Terraform:

Infra provisioning with terraform

Purpose:

- generally, terraform is used to create infrastructure
- to be more specific it creates virtual infrastructure

Do we have alternatives for Terraform?

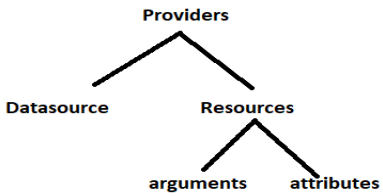
- **AWS =>** CloudFormation
- **Azure =>** ARM Templates

How to write terraform scripts?

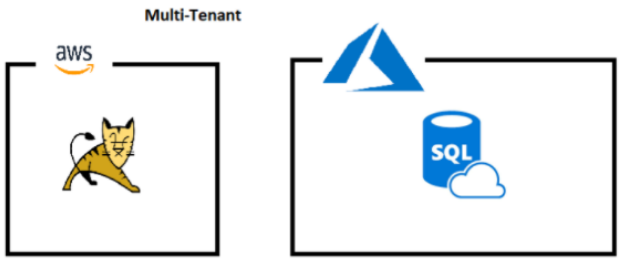
- Terraform has its own Domain Specific Language

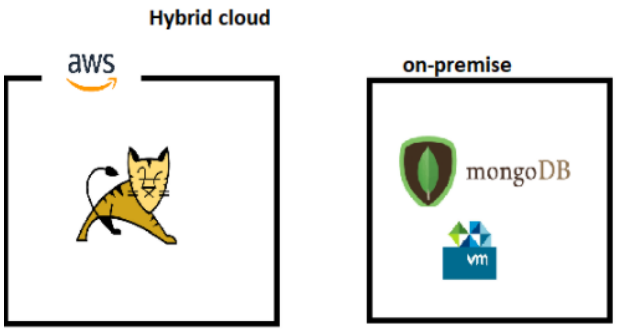
Terraform building blocks

1. Resource: atomic unit of creation
2. Variables: passing different values
3. Data sources:
4. Output: information to be shown to the user
5. Provider: where do you want to create infra



Popular use cases





Basic terraform DSL

- Input for terraform is folder/directory
- Terraform will pick all the files in the input folder with *.tf
- Provider syntax

```
provider "<provider name>" {
   arg1    = value1
   ...
   ...
   argn    = valuen
}
####### Example ###########

provider "aws" {
   region      = "us-west-2"
   secret_key  = "jkdsahfkjadsfhkjasdfhdkjahfdakj"
   access_key  = "kjsdhfkjsdfkjhdsf"
}
```

Resource syntax

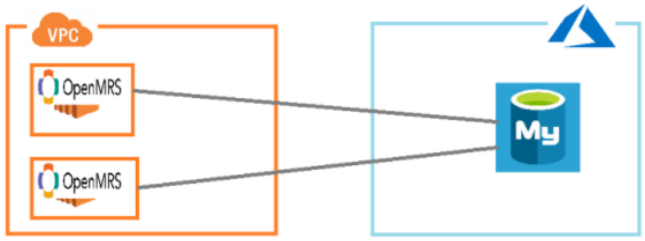
```
resource "<resource type>" "<your name for this resource>" {
   arg1    = value1
   ...
   ...
   argn    = valuen
}

################ Example ################

resource "aws_instance" "mywebserver" {
   ami           = "<ami id>"
   instance_type   = "t2.micro"
}
```

Execution steps

1. Create a directory with name hello-aws
2. Create a file inside a directory “hellow-aws” named “main.tf” with above contents

```
terraform init <dirpath of hello-aws>
terraform validate <dirpath of hello-aws>
terraform apply <dirpath of hello-aws>
```

Architecture to be realized



Multi- tenant
Deployment

AWS side of Architecture

- Manual Steps:
  - Create VPC with cidr range '10.10.0.0/16'
  - Add two subnets to VPC with cidr ranges '10.10.0.0/24' and '10.10.1.0/24'
  - Add internet gateway =>
  - Add a route to internet gateway to exisiting or new route table
  - Add a security group for openmrs server => 22,8080 opened for usage
  - create a key value pair
  - create 2 ec2 machine in 2 subnets created above

Using attributes of one resource in arguments of another resource

Syntax

```
"${resource_type.resource_name.attributename}"
${aws_vpc.my_network.id}
```

- Note: this also called as resource dependency
- Resource dependency determines the order of execution of resources

What happens when you execute terraform apply?

Terraform apply command does the following * Every terraform apply command creates a plan if you dont pass the pass. Plan can be created using `terraform plan` * Before terraform creates anything it will check state file. If state file is not present. It will create everything. * State will be maintained by terraform by making necessary calls to the provider

What terraform init does?

Terraform init downloads the provider executable into local directory ". terraform" in the working directory of terraform

Commands used in the session

```
terraform plan -out "./myplan" .
terraform apply  "./myplan"
# input the plan
```

Sample script without provider in Paris Region of AWS

for script finds the templates

Area to be improved

1. Making terraform script generic
2. Need to be provisioning

How to make terraform scripts generic

- **Variables**

Our scenario

In the case of sample architecture, I have created the following variables in a file called as vars.tf

To find vars.tf in templates

Using variables in terraform resources/providers/outputs

To find details check the templates

The commands used to create infra

```
terraform validate -var 'accesskey=<your access key>' -var 'secretkey=<yoursecretkey>' .
terraform apply -var 'accesskey=<your access key>' -var 'secretkey=<yoursecretkey>' .
```

What is that we have archived so far

- We have able to create ec2 machine and generalize the behaviour.
- Now we can use this script to create ec2 machines in any account & also in any region.

We need to install tomcat

Look at the script in terraform to create two aws ec2 machines & install tomcat

find the templates

Reusability in terraform using modules

- **Module:** reusable terraform script/template is called as module
- **Things to know:**

1. Module is nothing but organising the same terraform script in bit different
2. We have sample architecture terraform script. Now I want to convert that to module
3. Like all the components of terraform. Module also has arguments & attributes
4. Variables become arguments
5. Outputs become attributes
6. In the calling script you will import module
7. There are many modules which you can use. These modules are available in terraform registry

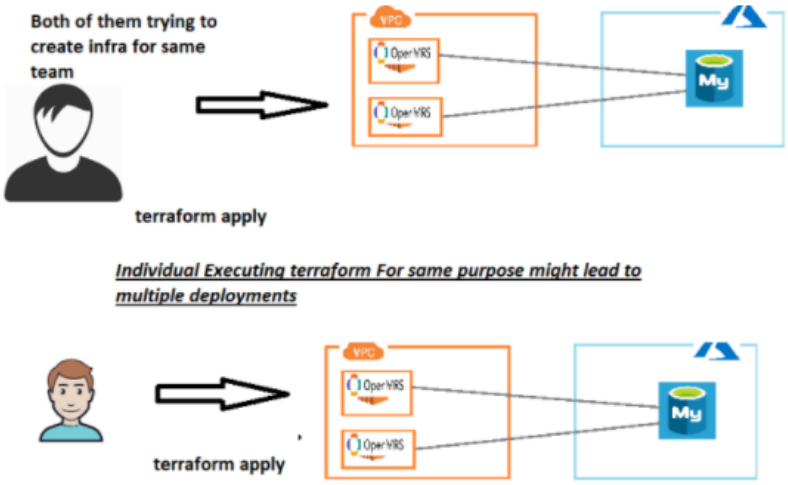
For modules find the templates

How to create module:

- Let's refactor sample architecture
- Ensure all the variables are written in a file called as variables.tf (recommendation)
- Ensure all the outputs are written in a file called as output.tf (recommended)
- Whenever you are using local modules try to use the following directory structure ➔ terraform script ➔ modules < module name > main.tf variables.tf output.tf

To find check the template

Multi user's scenario



As mentioned in the above images we need to solve the problem.

Why this problem

- Whenever terraform apply command is executed, a state file gets created.
- Default location of state file current directory (when you execute the terraform command)

Possible solution

- Try to store state file in common location which is accessible to both
- This is called as backend

What about backend with multiple environments

- Solutions to this is terraform work space