DOCUMENTATION OF ECS TERRAFORM

HashiCorp
Terraform

Presented By
SHUBHAM

What is Terraform?

HashiCorp Terraform is an infrastructure as code tool that lets you define both cloud and on-prem resources in human-readable configuration files that you can version, reuse, and share. You can then use a consistent workflow to provision and manage all of your infrastructure throughout its lifecycle. Terraform can manage low-level components like compute, storage, and networking resources, as well as high-level components like DNS entries and SaaS features.

What is Infrastructure as Code (IaC)?

Infrastructure as code (IaC) is a practice in which infrastructure is provisioned and managed using code and software development techniques, such as version control and continuous integration. The cloud's API-driven model enables developers and system administrators to interact with infrastructure programmatically, and at scale, instead of needing to manually set up and configure resources. Thus, engineers can interface with infrastructure using code-based tools and treat infrastructure in a manner similar to how they treat application code. Because they are defined by code, infrastructure and servers can quickly be deployed using standardized patterns, updated with the latest patches and versions, or duplicated in repeatable ways.

How does Terraform work?

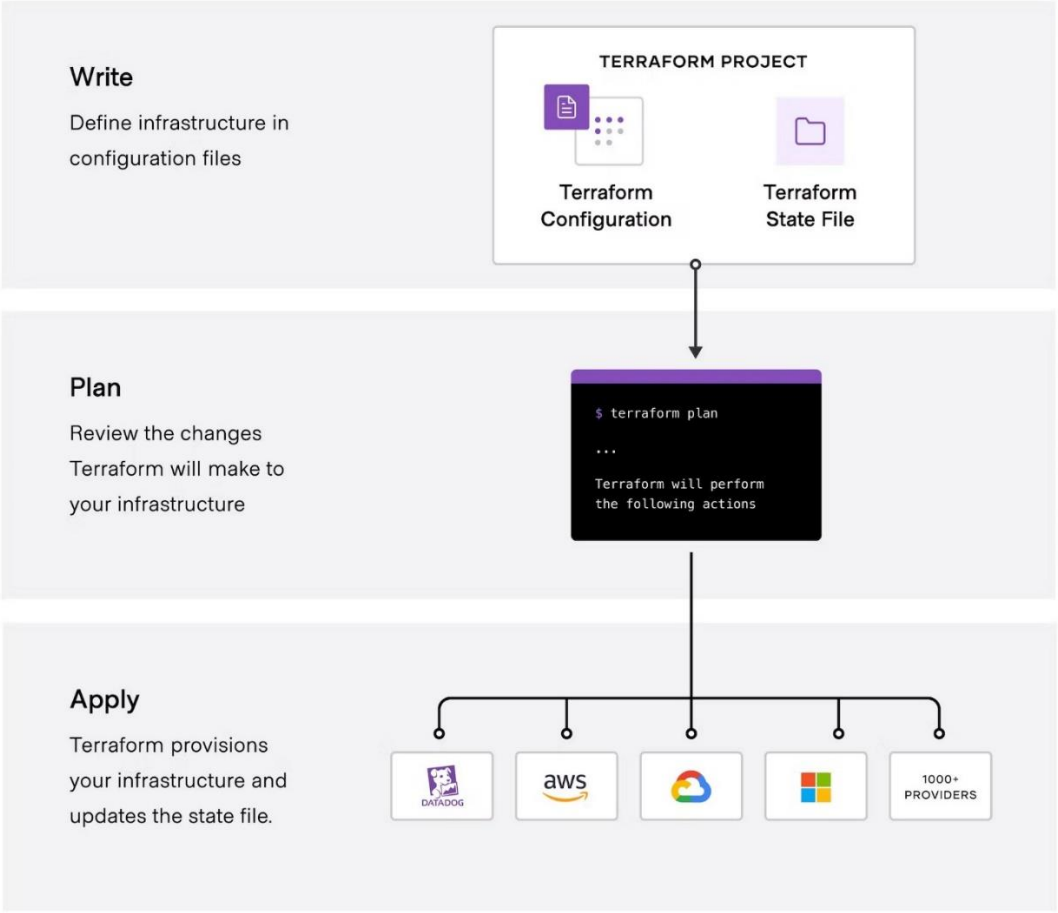
Terraform creates and manages resources on cloud platforms and other services through their application programming interfaces (APIs). Providers enable Terraform to work with virtually any platform or service with an accessible API.

HashiCorp and the Terraform community have already written **thousands of providers** to manage many different types of resources and services. You can find all publicly available providers on the Terraform Registry, including Amazon Web Services

(AWS), Azure, Google Cloud Platform (GCP), Kubernetes, Helm, GitHub, Splunk, DataDog, and many more.

The core Terraform workflow consists of three stages:

- **Write:** You define resources, which may be across multiple cloud providers and services. For example, you might create a configuration to deploy an application on virtual machines in a Virtual Private Cloud (VPC) network with security groups and a load balancer.
- **Plan:** Terraform creates an execution plan describing the infrastructure it will create, update, or destroy based on the existing infrastructure and your configuration.
- **Apply:** On approval, Terraform performs the proposed operations in the correct order, respecting any resource dependencies. For example, if you update the properties of a VPC and change the number of virtual machines in that VPC, Terraform will recreate the VPC before scaling the virtual machines.



What is Docker?

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications. By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Docker is a software platform that simplifies the process of building, running, managing and distributing applications. It does this by virtualizing the operating system of the computer on which it is installed and running.

What is Docker File?

A Docker file is a text document that contains all the commands a user could call on the command line to assemble an image. This page describes the commands you can use in a Docker file.

What is Microservices?

The microservices architecture is a design approach to build a single application as a set of small services. Each service runs in its own process and communicates with other services through a well-defined interface using a lightweight mechanism, typically an HTTP-based application programming interface (API). Microservices are built around business capabilities; each service is scoped to a single purpose. You can use different frameworks or programming languages to write microservices and deploy them independently, as a single service, or as a group of services.

What is Continuous Integration?

Continuous integration (CI) is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. The key goals of continuous integration are to find and address bugs quicker, improve software quality, and reduce the time it takes to validate and release new software updates. In more simple words: Continuous integration (CI) is the practice of automating the integration of code changes from multiple contributors into a single software project. It's a primary DevOps best practice, allowing developers to frequently merge code changes into a central repository where builds and tests then run.

About This Project.

In this documentation we have to make 6 microservices in a single cluster using ECS, it will have 6 pipelines for 6 microservices. Where we have to create services on a single cluster where we access VPN, load balancer, ECS ECR, IAM role and policies, and autoscaling.

VPC

Virtual Private Network (VPC) is a Virtual Network dedicated to your AWS account. It is logically isolated from other virtual network in the AWS cloud. You can specify an IP address range for the VPC, Subnet And gateways and associate security groups. A subnet is a range of IP Address in your VPC.

IAM Policies

It defines permissions for an action regardless of Method that you use to perform the operation.

Cluster

Amazon Cluster is a logical grouping of task or services. Your Tasks and service are run on the infrastructure that is registered to a cluster. The infrastructure capacity can be provided by AWS Faregate, which is server less infrastructure that AWS Manage.

Task Definition

A task is the instantiation of a task definition within a Cluster. After you create a task definition for your Application within Amazon ECS, you can specify the number of tasks to run on your cluster. An Amazon ECS Service run and maintains your tasks.

Auto Scaling Group

AWS Auto Scaling Monitors your applications and automatically adjusts capacity to maintain steady, Predictable performance at the lowest possible cost. Using ASG, It's easy to setup application scaling for Multiple resources across multiple services in minute.

Security Group

A security group acts as a virtual firewall for your EC2 Instance to control incoming and outgoing traffic. Inbound Rule control the incoming traffic to your instance and Outbound rule control the outgoing traffic from your Instance.

Load Balancer

A load balancer serves as the single point of contact for clients. The load balancer distributes incoming application traffic across multiple targets.

Target Group

A target Group tells a load balancer where to direct traffic. when creating a load balancer, you create one or more listeners and configure listener rule to direct the traffic to one

Listener

A listener is a process that check for connection requests, using the protocol and port that you configure. The rules that you define for listener determine how the load balancer routes requests to targets in one or more target groups.

Listener Rule

The rule that you define for your listener determine how the load balancer route requests to the targets in one or more target groups.

S3- Bucket

The S3 bucket used to storing the artifacts for a pipeline you can specify the name of an S3 bucket. A folder to contain the pipeline artifacts is created for you based on the name of the pipeline.

ECS

Amazon Elastic Container Service (Amazon ECS) is a highly scalable and fast container management service. You can use it to run, stop, and manage containers on a cluster. With Amazon ECS, your containers are defined in a task definition that you use to run an individual task or task within a service.

ECR

Amazon Elastic Container Registry (Amazon ECR) is an AWS managed container image registry service that is secure, scalable, and reliable. Amazon ECR supports private repositories with resource-based permissions using AWS IAM.

So, there are following steps that we have to followed for this infrastructure:

- Create a VPC with subnets and also required IAM roles.
- Create an ECS cluster in one of the public subnets & ECR for Docker images.
- Create an ELB in one of the public subnets.
- Configure the ELB with a Target Group for each of the 6 microservices.
- Create a Task Definition for each of the 6 microservices.
- Create a Service for each of the 6 microservices, associating it with the
- appropriate Target Group.

AWS Tools for CICD & Microservices

Continuous Integration and Continuous Delivery

AWS Code Pipeline:

AWS Code Pipeline is a continuous integration and continuous delivery service for fast and reliable application and infrastructure updates. Code Pipeline builds, tests, and deploys your code every time there is a code change, based on the release process models you define. This enables you to rapidly and reliably deliver features and updates.

AWS Code Build:

AWS Code Build is a fully managed build service that compiles source code, runs tests, and produces software packages that are ready to deploy. With Code Build, you don't need to provision, manage, and scale your own build servers. Code Build scales continuously and processes multiple builds concurrently, so your builds are not left waiting in a queue.

AWS Code Deploy:

AWS Code Deploy automates code deployments to any instance, including Amazon EC2 instances and on-premises servers. AWS Code Deploy makes it easier for you to rapidly release new features, helps you avoid downtime during application deployment, and handles the complexity of updating your applications.

AWS Code Commit:

AWS Code Commit is a fully-managed source control service that makes it easy for companies to host secure and highly scalable private Git repositories. You can use Code Commit to securely store anything from source code to binaries, and it works seamlessly with your existing Git tools.