Automate AWS infrastructure using Terraform

NOTE : The Terraform file bundle is attached with the assignment , extract the bundle and go through all the terrafiles modify the resource names, region, availability zone, image id and whatever changes required

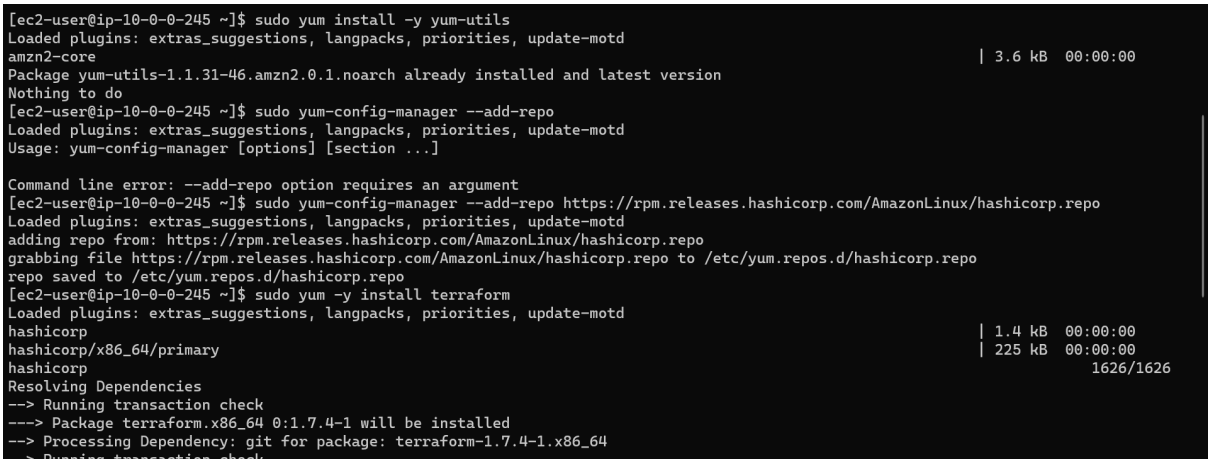
tf_bundle.zip

1.Download Terraform in any of your Amazon public instance using the following commands

sudo yum install -y yum-utils

Sudo yum-config-manager--add-repo
https://rpm.releases.hashicorp.com/AmazonLinux/hashicorp.repo

sudo yum -y install terraform
A:

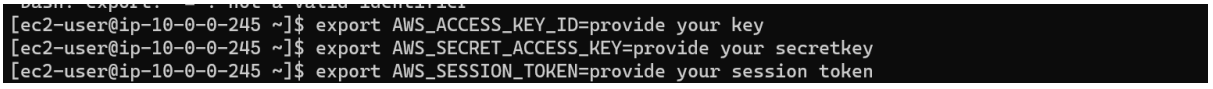


RUN the following commands to set the env variables for AWS (you need to have aws client installed in this instance)

export AWS_ACCESS_KEY_ID = provide your key

export AWS_SECRET_ACCESS_KEY= provide your secretkey

export AWS_SESSION_TOKEN= provide your session token
A:



2.Extract the TF bundle and copy all the files to your linux machine

3.Run the command terraform init from the directory where you copied the TF files

A:

```
[ec2-user@ip-10-0-0-245 terraformm]$ cd tf_bundle/
[ec2-user@ip-10-0-0-245 tf_bundle]$ terraform init

Initializing the backend...

Initializing provider plugins...
- Finding latest version of hashicorp/aws...
- Installing hashicorp/aws v5.38.0...
- Installed hashicorp/aws v5.38.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

4.Validate the terraform files using Terraform validate command

A:

```
[ec2-user@ip-10-0-0-245 tf_bundle]$ terraform validate
Success! The configuration is valid.
```

5.Execute the command terraform plan -out terraform.out and view the plans

A:

```
[ec2-user@ip-10-0-0-245 tf_bundle]$ terraform plan -out terraform.out

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the
following symbols:
  + create

Terraform will perform the following actions:

  # aws_instance.pranay-tf will be created
  + resource "aws_instance" "pranay-tf" {
      + ami                                  = "ami-0d8618c2b4aa22435"
      + arn                                  = (known after apply)
      + associate_public_ip_address          = (known after apply)
      + availability_zone                    = (known after apply)
      + cpu_core_count                       = (known after apply)
      + cpu_threads_per_core                 = (known after apply)
      + disable_api_stop                     = (known after apply)
      + disable_api_termination              = (known after apply)
      + ebs_optimized                        = (known after apply)
      + get_password_data                    = false
      + host_id                              = (known after apply)
      + host_resource_group_arn              = (known after apply)
      + iam_instance_profile                 = (known after apply)
      + id                                   = (known after apply)
      + instance_initiated_shutdown_behavior = (known after apply)
```

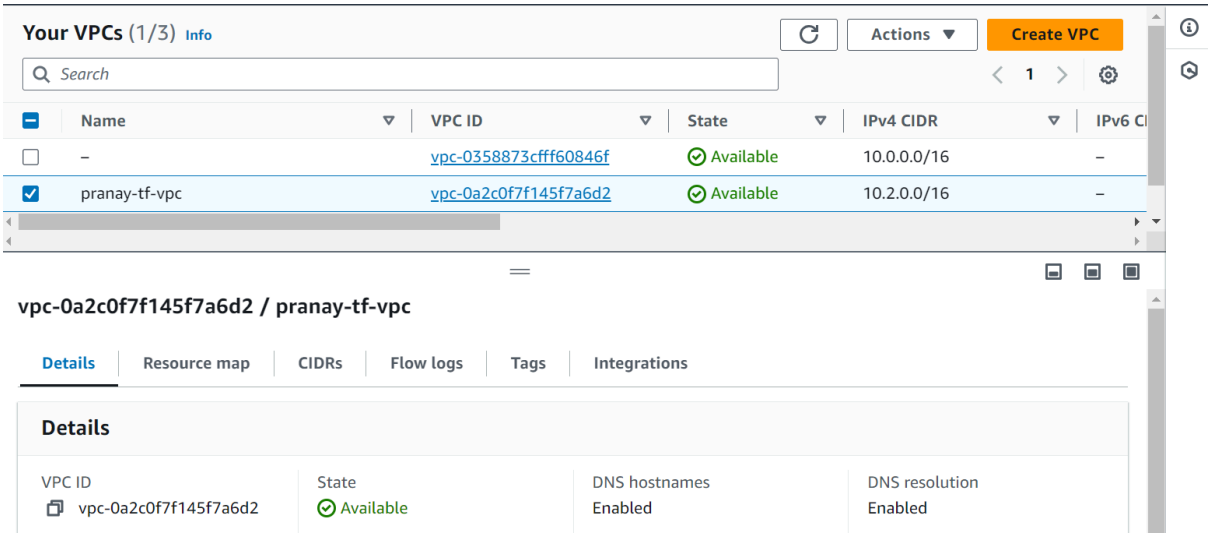
6.Now Execute the command terraform apply terraform.out and it will start creatingall the resources and verify all the resources in your aws console

A:

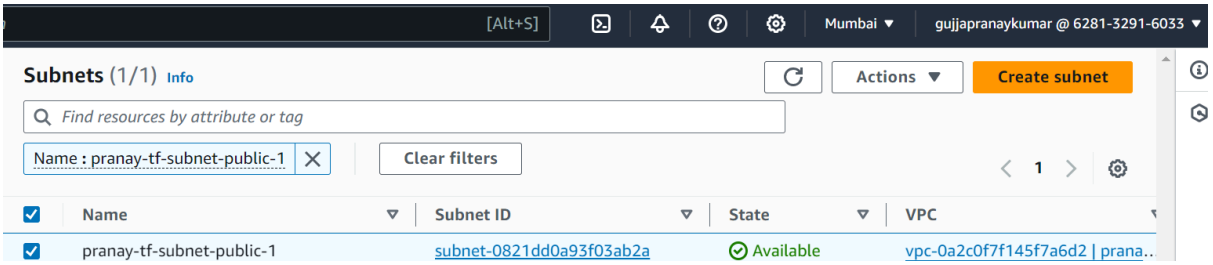
```
[ec2-user@ip-10-0-0-245 tf_bundle]$  terraform apply terraform.out
aws_instance.pranay-tf: Creating...
aws_instance.pranay-tf: Still creating... [10s elapsed]
aws_instance.pranay-tf: Still creating... [20s elapsed]
aws_instance.pranay-tf: Still creating... [30s elapsed]
aws_instance.pranay-tf: Creation complete after 31s [id=i-080742d42611f778e]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
[ec2-user@ip-10-0-0-245 tf_bundle]$ terraform plan -out terraform.out
aws_vpc.pranay-tf-vpc: Refreshing state... [id=vpc-0a2c0f7f145f7a6d2]
```

RESOURCES

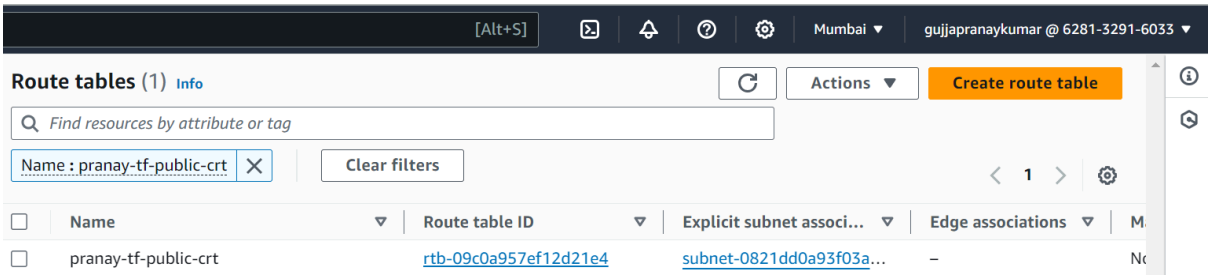
vpc



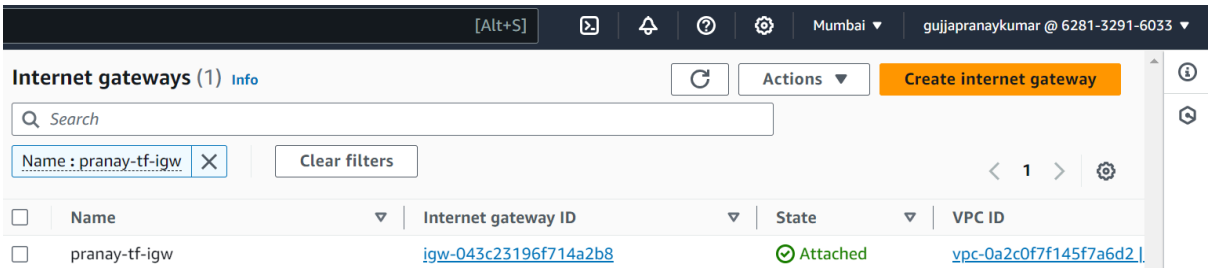
Subnet



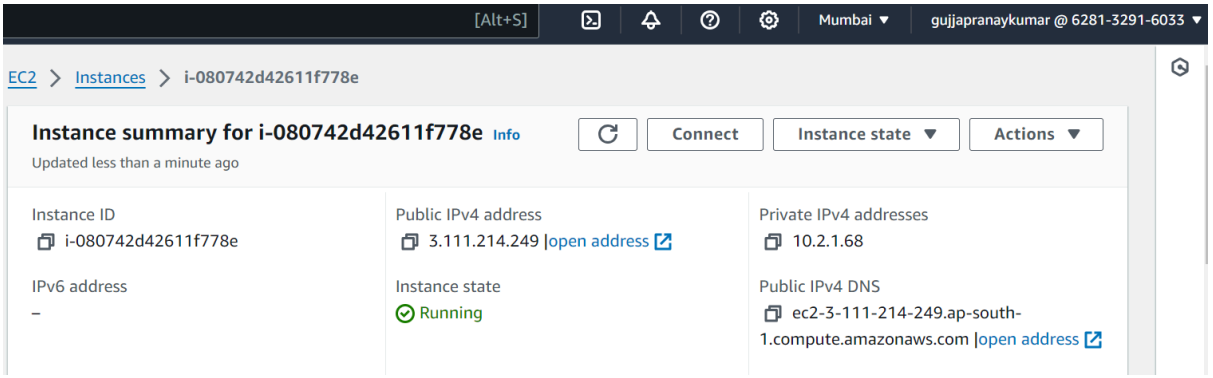
Route table



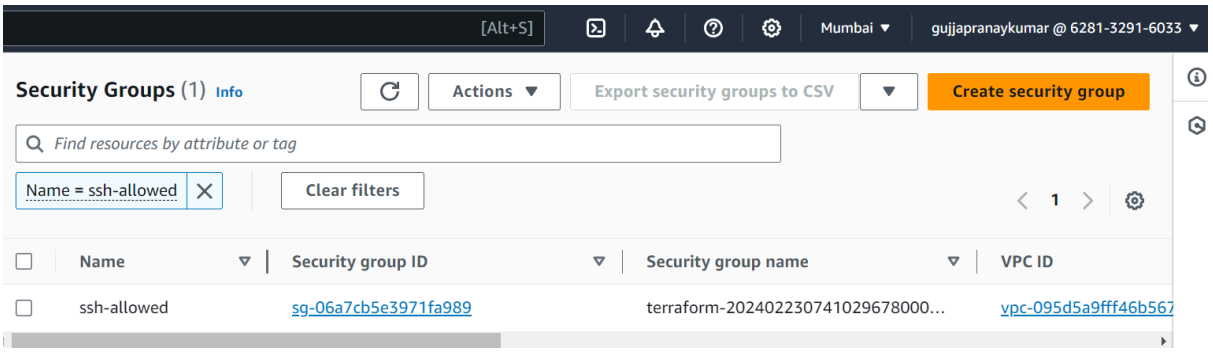
Internet gateway



Instance



Security group



7.Once it is created you can execute the following command to remove all the resources created terraform destroy

A:

```
[ec2-user@ip-10-0-0-245 tf_bundle]$ terraform destroy
aws_vpc.pranay-tf-vpc: Refreshing state... [id=vpc-095d5a9fff46b567d]
aws_security_group.ssh-allowed: Refreshing state... [id=sg-06a7cb5e3971fa989]
aws_subnet.pranay-tf-subnet-public-1: Refreshing state... [id=subnet-007bfea9fa8e2f146]
aws_internet_gateway.pranay-tf-igw: Refreshing state... [id=igw-0406f95b1fd2a1a14]
aws_route_table.pranay-tf-public-crt: Refreshing state... [id=rtb-07636bf004f863e7f]
aws_instance.pranay-tf: Refreshing state... [id=i-08f44bedccc413a33]
aws_route_table_association.pranay-tf-crta-public-subnet-1: Refreshing state... [id=rtbassoc-0e28a7c493451fb62]
```

END