    bucketRegionalDomainName = (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               = {
        "Name" = "Terraform-S3-Bucket"
    }
    tags_all           = {
        "Name" = "Terraform-S3-Bucket"
    }
    website_domain     = (known after apply)
    website_endpoint   = (known after apply)

    cors_rule (known after apply)
    grant (known after apply)
    lifecycle_rule (known after apply)
    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.

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.

Output:

A screenshot of a terminal window titled "terraformer-demo". The window shows the command "PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo> terraform apply" followed by the output of the "terraform apply" command. The output details the actions Terraform will perform to create an AWS S3 bucket named "my_bucket". The actions listed include creating the bucket, setting its name, and applying various configuration parameters like acceleration_status, acl, cors_rule, grant, lifecycle_rule, logging, and object_lock_configuration. The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, and a sidebar with options for powershell and powershell.

```
PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo> terraform apply
aws_instance.my_instance: Refreshing state... [id=deefb50d33c84a16]

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)
  + cors                    = (known after apply)
  + id                      = "s3-bucket-1-terraformer"
  + bucket_domain_name       = (known after apply)
  + bucket_prefix             = (known after apply)
  + bucketRegionalDomainName = (known after apply)
  + force_destroy            = false
  + hostedZoneId             = (known after apply)
  + id                       = (known after apply)
  + objectLockEnabled         = (known after apply)
  + policy                   = (known after apply)
  + region                   = (known after apply)
  + requestPayer              = (known after apply)
  + tags                     = {
      + "Name" = "Terraform-S3-Bucket"
    }
  + tags_all                 = {
      + "Name" = "Terraform-S3-Bucket"
    }
  + website_domain           = (known after apply)
  + website_endpoint          = (known after apply)

  + cors_rule (known after apply)
  + grant (known after apply)
  + lifecycle_rule (known after apply)
  + logging (known after apply)
  + object_lock_configuration (known after apply)
}
```

A screenshot of a terminal window titled "terraformer-demo". The window shows the command "PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo> terraform apply" followed by the execution plan and confirmation prompt. The output shows the creation of an AWS S3 bucket named "my_bucket" with the specified configuration. The terminal interface includes tabs for PROBLEMS, OUTPUT, DEBUG CONSOLE, and TERMINAL, and a sidebar with options for powershell and powershell.

```
PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo> terraform apply
aws_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Creation complete after 5s [id=s3-bucket-1-terraformer]

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

Enter a value: yes

aws_s3_bucket.my_bucket: Creating...
aws_s3_bucket.my_bucket: Creation complete after 5s [id=s3-bucket-1-terraformer]

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

PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo>
```

Step 7: Verify Resources:

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.

Output:

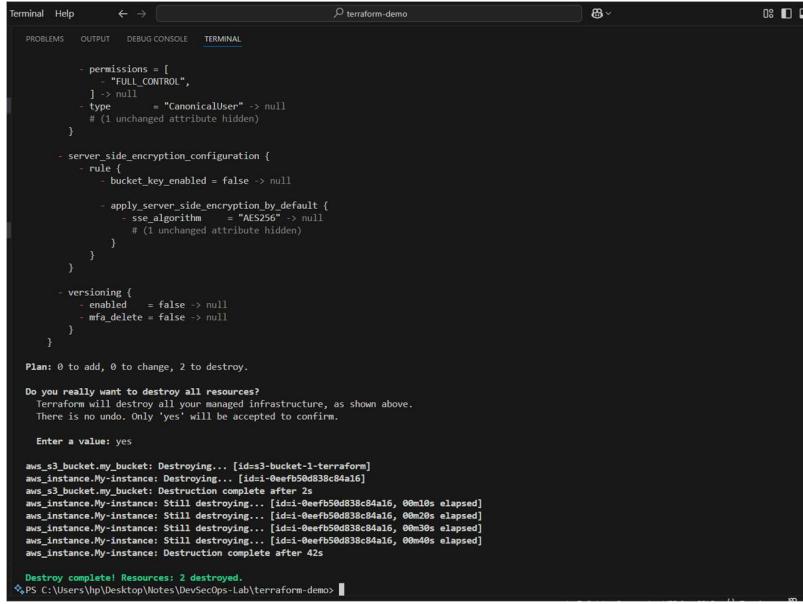
The screenshot shows the AWS S3 General purpose buckets dashboard. On the left, there's a sidebar with options like 'General purpose buckets', 'Directory buckets', 'Table buckets', etc. The main area shows a table for 'General purpose buckets (1)'. The table has columns for 'Name', 'AWS Region', and 'Creation date'. One row is listed: 's3-bucket-1-terraform' (Asia Pacific (Mumbai) ap-south-1, September 9, 2025, 22:45:38 (UTC+05:30)). There are buttons for 'Copy ARN', 'Empty', 'Delete', and 'Create bucket'. To the right of the table are two cards: 'Account snapshot' (updated daily) and 'External access summary - new' (updated daily). The URL in the address bar is https://ap-south-1.console.aws.amazon.com/s3/buckets/s3-bucket-1-terraform?region=ap-south-1.

Step 8: Cleanup Resources:

To remove the resources created, run the following command:

```
terraform destroy
```

When prompted, type yes to confirm.



The screenshot shows a terminal window titled "terraformer-demo" with the following content:

```
Terminal Help ← → terraformer-demo PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
permissions = [
  ] => null
  type      = "CanonicalUser" => null
  # (1 unchanged attribute hidden)
}

server_side_encryption_configuration {
  rule {
    bucket_key_enabled = false => null
    apply_server_side_encryption_by_default {
      sse_algorithm = "AES256" => null
      # (1 unchanged attribute hidden)
    }
  }
}

versioning {
  enabled     = false => null
  mfa_delete  = false => null
}
}

Plan: 0 to add, 0 to change, 2 to destroy.

Do you really want to destroy all resources?
Terraform will destroy all your managed infrastructure, as shown above.
There is no undo. Only 'yes' will be accepted to confirm.

Enter a value: yes

aws_s3_bucket.my_bucket: Destroying... [id:s3-bucket-1-terraform]
aws_instance.My-instance: Destroying... [id:i-0eef50d838c84a16]
aws_s3_bucket.my_bucket: Destruction complete after 2s
aws_instance.My-instance: Still destroying... [id:i-0eef50d838c84a16, 00m10s elapsed]
aws_instance.My-instance: Still destroying... [id:i-0eef50d838c84a16, 00m20s elapsed]
aws_instance.My-instance: Still destroying... [id:i-0eef50d838c84a16, 00m30s elapsed]
aws_instance.My-instance: Still destroying... [id:i-0eef50d838c84a16, 00m40s elapsed]
aws_instance.My-instance: Destruction complete after 42s

Destroy complete! Resources: 2 destroyed.
PS C:\Users\hp\Desktop\Notes\DevSecOps-Lab\terraformer-demo>
```

Resources destroyed.
