

**Academic Year: 2022-23 Semester: V Class / Branch: TE IT**

**Subject: Advanced Devops Lab (ADL)**

**Subject Lab Incharge: Prof. Manjusha K.**

**EXPERIMENT NO. 06**

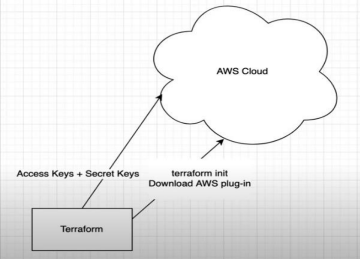
**Aim: To Build, change, and destroy AWS infrastructure Using Terraform.**

**Theory:**

Hashicorp's Terraform is an open-source tool for provisioning and managing cloud infrastructure. Terraform can provision resources on any cloud platform.

Terraform allows you to create infrastructure in configuration files(**tf files**) that describe the topology of cloud resources. These resources include virtual machines, storage accounts, and networking interfaces.

We will see how you can use Terraform to provision EC2 instance. Please do the below steps for provisioning EC2 instances on AWS:



**Department of Information Technology | APSIT**

****

**Pre-requistes:**

**1. Install the AWS CLI version 2 on Linux**

Follow these steps from the command line to install the AWS CLI on Linux.

**Install curl on linux**

****

**vishal@apsit:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86\_64.zip" -o "awscliv2.zip"**

****

**vishal@apsit:~$ sudo apt install unzip**

****

**vishal@apsit:~$ sudo unzip awscliv2.zip**

****

**vishal@apsit:~$ sudo ./aws/install**

**vishal@apsit:~$ aws --version**

it should display the below outout.

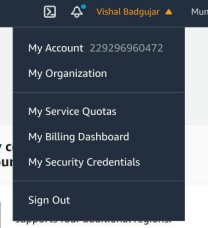
**aws-cli/2.1.29 Python/3.8.8 Linux/5.4.0-1038-aws exe/x86\_64.ubuntu.18 prompt/off**

**Department of Information Technology | APSIT**

****

**2. Create a new access key if you don't have one. Make sure you download the keys in your local machine.**

Login to AWS console, click on username and go to My security credentials.



Continue on security credentials, click on access keys

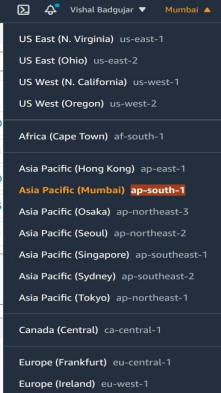
**Department of Information Technology | APSIT**

**Perform below commands in Linux where you have installed Terraform**

First setup your access keys, secret keys and region code locally.

**vishal@apsit:~$aws configure**

****You can check region as shown in below image :



**Department of Information Technology | APSIT**

****Create one Directory for Terraform project in which all files of terraform we can save

**vishal@apsit:~$ cd ~**

**vishal@apsit:~$ mkdir project-terraform**

**vishal@apsit:~$ cd project-terraform**

****

**Create Terraform Files**

**vishal@apsit:~$ sudo nano variables.tf**

In order to provide key name in variables first create key pair as shown:



**Department of Information Technology | APSIT**

****

Give name to key pair file as **terraform**

****Key pair is generated

**Department of Information Technology | APSIT**

****

Use your Region and Key name in variable.tf as shown and provide instance type which you want to create.



After creating variable terraform file note down the AMI ID of instance which u want to create which we will use to configure our instance in main.tf file.

**Department of Information Technology | APSIT**

****

**Now create main.tf file:**

****provider "aws" {

region = var.aws\_region

}

#Create security group with firewall rules

resource "aws\_security\_group" "security\_jenkins\_port" {

name = "security\_jenkins\_port"

description = "security group for jenkins"

ingress {

from\_port = 8080

to\_port = 8080

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

ingress {

from\_port = 22

to\_port = 22

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

# outbound from jenkis server

**Department of Information Technology | APSIT**

****

egress {

from\_port = 0

to\_port = 65535

protocol = "tcp"

cidr\_blocks = ["0.0.0.0/0"]

}

tags= {

Name = "security\_jenkins\_port"

}

}

resource "aws\_instance" "myFirstInstance" {

ami = "ami-0b9064170e32bde34"

key\_name = var.key\_name

instance\_type = var.instance\_type

security\_groups= [ "security\_jenkins\_port"]

tags= {

Name = "jenkins\_instance"

}

}

# Create Elastic IP address

resource "aws\_eip" "myFirstInstance" {

vpc = true

instance = aws\_instance.myFirstInstance.id

**Department of Information Technology | APSIT**

****

tags= {

Name = "jenkins\_elstic\_ip"

}

}

Put AMI-ID in above highlighted space and Now execute the below command:

you should see like below screenshot.

**Department of Information Technology | APSIT**

****

**Execute the below command**

****

the above command will show how many resources will be added.

Plan: 3 to add, 0 to change, 0 to destroy.

**Execute the below command**

****Provide the value as Yes for applying terraform

**Department of Information Technology | APSIT**

****

Plan: 3 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

Apply complete! Resources: 3 added, 0 changed, 0 destroyed.



Now login to EC2 console, to see the new instances up and running, you can see Jenkins\_instance is up and running which we deploy from terraform.

**Department of Information Technology | APSIT**

****You can also check the security group resource details which you created from terraform : 

**Terraform destroy**

you can also destroy or delete your instance by using terraform destroy command :

**Department of Information Technology | APSIT**

****

Now you can see instance which you created by using terraform is deleted successfully from aws console also you can check it will removed successfully:



All the Resources including Security groups, EC2 instances using terraform will be deleted. In this way we can automate infrastructure set up using terrform in aws cloud.

**Conclusion: Write your own findings.**

**Department of Information Technology | APSIT**