**16.DevOps-B24-Terraform-Part-2**

--- **note** – in this lesion, we will talk about data source.

--- **resource** – previously, we used resource block to create vpc, if you want to create resource using terraform then we will use this resource block.

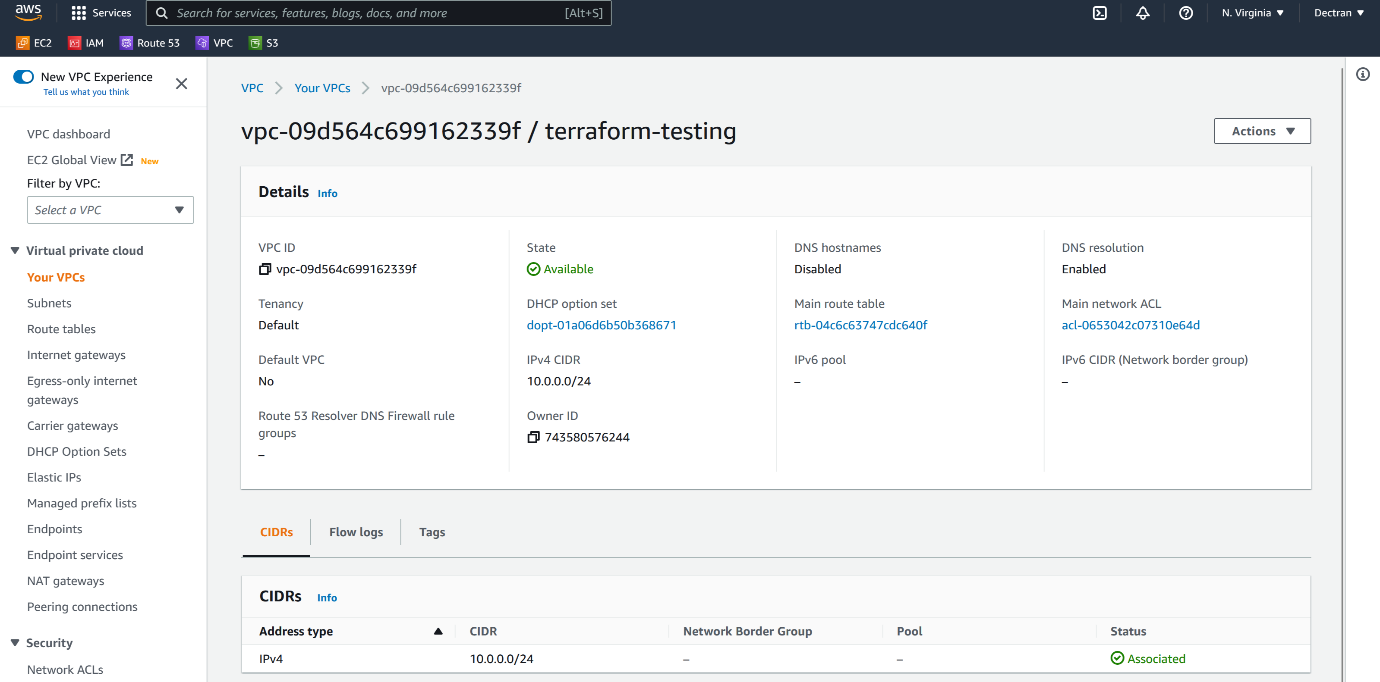
--- **note** – some times, you want to create resource and terraform does not have then in this type of scenario we will data source.

**Data Source**

--- Reference - <https://www.terraform.io/language/data-sources>

--- aws vpc datasource Reference - <https://registry.terraform.io/providers/hashicorp/aws/2.36.0/docs/data-sources/vpc>

--- **scenario** – single vpc is already created in aws cloud in some region and I was asked to create a subnet in that vpc. In this type of scenario, we will create datasource.



--- **note** – I need to create a vpc under this vpc so, copy the vpc id which will be used future usage.

**Vpc data source**

data "aws\_vpc" "terraform-testing" {

  id = "vpc-06bdd7c2d085a8b4d"

}

resource "aws\_subnet" "subnet1-public" {

    vpc\_id = "${data.aws\_vpc.terraform-testing.id}"

    cidr\_block = "10.1.0.0/16"

    availability\_zone = "us-east-1a"

    tags = {

        Name = "public-subnet-1"

    }

}

resource "aws\_subnet" "subnet2-public" {

    vpc\_id = "${data.aws\_vpc.terraform-testing.id}"

    cidr\_block = "10.2.0.0/16"

    availability\_zone = "us-east-1b"

    tags = {

        Name = "public-subnet-2"

    }

}

--- **what is data source…?**

You need to deploy something which is outside of the terraform then we will use data source. In my case we have vpc which is already created in aws and we are tasked to create subnets in that vpc the we will use data source.

**Remote Datasource**

**S3backend**

--- **note** – create bucket in s3 and enable versioning on that bucket.

--- **note** – we are going to store our state file in amazon s3 bucket.

--- s3backend.tf

terraform {

  backend "s3" {

    bucket = "stacksimplify-terraformbucket"

    key    = "myterraform.tfstate"

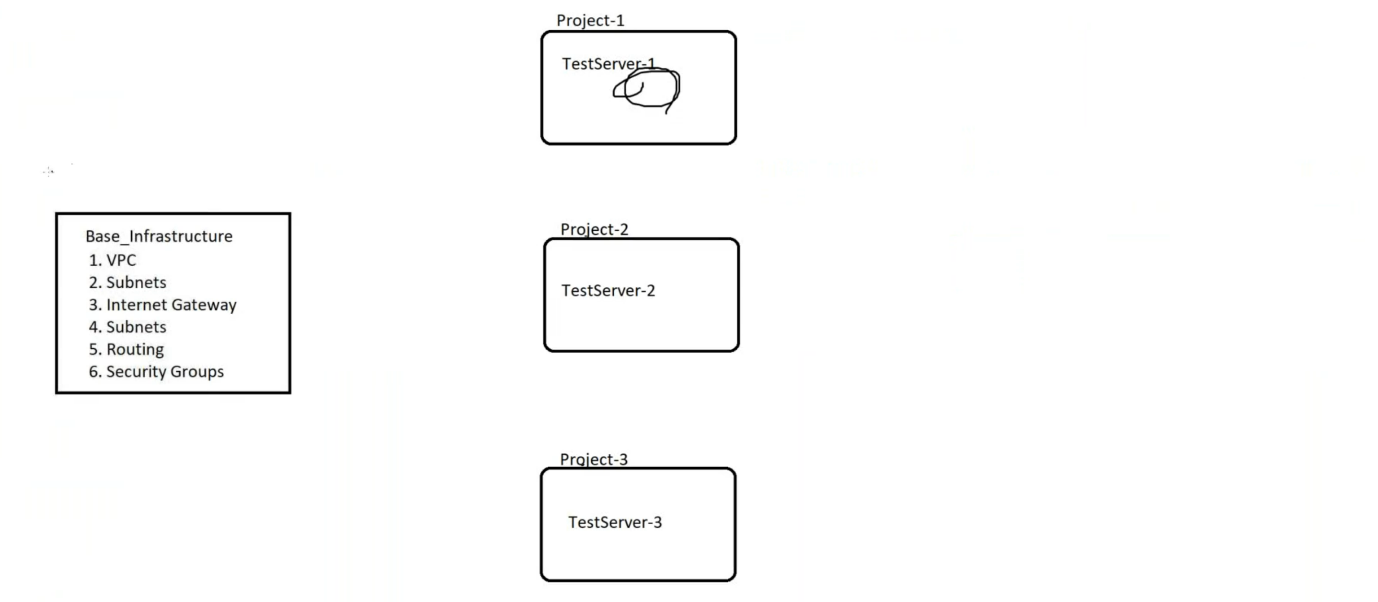
    region = "us-east-1"

  }

}

--- this way, we will save our state file in s3.

**Base infra**



--- **scenario** – I have created a base infra project and this base infra project contains vpc, subnets, internet gateway, routing, security groups.

--- I will create another project and will deploy instance in base infra subnets.

--- **main.tf**

#This Terraform Code Deploys Basic VPC Infra.

provider "aws" {

    #access\_key = "${var.aws\_access\_key}"

    #secret\_key = "${var.aws\_secret\_key}"

    region = "${var.aws\_region}"

}

terraform {

  required\_version = "<= 1.2" #Forcing which version of Terraform needs to be used

  required\_providers {

    aws = {

      version = "<= 3.0.0" #Forcing which version of plugin needs to be used.

      source = "hashicorp/aws"

    }

  }

}

terraform {

  backend "s3" {

    bucket = "stacksimplify-terraformbucket"

    key    = "baseinfra.tfstate"

    region = "us-east-1"

  }

}

resource "aws\_vpc" "default" {

    cidr\_block = "${var.vpc\_cidr}"

    enable\_dns\_hostnames = true

    tags = {

        Name = "${var.vpc\_name}"

    Owner = "Sreeharsha Veerapalli"

    environment = "${var.environment}"

    }

}

resource "aws\_internet\_gateway" "default" {

    vpc\_id = "${aws\_vpc.default.id}"

    tags = {

        Name = "${var.IGW\_name}"

    }

}

resource "aws\_subnet" "subnet1-public" {

    vpc\_id = "${aws\_vpc.default.id}"

    cidr\_block = "${var.public\_subnet1\_cidr}"

    availability\_zone = "us-east-1a"

    tags = {

        Name = "${var.public\_subnet1\_name}"

    }

}

resource "aws\_subnet" "subnet2-public" {

    vpc\_id = "${aws\_vpc.default.id}"

    cidr\_block = "${var.public\_subnet2\_cidr}"

    availability\_zone = "us-east-1b"

    tags = {

        Name = "${var.public\_subnet2\_name}"

    }

}

resource "aws\_subnet" "subnet3-public" {

    vpc\_id = "${aws\_vpc.default.id}"

    cidr\_block = "${var.public\_subnet3\_cidr}"

    availability\_zone = "us-east-1c"

    tags = {

        Name = "${var.public\_subnet3\_name}"

    }

}

resource "aws\_route\_table" "terraform-public" {

    vpc\_id = "${aws\_vpc.default.id}"

    route {

        cidr\_block = "0.0.0.0/0"

        gateway\_id = "${aws\_internet\_gateway.default.id}"

    }

    tags = {

        Name = "${var.Main\_Routing\_Table}"

    }

}

resource "aws\_route\_table\_association" "terraform-public" {

    subnet\_id = "${aws\_subnet.subnet1-public.id}"

    route\_table\_id = "${aws\_route\_table.terraform-public.id}"

}

resource "aws\_security\_group" "allow\_all" {

  name        = "allow\_all"

  description = "Allow all inbound traffic"

  vpc\_id      = "${aws\_vpc.default.id}"

  ingress {

    from\_port   = 0

    to\_port     = 0

    protocol    = "-1"

    cidr\_blocks = ["0.0.0.0/0"]

  }

  egress {

    from\_port       = 0

    to\_port         = 0

    protocol        = "-1"

    cidr\_blocks     = ["0.0.0.0/0"]

    }

}

--- **variable.tf**

variable "aws\_region" {}

variable "amis" {

    description = "AMIs by region"

    default = {

        us-east-1 = "ami-97785bed" # ubuntu 14.04 LTS

        us-east-2 = "ami-f63b1193" # ubuntu 14.04 LTS

        us-west-1 = "ami-824c4ee2" # ubuntu 14.04 LTS

        us-west-2 = "ami-f2d3638a" # ubuntu 14.04 LTS

    }

}

variable "vpc\_cidr" {}

variable "vpc\_name" {}

variable "IGW\_name" {}

variable "key\_name" {}

variable "public\_subnet1\_cidr" {}

variable "public\_subnet2\_cidr" {}

variable "public\_subnet3\_cidr" {}

variable "public\_subnet1\_name" {}

variable "public\_subnet2\_name" {}

variable "public\_subnet3\_name" {}

variable Main\_Routing\_Table {}

variable "azs" {

  description = "Run the EC2 Instances in these Availability Zones"

  default = ["us-east-1a", "us-east-1b", "us-east-1c"]

}

variable "environment" { default = "dev" }

variable "instance\_type" {

  default = {

    dev = "t2.nano"

    test = "t2.micro"

    prod = "t2.medium"

    }

}

--- **output.tf**

--- **terraform.tfvars**

#aws\_access\_key = "xxxxxx"

#aws\_secret\_key = "yyyyyyy"

aws\_region = "us-east-1"

vpc\_cidr = "10.1.0.0/16"

public\_subnet1\_cidr = "10.1.1.0/24"

public\_subnet2\_cidr = "10.1.2.0/24"

public\_subnet3\_cidr = "10.1.3.0/24"

vpc\_name = "terraform-aws-testing"

IGW\_name = "terraform-aws-igw"

public\_subnet1\_name = "Terraform\_Public\_Subnet1-testing"

public\_subnet2\_name = "Terraform\_Public\_Subnet2-testing"

public\_subnet3\_name = "Terraform\_Public\_Subnet3-testing"

Main\_Routing\_Table = "Terraform\_Main\_table-testing"

key\_name = "LaptopKey"

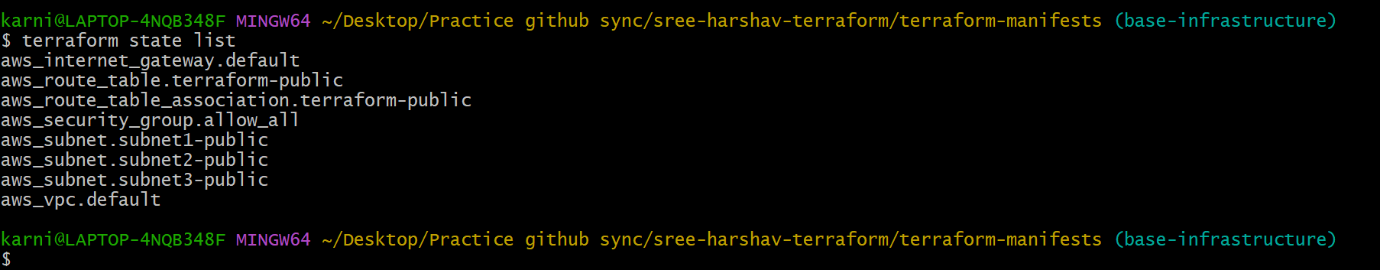
environment = "dev"

**project-1**

--- **scenario** – I want to deploy an instance in base infra subnet-1. Here I do not have base infra subnet details.

**# List all resources which are crated by terraform.**

--- terraform state show



--- **note** – there is 2 ways we can do that

1. We can import vpc details from aws using datasource.
2. We can use the state file, which is present in s3 bucket.

**1. We can import vpc details from aws using datasource.**

--- terraform data source vpc reference - <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/vpc>

--- terraform data source subnet reference - <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/subnet>

--- terraform data source security group reference - <https://registry.terraform.io/providers/hashicorp/aws/latest/docs/data-sources/security_group>

--- main.tf

 resource "aws\_instance" "web-1" {

     ami = "ami-052efd3df9dad4825"

     #ami = "ami-0d857ff0f5fc4e03b"

     #ami = "${data.aws\_ami.my\_ami.id}"

     availability\_zone = "us-east-1a"

     instance\_type = "t2.micro"

     key\_name = "terraform-key"

     subnet\_id = "${data.aws\_subnet.Terraform\_Public\_Subnet1-testing.id}"

     vpc\_security\_group\_ids = ["${data.aws\_security\_group.allow-all.id}"]

     associate\_public\_ip\_address = true

     tags = {

         Name = "Server-1"

         Env = "Prod"

         Owner = "Sree"

    CostCenter = "ABCD"

     }

 }

--- vpc-datasource.tf

data "aws\_vpc" "terraform-aws-testing" {

    id = "vpc-0eaf071b506036018"

}

data "aws\_subnet" "Terraform\_Public\_Subnet1-testing" {

    id = "subnet-0846d0b4aea9f8197"

}

data "aws\_security\_group" "allow-all" {

  id = "sg-003e1518515e691c1"

}

--- s3backend.tf

terraform {

  backend "s3" {

    bucket = "stacksimplify-terraformbucket"

    key    = "project1.tfstate"

    region = "us-east-1"

  }

}

**2. We can use the state file, which is present in s3 bucket**

--- **note** - The **terraform\_remote\_state** data source uses the latest state snapshot from a specified state backend to retrieve the root module output values from some other Terraform configuration.

You can use the terraform\_remote\_state data source without requiring or configuring a provider. It is always available through a built-in provider with the source address **terraform.io/builtin/terraform**. That provider does not include any other resources or data sources.

--- terraform data source remote state reference - <https://www.terraform.io/language/state/remote-state-data>

--- remote-state.tf

data "terraform\_remote\_state" "vpc" {

 backend = "s3"

    config = {

    key    = "baseinfra.tfstate"

    region = "us-east-1"

    bucket = "stacksimplify-terraformbucket"

  }

}

--- ec2-instance.tf

 resource "aws\_instance" "web-1" {

     ami = "ami-052efd3df9dad4825"

     #ami = "ami-0d857ff0f5fc4e03b"

     #ami = "${data.aws\_ami.my\_ami.id}"

     availability\_zone = "us-east-1a"

     instance\_type = "t2.micro"

     key\_name = "terraform-key"

     subnet\_id = data.terraform\_remote\_state.vpc.outputs.subnet\_id\_1

     vpc\_security\_group\_ids = ["${data.terraform\_remote\_state.vpc.outputs.sd\_id\_1}"]

     associate\_public\_ip\_address = true

     tags = {

         Name = "Server-1"

         Env = "Prod"

         Owner = "Sree"

    CostCenter = "ABCD"

     }

 }

--- s3backend.tf

terraform {

  backend "s3" {

    bucket = "stacksimplify-terraformbucket"

    key    = "project-2.tfstate"

    region = "us-east-1"

  }

}

--- **note** – we can use remote state file to deploy the resources on other projects.