**What is Terraform:** Terraform is an IAAC tool created by Hashicorp open source community. Terraform can be setup on premise as well as on cloud or any OS. It is an infrastructure provisioning tool which reduces manual efforts and helps in provisioning

Machines in automatic way. Terraform uses Hashicorp configuration language. So basically it’s a document in which we define the respective provider details along all the requirements like GW, cloud provider ,subnet , Security Group etc that all needs to be provisioned.

**Terraform Overview:** All the codes will be stored in .tf format. Terraform.io gives the list of cloud providers, If you want to use custom cloud vendor, that is also possible by downloading the binary of custom cloud and put it in provider list and run terraform init.

* Terraform Resources: Terraform is written in a declarative manner and if someone again applies or copy terraform resource data again, terraform apply command, terraform will not spin another cluster with same data set. If we add any tag to an existing resource, it will simply attach to the existing spined up cluster.
* Terraform Referencing Resources: In terraform we can refer other created resources which were created earlier using name.id. For example for a subnet to created, VPC should exist first. So in subnet resource, we can tag the vpc id in subnet resource. While refencing, we can place any resource above and any resource beneath. So it means order doesn’t really matter for vpc to be created.

**Terraform Most Basic Commands:**

Terraform init

Terraform plan

Terraform apply

Terraform destroy

Terraform destroy: It will destroy the instances which was created. For destroying a single instance, use particular parameters. Terraform checks for state file in next run, if someone mistakenly deleted all the config from terraform file, and as changes are still in backup state file, Terraform can turn the instances on as per backup state file.

* Terraform Files: This folder is created when we initialize any directory. All the code for a resource get stored here. Terraform init creates these files, if it gets deleted, it downloads again using terraform init. Terraform.tfstate is very important file and playing wrongly with it can ruin the project too. All the modifications are reflected in the file.
* Terraform State Commands
* Terraform Output
* Target Resources
* Terraform Variables

Why to use terraform: Letz say, we have a large infra to spin up in which we need to have multiple instances to spin up. It helps in creating repeatable infrastructure, define once and change certain variables only depending on the requirement. We can destroy them easily as well. It also takes care of managing the infra means after spinning up the infra, we can modify as per new requirements as well.

* Provision Infra using human and machine readable code
* Coded workflow
* Change/Update Infra
* Plan Before applying
* Reusability

Prerequisites for Terraform

Cloud provisioning with Terraform

Demo:

1. Create your first Instance:

Write below config and save it in .tf format

*provider "aws" {*

*region = "us-east-1"*

*access\_key = "AKIARKRKKBQYV63ZDAN2"*

*secret\_key = "8N2I47RQAymAfuM6yGAqJW1FaaZ72clmRhTm+tAl"*

*}*

*resource "aws\_instance" "my-first-server" {*

*ami = "ami-085925f297f89fce1"*

*instance\_type = "t2.micro"*

*availability\_zone = "us-east-1a"*

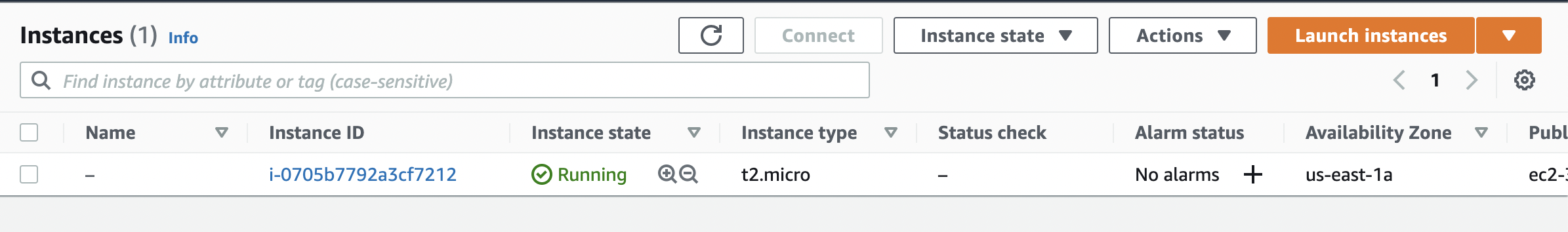
*}*

*Run*

*#terraform init*

*#terraform plan*

*#terraform apply –auto-approve*

**