Logotipo, Esquemático

Descripción generada automáticamente



**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)*

Contenido

[INTRODUCCION 4](#_Toc159921563)

[PRERREQUISITOS 4](#_Toc159921564)

[EJERCICIO 5: 5](#_Toc159921565)

# INTRODUCCION

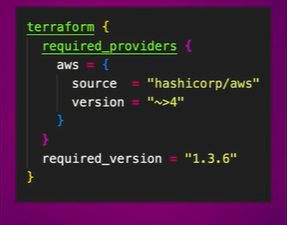
El siguiente documento proporciona una introducción detallada a los ejercicios realizados en el laboratorio.

# PRERREQUISITOS

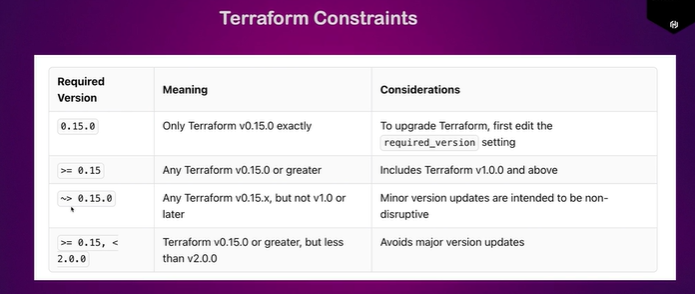
* Tener entendimiento de los temas vistos anteriormente.

# EJERCICIO 5:

Manejo de versiones de terraform o de los providers.



Terraform constraints



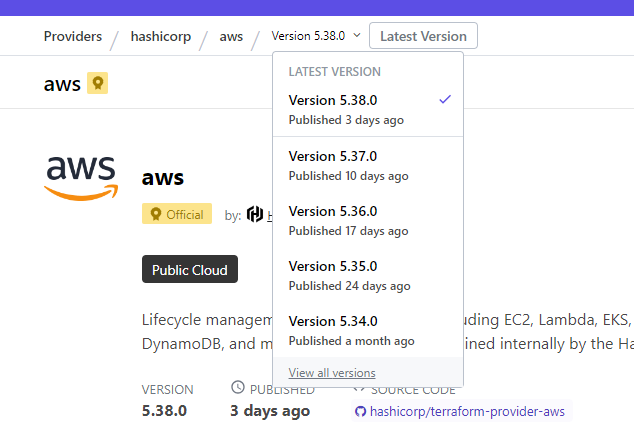
Nota: Lo recomendado por Terraform es usar la opción 3: “**~>**”.

En este ejercicio vamos a trabajar las VPC.

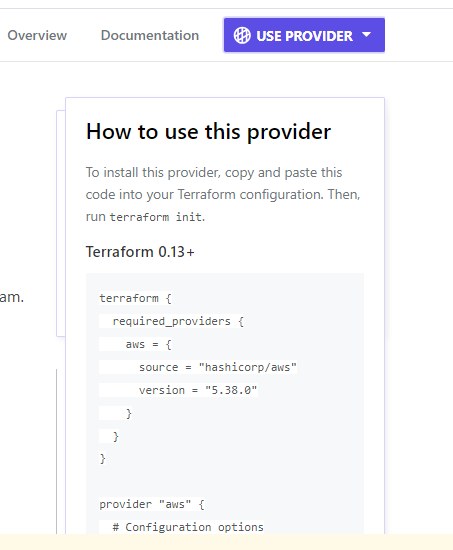
VPC es un servicio de redes virtuales que te permite crear y administrar redes privadas en la nube de AWS. Puedes utilizar Terraform para definir VPCs, subredes, tablas de ruteo, grupos de seguridad y otros componentes de red.

Nota: Este ejemplo no tiene costo alguno de uso de nube.

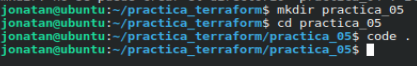
1. Vamos a la página de documentación, al provider de AWS.
   * <https://registry.terraform.io/providers/hashicorp/aws/latest>
2. Estando en la página, podemos mirar que versión usar.



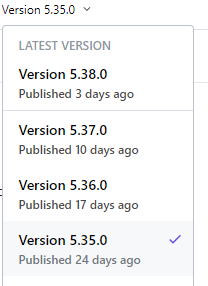
1. Al darle al botón de “USE PROVIDER”, nos muestra un ejemplo de cómo usar una determinada versión.



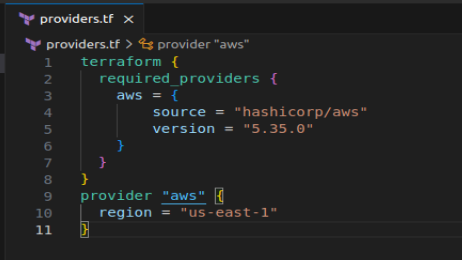
1. Vamos a crear una nueva carpeta, para el ejercicio 5.



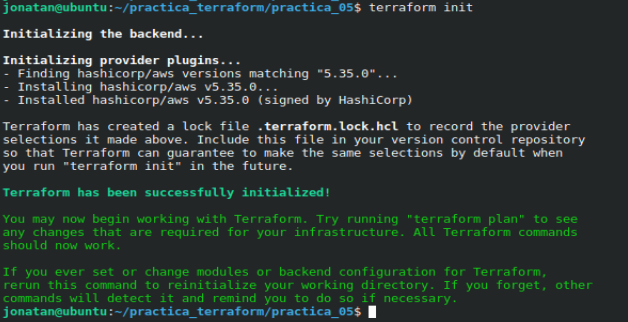
1. Estando en Visual Studio Code, creamos un archivo “providers.tf”, Vamos a la página visitada anteriormente seleccionamos una versión un poco más antigua a la última disponible en el vamos a copiar el ejemplo que nos da al seleccionar “USE PROVIDER” y le añadimos la región en el provider “aws” (más adelante veras el ejemplo).
   * En mi caso usare la “5.38.0” del provider aws.



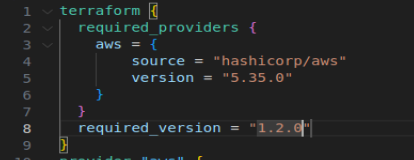
1. El archivo en Visual Studio Code, nos debe quedar así con el agregado de la región.



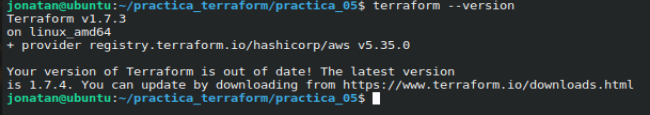
1. Vamos a la consola, estando en la carpeta ejecutamos “terraform init”.
   * Como vemos, nos descarga la versión que le pedimos anteriormente del provider de aws “5.35.0”.



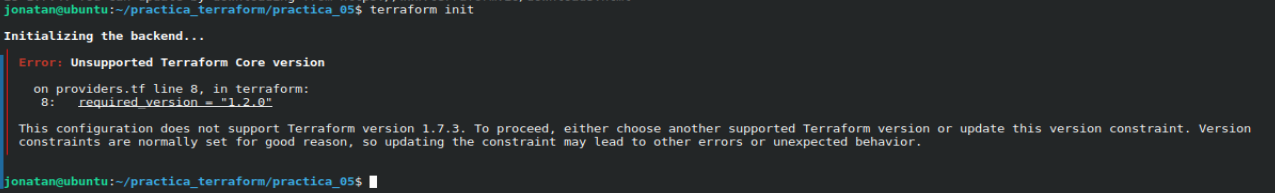
1. Ahora vamos a requerir una versión de terraform especifica.
2. Vamos al Visual Studio Code y vamos a editar el archivo “providers.tf” y añadimos la línea 8 que se ve en la siguiente imagen.



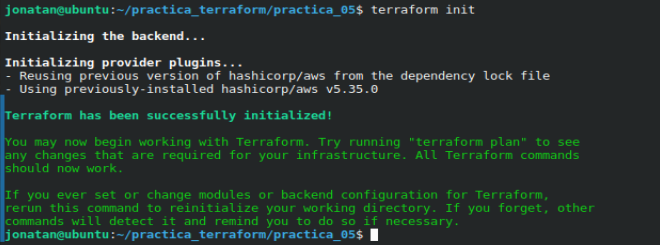
1. Volvemos a la consola y ejecutamos el comando “terraform --version”
   * En mi caso tengo la version “1.7.3”



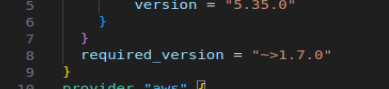
1. Ejecutamos el comando “terraform init”.
   * Esto nos va a generar un error en caso de que sean diferentes, la versión instala y la versión requerida en el código.



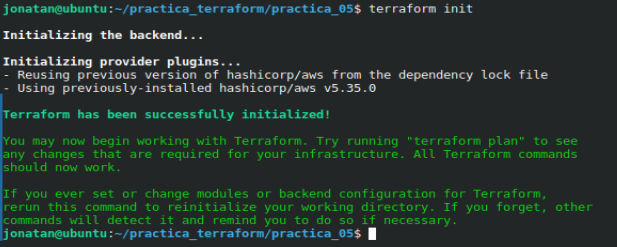
1. Vamos al Visual Studio Code y editamos el archivo “terraform.tf”, y colocamos la versión local instalada (terraform --version).
2. Ejecutamos el comando “terraform init”.
   * Salió todo correcto.



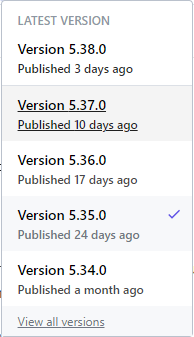
1. Ahora vamos a manejar los Terraform constraints.
   * “~>1.7.3”, esto nos quiere decir que cualquier subversión en la 1.7.0, por ejemplo “1.7.1”, “1.7.2”, son aceptadas.
   * Vamos al Visual Studio Code y editamos el archivo “terraform.tf”
   * En mi caso tengo la version “1.7.3” instalada de manera local, pero vamos a escribir “~>1.7.0” para decirle que acepta todas sus subversiones.



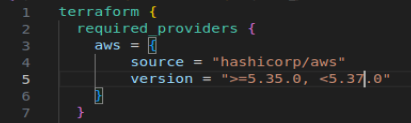
1. Volvemos a la consola, ejecutamos el comando “terraform init”.
   * Como veremos, todo funciona bien.



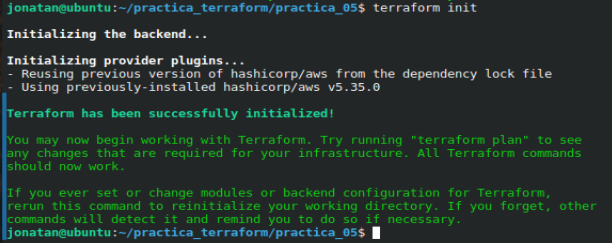
1. Ahora vamos a agregar condicionales con la versión del provider de aws.



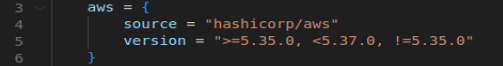
* + La versión debe ser mayor o igual a la “5.35.0” y menor a la “5.37.0”



1. Ahora vamos a la consola, y ejecutamos de nuevo “terraform init”.



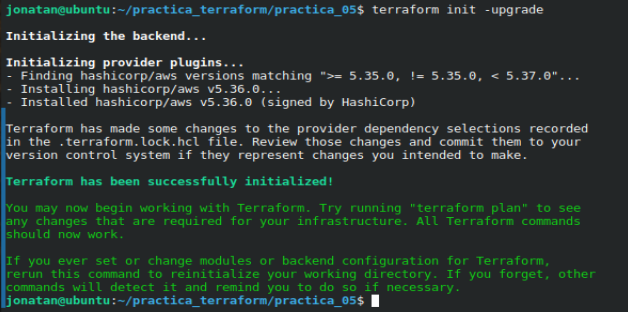
* + Como vemos sigue con la misma versión que teníamos, pero cumpliendo con la condicional que pusimos.
  + Ahora en el condicional vamos a colocar que sea una diferente a la que tenemos, en este caso tenemos la “5.35.0”, pero que siga con los parámetros anteriores.



* + Volvemos a la consola, y ahora ejecutamos “terraform init -upgrade”

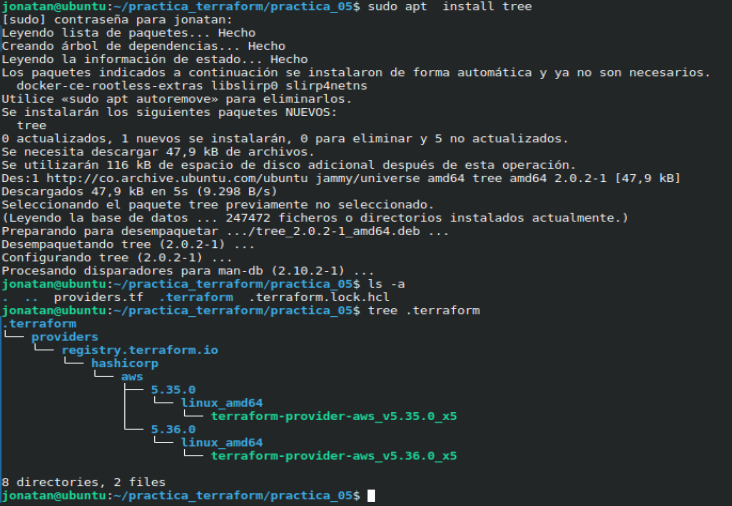
Nota: terraform init –upgrade : se utiliza en terraform para actualizar los plugins y módulos a la versión compatible con el proyecto.

* + Volvemos a la consola, y ahora ejecutamos “terraform init -upgrade”.

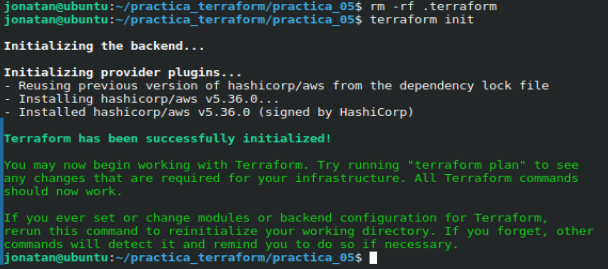


* + Como vemos se instaló la versión “5.36.0”.

1. Ahora ejecutamos los siguientes comandos:
   * sudo apt install tree
   * ls –a
   * tree .terraform
   * Veremos las versiones del provider aws instaladas.



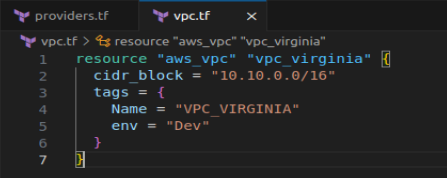
1. Ahora si queremos borrar, y dejar solo una versión, hacemos lo siguiente:
   * rm –rf .terraform
   * terraform init
   * tree .terraform
   * Veremos que ahora solo tendremos una versión del provider aws



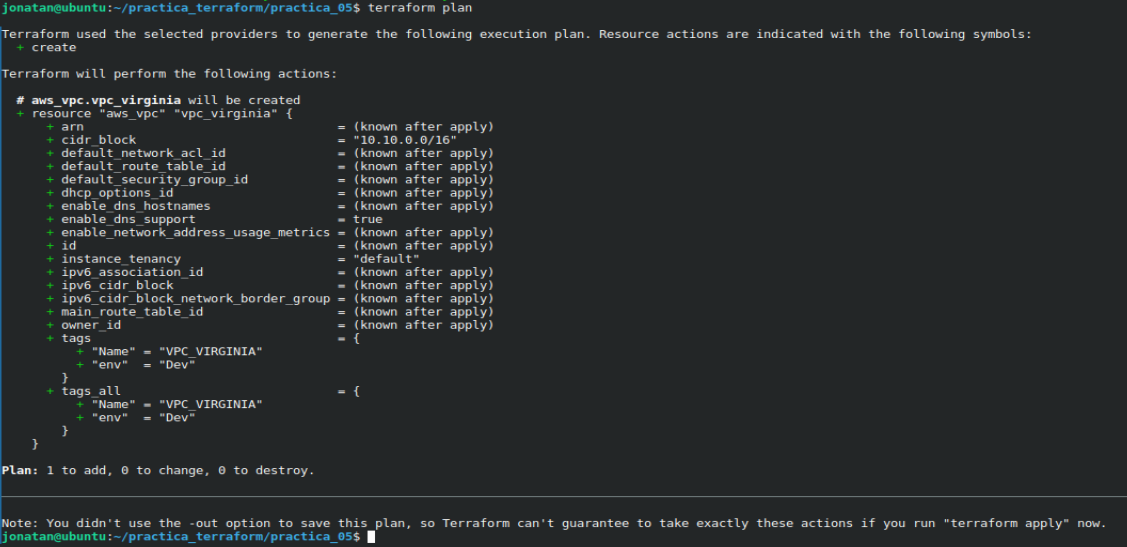
1. Ahora vamos a usar VPC, para ello nos dirigimos a la documentación:
   * [https://registry.terraform.io/providers/hashicorp/aws/5.35.0/docs/resources/vpc](https://registry.terraform.io/providers/hashicorp/aws/5.35.0/docs/resources/vpc:)
2. Copiamos el ejemplo básico:



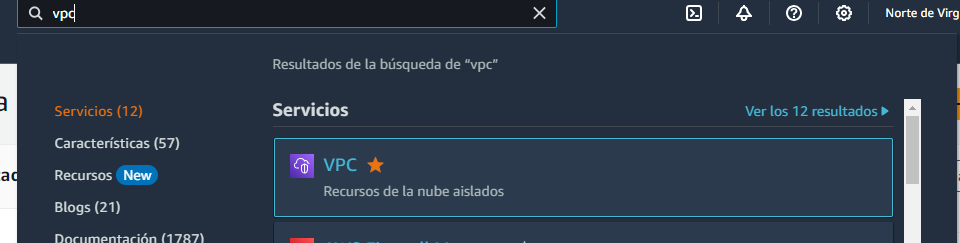
1. Creamos un archivo “vpc.tf”, pegamos lo anteriormente copiado y lo editamos de la siguiente manera.



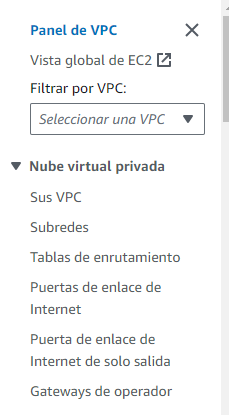
1. Nos dirigimos a la consola y ejecutamos “terraform plan”.



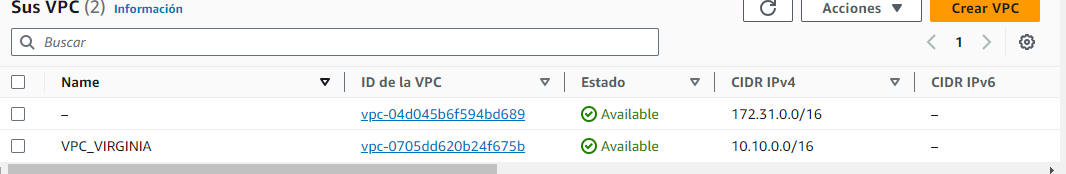
1. Vamos a la página de AWS, ingresamos con el usuario IAM, buscamos VPC, le damos en la estrella, y damos click en VPC.



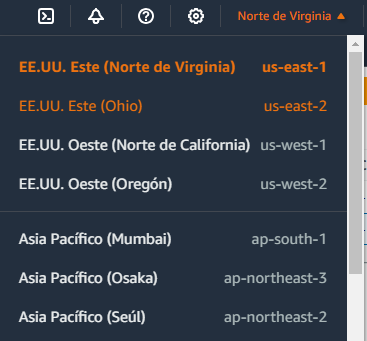
1. Ahora en el menú del lado izquierdo, damos click en sus VPC.



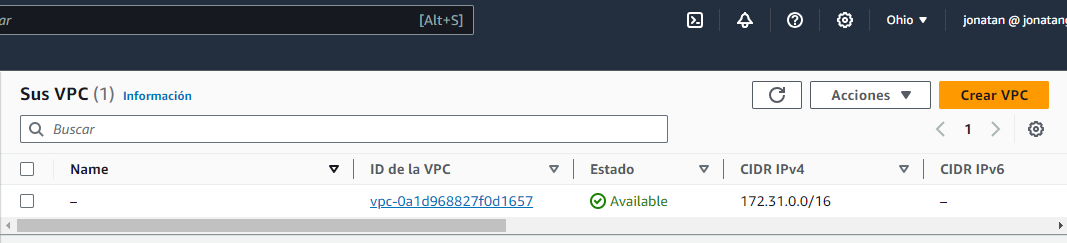
1. Volvemos a la consola, y ejecutamos el comando “terraform apply”.
2. En la página de aws, actualizamos y veremos la VPC creada con una vpc predeterminada.



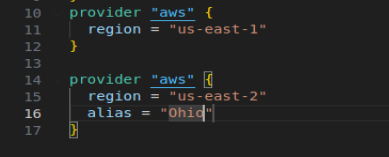
1. Ahora queremos cambiar de región, damos click en (Ohio).



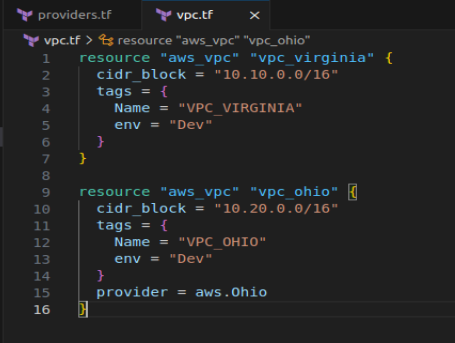
1. Ahora queremos cambiar de región, damos click en (Ohio), veremos que ya no está la anterior VPC creada y solo nos muestra la predeterminada.



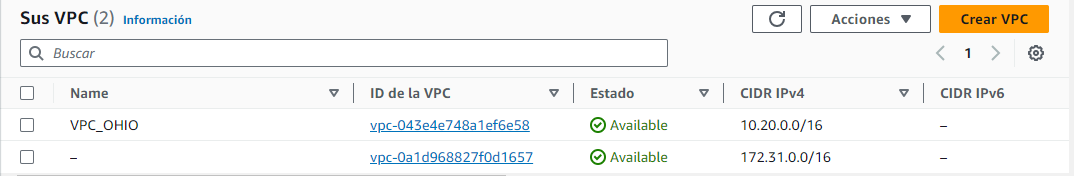
1. Para crear una VPC en otra región al tiempo, haremos lo siguiente, nos dirigimos a Visual Studio Code en el archivo “providers.tf” y agregamos lo siguiente (línea 14-17).



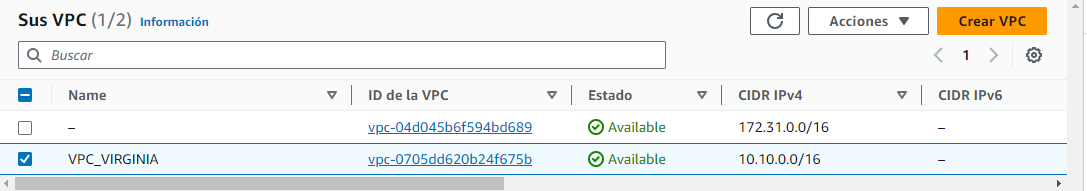
1. Ahora nos dirigimos al archivo “vpc.tf” y nos debe quedar así:
   * Cambiamos el id
   * Editamos la subnet, para evitar problemas si las llegamos a conectar.
   * Cambiamos el tag Name
   * Agregamos la etiqueta provider = aws.Ohio



1. Ahora nos dirigimos a consola, ejecutamos:
   * terraform plan
   * terraform apply
2. Ahora nos dirigimos a la página de aws, en la región de Ohio actualizamos:



1. Ahora nos dirigimos a la región de Virginia y vemos que la vpc de esta región sigue ahí.



1. Ahora aplicamos “terraform destroy”.

Nota: Las VPC no generan ningún costo a menos de que sean más de 5 por región.