**22.DevOps-B24-Terraform-Functions-Part-1**

--- **note** - in this session, we will discuss about terraform functions.

--- we will learn below topics in terraform functions.

1. Interpolation
2. List
3. Map
4. Count
5. Count.index
6. Lookup
7. Length
8. Element
9. Condition? True:false

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

--- go to collection functions.

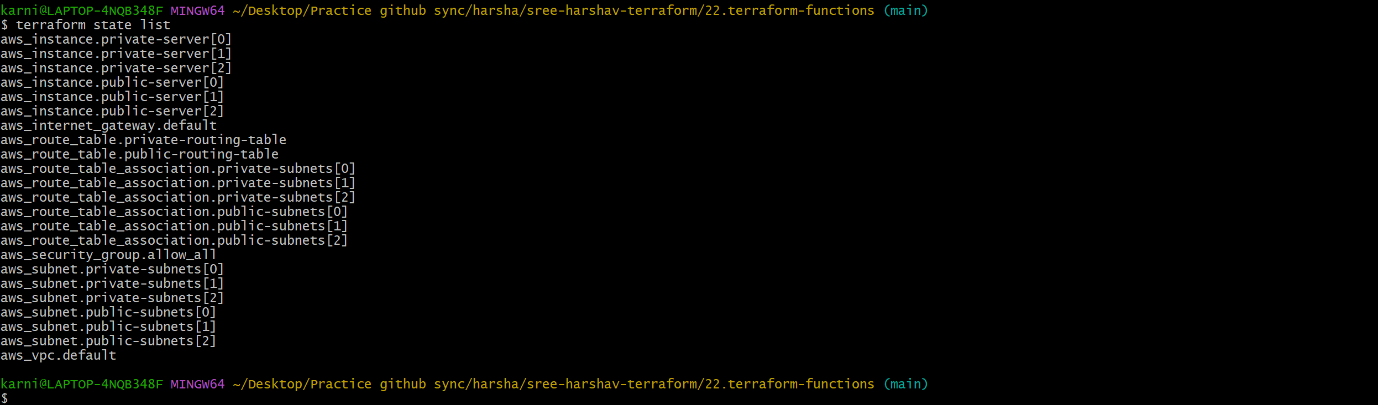
**Terraform taint**

--- **scenario** – you have already deployed public servers into cloud and you want to recreate the public server one more time. Then you can use taint

--- **note** – after applying taint to public server’s resources, then we need to do **terraform apply**. So, terraform will recreate those instances or resources.

**# List the terraform resources**

--- terraform state list



**# Apply taint to multiple resources**

--- terraform taint aws\_instance.public-server[2]

**Conditions**

--- scenario –

if ["env" = "prod" || "env" = "production"]

then

deploy 3 servers

else

deploy 1 server

fi

--- **note** – if env is prod then create 3 servers or deployed 1 server.

--- ec2.tf

resource "aws\_instance" "private-server" {

  count = "${var.environment == "prod" ? 3 : 1}"

  ami                         = lookup(var.amis, var.aws\_region)

  instance\_type               = var.instance\_type

  key\_name                    = var.key\_name

  subnet\_id                   = element(aws\_subnet.private-subnets.\*.id, count.index)

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

  associate\_public\_ip\_address = true

  tags = {

    Name        = "${var.vpc\_name}-Private-Server-${count.index + 1}"

    Deployed\_by = local.Deployed\_by

    owner       = local.owner

    costcenter  = local.costcenter

    TeamDL      = local.TeamDL

    environment = "${var.environment}"

  }

  user\_data = <<-EOF

#! /bin/bash

sudo apt-get update

sudo apt-get install -y nginx

echo "<h1>${var.vpc\_name}-private-server-${count.index}</h1>" | sudo tee /var/www/html/index.html

EOF

}

**Provisioner**

--- **scenario** – you want to change some user date of a server, that is changing form time to time. Then you can use provisioners to update the server date.

--- we need to update the data on cluster of servers.

--- null\_resources.tf

resource "null\_resource" "cluster" {

  count = var.environment == "prod" ? 3 : 1

  provisioner "file" {

    source      = "user\_data.sh"

    destination = "/tmp/user\_data.sh"

    connection  {

      type        = "ssh"

      user        = "ubuntu"

      private\_key = file("terraform-key.pem")

      host        = element(aws\_instance.public-server.\*.public\_ip, count.index)

    }

  }

}