



System Provisioning and Configuration Management LAB

SUBMITTED TO

Dr. Hitesh Kumar Sharma

SUBMITTED BY

Siddharth Agarwal

500107594

R2142220663

Btech CSE DevOps B1

Lab Exercise 2– Terraform AWS Provider and IAM User Setting

Prerequisites: Terraform Installed: Make sure you have Terraform installed on your machine. Follow the official installation guide if needed.

AWS Credentials: Ensure you have AWS credentials (Access Key ID and Secret Access Key) configured. You can set them up using the AWS CLI or by setting environment variables.

Exercise Steps:

Step 1: Create a New Directory:

Create a new directory for your Terraform configuration:

```
mkdir aws-terraform-demo  
  
cd aws-terraform-demo
```

Step 2: Create 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    = "ap-south-1"

  access_key = "your IAM access key"

  secret_key = "your secret access key"

}

```

The screenshot shows the AWS IAM console 'Create user' wizard, specifically the 'Review and create' step. The user name is 'sid-terraform-lab'. The console password type is 'None' and 'Require password reset' is 'No'. The permissions summary shows 'AdministratorAccess' as the policy. The 'Tags' section is optional and currently empty. At the bottom, there are 'Cancel', 'Previous', and 'Create user' buttons.

Review and create
Review your choices. After you create the user, you can view and download the autogenerated password, if enabled.

User details

User name sid-terraform-lab	Console password type None	Require password reset No
--------------------------------	-------------------------------	------------------------------

Permissions summary

Name	Type	Used as
AdministratorAccess	AWS managed - job function	Permissions policy

Tags - optional
Tags are key-value pairs you can add to AWS resources to help identify, organize, or search for resources. Choose any tags you want to associate with this user.
No tags associated with the resource.
[Add new tag](#)
You can add up to 50 more tags.

Cancel Previous **Create user**

```

main.tf
lab1 > 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    = "us-east-1"
12   access_key = "AKIA2BRNT5GDCGJAXCPZ"
13   secret_key = "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"
14 }
15

```

This script defines an AWS provider and provisions an EC2 instance.

Step 3: Initialize Terraform:

Run the following command to initialize your Terraform working directory:

```
terraform init
```

```
● PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SP\lab\lab1> 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.
○ PS C:\SID_DATA\SIDDHARTH\UPES COLLEGE STUDY MATERIAL\SEM6\SP\lab\lab1> █
```