Semester	T.E. Semester V – Information Technology
Subject	Advance DevOps Lab
Subject Professor In-charge	Prof. Indu Anoop
Laboratory	(Leave blank for now)

Student Name	Rahul Chougule	
Roll Number	20101A0055	
Grade and Subject Teacher's Signature		

Experiment	5	
Problem Statement	To understand terraform lifecycle, core concepts/terminologies and install it on a Linux Machine	
Resources / Apparatus Required	Hardware: Computer System	Software: Web Browser
Details	**Terraform** Terraform is an infrastructure as code (IaC) tool that allows you to build, change, and version infrastructure safely and efficiently. This includes low-level components such as compute instances, storage, and networking, as well as high-level components such as DNS entries, SaaS features, etc. Terraform can manage both existing service providers and custom in-house solutions. **Key Features** **Infrastructure as Code**: You describe your infrastructure using Terraform's high-level configuration language in human-readable, declarative configuration files. This allows you to create a blueprint that you can version, share, and reuse.	

Resource Graph

Terraform builds a resource graph and creates or modifies non-dependent resources in parallel. This allows Terraform to build resources as efficiently as possible and gives you greater insight into your infrastructure.

Change Automation

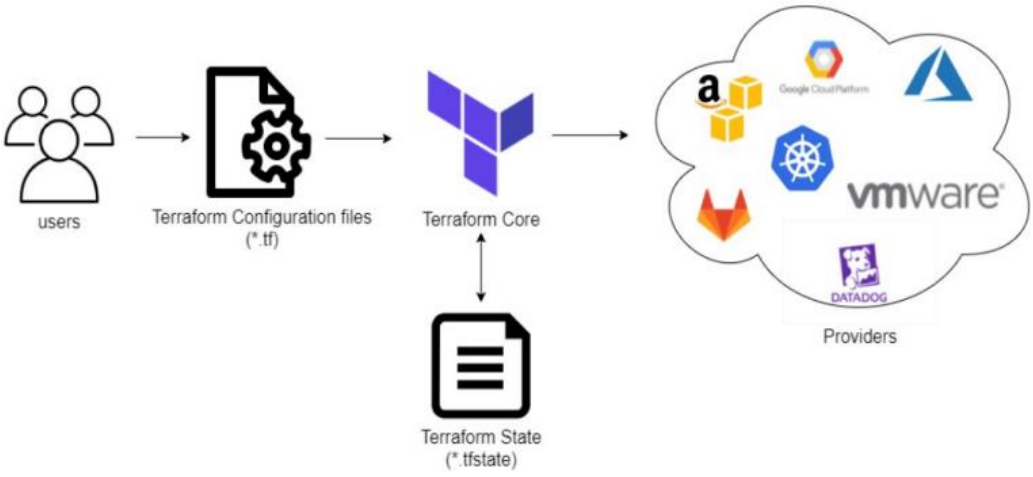
Terraform can apply complex changesets to your infrastructure with minimal human interaction. When you update configuration files, Terraform determines what changed and creates incremental execution plans that respect dependencies.

How does Terraform work?

Terraform works with two major components:

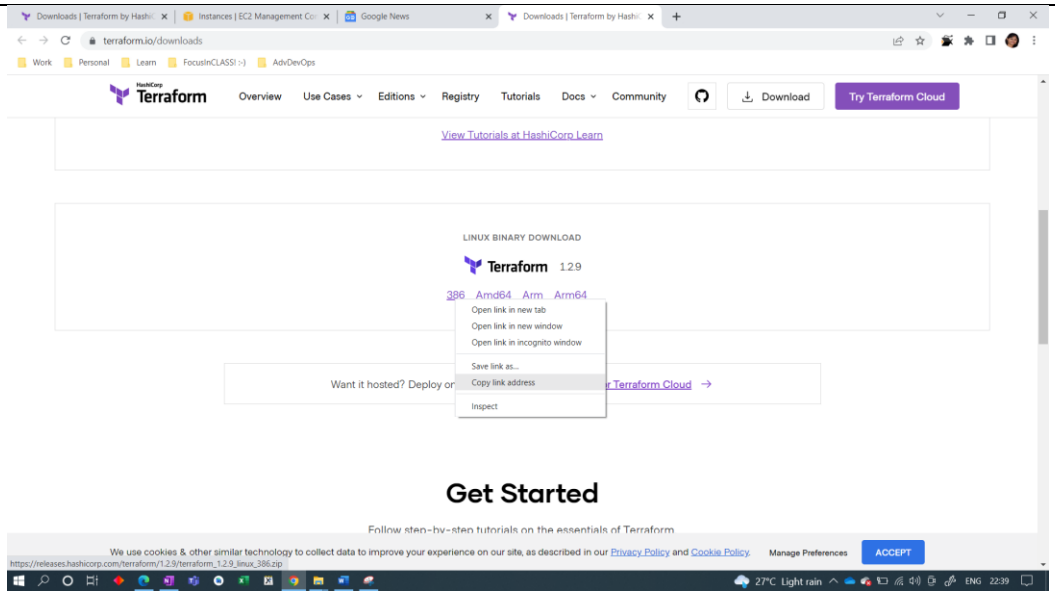
one is the **terraform core**: it takes the terraform configuration which is being provided by the user and then takes the terraform state which is managed by terraform itself. As such, this gets fed into the core that is responsible for figuring out what is that graph of our different resources for example how these different pieces relate to each other or what needs to be created/updated/destroyed, it does all the essential lifecycle management.

On the backside, terraform supports many different **providers**, such as: cloud providers (AWS,GCP,AZURE) and they also could be on-premise infrastructure (VMware, OpenStack.) But this support is not restricted or limited only to Infrastructure As A Service , terraform can also manage higher level like Platform As A Service(Kubernetes, Lambdas..)or even Software As A Service (DataDog, GitHub..)



All of these are important pieces of the infrastructure, they are all part of the logical end-to-end delivery.

	Terraform has over a hundred providers for different technologies, and each provider gives terraform users access to their resources. It also gives you the ability to create infrastructure at different levels. **Terraform Core Concepts:** Below are the core concepts/terminologies used in Terraform: •**Variables:** Also used as input-variables, it is a key-value pair used by Terraform modules to allow customization. •**Provider:** It is a plugin to interact with APIs of service and access its related resources. •**Module:** It is a folder with Terraform templates where all the configurations are defined •**State:** It consists of cached information about the infrastructure managed by Terraform and its related configurations. •**Resources**: It refers to a block of one or more infrastructure objects (compute instances, virtual networks, etc.), which are used in configuring and managing the infrastructure. •**Data Source**: It is implemented by providers to return information on external objects to terraform. •**Output Values**: These are return values of a terraform module that can be used by other configurations. •**Plan:** It is one of the stages where it determines what needs to be created, updated, or destroyed to move from the real/current state of the infrastructure to the desired state. •**Apply:** It is one of the stages where it applies the changes in the real/current state of the infrastructure to move to the desired state.
Code	Terraform Installation Steps on Ubuntu20 **Step: 1** (Ensure root user privileges)Terraform uses HashiCorp Configuration Language (HCL) to manage environments of Operators and Infrastructure teams. To **download** go to site https://www.terraform.io/downloads.html Select the appropriate package for your operating system and architecture. Copy link address



Use wget command to download from the link

wget https://releases.hashicorp.com/terraform/1.2.9/terraform_1.2.9_linux_386.zip

Step:2 unzip the archive by using below command

unzip terraform_1.2.9_linux_386.zip

The archive will extract a single binary called **terraform**
and Move the terraform binary to a directory included in your system's PATH in my case usr/local/bin/

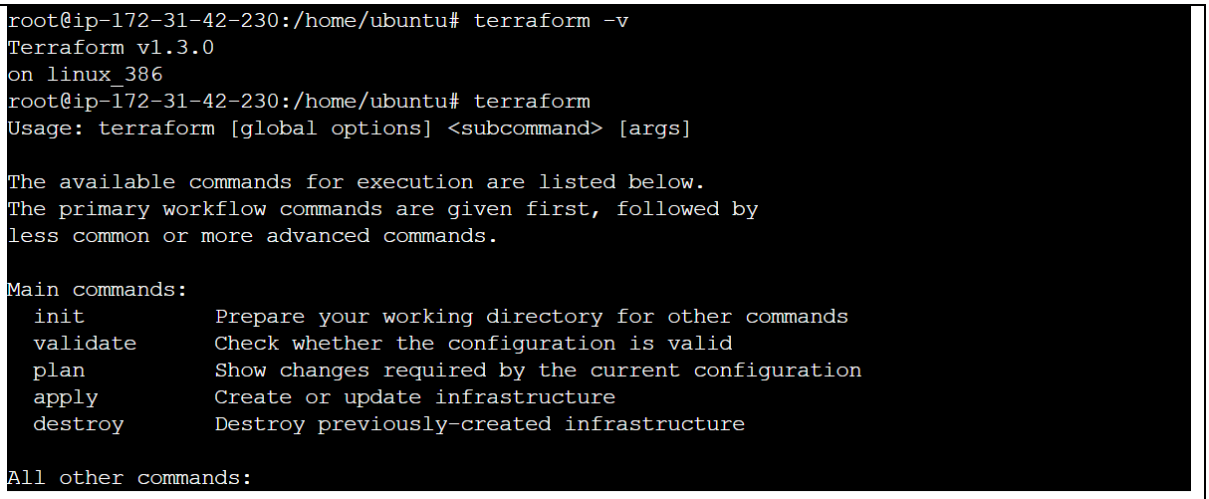
mv terraform /usr/local/bin/

Step 4: To check whether Terraform is installed, run:

terraform -v

or
type

terraform
to see options list for terraform

	
Conclusion	Successfully understood terraform and it's installation