

ASSIGNMENT 1

**Write Terraform script to do perform following tasks on
AWS cloud Platform**

Name: Aditya Tomar

SAP:500106015

R.NO=R2142221060

Batch:DevOps(2 NH)

In this assignment, terraform — an open-source Infrastructure as Code (IaC) tool — is used to automate the provisioning of cloud resources on Amazon Web Services (AWS). The goal is to write a reusable, version-controlled script that sets up a basic cloud infrastructure in a matter of seconds.

The tasks covered in this assignment include:

- **Creating two EC2 instances of type t2.micro**
- **Provisioning a Customer Gateway and a VPN Gateway to simulate a VPN setup**
- **Creating an S3 bucket for cloud-based storage**

1. Creating IAM User

An IAM user named terraform-assign was created in the AWS Console with programmatic access. This user will be used to authenticate Terraform with AWS for infrastructure provisioning.

Step 1

Step 2

Step 3

Step 4

Specify user details

Set permissions

Review and create

Retrieve password

Specify user details

User details

User name

terraform-assign

The user name can have up to 64 characters. Valid characters: A-Z, a-z, 0-9, and +, -, @, _ (hyphen)

☒ Provide user access to the AWS Management Console - optional

If you're providing console access to a person, it's a [best practice](#) to manage their access in IAM Identity Center.

Are you providing console access to a person?

User type

☐ Specify a user in Identity Center - Recommended

We recommend that you use Identity Center to provide console access to a person. With Identity Center, you can centrally manage user access to their AWS accounts and cloud applications.

☒ I want to create an IAM user

We recommend that you create IAM users only if you need to enable programmatic access through access keys, service-specific credentials for AWS CodeCommit or Amazon Keyspaces, or a backup credential for emergency account access.

Console password

☐ Autogenerated password

You can view the password after you create the user.

☒ Custom password

Enter a custom password for the user.

☐ Show password

☒ Users must create a new password at next sign-in - Recommended

Users automatically get the [IAMUserChangePassword](#) policy to allow them to change their own password.

User created successfully

You can view and download the user's password and email instructions for signing in to the AWS Management Console.

View user

Step 1

Step 2

Step 3

Step 4

Specify user details

Set permissions

Review and create

Retrieve password

Retrieve password

You can view and download the user's password below or email users instructions for signing in to the AWS Management Console. This is the only time you can view and download this password.

Console sign-in details

Console sign-in URL

<https://248189932992.signin.aws.amazon.com/console>

User name

terraform-assign

Console password

***** [Show](#)

Cancel

Download .csv file

Return to users list

2. Creating Access Key for IAM User

An Access Key and Secret Access Key were generated for the IAM user. These credentials were configured using the `aws configure` command to securely connect Terraform with the AWS account.

Step 1

Access key best practices & alternatives

Step 2 - optional

Set description tag

Step 3

Retrieve access keys

Access key best practices & alternatives [Info](#)

Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.

Use case

☐ Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.

☐ Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.

☐ Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.

☐ Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.

☐ Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.

☒ Other
Your use case is not listed here.

It's okay to use an access key for this use case, but follow the best practices:

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access keys when no longer needed.
- Enable least-privilege permissions.

Step 1

Access key best practices & alternatives

Step 2 - optional

Set description tag

Step 3

Retrieve access keys

Access key created

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Retrieve access keys [Info](#)

Access key

If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key

Secret access key

AKIATTSKF2XABY2UE6VS

Show

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

Download .csv file

Done

3. Writing Terraform Code in VS Code

The Terraform configuration was written in VS Code. It includes resources for EC2 instances, a VPN setup, and an S3 bucket — all defined as Infrastructure as Code in a single main.tf file.

```
main.tf > resource "aws_s3_bucket" "bucket"
4
5 # Step 1: Create 2 EC2 instances
6 resource "aws_instance" "example1" {
7     ami           = "ami-0c55b159cbf1f0" # Amazon Linux 2 (Update if needed)
8     instance_type = "t2.micro"
9     tags = {
10         Name = "Instance-1"
11     }
12 }
13
14 resource "aws_instance" "example2" {
15     ami           = "ami-0c55b159cbf1f0"
16     instance_type = "t2.micro"
17     tags = {
18         Name = "Instance-2"
19     }
20 }
21
22 # Step 2: Create VPN resources
23 resource "aws_customer_gateway" "vpn" {
24     bgp_asn     = 65000
25     ip_address  = "1.1.1.1" # Replace with real IP or dummy for now
26     type        = "ipsec.1"
27     tags = {
28         Name = "VPN-CGW"
29     }
30 }
31
32 resource "aws_vpn_gateway" "vpn" {
33     amazon_side_asn = 64512
34     tags = {
35         Name = "VPN-GW"
36     }
37 }
38
39 # Step 3: Create an S3 bucket
40 resource "aws_s3_bucket" "bucket" {
41     bucket = "your-unique-bucket-name-2025" # Must be globally unique
42     tags = {
43         Name        = "AssignmentBucket"
44         Environment = "Dev"
45     }
46 }
```

4. Running terraform init, plan, and apply

Terraform CLI commands were used to initialize the working directory, preview the execution plan, and apply the infrastructure changes. This screenshot shows the commands being executed in the terminal.

```
adityatomar@Adityas-MacBook-Air-3 Assignment-1 % terraform init
Initializing the backend...
Initializing provider plugins...
- Reusing previous version of hashicorp/aws from the dependency lock file
- Using previously-installed hashicorp/aws v5.94.1

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.
]

adityatomar@Adityas-MacBook-Air-3 Assignment-1 % terraform apply
aws_vpn_gateway.vpn: Refreshing state... [id=vgw-04f0f5bd211dd5158]
aws_customer_gateway.vpn: Refreshing state... [id=cgw-0b039eaa40ec05088]

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_instance.example1 will be created
+ resource "aws_instance" "example1" {
  + ami              = "ami-051f7e7f6c2f40dc1"
  + arn              = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)
  + cpu_core_count   = (known after apply)
  + cpu_threads_per_core = (known after apply)
  + disable_api_stop  = (known after apply)
  + disable_api_termination = (known after apply)
  + ebs_optimized     = (known after apply)
  + enable_primary_ipv6 = (known after apply)
  + get_password_data  = false
  + host_id           = (known after apply)
}

Enter a value: yes

aws_s3_bucket.bucket: Creating...
aws_instance.example1: Creating...
aws_instance.example2: Creating...
aws_instance.example2: Still creating... [10s elapsed]
aws_instance.example1: Still creating... [10s elapsed]
aws_instance.example1: Creation complete after 16s [id=i-016c3b803027bdcd7]
aws_instance.example2: Creation complete after 16s [id=i-00786ee9327822733]
```

5. AWS Console Showing Resources Created

This screenshot captures the AWS Console showing all the resources created — including EC2 instances, VPN components, and the S3 bucket — confirming successful provisioning using Terraform.

Instance-2

i-00786ee9327822733

Running

t2.micro

2/2 checks passed

View alarms +

us-east-1a

ec2-

Instance-1

i-016c3b803027bdcd7

Running

t2.micro

2/2 checks passed

View alarms +

us-east-1a

ec2-

Account snapshot - updated every 24 hours

All AWS Regions

View Storage Lens dashboard

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. [Learn more](#)

General purpose buckets

Directory buckets

General purpose buckets (1)

Info

All AWS Regions

Refresh

Copy ARN

Empty

Delete

Create bucket

Buckets are containers for data stored in S3.

Find buckets by name

< 1 >

Settings

Name	AWS Region	IAM Access Analyzer	Creation date
aditya-assignment-bucket-2025	US East (N. Virginia) us-east-1	View analyzer for us-east-1	April 11, 2025, 00:41:08 (UTC+05:30)

Customer gateways (1/1)

Info

Refresh

Actions

Create customer gateway

CustomerGateways

Source by attribute or tag

< 1 >

Settings

Name	Customer gateway ID	State	BGP ASN	IP address	Type
VPN-CGW	cgw-0b039eaa40ec05088	Available	65000	1.1.1.1	ipsec.1

Customer gateway cgw-0b039eaa40ec05088 / VPN-CGW

Details

Tags

Details

Customer gateway ID

cgw-0b039eaa40ec05088

BGP ASN

65000

State

Available

Certificate ARN

-

Type

ipsec.1

Device

-

IP address

1.1.1.1