What are we going to see in this session?

- Terraform state file
  - Understanding the use of remote state
  - How to configure remote state
  - Configure S3 bucket
  - Configure AWS CLI to store credentials
  - code

Terraform state file

- Terraform keeps the remote state of the Infrastructure.
- You might have noticed, two file in terraform directory.
  - terraform.tfstate.
  - terraform.tfstate.backup
- When you run terraform apply, these files will be created and written by terraform.
- tfstate file is used to store the current state of infrastructure.
- tfstate.backup file is used to store the backup of previous state [N-1]
- This is how terraform keeps track of the remote state.
- This work in 2 ways :

Assume, you have written [main.tf] file to create instance with Tag name [HTTP SERVER]
now you are rewriting [main.tf] file to change the Tag name as [HTTPS SERVER] and hit terraform apply. Now this is not going to disturb your machine by any means instead it will just modify the Tag name.

In other way around, lets say if you terminate the instance from console which is managed by terraform, after terraform apply it will be started again.

Understating the use remote state

You can keep terraform.tfstate file in version control. Eg : Git, Bitbucket etc..

- ✓ It gives you a history of your terraform.tfstate file [which is just a big JSON file]
- ✓ It allows you to collaborate with team members.
  - But drawback here is when multiple people working, there is chances of collusion.
  - Many people might try to update the state of file.

Keeping state file locally might help in beginning, but when your project become bigger you might want to store your state in remote location.

- ✓ The terraform state file can be saved in remote location, using backend functionality in terraform.
- ✓ By default backend = local [hence it will just save your terraform state file locally]
- ✓ You can keep your backend of below functionalities as well.
  - S3 bucket AWS service [with Locking mechanism using DynamoDB]
  - Consul vault from HashiCorp [with Locking mechanism enabled] :
  - Terraform also provides native solution which comes along with commercial solution.

Using backend functionality benefits in collaboration, because remote state will be available for whole team. Since there is locking enabled only one person can update the state of file at single point in time. Moreover reducing the risks of missing the state file by mistake

How to configure remote state

There are 2 steps to configure remote state :

- Add backend code to backend.tf file
- Run the initialization process

- Point to be noted :
  - ✓backend.tf file will not accept variables.
  - ✓backend.tf file will not accept credentials from terraform.tfvars
  - ✓terraform init should be done after every modification in backend.tf file

In this Example we shall create one new S3 bucket in AWS and push all remote state files over there.

Configure S3 bucket

Lets first configure S3 bucket in AWS.

Short intro about S3 bucket : It is an object storage service provided by Amazon for customers.

- S3 bucket should contain unique name.
- Permission should be kept public.
- Create folder called terraform in bucket

Configure AWS CLI to store credentials

Here we are going to configure AWS CLI to store credentials, since backend.tf file is not going to accept credentials as variables.

Download AWS cli from this link to your local machine :

- curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
- unzip awscliv2.zip
- Install unzip if in case not there “yum install unzip* -y]
- ./aws/install
- aws configure
- Enter your AWS credentials asked for in command line.
- Then you can see credentials are stored as environment variable under file ~/.aws/credentials
- Just do cat ~/.aws/credentials to view the file.

Code

Code : git-repo [file path : Terraform/Codes/E9_remote_state

- Once after creating this file along with other terraform project files just hit terraform init.
- After every terraform apply command is going to save the tfstate file in S3 bucket and maintain the versioning as well.
- You can just see the versions by hitting version “show” in s3 bucket.
- You can use terraform state pull to which will load the remote state file and output it to stdout.

End of this topic !

Any questions?