# Terraform

# Table of Contents

1

## What is Terraform?

# What is Terraform?

- **Terraform** is the infrastructure as code offering from HashiCorp.

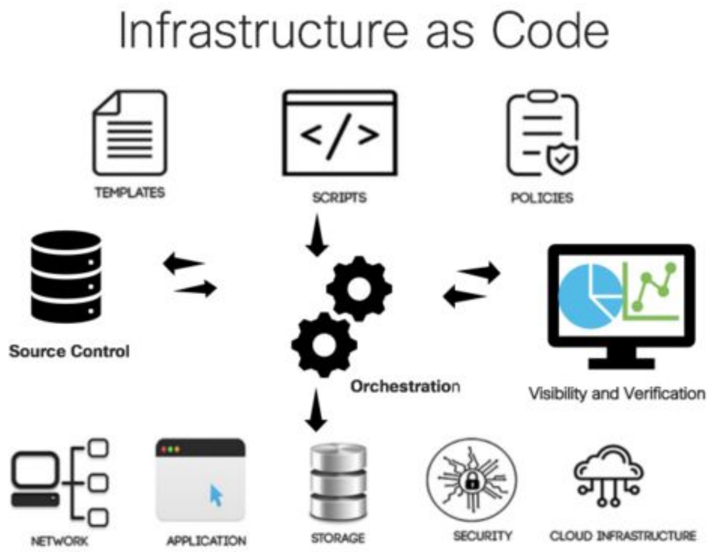
- It is a tool for building, changing, and managing infrastructure in a safe, repeatable way.

2

## What is Infrastructure as Code?

# What is Infrastructure as Code (IaC)?

- **IaC** is the process of managing infrastructure in a file or files rather than manually configuring resources in a user interface.



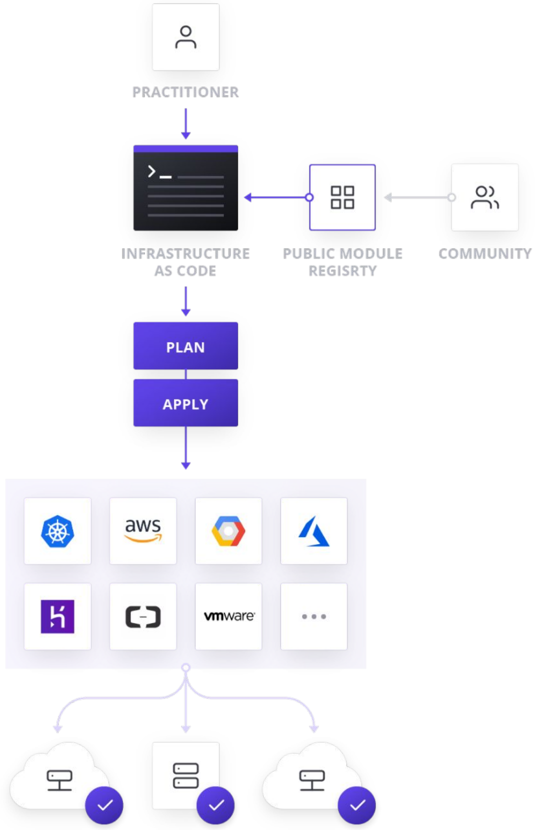
3

## How Terraform Works

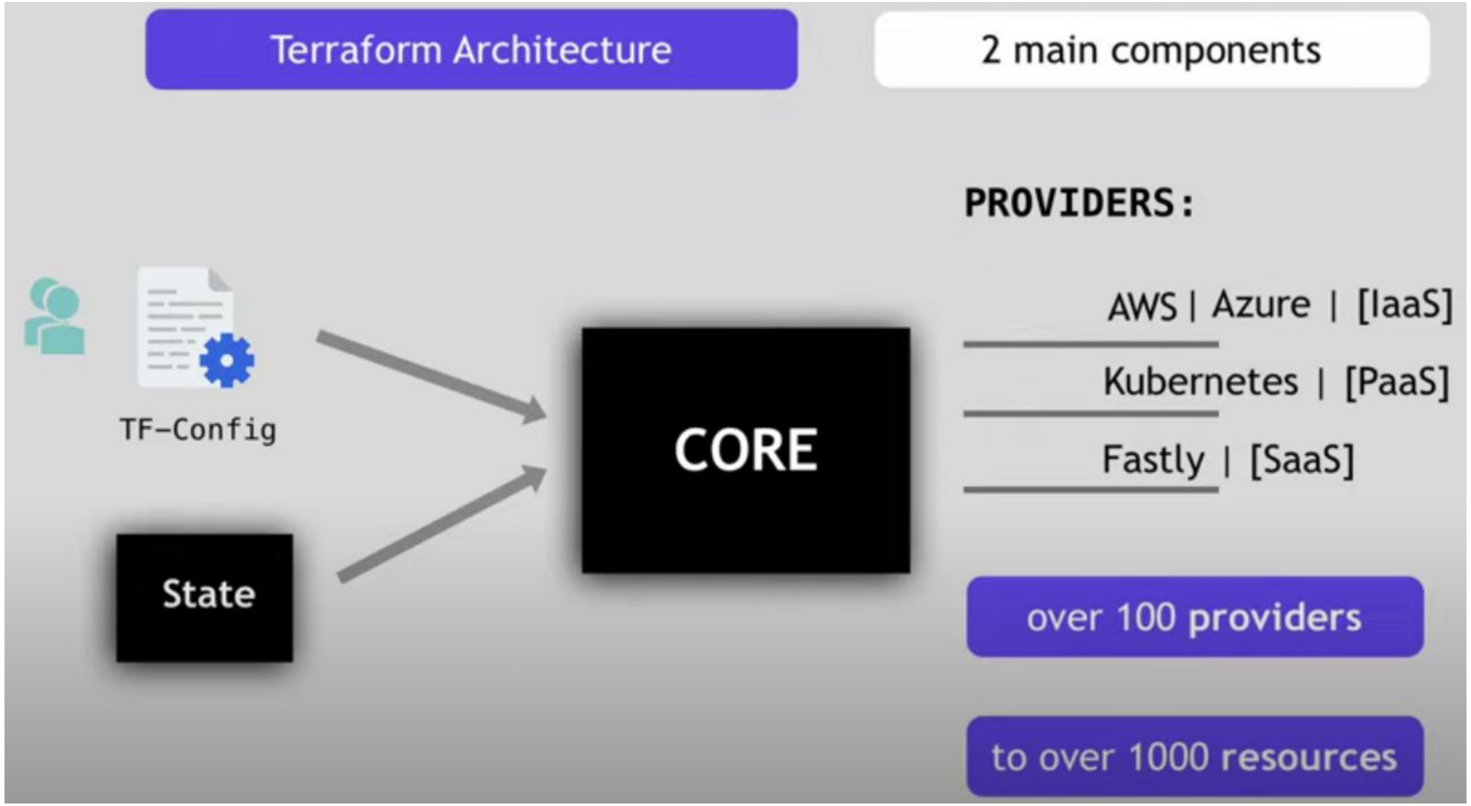
# How Terraform Works

- Terraform allows infrastructure to be expressed as code in a simple, human readable language called **HCL** (HashiCorp Configuration Language).

- Terraform reads configuration files and provides an execution plan of changes, which can be reviewed for safety and then applied and provisioned.



## How Terraform Works



4

## Workflows

# Workflows

A **simple workflow** for deployment will follow closely to the steps below.

**Scope:** Confirm what resources need to be created for a given project

**Author:** Create the configuration file in HCL based on the scoped parameters

**Initialize:** Run terraform init in the project directory with the configuration files. This will download the correct provider plug-ins for the project.

**Plan & Apply:** Run terraform plan to verify creation process and then terraform apply to create real resources as well as state file that compares future changes in your configuration files to what actually exists in your deployment environment.

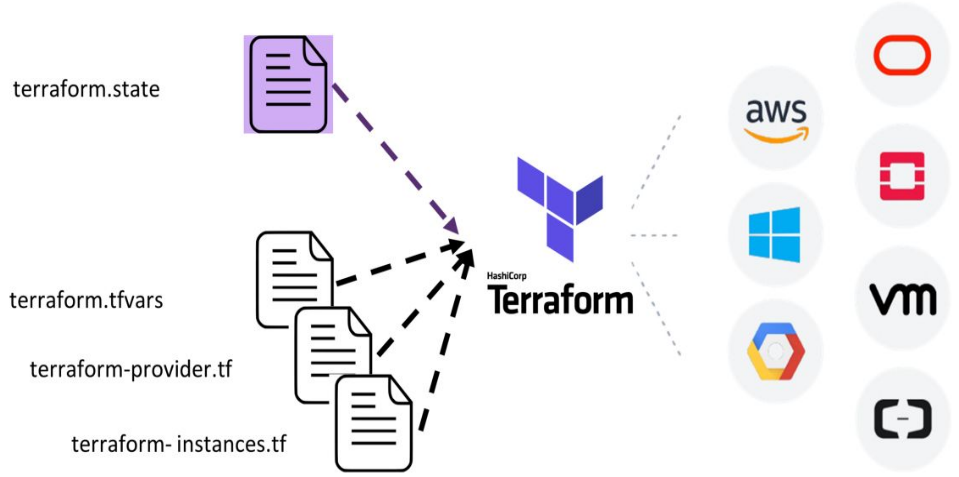
5

## Terraform Elements

# Terraform Elements

**State**

This state is used by Terraform to map real world resources to your configuration, keep track of metadata, and to improve performance for large infrastructures. Terraform uses this local state to create plans and make changes to your infrastructure. State is a necessary requirement for Terraform to function.



# Terraform Elements

**Providers**

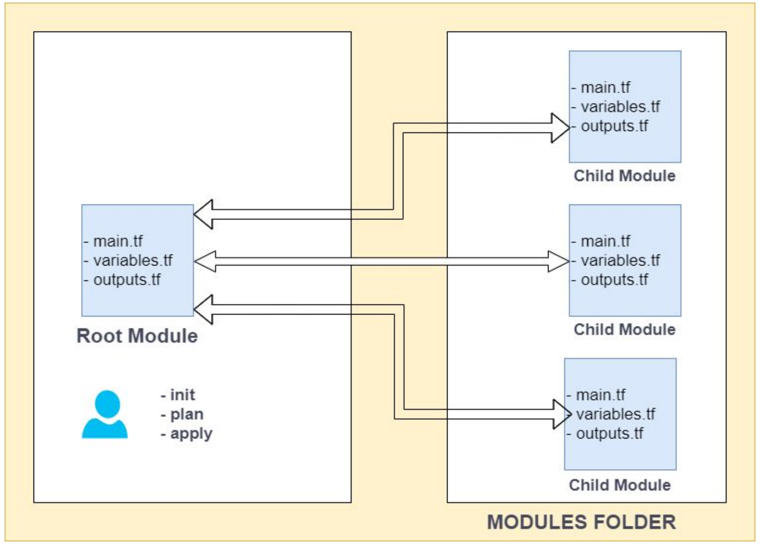
A provider is responsible for understanding API interactions and exposing resources. Every Terraform provider has its own documentation, describing its resource types and their arguments.

amazon web services
Google Cloud Platform
Azure
kubernetes
HashiCorp Nomad
openstack
PostgreSQL
Bitbucket
docker
CONSUL
GitHub
New Relic.
DATADOG
dnsimple
vmware vSphere

# Terraform Elements

**Modules**

A Terraform module is a set of Terraform configuration files in a single directory. Even a simple configuration consisting of a single directory with one or more «.tf» files is a module. When you run Terraform commands directly from such a directory, it is considered the root module. So in this sense, every Terraform configuration is part of a module.



# Terraform Elements

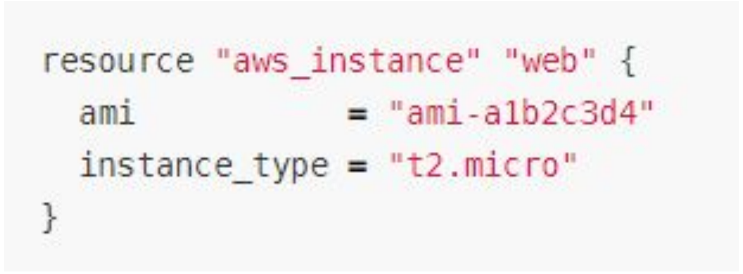
**Backends**

A "backend" in Terraform determines how state is loaded and how an operation such as <apply> is executed. By default, Terraform uses the "local" backend, which is the normal behavior of Terraform you're used to. Backends are completely optional. You can successfully use Terraform without ever having to learn or use backends. Backends are used for keeping sensitive information off disk.

# Terraform Elements

## Resource Blocks

*Resources* are the most important element in the Terraform language. Each resource block describes one or more infrastructure objects, such as virtual networks, compute instances, or higher-level components such as DNS records.



```
resource "aws_instance" "web" {
  ami           = "ami-a1b2c3d4"
  instance_type = "t2.micro"
}
```

# Terraform Elements

## Resource Blocks

```
resource "aws_instance" "web" {

  ami           = "ami-a1b2c3d4"

  instance_type = "t2.micro"

}
```

* A `resource` block declares a resource of a given type ("aws_instance") with a given local name ("web"). The name is used to refer to this resource from elsewhere in the same Terraform module, but has no significance outside that module's scope.
* The resource type and name together serve as an identifier for a given resource and so must be unique within a module.
* Resource names must start with a letter or underscore, and may contain only letters, digits, underscores, and dashes.

6

## Advantages of Terraform

# Advantages of Terraform

**Platform Agnostic**

- In a modern datacenter, you may have several different clouds and platforms to support your various applications.

- With Terraform, you can manage a **heterogeneous environment** with the same workflow by creating a configuration file to fit the needs of your project or organization.

## Advantages of Terraform

**State Management**

- Terraform creates **a state file** when a project is first initialized. Terraform uses this local state to create plans and make changes to your infrastructure.
- Prior to any operation, Terraform does a refresh to update the state with the real infrastructure. This means that Terraform state is the source of truth by which configuration changes are measured.
- If a change is made or a resource is appended to a configuration, Terraform compares those changes with the state file to determine what changes result in a new resource or resource modifications.

# Advantages of Terraform

## Operator Confidence

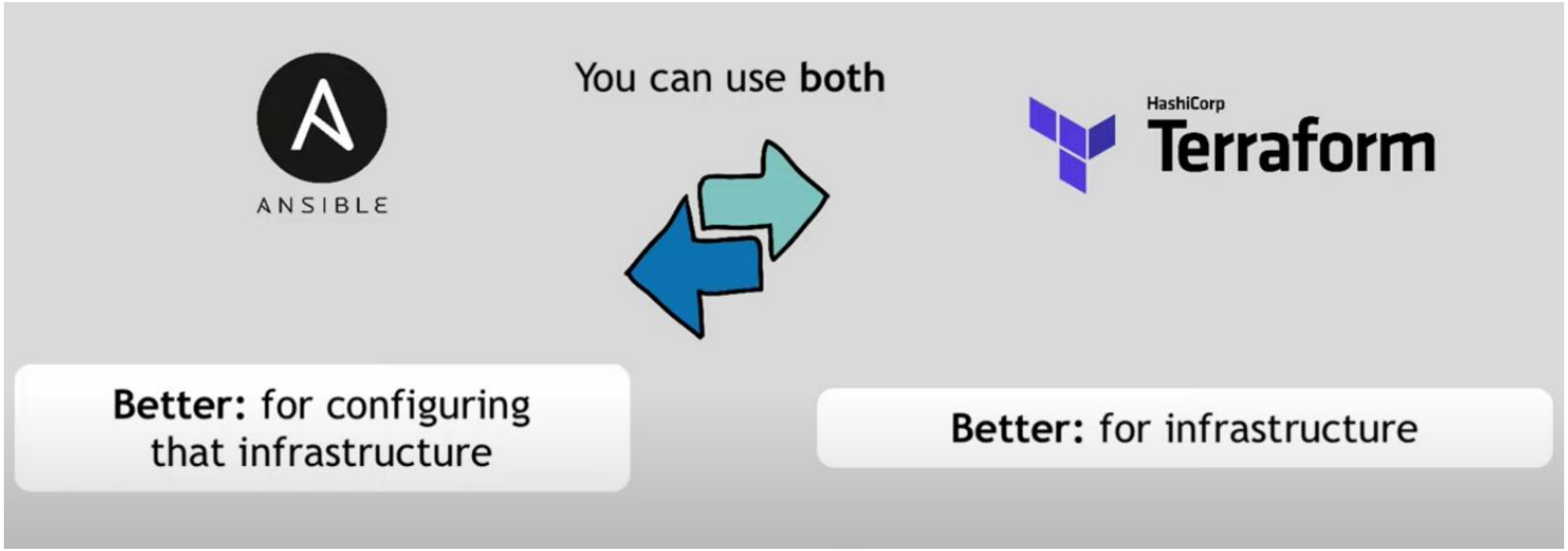
- The workflow built into Terraform aims to instill confidence in users by promoting easily repeatable operations and a planning phase to allow users to ensure the actions taken by Terraform will not cause disruption in their environment.

- Upon **terraform apply**, the user will be prompted to review the proposed changes and must affirm the changes or else Terraform will not apply the proposed plan.

7

## Terraform & Ansible

# Terraform & Ansible



You can use **both**

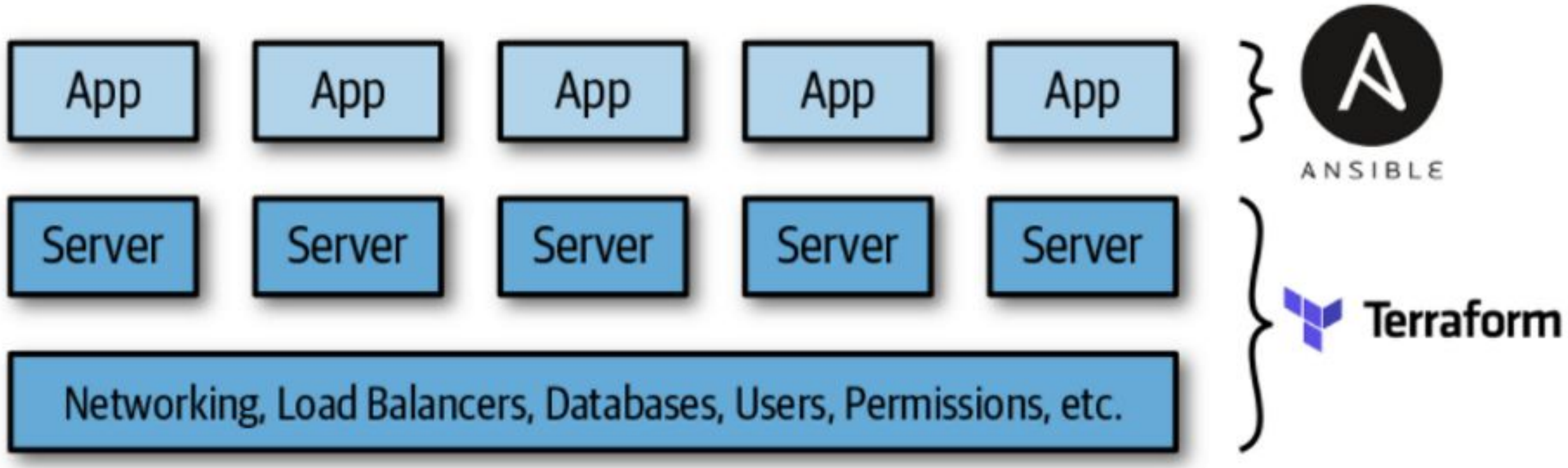
ANSIBLE

HashiCorp Terraform

**Better:** for configuring that infrastructure

**Better:** for infrastructure

# Terraform & Ansible


App
App
App
App
App
ANSIBLE
Server
Server
Server
Server
Server
Terraform
Networking, Load Balancers, Databases, Users, Permissions, etc.

8

## Learn More Terraform

# Learn More Terraform

- Terraform Documentation
- Hashicorp/terraform (Github Page)
- Shuaibiyy/awesome-terraform
- tfutils/tfenv
- gruntwork-io/terragrunt
- 28mm/blast-Radius
- Terraform Registry

# THANKS!

## Any questions?

You can find me at:

- sumod@clarusway.com