Terraform Networking - Part 3

Provisioning more Terraform resources

Be sure you have completed the "terraform destroy" command in the previous step before proceeding

A set of additional Terraform files are available to help you provision other parts of the infrastructure

Change directory to tflab2

```
cd ~/environment/tfekscode/primer/tflab2
```

Now take some time to look through the various terraform files that have been provided

Terraform variables

look at the variables.tf file it contains the following:

```
variable "mycount" {
  default = 1
}

variable "aws_vpc" {
  type    = list
  default = ["vpc-10-1", "vpc-10-2"]
}

variable "aws_cidr" {
  default = {
    "vpc-10-1"       = "10.1.0.0/16"
    "vpc-devt-proja" = "10.3.0.0/16"
    "vpc-10-2"       = "10.2.0.0/16"
    "vpc-devt-projx" = "10.4.0.0/16"
  }
}
```

- The "mycount" variable controls how many resources of a given type that reference it are deployed using the special "count" variable (see later).
- The variable aws_vpc is a list of strings that will be used to name your VPCs.
- The variable aws_cidr is a "mapping" that uses the vpc name as defined in the variable aws_vpc to pick up an associated CIDR range.

Lets now look how these are utilized in our other Terraform files

The file vpc.tf has these contents:

```
resource "aws_vpc" "VPC" {
  count                            = var.mycount
  assign_generated_ipv6_cidr_block = false
  cidr_block                       = lookup(var.aws_cidr, var.aws_vpc[count.index])
  enable_dns_hostnames             = false
  enable_dns_support               = true
  instance_tenancy                 = "default"
  tags = {
    "Name" = var.aws_vpc[count.index]
  }
}
```

- **count= var.mycount** - this assigns the special variable "count" to our variable "mycount". The Terraform variable count is an iterator - it says "create this resource count times"
- **"Name" = var.aws_vpc[count.index]** - This line looks up the Nth value in the aws_vpc list and uses it as the value in the tag's Key (Name) , Value (aws_vpc[count.index]) pair.
- **cidr_block = lookup(var.aws_cidr, var.aws_vpc[count.index])** - This is saying look into our map "var.aws_cidr" - use the index value "var.aws_vpc" using the Nth value in the aws_vpc list - where N = the current value of count

Using variables like this enables us to predefine VPC names and CIDR's in advance. And check them into source control systems (github, gitlab etc). Something we will do later.

Other Terraform functions

Another useful technique is to use the Terraform "format" directive to construct strings from variables

in the file subnets.tf we see the line:

- **cidr_block= format("10.%s.1.0/24", count.index + 1)** - The %s in the format statement means "substitute a variable here as a string" - in this case we use the current value of count plus one (as computers start counting at 0 - be we humans like ot start counting at 1)

Also notice in this file how the VPC we just created with the vpc.tf file is referenced:

- **vpc_id = aws_vpc.VPC[count.index].id** - the "VPC" structure is now indexed using "count". As our vpc.tf file is using the count variable to iterate and potentially create multiple VPCs they now are referenced as an Array.

Understanding order and dependencies

One final thing to appreciate is that resources have dependencies. This is automatically handled for you by Terraform, but when reading Terraform code it's important to read things in the right order:

- aws.tf - Specifies how we connect to Terraform , where the credentials are - and optionally where to store the state file (we are using the default - the local directory).
- variables.tf - Defines the variables our terraform code will use.

- subnets.tf and security groups mysg*.tf - depends on the vpc.
- nat_gateway.tf - depends on subnets and elastic ip.

.. and so on ..

Finally

- instance.tf - has the most dependencies and thus is one of the last resources to be created.

Ask the workshop host if you are unsure about any of the file contents before proceeding....

Terraform's documentation for AWS can also be found here: https://www.terraform.io/docs/providers/aws/index.html

Initialize Terraform

```
terraform init
```

As before perform these steps:

```
terraform plan -out tfplan
```

Take some time to study the plan and what Terraform intends to do

It will take a few minutes for the resources to create.

Go to the AWS console as before and find:

- The new VPC.
- New subnets.
- Observe the relevant routing table entries for the public and private subnets.
- Is there a new instance running ?

Previous Next