



**Laboratorio**  
PRACTICA  
TERRAFORM



## CONTROL DE VERSIONES

|  |                              |
|--|------------------------------|
| <b>Elaborado por:</b> Jonatan Stiven Gutierrez | <b>No. de Versión:</b> 1.0.0 |
| <b>Revisado por:</b>                           | <b>Fecha de revisión:</b>    |
| <b>Aprobado por:</b>                           | <b>Fecha de Aprobación:</b>  |

### Historia de Modificaciones

| No. de Versión | Fecha de Versión | Autor                    | Revisado por | Aprobado por | Descripción        |
|----------------|------------------|--------------------------|--------------|--------------|--------------------|
| 1.0.0          | 21/02/2024       | Jonatan Stiven Gutierrez |              |              | Documento Original |
|                |                  |                          |              |              |                    |
|                |                  |                          |              |              |                    |
|                |                  |                          |              |              |                    |
|                |                  |                          |              |              |                    |

### Lista de distribución

| Para | Acción* | Empresa | Firma/Medio de Entrega |
|------|---------|---------|------------------------|
|      |         |         |                        |
|      |         |         |                        |
|      |         |         |                        |
|      |         |         |                        |

\* Tipos de acción: Aprobar, Revisar, Informar, Archivar, Complementar, Asistir a junta, Otras (por favor especificar)

**Este documento fue elaborado por SETI. Prohibida su reproducción total o parcial sin previa autorización del autor.**



## Contenido

|                             |          |
|-----------------------------|----------|
| <b>INTRODUCCION .....</b>   | <b>4</b> |
| <b>PRERREQUISITOS .....</b> | <b>4</b> |
| <b>EJERCICIO 11:.....</b>   | <b>5</b> |



## **INTRODUCCION**

El siguiente documento proporciona una introducción detallada a los ejercicios realizados en el laboratorio.

## **PRERREQUISITOS**

- Tener entendimiento de los temas vistos anteriormente.



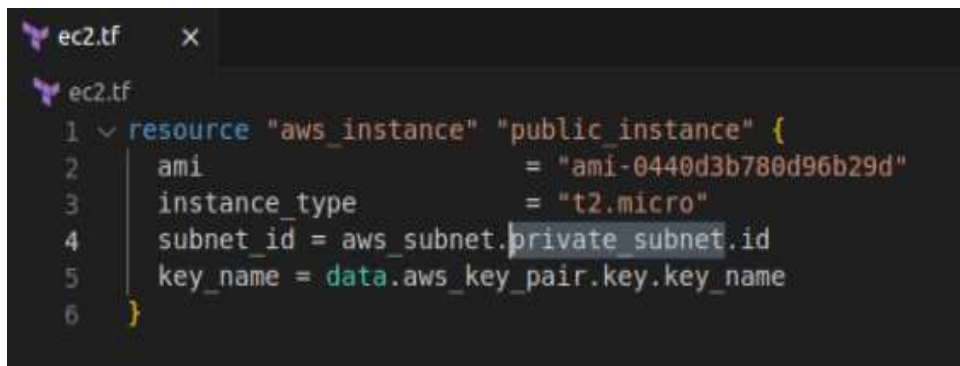
## EJERCICIO 11:

En este ejercicio vamos a trabajar **Lifecycles**.

Los lifecycles (ciclos de vida) se refieren a las fases y acciones que Terraform puede llevar a cabo durante el ciclo de vida de un recurso. Estas fases y acciones se definen en los bloques de ciclo de vida dentro de la configuración de Terraform y pueden incluir acciones como la creación, actualización o eliminación de recursos, así como acciones específicas de provisionamiento o gestión de estado.

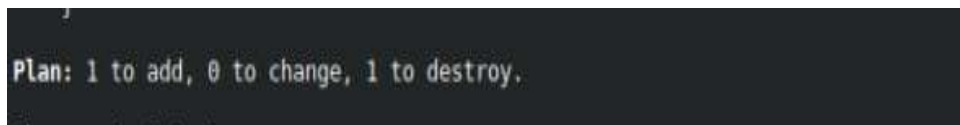
Para este hacemos una copia del ejercicio 10 con el comando: "cp -r practica\_10 practica\_11"

1. Vamos a la consola y ejecutamos "terraform apply"
2. Vamos al Visual Studio Code y nos dirigimos al archivo "ec2.tf" y modificamos lo siguiente a la subnet privada.



```
ec2.tf
ec2.tf
1 resource "aws_instance" "public_instance" {
2   ami           = "ami-0440d3b780d96b29d"
3   instance_type = "t2.micro"
4   subnet_id     = aws_subnet.private_subnet.id
5   key_name      = data.aws_key_pair.key.key_name
6 }
```

3. Vamos a la consola y ejecutamos "terraform plan".
4. Como veremos hacen 1 add y 1 destroy que es el comportamiento por defecto, pero no queremos que eso, para ello usaremos el Lifecycles.



```
Plan: 1 to add, 0 to change, 1 to destroy.
```



5. Volvemos al archivo "ec2.tf" y escribimos lo siguiente, (línea 7 - 11):

```
ec2.tf
ec2.tf
1 resource "aws_instance" "public_instance" {
2     ami                = "ami-0440d3b780d96b29d"
3     instance_type      = "t2.micro"
4     subnet_id = aws_subnet.private_subnet.id
5     key_name = data.aws_key_pair.key.key_name
6
7     lifecycle {
8         ignore_changes = [
9             subnet_id
10        ]
11    }
12 }
```

6. Vamos a la consola y ejecutamos "terraform plan".

7. Como veremos ya ignora el cambio que hicimos y nos muestra 0 cambios.

```
jonatan@ubuntu:~/Laboratorio/aws/practica_11$ terraform plan
data.aws_key_pair.key: Reading...
aws_vpc.vpc_virginia: Refreshing state... [id=vpc-0f1b3541816177a2b]
data.aws_key_pair.key: Read complete after 0s [id=key-05793e2482f866ad8]
aws_subnet.public_subnet: Refreshing state... [id=subnet-00bfcab106b9d0bca]
aws_subnet.private_subnet: Refreshing state... [id=subnet-053a160f498770f0c]
aws_instance.public_instance: Refreshing state... [id=i-0dbb1f9fbb638d438]

No changes. Your infrastructure matches the configuration.

Terraform has compared your real infrastructure against your configuration and found no differences, so no changes are needed.
jonatan@ubuntu:~/Laboratorio/aws/practica_11$
```

8. Volvemos al archivo "ec2.tf" y editamos a la public\_subnet.

```
ec2.tf
1 resource "aws_instance" "public_instance" {
2     ami                = "ami-0440d3b780d96b29d"
3     instance_type      = "t2.micro"
4     subnet_id = aws_subnet.public_subnet.id
5     key_name = data.aws_key_pair.key.key_name
6 }
```



9. Modificamos lo siguiente:

- Comentamos el espacio de "ignore\_changes" y agregamos "replace\_triggered\_by", este nos hace dependiente la instancia a la private subnet, si tiene algún cambio se reconstruirá la instancia.

```
ec2.tf  X
ec2.tf
1  resource "aws_instance" "public_instance" {
2      ami                    = "ami-0440d3b780d96b29d"
3      instance_type         = "t2.micro"
4      subnet_id = aws_subnet.public_subnet.id
5      key_name = data.aws_key_pair.key.key_name
6
7      lifecycle {
8
9          // ignore_changes = [
10             //     subnet_id
11             // ]
12
13             replace_triggered_by = [
14                 aws_subnet.private_subnet
15             ]
16         }
17     }
```

10. Para mirar su funcionalidad, vamos al archivo "vpc.tf" y editamos la private subnet:

```
18  resource "aws_subnet" "private_subnet" {
19      vpc_id = aws_vpc.vpc_virginia.id
20      cidr_block = var.subnets[1]
21      tags = {
22          Name = "Private subnet jonatan"
23      }
24      depends_on = [
25          aws_subnet.public_subnet
26      ]
27  }
```



11. Ejecutamos el comando terraform plan, y como veremos no muestra que hubo cambios en el `replace_triggered_by`:

```
jonatan@ubuntu:~/Laboratorio/aws/practica_11$ terraform plan
data.aws_key_pair.key: Reading...
aws_vpc.vpc_virginia: Refreshing state... [id=vpc-0f1b3541816177a2b]
data.aws_key_pair.key: Read complete after 1s [id=key-05793e2482f866ad8]
aws_subnet.public_subnet: Refreshing state... [id=subnet-00bfcab106b9d8bca]
aws_subnet.private_subnet: Refreshing state... [id=subnet-053a160f488770f6c]
aws_instance.public_instance: Refreshing state... [id=i-0dbb1f9fbb638d430]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  ~ update in-place
  + destroy and then create replacement

Terraform will perform the following actions:

# aws_instance.public_instance will be replaced due to changes in replace_triggered_by
/+ resource "aws_instance" "public_instance" {
```

12. Vamos al archivo `ec2.tf` y comentaríamos todo el bloque de `lifecycle`.

```
vpc.tf  ec2.tf  x
ec2.tf
1 resource "aws_instance" "public_instance" {
2   ami           = "ami-0440d3b780d96b29d"
3   instance_type = "t2.micro"
4   subnet_id     = aws_subnet.public_subnet.id
5   key_name      = data.aws_key_pair.key.key_name
6
7   //lifecycle {
8
9   //   ignore_changes = [
10    //     subnet_id
11    //   ]
12
13    //   replace_triggered_by = [
14    //     aws_subnet.private_subnet
15    //   ]
16  //}
17 }
```

13. Volvemos al archivo `vpc.tf` y editamos la private subnet:

```
18 resource "aws_subnet" "private_subnet" {
19   vpc_id = aws_vpc.vpc_virginia.id
20   cidr_block = var.subnets[1]
21   tags = {
22     Name = "Private subnet"
23   }
24   depends_on = [
25     aws_subnet.public_subnet
26   ]
27 }
```





14. Por último, terraform destroy.