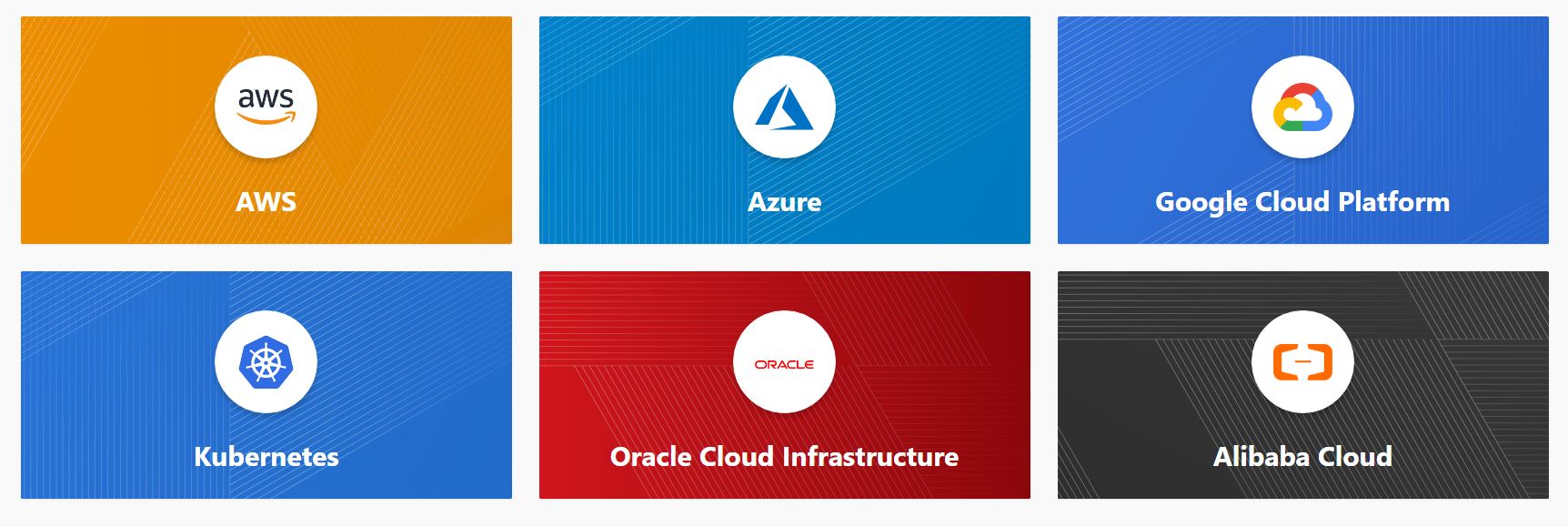
# **Deploying Infrastructure with Terraform**

## **Provider and Resources**

1.1 Overview of Providers:

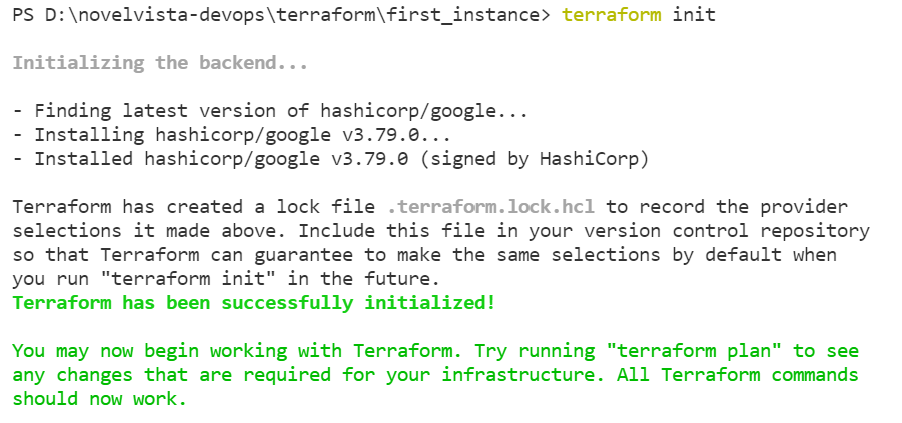
Terraform supports multiple providers.

Depending on what type of infrastructure we want to launch, we have to use appropriate providers accordingly.



1.2 Initialization Phase

Upon adding a provider, it is important to run terraform init which in-turn will download plugins associated with the provider.

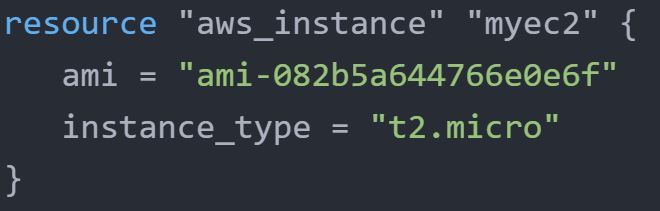


1.3 Resources

Resources are the reference to the individual services which the provider has to offer

Example:

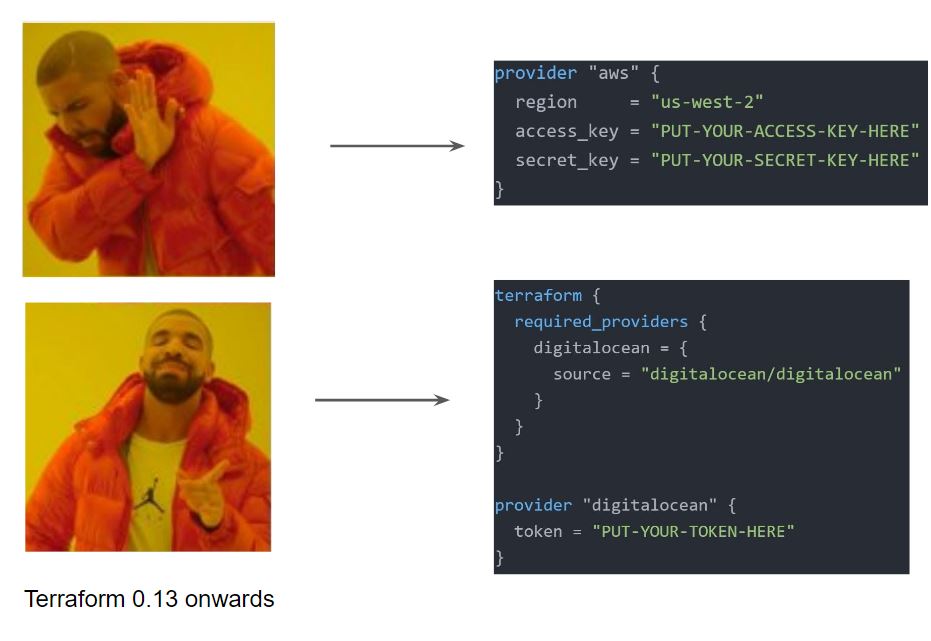
* resource aws\_instance
* resource aws\_alb
* resource iam\_user
* resource digitalocean\_droplet



1.4 Important Update - Newer Version

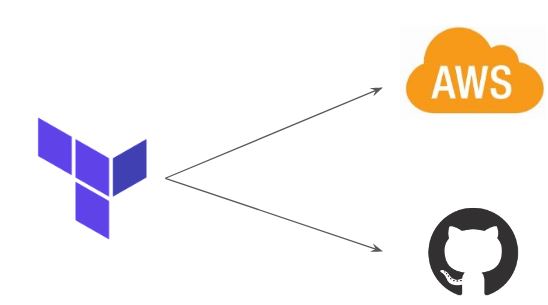
From 0.13 onwards, Terraform requires explicit source information for any providers that are not HashiCorp-maintained, using a new syntax in the required\_providers nested block inside the Terraform configuration block



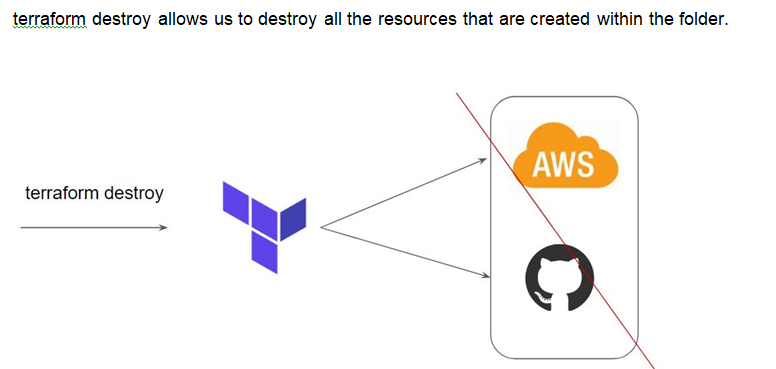


## **Module 2: Destroying Infrastructure with Terraform (NEW)**

If you keep the infrastructure running, you will get charged for it. Hence it is important for us to also know how we can delete the infrastructure resources created via terraform.

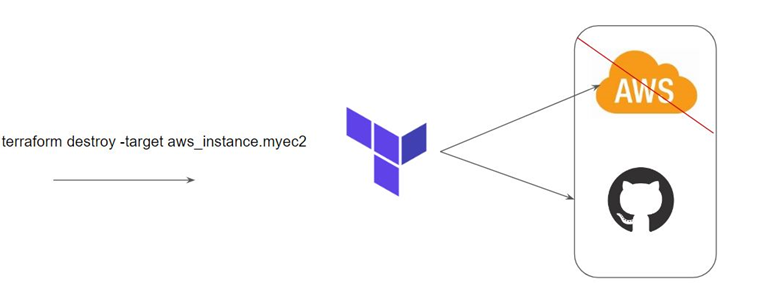


2.1 Approach 1



2.2 Approach 2

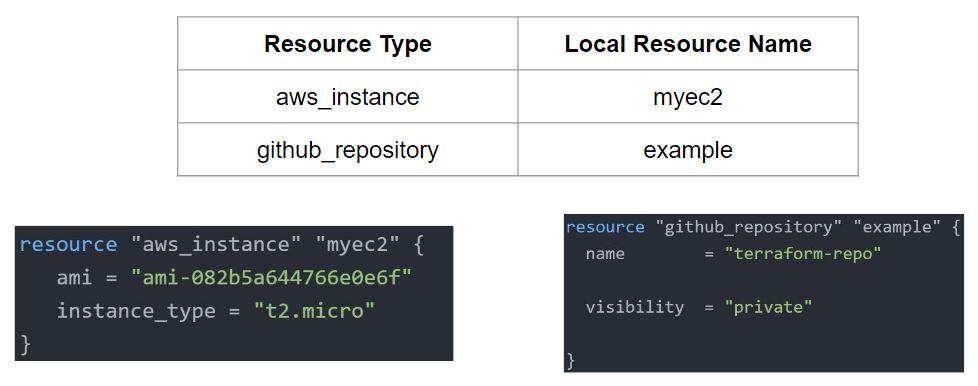
terraform destroy with -target flag allows us to destroy the specific resource.



2.3 Terraform Destroy with Target

The -target option can be used to focus Terraform's attention on only a subset of resources.

Combination of: Resource Type + Local Resource Name

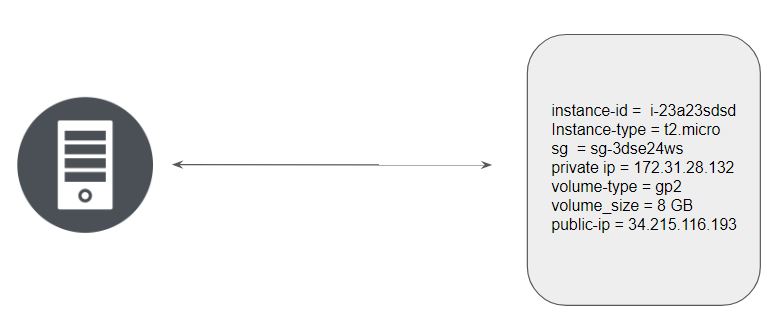


## **Module 3: Terraform State File**

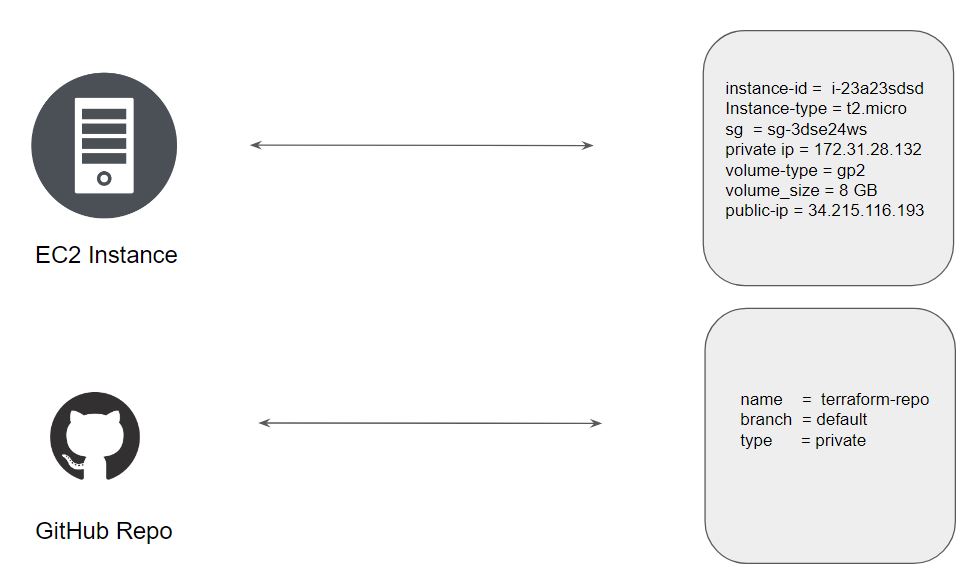
3.1 Overview of State Files:

Terraform stores the state of the infrastructure that is being created from the TF files.

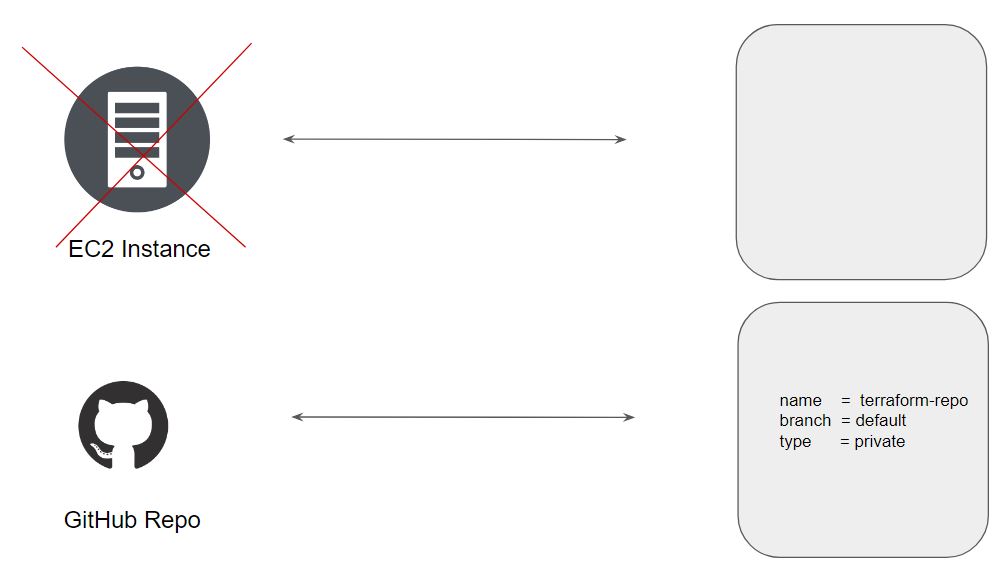
This state allows terraform to map real-world resources to your existing configuration.



Multiple resources in Terraform will have a separate block with the state file.



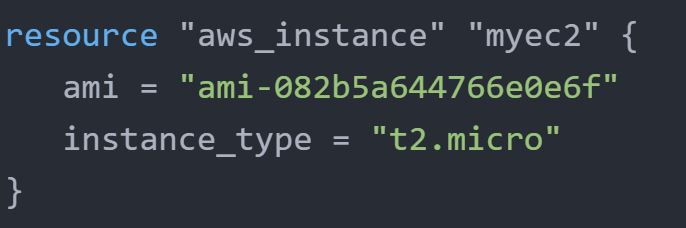
Whenever a resource is removed, its corresponding entry under the state file is also removed.



## **Module 4: Desired & Current State**

4.1 Desired State

Terraform's primary function is to create, modify, and destroy infrastructure resources to match the desired state described in a Terraform configuration



4.2 Current State

The current state is the actual state of a resource that is currently deployed.



4.3 Important Pointer

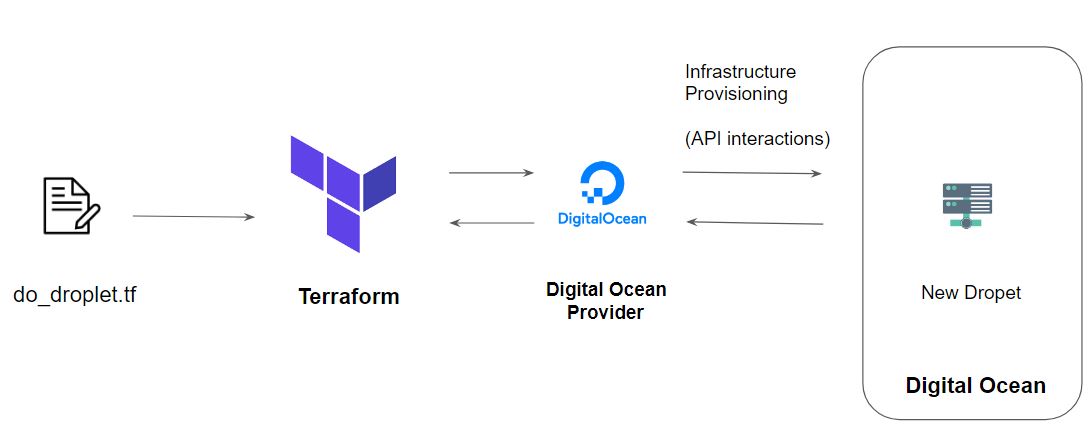
Terraform tries to ensure that the deployed infrastructure is based on the desired state.

If there is a difference between the two, terraform plan presents a description of the changes necessary to achieve the desired state.



**Module 5 Provider Versioning**

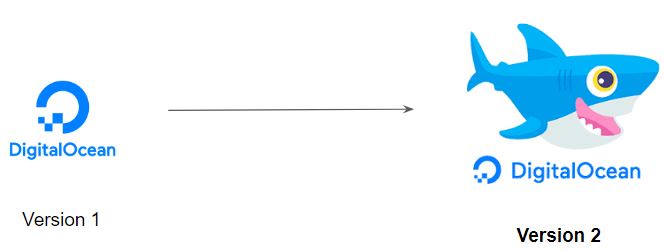
5.1 Overview of Provider Architecture



5.2 Provider Versioning

Provider plugins are released separately from Terraform itself.

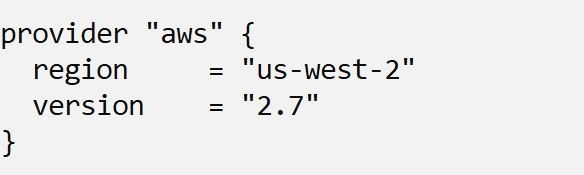
They have a different set of version numbers.



5.3 Explicitly Setting Provider Version

During terraform init, if the version argument is not specified, the most recent provider will be downloaded during initialization.

For production use, you should constrain the acceptable provider versions via configuration, to ensure that new versions with breaking changes will not be automatically installed



5.4 Arguments for Specifying the provider

There are multiple ways of specifying the version of a provider.

