



Laboratorio
PRACTICA
TERRAFORM



CONTROL DE VERSIONES

Elaborado por: Jonatan Stiven Gutierrez	No. de Versión: 1.0.0
Revisado por:	Fecha de revisión:
Aprobado por:	Fecha de Aprobación:

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

INTRODUCCION	4
PRERREQUISITOS	4
EJERCICIO 9:.....	5



INTRODUCCION

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

PRERREQUISITOS

- Tener entendimiento de los temas vistos anteriormente.



EJERCICIO 9:

En este ejercicio vamos a usar Target Resources.

En Terraform, el concepto de "Target Resources" se refiere a la capacidad de limitar las operaciones de Terraform a un conjunto específico de recursos dentro de tu configuración. Esto es útil cuando deseas realizar operaciones como la creación, actualización o destrucción solo en un subconjunto de recursos en lugar de todos los recursos definidos en tu configuración.

Hacemos una copia del ejercicio anterior, ya agregamos lo siguiente en el archivo "vpc.tf", línea 24 a la 26 (depends_on esta explicado, Tema: **Dependencias**, en el archivo "teoria.pdf").

```
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 }
```

1. Ejecutamos terraform plan
2. Ejecutamos terraform apply --auto-approve=true
 - Esto nos permite omitir el proceso de aprobación, no es recomendado hacerlo.
3. Ingresamos a la página de AWS, con el usuario IAM y veremos la creación de las dos VPC y las dos subredes.



4. Ahora nos dirigimos al archivo "vpc.tf" y editamos los nombres de las subredes agregándole un temp al final:

```
7 resource "aws_subnet" "public_subnet" {
8   vpc_id = aws_vpc.vpc_virginia.id
9   cidr_block = var.subnets[0]
10  # para asignar ip publica: map_public_ip_on_launch
11  # por defecto siempre esta en false
12  map_public_ip_on_launch = true
13  tags = {
14    Name = "Public subnet temp"
15  }
16 }
17
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 temp"
23   }
24   depends_on = [
25     aws_subnet.public_subnet
26   ]
27 }
```

5. Volvemos a la consola y ejecutamos "terraform plan" y veremos que nos marca los 2 cambios hechos.

```
Terraform will perform the following actions:

# aws_subnet.private_subnet will be updated in-place
- resource "aws_subnet" "private_subnet" {
  id = "subnet-090d230d27934793e"
  - tags = {
    - "Name" = "Private subnet" -> "Private subnet temp"
  }
  - tags_all = {
    - "Name" = "Private subnet" -> "Private subnet temp"
    # (5 unchanged elements hidden)
  }
  # (15 unchanged attributes hidden)
}

# aws_subnet.public_subnet will be updated in-place
- resource "aws_subnet" "public_subnet" {
  id = "subnet-06c94b9ad2bac5d94"
  - tags = {
    - "Name" = "Public subnet" -> "Public subnet temp"
  }
  - tags_all = {
    - "Name" = "Public subnet" -> "Public subnet temp"
    # (5 unchanged elements hidden)
  }
  # (15 unchanged attributes hidden)
}

Plan: 0 to add, 2 to change, 0 to destroy.
```



6. Ahora vamos a ejecutar un solo cambio, No los 2, para ello ejecutamos el siguiente comando:

- **"terraform apply --target aws.subnet.public_subnet"**.
- Nos indica una advertencia de solo aplicamos un cambio y no aplicamos todos los cambios.

```
}
# (1) unchanged attributes:
}

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

Warning: Resource targeting is in effect

You are creating a plan with the -target option, which means that the result of this plan may not represent all of the changes requested by the current configuration.

The -target option is not for routine use, and is provided only for exceptional situations such as recovering from errors or mistakes, or when Terraform specifically suggests to use it as part of an error message.

Do you want to perform these actions?
Terraform will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

aws_subnet.public_subnet: Modifying... [id=subnet-96c94b9ad2bac3d94]
aws_subnet.public_subnet: Modifications complete after 0s [id=subnet-96c94b9ad2bac3d94]

Warning: Applied changes may be incomplete

The plan was created with the -target option in effect, so some changes requested in the configuration may have been ignored and the output values may not be fully updated. Run the following command to verify that no other changes are pending:

    terraform plan

Note that the -target option is not suitable for routine use, and is provided only for exceptional situations such as recovering from errors or mistakes, or when Terraform specifically suggests to use it as part of an error message.

apply complete! Resources: 0 added, 1 changed, 0 destroyed.

Outputs:
ec2_public_ip = "3.93.6.7"
instance_public_ip = "3.93.6.7"
```

7. Ahora vamos a la página y verificamos las subredes y veremos el cambio.

	Name	ID de subred	Estado	VPC	CIDR IP
<input type="checkbox"/>	-	subnet-0867830cac66c5173	Available	vpc-04d045b6f594b0689	172.31.
<input type="checkbox"/>	Private subnet	subnet-090d130a27934293a	Available	vpc-031db2b33e1ca83f5	10.10.1
<input type="checkbox"/>	-	subnet-094293857d496d4e1	Available	vpc-04d045b6f594b0689	172.31.
<input type="checkbox"/>	-	subnet-067efc360a4794aef	Available	vpc-04d045b6f594b0689	172.31.
<input type="checkbox"/>	Public subnet temp	subnet-06c94b9ad2bac3d94	Available	vpc-031db2b33e1ca83f5	10.10.0
<input type="checkbox"/>	-	subnet-002e11192c3d811af	Available	vpc-04d045b6f594b0689	172.31.
<input type="checkbox"/>	-	subnet-038648e54ce1ed507	Available	vpc-04d045b6f594b0689	172.31.
<input type="checkbox"/>	-	subnet-0929ca2cb541d54	Available	vpc-04d045b6f594b0689	172.31.



8. En consola si ejecutamos un terraform plan, veremos que tenemos un cambio todavía pendiente.

```
jonatan@ubuntu:~/Laboratorio/aws/practica_01$ terraform plan
aws_vpc.vpc_virginia: Refreshing state... [id=vpc-031d02b33e1ca83f5]
aws_subnet.public_subnet: Refreshing state... [id=subnet-06c94b9ad2bac5d94]
aws_subnet.private_subnet: Refreshing state... [id=subnet-090d230d27934793e]
aws_instance.public_instance: Refreshing state... [id=i-0ebfb0f3fd052a185]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
- update in-place

Terraform will perform the following actions:

# aws_subnet.private_subnet will be updated in-place
- resource "aws_subnet" "private_subnet" {
  id = "subnet-090d230d27934793e"
  tags = {
    Name = "Private subnet" -> "Private subnet temp"
  }
  tags_all = {
    Name = "Private subnet" -> "Private subnet temp"
  }
  # ($$ unchanged elements hidden)
  # ($$ unchanged attributes hidden)
}

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

Note: You didn't use the -out option to save this plan, so Terraform can't guarantee to take exactly these actions if you run "terraform apply" now.
jonatan@ubuntu:~/Laboratorio/aws/practica_01$
```

9. Opcional, Nos dirigimos al archivo "vpc.tf" y dejamos los nombres como los teníamos al principio.

```
7 resource "aws_subnet" "public_subnet" {
8   vpc_id = aws_vpc.vpc_virginia.id
9   cidr_block = var.subnets[0]
10  # para asignar ip publica: map_public_ip_on_launch
11  # por defecto siempre esta en false
12  map_public_ip_on_launch = true
13  tags = {
14    Name = "Public subnet"
15  }
16 }
17
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 }
```




10. En consola ejecutamos "terraform plan", después ejecutamos "terraform apply", ingresamos a las subredes y verificamos que los nombres volvieron a como estaban al principio.

The screenshot shows the AWS Management Console interface for the 'Subnetes (8)' page. At the top, there is a search bar with the placeholder text 'Find resources by attribute or tag'. To the right of the search bar are buttons for 'Acciones' and 'Crear subred'. Below the search bar is a table listing the subnets. The table has columns for 'Name', 'ID de subred', 'Estado', 'VPC', and 'CIDR IP'. The subnets listed include 'Private subnet' and 'Public subnet', both with a status of 'Available'.

Name	ID de subred	Estado	VPC	CIDR IP
-	subnet-0867830ca0665173	Available	vpc-04d045b6f594bd689	172.31.
Private subnet	subnet-090d230427934795e	Available	vpc-031db2b33e1ca83f5	10.10.1
-	subnet-094293857e4a0d4e1	Available	vpc-04d045b6f594bd689	172.31.
-	subnet-067e6360a4794ea1	Available	vpc-04d045b6f594bd689	172.31.
Public subnet	subnet-06c94b9ad2ba5c094	Available	vpc-031db2b33e1ca83f5	10.10.0
-	subnet-002e11392c5d8114f	Available	vpc-04d045b6f594bd689	172.31.
-	subnet-038646e64ce1ed5f7	Available	vpc-04d045b6f594bd689	172.31.
-	subnet-0929ca02cb541d154	Available	vpc-04d045b6f594bd689	172.31.

11. Por último ejecutamos el comando terraform destroy.