1# new configuration

```
terraform project
$HOME/terraform/
|-travelgo (project directory)
  |-main.tf
  |-variables.tf
  |-inputs.tfvars
  |-outputs.tf
  |-.terraform/
     |-providerplugins
```

1. keypair

```
resource “aws_key_pair” “travelgokp” {
  key_name=“travelgokp”
  public_key = file(“~/.ssh/terraformkp.pub”)
}
```

2. ec2

```
resource “travelgoec2” {
  instance_type=“t2.micro”
  ami = “ami-9389904”
  key_name = aws_key_pair.travelgokp.key_name
  tags = {
    “Name” =“travelgoec2”
  }
}
```

----

there is no terraform state as it is an new terraform configuration.

➜ terraform plan

since there is no terraform state file, terraform identifies that all the resources that are declared in the terraform configuration should be created newly.

➜ terraform apply

upon applying the terraform configuration, the terraform provisions the resources on the cloud platform
here it is keypair and ec2 instance
then captures the id’s of the resources that are created and stores in the terraform statefile by generating it in the project directory

```
terraform.tfstate
resources {
  @resource = #id
}
```

#2. modified the terraform configuration for eg.. added a new resource into the above configuration.

```
resource “aws_security_group” “travelgosg” {
  ingress {
    from_port=22
    to_port=22
    procotol=“tcp”
    cidr_block = “0.0.0.0/0”
  }
  egress {
    from_port=0
    to_port=0
    procotol=“-1”
    cidr_block = “0.0.0.0/0”
  }
  tags = {
    “Name” = “travelgosg”
  }
}
resource “travelgoec2” {
  instance_type=“t2.nano”
  ami = “ami-9389904”
  key_name = aws_key_pair.travelgokp.key_name
  vpc_security_group_ids = [aws_security_group.travelgosg.id]
  tags = {
    “Name” =“travelgoec2”
  }
}
```

➜ terraform plan

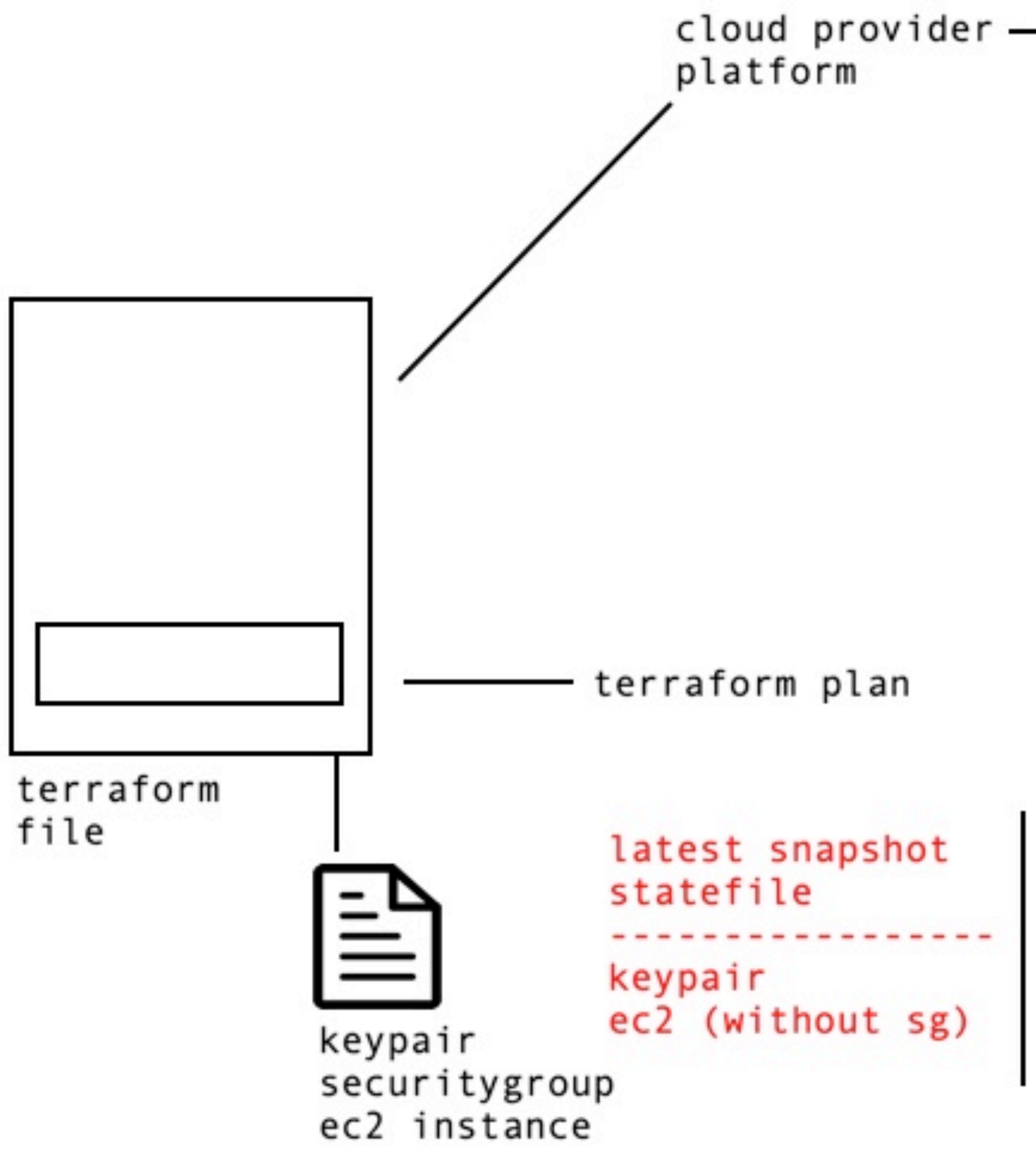
terraform.tfstate

```
terraform.tfstate
resources: [
  {
    “name” : “travelgoec2”
    “instance_type”: “t2.micro”
    “ami” : “ami-9389904”
    “…”
    “id: “#id”
  }
]
```

apply -> what we have -> what should be done?

What will happens when we run terraform plan now?
Stage#1: [refresh state]
in refresh phase, the terraform refreshes the state of the existing resources with that latest snapshot of resource from the cloud provider.
1.1 goes to the terraform configuration file, it picks each resource declaration and checks in the terraform statefile, is the resource is already provisioned and resource id exists for the resource declaration or not
- if the resource state is not available in tfstate file, it ignores and moves to the next resource declaration
- if the resource is found in terraform.tfstate file, then it picks the id of the resource and goes to the cloud provider platform and queries the latest snapshot of the resource metadata and updates in the terraform state file.

Stage#2: [Compute delta]
2.1 Now goes to the terraform configuration file, again picks up the resource declaration and compares with the latest [refreshed] state of the resource in terraform state file. this comparision results in #3 possible outcomes:
  2.1.1 no resource found in tfstate file for that resource declaration:
    - it marks the resource to be newly created
  2.1.2 resource found in tfstate file and state of the resource is matching with terraform resource declaration
    - dont do anything as the resource already provisioned.
  2.1.2 resource found, but the resource declaration and the state is deviating
    - see what type of delta it has based on that
      – it might update the existing resource if allowed by provider.
      – if the delta cannot be applied on an existing instance, the terraform destroy the resource and provisions it with new terraform resource declaration to meet the final state of the system declared.



cloud
console and changed the resource manually.
1. security group (deleted)
2. ec2 instance
   - storage
   - removed security group
****** [not recommended] *******

latest snapshot
statefile
----
keypair
ec2 (without sg)

delta
new security group
modify ec2 with sg
new subnet