**Terraform with AWS**

**Introduction**

Terraform is the **infrastructure as code** offering from **HashiCorp**. It is a tool for building, changing, and managing infrastructure in a safe, repeatable way. Operators and Infrastructure teams can use **Terraform** to manage environments with a configuration language called the **HashiCorp Configuration Language (HCL)** for human-readable, automated deployments.



**Infrastructure as Code**

**Infrastructure as Code** is the process of managing infrastructure in a file or files rather than manually configuring resources in a user interface. A resource in this instance is any piece of infrastructure in a given environment, such as a virtual machine, security group, network interface, etc.

At a high level, Terraform allows operators to use HCL (HashiCorp Configuration Language) to author files containing definitions of their desired resources on almost any provider (AWS, GCP, GitHub, Docker, etc) and automates the creation of those resources at the time of apply.

Video:

<https://youtu.be/ItWenVs22Q0>

**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.

**Advantages of Terraform**

While many of the current offerings for infrastructure as code may work in your environment, Terraform aims to have a few advantages for operators and organizations of any size.

**1. 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.

**2. 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.

**3. 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.

Complementary Lesson about Introduction to Terraform;

<https://youtu.be/bsl0TrQv9EY>