Table of Contents

- Table of Contents
  - Data Sources
  - Using Count Meta Argument
  - Launching EC2 instances inside VPC
  - Creating Multiple Environments
    - Changes in Multiple Environments
    - Destroy Environments after testing
  - Terraform Best Practices
  - Terraform Assignment
    - Notes

Data Sources

- A data source is accessed via a special kind of resource known as a data resource, declared using a **data** block:

```
# https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/availability_zones
data "aws_availability_zones" "available" {}
```

- For CLI based command:

```
aws ec2 describe-availibility-zones --region us-east-1
```

- If value of the AZ is hardcoded, this is not a good practice.
- Execute **terraform console**, this command requires the details of the infra created inside the state file.

```
data.aws_availablity_zones.available
data.aws_availablity_zones.available.names
data.aws_availablity_zones.available.names[0]
exit
```

- Add network resources i.e subnets along with resource creation.

Using Count Meta Argument

- The resource block for multiple subnet creation can be added one by one, or there can be one single resource block with **count**

- Check : count
- **count.index** — The distinct index number (starting with 0)
- Add list variable types in variable declaration file.

--

```
# https://developer.hashicorp.com/terraform/language/values/variables

variable "public_cidrs" {
  type = list(string)
  default = ["172.31.3.0/24","172.31.4.0/24"]
}

variable "private_cidrs" {
  type    = list(string)
  default = ["172.31.5.0/24","172.31.6.0/24"]
}
```

```
resource "aws_subnet" "terraform_public_test_subnet" {
  count                   = 2
  //count                   = length(var.public_cidrs)
  vpc_id                  = aws_vpc.terraform_test_vpc.id
  cidr_block              = var.public_cidrs[count.index]
  map_public_ip_on_launch = true
  availability_zone       =
data.aws_availablity_zones.available.names[count.index]

  tags = {
    Name = "terraform_public_test_subnet"
  }
}
```

Launching EC2 instances inside VPC

- Current file structure can be modified as:

```
[ec2-user@ip-172-31-20-228 terraform_scripts]$ tree .
.
├── backends.tf
├── compute.tf
├── dev.tfvars
├── networking.tf
├── providers.tf
├── terraform.tfstate
├── terraform.tfstate.backup
└── variables.tf
```

- **networking.tf** : This file will contain all VPC and Networking related resource definitions.
- **compute.tf** : This file will contain all EC2 and Computing related resource definitions.
- For launching EC2 instance, there is a different AMI ID present in each region.

```
aws ec2 describe-images --image-ids IMAGE_ID --region us-east-1
```

--

- Use data source **aws_ami** to dynamically fetch the AMI Id of an Operating System

```
data "aws_ami" "server_ami" {
  most_recent = true

  owners = ["099720109477"]

  filter {
    name   = "name"
    values = ["ubuntu/images/hvm-ssd/ubuntu-focal-20.04-amd64-server-*"]
  }
}
```

- Use **aws_instance** as a resource type for creating an EC2 instance.
- Use Resource Referencing for attributes defined as below.

```
resource "aws_instance" "terraform_test_ec2" {
  count         = var.instance_count
  instance_type = var.instance_type
  ami           = data.aws_ami.server_ami.id
  key_name      = var.instance_keypair_name
  tags = {
    Name = "terraform_test_ec2"
  }
  vpc_security_group_ids = [aws_security_group.terraform_test_sg.id]
  subnet_id              = aws_subnet.terraform_public_test_subnet[count.index].id

  root_block_device {
    volume_size = var.vol_size
  }
}
```

Creating Multiple Environments

- Create an environment specific directory, `dev/qa/prod` and copy all TF files required for provisioning resources.

- Create **dev_tf_resources_ws**,**qa_tf_resources_ws**, **prod_tf_resources_ws** in Terraform Cloud and update the **dev/backends.tf**, **qa/backends.tf**, **prod/backends.tf** file as per workspace name.
- Modify the specific variables files i.e **.tfvars, variables.tf** as per environment specifications.
    - **dev/qa/prod** environment should be in **ap-south-1**, modify the **providers.tf** file for region specification.
    - Modify the **variables.tf** to include tags as per environment prefix.
    - Modify ec2 keypair name as per region.

--

- Below will be ideal structure of all terraform Scripts.

```
├── dev
│   ├── backends.tf
│   ├── compute.tf
│   ├── dev.tfvars
│   ├── networking.tf
│   ├── providers.tf
│   └── variables.tf
├── qa
│   ├── backends.tf
│   ├── compute.tf
│   ├── qa.tfvars
│   ├── networking.tf
│   ├── providers.tf
│   └── variables.tf
├── prod
│   ├── backends.tf
│   ├── compute.tf
│   ├── prod.tfvars
│   ├── networking.tf
│   ├── providers.tf
│   └── variables.tf
```

--

- Execute the **terraform init**, **terraform plan**, **terraform apply** from the specific environment directory.

```
[ec2-user@ip-172-31-20-228 terraform_scripts]$ pwd
/home/ec2-user/terraform_scripts
[ec2-user@ip-172-31-20-228 terraform_scripts]$ cd dev
[ec2-user@ip-172-31-20-228 dev]$ ls
backends.tf  compute.tf  dev.tfvars  networking.tf  providers.tf  variables.tf
[ec2-user@ip-172-31-20-228 dev]$ terraform init
[ec2-user@ip-172-31-20-228 dev]$ terraform plan

data.aws_availability_zones.available: Reading...
data.aws_ami.server_ami: Reading...
data.aws_availability_zones.available: Read complete after 0s [id=ap-south-1]
data.aws_ami.server_ami: Read complete after 0s [id=ami-0340ea71c538887c3]
```

```
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:
Plan: 17 to add, 0 to change, 0 to destroy.

[ec2-user@ip-172-31-20-228 dev]$ terraform apply --auto-approve
Plan: 17 to add, 0 to change, 0 to destroy.
aws_vpc.terraform_test_vpc: Creating...
aws_vpc.terraform_test_vpc: Still creating... [10s elapsed]
aws_vpc.terraform_test_vpc: Creation complete after 11s [id=vpc-0f3380e61f87c87d6]
aws_subnet.terraform_private_test_subnet[0]: Creating...
aws_internet_gateway.terraform_test_internet_gateway: Creating...
aws_subnet.terraform_public_test_subnet[0]: Creating...
aws_default_route_table.terraform_private_rt: Creating...
aws_route_table.terraform_public_rt: Creating...
aws_security_group.terraform_test_sg: Creating...
aws_subnet.terraform_public_test_subnet[1]: Creating...
aws_subnet.terraform_private_test_subnet[1]: Creating...
aws_default_route_table.terraform_private_rt: Creation complete after 0s [id=rtb-
07ac6cddceb1b41a1]
aws_internet_gateway.terraform_test_internet_gateway: Creation complete after 0s
[id=igw-03142eae7818d12af]
aws_route_table.terraform_public_rt: Creation complete after 0s [id=rtb-
0d7a2b55d3aedd409]
aws_route.terraform_test_route: Creating...
aws_subnet.terraform_private_test_subnet[0]: Creation complete after 0s
[id=subnet-0fe9114f4af44ee39]
aws_subnet.terraform_private_test_subnet[1]: Creation complete after 0s
[id=subnet-09369fb9540616319]
aws_route_table_association.terraform_private_subnet_association[1]: Creating...
aws_route_table_association.terraform_private_subnet_association[0]: Creating...
aws_route_table_association.terraform_private_subnet_association[0]: Creation
complete after 0s [id=rtbassoc-0a869ba078f8279bb]
aws_route_table_association.terraform_private_subnet_association[1]: Creation
complete after 0s [id=rtbassoc-090530895336b93b2]
aws_route.terraform_test_route: Creation complete after 0s [id=r-rtb-
0d7a2b55d3aedd4091080289494]
aws_security_group.terraform_test_sg: Creation complete after 1s [id=sg-
0e846c44693897ec8]
aws_security_group_rule.egress_all: Creating...
aws_security_group_rule.ingress_all: Creating...
aws_security_group_rule.ingress_all: Creation complete after 0s [id=sgrule-
3738274526]
aws_security_group_rule.egress_all: Creation complete after 1s [id=sgrule-
196345404]
aws_subnet.terraform_public_test_subnet[0]: Still creating... [10s elapsed]
aws_subnet.terraform_public_test_subnet[1]: Still creating... [10s elapsed]
aws_subnet.terraform_public_test_subnet[0]: Creation complete after 10s
[id=subnet-037fa1c9e15812346]
```

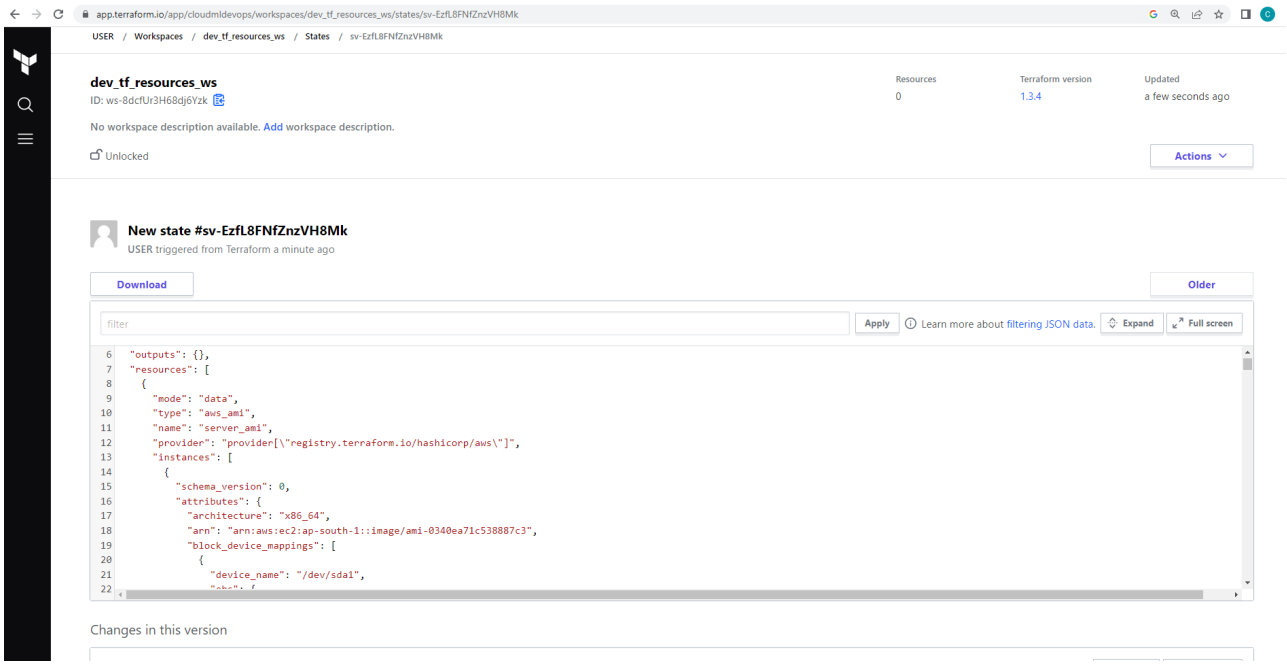
```
aws_instance.terraform_test_ec2[0]: Creating...
aws_subnet.terraform_public_test_subnet[1]: Creation complete after 10s
[id=subnet-011ec3f389244351d]
aws_route_table_association.terraform_public_subnet_association[0]: Creating...
aws_route_table_association.terraform_public_subnet_association[1]: Creating...
aws_route_table_association.terraform_public_subnet_association[0]: Creation
complete after 1s [id=rtbassoc-037be55db3bb8d313]
aws_route_table_association.terraform_public_subnet_association[1]: Creation
complete after 1s [id=rtbassoc-041092aa10d7f5f61]
aws_instance.terraform_test_ec2[0]: Still creating... [10s elapsed]
aws_instance.terraform_test_ec2[0]: Still creating... [20s elapsed]
aws_instance.terraform_test_ec2[0]: Still creating... [30s elapsed]
aws_instance.terraform_test_ec2[0]: Creation complete after 31s [id=i-
0e5f3aceb1a6765ac]
Releasing state lock. This may take a few moments...

Apply complete! Resources: 17 added, 0 changed, 0 destroyed.

# Validate all the above resources in AWS Environment.
# Navigate to AWS Account for validating the resource created by Terraform.
```

--

- Validate the state file in Terraform Cloud under the specific Workspace



--

Changes in Multiple Environments

- Ideal steps for changing any configuration:
  - modify the dev/networking.tf -> terraform apply -> validate the create/update/destroy in console in dev environment
  - modify the qa/networking.tf -> terraform apply -> validate the create/update/destroy in console in qa environment

- modify the prod/networking.tf -> terraform apply -> validate the create/update/destroy in console in prod environment

Destroy Environments after testing

- Since there are multiple environments created, if you do not need the resources and don't want any cost implications, after trying resource creation for multiple environments, execute the **terraform destroy** command for each folder to destroy all resources.

Terraform Best Practices

- **Identify what should be declared in variable and resource block?**
  - Code should be generic (reused across environments, regions, accounts)
  - Any variable that will change across environments, regions, accounts should be declared in variables.tf file.
    - Static reference : definition of variables.
    - Dynamic : Resource reference : resource_type.resource_logical_name.id
- **Terraform configurations files separation**
  - **compute.tf** - define data sources to create all compute resources.
  - **networking.tf** - define data sources to create all networking resources.
  - **variables.tf** - contains declarations of variables used in other terraform files.
  - **terraform.tfvars** - contains variables values and should not be used anywhere and set by default.

--

- **Use separate directories for each environment:**
  - Use separate directory for each environment (dev, qa, prod).
  - Each environment directory corresponds to a default Terraform workspace and deploys a version of the service to that environment.
- **Variables Conventions**
  - Declare all variables in **variables.tf**.
  - Provide meaningful description for all variables.
  - Order keys in a variable block like this: **description , type, default**.
- **Better Security practices**
  - Use remote state:
    - Never to store the state file on your local machine or version control.
    - With remote state, Terraform writes the state data to a remote data store, which can be shared between all team members. This approach locks the state to allow for collaboration as a team.
    - Configure Terraform backend using remote state (shared locations) services such as Terraform Cloud, Amazon S3, Azure Blob Storage, GCP Cloud Storage.

Terraform Assignment

- Provision a VPC Network Resources having 2 public subnets and 2 private subnets, IGW attached to VPC, VPC Gateway Endpoint for S3 Service.
- Create an S3 Bucket with sdlc name as prefix.
- Provision RDS Instance in VPC private subnet launched in the previous step (network resources)
- Provision an EC2 instance having access to IAM Role, that contains IAM Permissions to read and write data to S3 buckets.
    - Validate the data copy from ec2 instance to/from S3 bucket
    - Validate network to connect with RDS instance.
- Validate the connection to RDS Instance from EC2 instance by executing mysql commands
- Document all steps with AWS Service Screenshots into a Word File.

Notes

- Use RDS Free tier instance type to avoid cost
- Use Terraform to create above resources.
- Terraform Code used to create above resources should be generic to create multiple environments in Multiple Region/Accounts.
- Use **terraform destroy** once resources are created and tested to avoid cost.