DigitalHouse>

Certified Tech Developer
The Ultimate Degree

Infraestructura II

Actividad obligatoria e individual
Dificultad: Alta

Ejercitación Terraform - Parte 1

En las próximas clases vamos a realizar una serie de ejercicios que nos permitirán obtener un producto terminado al final de ellas. En esta primera práctica vamos a crear tres archivos:

- main.tf
- variables.tf
- output.tf

En el caso de **variables.tf**, únicamente vamos a definir una variable de tipo *String* para que solicite el nombre del recurso a crear. Esto nos permitirá crear N cantidad de recursos iguales con distintos nombres. El valor se ingresa por consola/terminal cuando ejecutamos "terraform apply".

Dentro de **outputs.tf** vamos a definir la salida. La información seleccionada va a ser a elección de cada uno, según lo que interese ver del recurso creado.

¡Importante! ¿Qué ejecutamos dentro de **main.tf**? Nos va a permitir:

- Crear una VPC.
- Definir el nombre de dicha VPC.
- Crear un Internet Gateway y un NAT Gateway.
- Dos subnets públicas y dos subnets privadas.
- Crear dos grupos de seguridad con los siguientes accesos, uno de acceso público y otro privado. El público va a contener el siguiente código. ¿Te animás a pensar cómo será el privado?

```
ingress {

  description = "SSH from the internet"

  from_port   = 22

  to_port     = 22

  protocol    = "tcp"

  cidr_blocks = ["0.0.0.0/0"]

}


egress {

  from_port   = 0

  to_port     = 0

  protocol    = "-1"

  cidr_blocks = ["0.0.0.0/0"]

}
```

A continuación vamos a mostrar la estructura del archivo para que puedan seguir completando. Vemos solo del archivo main.tf que es el más extenso:

```
data "aws_availability_zones" "available" {}


module "vpc" {


}


resource "aws_security_group" "allow_ssh_pub" {

 name        = "${var.namespace}-allow_ssh"

 description = "Allow SSH inbound traffic"
```

```
 vpc_id      = module.vpc.vpc_id


 ingress {
   description = "SSH from the internet"
   from_port   = 22
   to_port     = 22
   protocol    = "tcp"
   cidr_blocks = ["0.0.0.0/0"]
 }


 egress {
   from_port   = 0
   to_port     = 0
   protocol    = "-1"
   cidr_blocks = ["0.0.0.0/0"]
 }


 tags = {
   Name = "${var.namespace}-allow_ssh_pub"
 }
}


resource "aws_security_group" "allow_ssh_priv" {


}
```

¡Recordá que tenés que crear **outputs.tf** y **variables.tf** desde cero!

En la siguiente página se encuentra la resolución. Continúa únicamente para realizar una autoevaluación.

Resolución

Antes de comenzar, recordá tener instalado Terraform en la computadora o máquina virtual donde lo vas a ejecutar. Vamos a comenzar por el final.

Los cuatro comandos a ejecutar son:

- terraform init
- terraform plan
- terraform apply
- terraform destroy

Al momento de ejecutar, **terraform init**, únicamente va a descargar nuestros módulos desde el repositorio de Terraform:

```
Initializing modules...
Downloading terraform-aws-modules/vpc/aws 3.6.0 for vpc...
- vpc in .terraform/modules/vpc

Initializing the backend...

Initializing provider plugins...
- Finding hashicorp/aws versions matching ">= 3.28.0"...
- Installing hashicorp/aws v3.55.0...
- Installed hashicorp/aws v3.55.0 (signed by HashiCorp)

Terraform has created a lock file .terraform.lock.hcl to record the provider
selections it made above. Include this file in your version control repository
so that Terraform can guarantee to make the same selections by default when
you run "terraform init" in the future.

Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
```

Cuando ejecutamos **terraform plan**, vamos a poder comprobar que las acciones que vamos a realizar son posibles y no nos devuelve ningún tipo de error. Recordemos que este comando no realiza ninguna acción, sino que solo comprueba.

```
var.namespace
  Enter a value: digitalhouse

provider.aws.region
  The region where AWS operations will take place. Examples
  are us-east-1, us-west-2, etc.

  Enter a value: us-west-1


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:

  # aws_security_group.allow_ssh_priv will be created
  + resource "aws_security_group" "allow_ssh_priv" {
      + arn                    = (known after apply)
      + description            = "Allow SSH inbound traffic"
      + egress                 = [
          + {
              + cidr_blocks      = [
                  + "0.0.0.0/0",
                ]
              + description      = ""
              + from_port        = 0
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "-1"
              + security_groups  = []
              + self             = false
              + to_port          = 0
            },
        ]
      + id                     = (known after apply)
      + ingress                = [
          + {
              + cidr_blocks      = [
                  + "10.0.0.0/16",
                ]
              + description      = "SSH only from internal VPC clients"
              + from_port        = 22
              + ipv6_cidr_blocks = []
              + prefix_list_ids  = []
              + protocol         = "tcp"
              + security_groups  = []
              + self             = false
              + to_port          = 22
            },
        ]
      + name                   = "digitalhouse-allow_ssh_priv"
      + name_prefix            = (known after apply)
      + owner_id               = (known after apply)
      + revoke_rules_on_delete = false
      + tags                   = {
          + "Name" = "digitalhouse-allow_ssh_priv"
        }
      + tags_all               = {
          + "Name" = "digitalhouse-allow_ssh_priv"
        }
      + vpc_id                 = (known after apply)
```

Al ejecutar **terraform apply**, la salida es más completa. Para iniciar, vamos a tener que ingresar los datos solicitados en la consigna: el nombre de nuestros recursos.

```
var.namespace
  Enter a value: digitalhouse

provider.aws.region
  The region where AWS operations will take place. Examples
  are us-east-1, us-west-2, etc.

  Enter a value: us-west-1
```

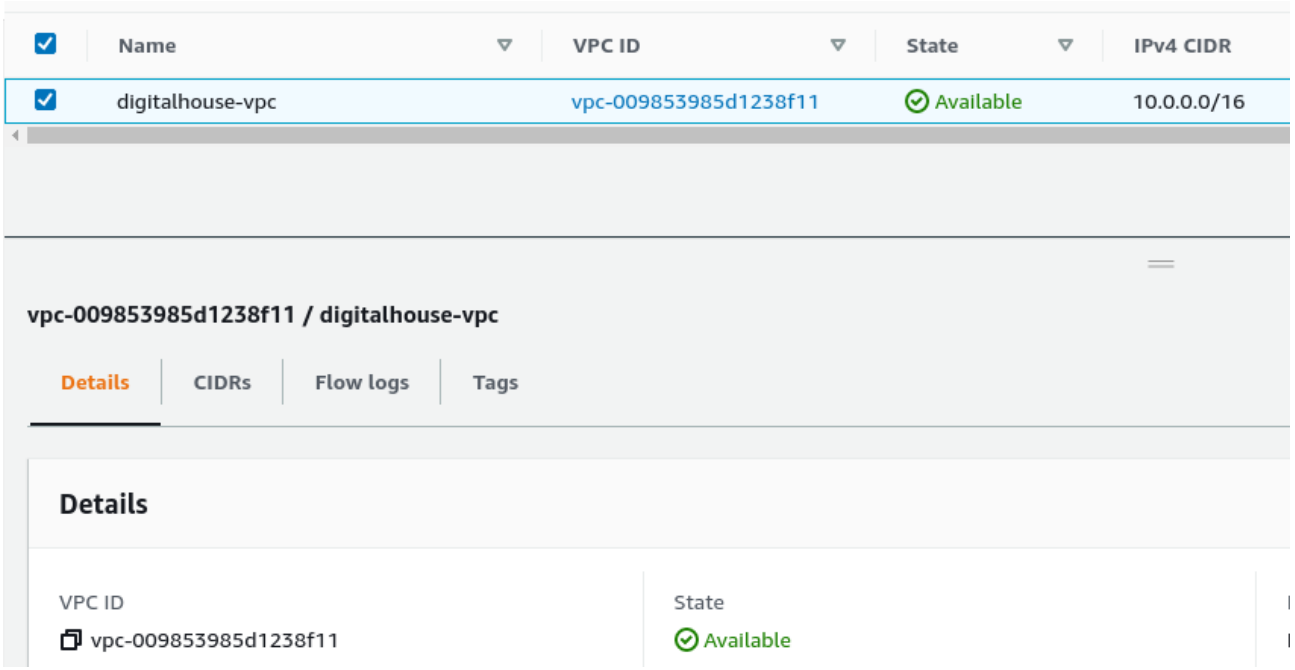
En la salida notamos que ingresamos el valor solicitado, además, el módulo oficial de aws nos pide que ingresemos la región donde lo queremos crear.

Después de realizar toda la creación de los recursos e informar el estado de cada uno, vamos a ver el resumen y la salida de "outputs.tf".

```
Apply complete! Resources: 18 added, 0 changed, 0 destroyed.

Outputs:
```

Si chequeamos nuestra consola de AWS, vamos a ver los recursos creados:



Al finalizar y para saber qué funcionó todo bien, vamos a destruir los recursos para no generar costos extras. Para hacerlo, volvemos a ingresar los mismos datos, con el fin de identificar qué recursos queremos eliminar.

```
var.namespace
  Enter a value: digitalhouse

provider.aws.region
  The region where AWS operations will take place. Examples
  are us-east-1, us-west-2, etc.

  Enter a value: us-west-1
```

Ingresamos, también, la confirmación de ello:

```
Do you really want to destroy all resources?
  Terraform will destroy all your managed infrastructure, as shown above.
  There is no undo. Only 'yes' will be accepted to confirm.

  Enter a value: yes

module.vpc.aws_route_table_association.public[0]: Destroying... [id=rtbassoc-0d501332b68018803]
module.vpc.aws_route_table_association.private[1]: Destroying... [id=rtbassoc-04f0149368e14523f]
module.vpc.aws_route_table_association.public[1]: Destroying... [id=rtbassoc-07d808319e5409403]
module.vpc.aws_route_table_association.private[0]: Destroying... [id=rtbassoc-061cbdc8c9f1657e0]
module.vpc.aws_route.public_internet_gateway[0]: Destroying... [id=r-rtb-0b8d61ff0045b420a1080289494]
aws_security_group.allow_ssh_priv: Destroying... [id=sg-00f5d4a8a3e641575]
module.vpc.aws_route.private_nat_gateway[0]: Destroying... [id=r-rtb-0d0c36258a2247e1b1080289494]
aws_security_group.allow_ssh_pub: Destroying... [id=sg-090780f6260d2898e]
module.vpc.aws_route_table_association.public[1]: Destruction complete after 1s
```

Finalmente, quedó todo eliminado:

```
module.vpc.aws_nat_gateway.this[0]: Destruction complete after 55s
module.vpc.aws_internet_gateway.this[0]: Destroying... [id=igw-07480bf36ec78e78a]
module.vpc.aws_eip.nat[0]: Destroying... [id=eipalloc-9eafce9e]
module.vpc.aws_subnet.public[1]: Destroying... [id=subnet-0538b8c7134fcd82f]
module.vpc.aws_subnet.public[0]: Destroying... [id=subnet-01950f8e7da1279d8]
module.vpc.aws_subnet.public[1]: Destruction complete after 2s
module.vpc.aws_subnet.public[0]: Destruction complete after 2s
module.vpc.aws_eip.nat[0]: Destruction complete after 2s
module.vpc.aws_internet_gateway.this[0]: Still destroying... [id=igw-07480bf36ec78e78a, 10s elapsed]
module.vpc.aws_internet_gateway.this[0]: Destruction complete after 12s
module.vpc.aws_vpc.this[0]: Destroying... [id=vpc-009853985d1238f11]
module.vpc.aws_vpc.this[0]: Destruction complete after 2s

Destroy complete! Resources: 18 destroyed.
```

El código completo de estos recursos es:

```
data "aws_availability_zones" "available" {}



module "vpc" {

 source = "terraform-aws-modules/vpc/aws"

 name                                = "${var.namespace}-vpc"

 cidr                                = "10.0.0.0/16"

 azs                                 =
data.aws_availability_zones.available.names

 private_subnets                     = ["10.0.1.0/24", "10.0.2.0/24"]

 public_subnets                      = ["10.0.101.0/24", "10.0.102.0/24"]
```

```
 create_database_subnet_group     = true
 enable_nat_gateway               = true
 single_nat_gateway               = true
}


resource "aws_security_group" "allow_ssh_pub" {
 name        = "${var.namespace}-allow_ssh"
 description = "Allow SSH inbound traffic"
 vpc_id      = module.vpc.vpc_id


 ingress {
   description = "SSH from the internet"
   from_port   = 22
   to_port     = 22
   protocol    = "tcp"
   cidr_blocks = ["0.0.0.0/0"]
 }


 egress {
   from_port   = 0
   to_port     = 0
   protocol    = "-1"
   cidr_blocks = ["0.0.0.0/0"]
 }
```

```
 tags = {
   Name = "${var.namespace}-allow_ssh_pub"
 }
}


resource "aws_security_group" "allow_ssh_priv" {
 name        = "${var.namespace}-allow_ssh_priv"
 description = "Allow SSH inbound traffic"
 vpc_id      = module.vpc.vpc_id

 ingress {
   description = "SSH only from internal VPC clients"
   from_port   = 22
   to_port     = 22
   protocol    = "tcp"
   cidr_blocks = ["10.0.0.0/16"]
 }

 egress {
   from_port   = 0
   to_port     = 0
   protocol    = "-1"
   cidr_blocks = ["0.0.0.0/0"]
 }
```

```
 tags = {

   Name = "${var.namespace}-allow_ssh_priv"

 }

}
```