**Terraform Provisioners**

Provisioners address the limitation that terraform has and provide some actions after the resources are created such as executing script, commands

Provisioners are not a good approach as this is just a addon and it should be a last resort

Local exec / Remote exec / file

Local exec:

Allows to execute commands and scripts in the local system

Remote exec:

Null resource does not do anything, It will instal a null provider that does not do anything

Remote-exec provisioner will invoke a script like bootstrap once terraform creates the resource the remote-exec will be invoked and add the configuration

Ssh / winrm are the connections

Can be enabled with create / destroy time provisioner

**What happen if the provisioner fails?**

If the provisioner fails then the terraform will consider the resources as tainted. Meaning it is (incomplete or unknown state)

So everytime the remote-exec fails then we need to re-run terraform plan and terraform apply which will re-create the resources and then execute the remote-exec provisioner.

Consider this as the last-resort as deleting and re-creating the instance is not a good approach.

**Terraform workspaces**

Local workspaces to run multiple state files associated with same configuration in the same root module

Resources cannot be viewed or shared across workspaces

Starts with the default workspace

Terraform workspace new (dev)

Terraform workspace select (dev)

Terraform workspace delete (dev)

This can be used in the version control systems

Mapping a VCS (branch) to a workspace

Eg: dev 🡪 Dev branch / UAT 🡪 UAT Branch / Prod 🡪 PROD Branch

We can still delete the default workspace by running

Terraform workspace delete default -force

**Terraform Taints:**

Terraform will taint a resource whenever a terraform apply is failed or if the provisioner is failed we need to taint the resource and then execute terraform plan and apply.

Tainting marks the resources as degraded or damaged.

Terraform new version for taint after 0.15 and latest

Terraform apply -replace=azurerm.

terraform apply -replace=azurerm\_virtual\_machine.my\_vm

terraform untaint azurerm\_virtual\_machine.my\_vm