

# Reporte de Laboratorio: Desafío de Terraform en Google Cloud (GSP345)

Autor: Christhian Rodriguez

Fecha: 14 de Diciembre, 2025

Tema: Infraestructura como Código, Importación de Recursos, Módulos de Registro y Troubleshooting Avanzado.

## 1. Introducción

En este laboratorio de desafío ("Challenge Lab"), el objetivo fue demostrar la competencia en Terraform para construir, importar y gestionar infraestructura en Google Cloud bajo un escenario semi-realista sin instrucciones paso a paso.

El proyecto abarcó el ciclo de vida completo de la infraestructura:

- Inicialización:** Configuración de un entorno modular.
- Importación:** Toma de control de recursos "legacy" (creados manualmente).
- Gestión de Estado:** Migración a un backend remoto seguro.
- Redes y Seguridad:** Implementación de VPC modular y Firewalls.
- Troubleshooting:** Resolución de conflictos de versiones de proveedores y condiciones de carrera (race conditions).

## 2. Tarea 1: Inicialización e Importación de Infraestructura

El primer reto consistió en tomar control de dos instancias de Compute Engine ( `tf-instance-1` y `tf-instance-2` ) que ya existían en el proyecto pero no estaban gestionadas por Terraform.

### 2.1 Configuración del Entorno (Scaffolding)

Se creó la estructura de directorios modular ( `instances` , `storage` ) y se definieron los archivos de configuración iniciales, estableciendo las variables de región y zona.

```
CLOUD SHELL
Terminal (qwiklabs-gcp-04-d687c34755a8) x + v

(i) Gemini CLI is available in Cloud Shell terminal! Type gemini to try it. Learn more

student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ touch main.tf variables.tf
mkdir -p modules/instances modules/storage
touch modules/instances/instances.tf modules/instances/outputs.tf modules/instances/variables.tf
touch modules/storage/storage.tf modules/storage/outputs.tf modules/storage/variables.tf
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ terraform init

Initializing the backend...
Initializing modules...
- instances in modules/instances

Initializing provider plugins...
- Finding hashicorp/google versions matching "4.51.0"...
- Installing hashicorp/google v4.51.0...
- Installed hashicorp/google v4.51.0 (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.
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$
```

Fig 1. Scaffolding: Inicialización del proyecto con `terraform init` y creación de la estructura de archivos ( `main.tf` , `variables.tf` , módulos).

## 2.2 Importación de Estado (State Import)

Utilizando ingeniería inversa, escribimos el código `instances.tf` para coincidir con la configuración real ( `e2-micro` , `debian-11` ). Usamos el comando `terraform import` para vincular los IDs de las instancias reales con nuestro código local.

```
commands will detect it and remind you to do so if necessary.
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ terraform import module.instances.google_compute_instance.tf_instance_1 734460039423757792
module.instances.google_compute_instance.tf_instance_1: Importing from ID "734460039423757792"...
module.instances.google_compute_instance.tf_instance_1: Import prepared!
Prepared google_compute_instance for import
module.instances.google_compute_instance.tf_instance_1: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/734460039423757792]

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.

student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ terraform import module.instances.google_compute_instance.tf_instance_2 1940607020735565280
module.instances.google_compute_instance.tf_instance_2: Importing from ID "1940607020735565280"...
module.instances.google_compute_instance.tf_instance_2: Import prepared!
Prepared google_compute_instance for import
module.instances.google_compute_instance.tf_instance_2: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/1940607020735565280]

Import successful!

The resources that were imported are shown above. These resources are now in
your Terraform state and will henceforth be managed by Terraform.

student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$
```

Fig 2. Importación: Ejecución exitosa de `terraform import` , transfiriendo la gestión de las instancias al estado de Terraform ( `Import successful!` ).

*Nota:* Al agregar el script de inicio ( `metadata_startup_script` ), Terraform detectó correctamente que debía recrear las instancias para aplicar este cambio estructural, asegurando la inmutabilidad de la infraestructura.

### 3. Tarea 2: Backend Remoto y Modificación de Recursos

Para asegurar el estado, creamos un bucket de Cloud Storage y configuramos el bloque backend "gcs", migrando el estado local ( terraform.tfstate ) a la nube para permitir colaboración y bloqueo de estado.

#### 3.1 Escalado Vertical y Horizontal

Se modificó el tipo de máquina de las instancias existentes a e2-standard-2 (escalado vertical) y se agregó una tercera instancia tf-instance-752620 (escalado horizontal).

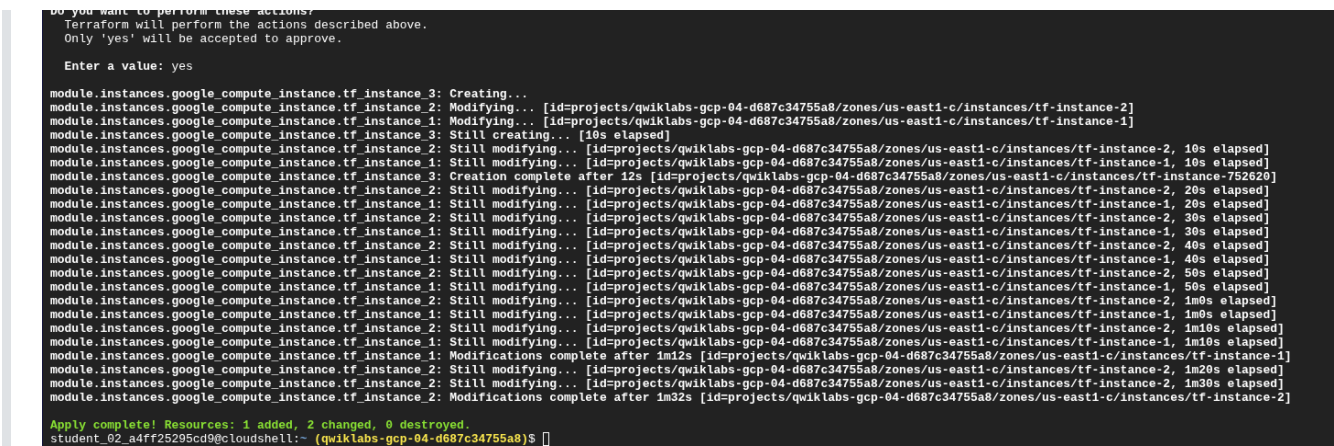


Fig 3. Ejecución de Cambios: Terraform aplica cambios "in-place" para actualizar las instancias existentes y crea la tercera nueva.

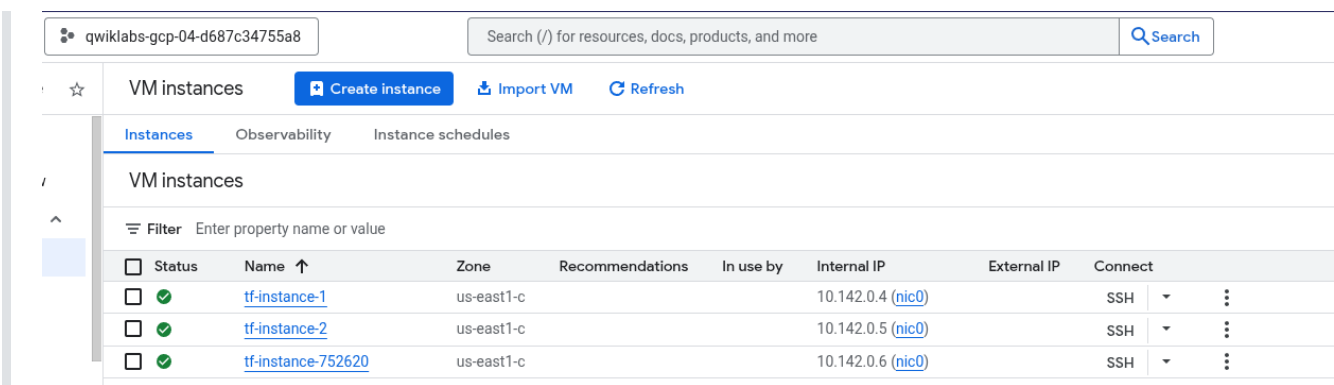


Fig 4. Verificación: Las tres instancias corriendo correctamente con la nueva configuración en la consola de Compute Engine.

#### 3.2 Ciclo de Vida (Destrucción Controlada)

Siguiendo los requerimientos del desafío, eliminamos la tercera instancia mediante código para verificar la capacidad de limpieza de recursos y gestión de costos.

```

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

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

module.instances.google_compute_instance.tf_instance_3: Destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 10s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 20s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 30s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 40s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 50s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m0s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m10s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m20s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m30s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m40s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 1m50s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 2m0s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 2m10s elapsed]
module.instances.google_compute_instance.tf_instance_3: Still destroying... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620, 2m20s elapsed]
module.instances.google_compute_instance.tf_instance_3: Destruction complete after 2m22s

Apply complete! Resources: 0 added, 0 changed, 1 destroyed.
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8) $

```

**Fig 5. Destroy:** Eliminación controlada de la tercera instancia ( 1 to destroy ), confirmando el control total del ciclo de vida.

## 4. Tarea 3: Redes y Troubleshooting (Análisis de Incidentes)

Esta sección presentó la mayor complejidad técnica. Se requería implementar una VPC personalizada usando un módulo del Terraform Registry y configurar un Firewall. Durante este proceso, enfrentamos y resolvimos dos incidentes críticos.

### 4.1 Incidente 1: Conflicto de Dependencias (Dependency Hell)

Al intentar usar el módulo de red `terraform-google-modules/network/google` versión `10.0.0` , la inicialización falló.

- **Síntoma:** Error `Inconsistent dependency lock file` .
- **Causa Raíz:** El archivo `main.tf` original bloqueaba el proveedor `google` a la versión `4.51.0` , pero el módulo de red moderno requería una versión superior ( `>= 6.0` ).
- **Resolución:** Se eliminó el archivo de bloqueo con `rm .terraform.lock.hcl` y se forzó la actualización de proveedores con `terraform init -upgrade` .

```

student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ terraform apply -auto-approve

Error: Inconsistent dependency lock file

The following dependency selections recorded in the lock file are inconsistent with the current configuration:
- provider registry.terraform.io/hashicorp/google-beta: required by this configuration but no version is selected
- provider registry.terraform.io/hashicorp/google: locked version selection 4.51.0 doesn't match the updated version constraints ">= 3.33.0, >= 3.83.0, >= 4.25.0, 4.51.0, >= 4.64.0, < 7.0.0"

To update the locked dependency selections to match a changed configuration, run:
  terraform init -upgrade

student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ rm .terraform.lock.hcl
terraform init -upgrade

Initializing the backend...
Upgrading modules...
- instances in modules/instances
- storage in modules/storage
  Downloading registry.terraform.io/terraform-google-modules/network/google 10.0.0 for vpc...
- vpc in .terraform/modules/vpc
- vpc.firewall_rules in .terraform/modules/vpc/modules/firewall-rules
- vpc.routes in .terraform/modules/vpc/modules/routes
- vpc.subnets in .terraform/modules/vpc/modules/subnets
- vpc.vpc in .terraform/modules/vpc/modules/vpc

Initializing provider plugins...
- Finding hashicorp/google-beta versions matching ">= 4.64.0, >= 6.13.0, < 7.0.0"...
- Finding hashicorp/google versions matching ">= 3.33.0, >= 3.83.0, >= 4.25.0, >= 4.51.0, >= 4.64.0, < 7.0.0"...
- Installing hashicorp/google-beta v6.50.0...
- Installed hashicorp/google-beta v6.50.0 (signed by HashiCorp)
- Installing hashicorp/google v6.50.0...
- Installed hashicorp/google v6.50.0 (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.
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8)$ terraform apply -auto-approve
module.instances.google_compute_instance.tf_instance_1: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-1]
module.storage.google_storage_bucket.storage_bucket: Refreshing state... [id=tf-bucket-534723]
module.instances.google_compute_instance.tf_instance_2: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-752620]
module.instances.google_compute_instance.tf_instance_2: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-2]

Terraform used the selected providers to generate the following execution plan. Resource actions are indicated with the following symbols:
  create

```

**Fig 6. Diagnóstico y Solución:** Captura del error de bloqueo y la ejecución de los comandos de remediación en la terminal.

## 4.2 Incidente 2: Condición de Carrera (Race Condition)

Al aplicar la configuración de red y firewall simultáneamente, ocurrió un error `404 Not Found` al crear la regla de firewall.

- **Causa Raíz:** Terraform intentó crear el recurso `google_compute_firewall` antes de que el módulo de VPC terminara de aprovisionar la red `tf-vpc-810706`. Aunque Terraform paraleliza operaciones, no detectó esta dependencia implícita a través del módulo.
- **Resolución:** Se agregó explícitamente el meta-argumento `depends_on = [module.vpc]` en el recurso del firewall para forzar el orden de creación secuencial.

```
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8) $ terraform apply -auto-approve
module.vpc.module.vpc.google_compute_network.network: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/global/networks/tf-vpc-810706]
module.instances.google_compute_instance.tf_instance_1: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-1]
module.instances.google_compute_instance.tf_instance_2: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/zones/us-east1-c/instances/tf-instance-2]
module.storage.google_storage_bucket.storage-bucket: Refreshing state... [id=tf-bucket-534723]
module.vpc.module.subnets.google_compute_subnetwork.subnetwork["us-east1/subnet-01"]: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/regions/us-east1/subnetworks/subnet-01]
module.vpc.module.subnets.google_compute_subnetwork.subnetwork["us-east1/subnet-02"]: Refreshing state... [id=projects/qwiklabs-gcp-04-d687c34755a8/regions/us-east1/subnetworks/subnet-02]

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:

# google_compute_firewall.tf-firewall will be created
+ resource "google_compute_firewall" "tf-firewall" {
  + creation_timestamp = (known after apply)
  + destination_ranges = (known after apply)
  + direction          = (known after apply)
  + enable_logging     = (known after apply)
  + id                = (known after apply)
  + name              = "tf-firewall"
  + network            = "projects/qwiklabs-gcp-04-d687c34755a8/global/networks/tf-vpc-810706"
  + priority           = 1000
  + project            = "qwiklabs-gcp-04-d687c34755a8"
  + self_link          = (known after apply)
  + source_ranges      = [
    + "0.0.0.0/0",
  ]
  + allow {
    + ports = [
      + "80",
    ]
    + protocol = "tcp"
  }
}

Plan: 1 to add, 0 to change, 0 to destroy.
google_compute_firewall.tf-firewall: Creating...
google_compute_firewall.tf-firewall: Still creating... [10s elapsed]
google_compute_firewall.tf-firewall: Creation complete after 11s [id=projects/qwiklabs-gcp-04-d687c34755a8/global/firewalls/tf-firewall]

Apply complete! Resources: 1 added, 0 changed, 0 destroyed.
student_02_a4ff25295cd9@cloudshell:~ (qwiklabs-gcp-04-d687c34755a8) $
```

**Fig 7. Despliegue Final:** Ejecución exitosa creando el Firewall ( 1 to add ) después de asegurar que la red y las instancias ya estaban listas.

### 4.3 Resultado Final de Red

Las instancias fueron movidas exitosamente de la red default a la nueva red privada tf-vpc-810706 con sus respectivas subredes.

SMTP port 25 disallowed in this project. [Learn more](#)

VPC networks

Filter

Enter property name or value

| <input type="checkbox"/> | Name ↑        | Subnets | MTU ⓘ | Mode   | IPv6 ULA range | Gateways | Flow log configs ⓘ | Firewall rules | Global dynamic routing | Netw |
|--------------------------|---------------|---------|-------|--------|----------------|----------|--------------------|----------------|------------------------|------|
| <input type="checkbox"/> | default       | 24      | 1460  | Auto   |                |          |                    | 4              | Off                    |      |
| <input type="checkbox"/> | tf-vpc-810706 | 2       | 1460  | Custom |                |          |                    | 1              | On                     |      |

**Fig 8. Arquitectura de Red:** La VPC personalizada desplegada correctamente con modo de subred personalizado.

### VPC firewall rules

Firewall rules control incoming and outgoing traffic to an instance. By default, all incoming traffic to your network is blocked. [Learn more](#)

Note: App Engine firewalls are managed in the [App Engine Firewall rules section](#)

SMTP port 25 disallowed in this project. [Learn more](#)

Refresh

Configure logs

Delete

Filter

Enter property name or value

| <input type="checkbox"/> | Name                                   | Type    | Targets      | Filters    | Protocols / ports                  | Action | Priority | Network                       | Logs | Hit count | Last hit |   |
|--------------------------|--|---------|--------------|------------|------------------------------------|--------|----------|-------------------------------|------|-----------|----------|---|
| <input type="checkbox"/> | <a href="#">default-allow-icmp</a>     | Ingress | Apply to all | IP ranges: | icmp                               | Allow  | 65534    | <a href="#">default</a>       | Off  | —         | —        | ▼ |
| <input type="checkbox"/> | <a href="#">default-allow-internal</a> | Ingress | Apply to all | IP ranges: | tcp:0-65535<br>udp:0-65535<br>icmp | Allow  | 65534    | <a href="#">default</a>       | Off  | —         | —        | ▼ |
| <input type="checkbox"/> | <a href="#">default-allow-rdp</a>      | Ingress | Apply to all | IP ranges: | tcp:3389                           | Allow  | 65534    | <a href="#">default</a>       | Off  | —         | —        | ▼ |
| <input type="checkbox"/> | <a href="#">default-allow-ssh</a>      | Ingress | Apply to all | IP ranges: | tcp:22                             | Allow  | 65534    | <a href="#">default</a>       | Off  | —         | —        | ▼ |
| <input type="checkbox"/> | <a href="#">tf-firewall</a>            | Ingress | Apply to all | IP ranges: | tcp:80                             | Allow  | 1000     | <a href="#">tf-vpc-810706</a> | Off  | —         | —        | ▼ |

### Network firewall policies

Firewall policies let you group several firewall rules so that you can update them all at once, effectively controlled by Identity and Access Management (IAM) roles. [Learn more](#)

**Fig 9. Seguridad:** La regla de firewall `tf-firewall` permitiendo tráfico TCP:80 y apuntando correctamente a la nueva red VPC.

## 5. Lecciones Aprendidas

- Gestión de Dependencias:** Los módulos externos del Registry evolucionan rápido. Es crítico gestionar el archivo `.terraform.lock.hcl` y entender cuándo usar `init -upgrade` para resolver conflictos entre versiones del proveedor y módulos.
- Orquestación de Recursos:** Terraform paraleliza la creación de recursos. Cuando existen dependencias que Terraform no detecta automáticamente (especialmente al mezclar recursos nativos con módulos complejos), el uso de `depends_on` es vital para evitar condiciones de carrera.
- Inmutabilidad vs Modificación:** Mover instancias de una red a otra implica recreación o reinicio de las mismas. Este comportamiento debe ser planificado en entornos productivos para evitar tiempos de inactividad no programados.

## 6. Anexo: Cheat Sheet de Comandos Utilizados

Referencia rápida de los comandos de CLI ejecutados durante el laboratorio y su propósito.



| Comando                                    | Propósito   | Contexto de Uso   |
|--|---|---|
| <code>terraform init</code>                | Inicializa el directorio de trabajo, descarga proveedores y configura el backend.                               | Primer paso obligatorio en cualquier proyecto.  |
| <code>terraform init -upgrade</code>       | Fuerza la actualización de plugins y proveedores a las versiones más nuevas permitidas, ignorando el lock file. | Solución al error "Inconsistent dependency lock file".                                  |
| <code>terraform import [addr] [id]</code>  | Vincula un recurso existente en la nube con un bloque de recurso en el código local.                            | Usado para traer <code>tf-instance-1</code> y <code>2</code> bajo control de Terraform. |
| <code>terraform plan</code>                | Genera un plan de ejecución mostrando qué cambios se harán (dry-run).   | Verificación de seguridad antes de aplicar cambios.                                     |
| <code>terraform apply</code>               | Aplica los cambios para alcanzar el estado deseado. Requiere confirmación <code>yes</code> .                    | Creación del bucket, VPC y modificación de instancias.                                  |
| <code>terraform apply -auto-approve</code> | Aplica los cambios sin pedir confirmación manual.   | Útil para iteraciones rápidas en entornos de prueba o pipelines CI/CD.                  |
| <code>terraform destroy</code>             | Destruye toda la infraestructura gestionada por la configuración.   | Usado implícitamente al remover la tercera instancia del código.                        |
| <code>rm .terraform.lock.hcl</code>        | Elimina el archivo de bloqueo de dependencias.  | Maniobra de emergencia para resolver conflictos de versiones severos.                   |

## 7. Análisis de Problemas: Laboratorio vs. Mundo Real

Comparativa de los desafíos enfrentados en este entorno controlado versus cómo se manifiestan en entornos de producción.

### 7.1 Problemas Comunes en Labs de Terraform



1. **Errores de Sintaxis HCL:** Olvidar comillas, llaves o referencias a variables ( `var.zona` vs `"var.zona"` ).
2. **IDs Incorrectos:** Copiar mal el ID del Proyecto o el ID de una instancia para el comando `import` .
3. **Versiones Hardcodeadas:** Los laboratorios antiguos suelen tener versiones de proveedores fijas ( `4.51.0` ) que rompen al usar módulos modernos.
4. **Resource Zombie:** Recursos que no se destruyen correctamente y bloquean la creación de nuevos recursos con el mismo nombre.

## 7.2 Desafíos Críticos en el Mundo Real

1. **State Locking en Equipos:** En un lab trabajas solo. En una empresa, si dos ingenieros ejecutan `apply` al mismo tiempo sin un Backend remoto (GCS/S3) con bloqueo configurado, pueden corromper el archivo de estado irreversiblemente.
2. **Drift de Configuración:** Alguien modifica un firewall manualmente "de emergencia" en la consola de GCP. Terraform no se enterará hasta el siguiente `plan` , donde intentará revertir ese cambio, potencialmente causando una interrupción.
3. **Dependency Hell en Módulos:** Actualizar un módulo centralizado de red puede romper la infraestructura de 50 equipos diferentes si no se usan versiones estrictas ( `version = "1.2.3"` ).
4. **Secret Management:** En los labs a veces ponemos credenciales o IDs en texto plano. En producción, esto es una vulnerabilidad grave; se deben usar herramientas como Vault o Google Secret Manager.

## 8. Conclusión

El laboratorio **GSP345 "Automating Infrastructure on Google Cloud with Terraform: Challenge Lab"** sirvió como una prueba integral de habilidades. Más allá de la simple provisión de recursos, el valor real de este ejercicio radicó en la resolución de problemas no guionados.

Lograr un puntaje de 100/100 requirió no solo conocimiento de la sintaxis HCL, sino una comprensión profunda de cómo Terraform gestiona las dependencias y el ciclo de vida de los recursos. La capacidad de diagnosticar un conflicto de versiones y mitigar una condición de carrera mediante `depends_on` son habilidades que distinguen a un operador básico de un Ingeniero de Nube competente. Este desafío valida la preparación práctica para certificaciones como la **Google Cloud Associate Cloud Engineer**.