

Tiempo para aprender juntos

Dirección de Servicios de
Infraestructura y Operaciones

Abril 2021

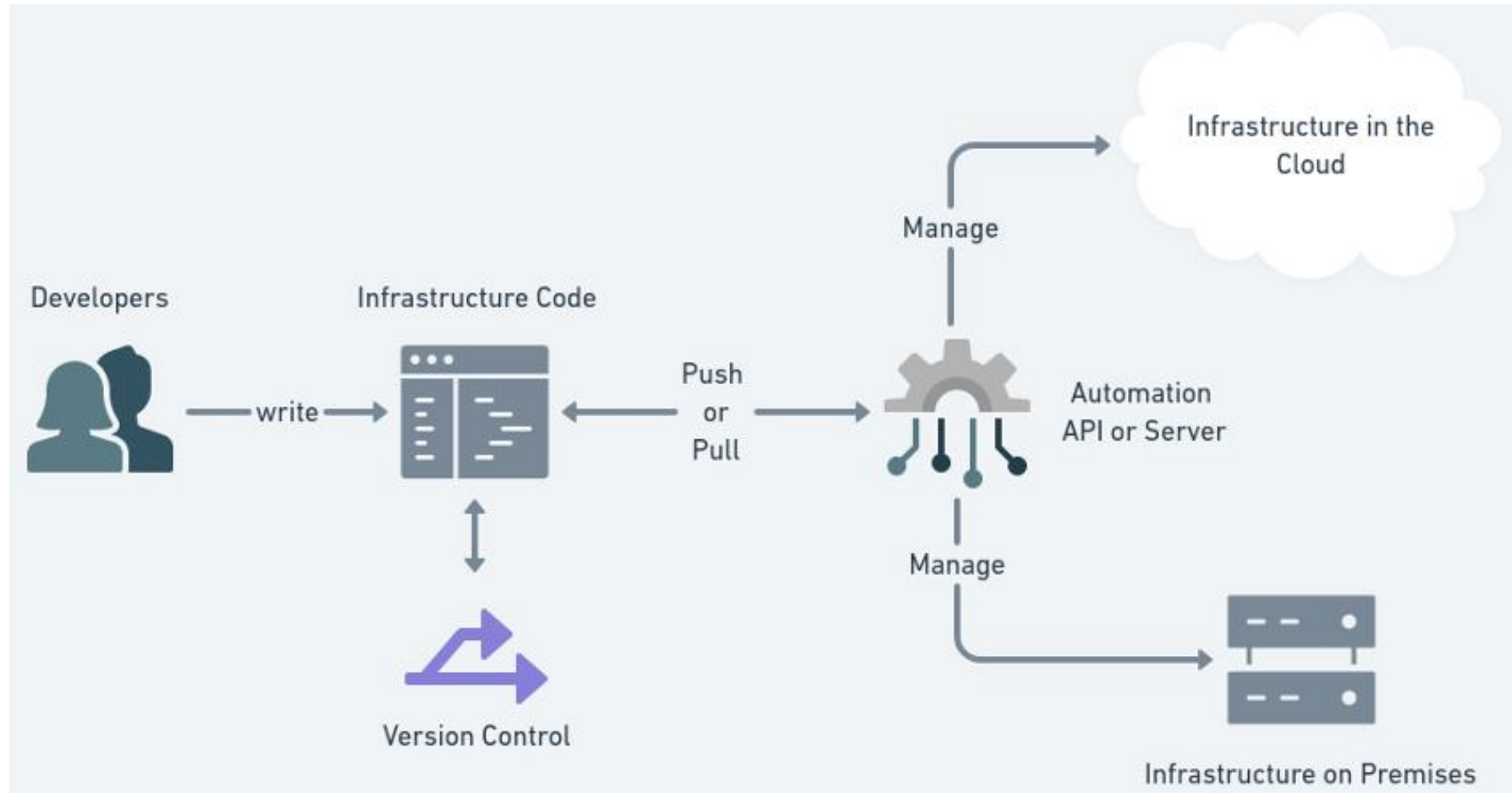
Terraform



- ¿Qué es Infraestructura como Código?
- Beneficios de Infraestructura como Código
- ¿Qué es Terraform?
- ¿Cómo aprender Terraform?
- ¿Cómo funciona Terraform?
- Terraform vs. Ansible
- ¿Cómo se integra con Ansible?
- ¿Cómo se integra con Ansible y Jenkins?
- Laboratorios

¿Qué es Infraestructura como Código?

- Es la **gestión de infraestructura (redes, máquinas virtuales, balanceadores de carga, firewalls)** de una manera **descriptiva** usando un **repositorio central de código**.

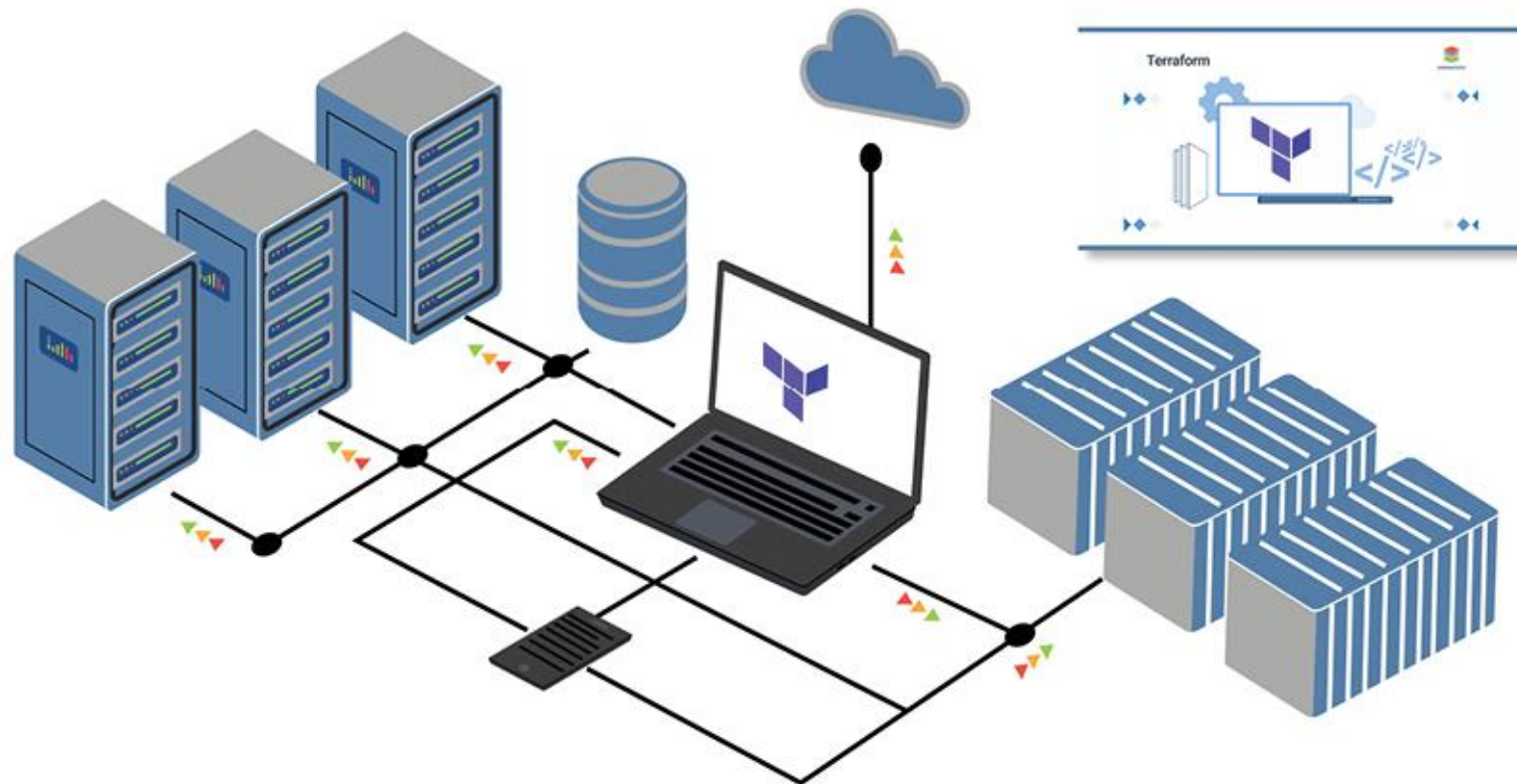


Beneficios de Infraestructura como Código

- **Rendimiento mejorado** del sistema con **prevención** de **riesgos** asegurada.
- Entrega de **software** a mayor **velocidad**.
- **Gestión** de **cambios** más **segura** y **eficiente**.
- **Escalabilidad**.
- Mejora la **satisfacción** del cliente.
- **Reducción** de **costos**.
- **Consistencia** en la **configuración**.

¿Qué es Terraform?

- Es un software **open-source** desarrollado por **HashiCorp** utilizado para **Infraestructura como Código** que permite a los usuarios **definir** y **provisionar** una **infraestructura** de centro de datos usando un lenguaje de configuración alto nivel conocido como *Hashicorp Configuration Language (HCL)*.



¿Qué es Terraform?



```
alakazam@vaibhavs-MacBook-Air auftr $ terraform plan
Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not
persisted to local or remote state storage.
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

Terraform will perform the following actions:

```
# aws_instance.example will be created
+ resource "aws_instance" "example" {
  + ami               = "ami-13be557e"
  + arn               = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)
  + cpu_core_count    = (known after apply)
  + cpu_threads_per_core = (known after apply)
  + get_password_data = false
  + host_id           = (known after apply)
  + id               = (known after apply)
  + instance_state    = (known after apply)
  + instance_type     = "t2.micro"
  + ipv6_address_count = (known after apply)
```

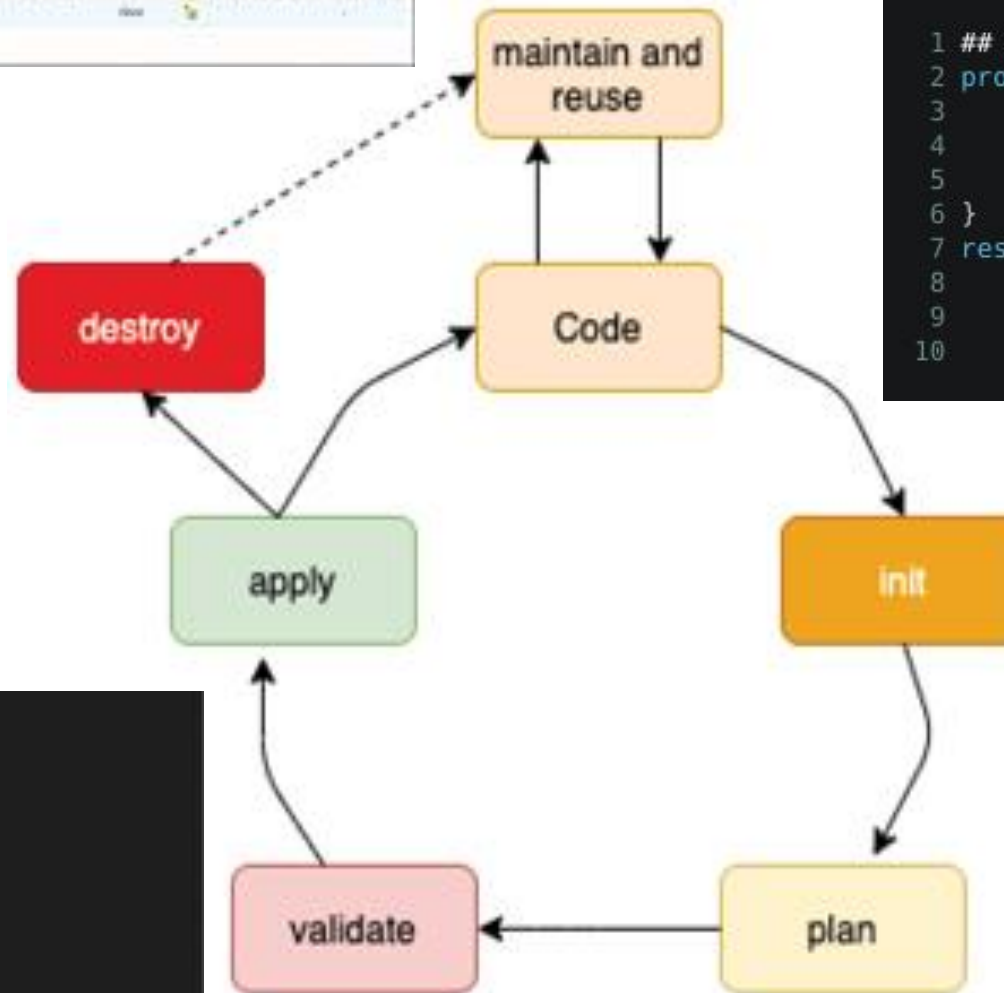
Plan: 1 to add, 0 to change, 0 to destroy.

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

```
aws_instance.example: Creating...
aws_instance.example: Still creating... [10s elapsed]
aws_instance.example: Still creating... [20s elapsed]
aws_instance.example: Still creating... [30s elapsed]
aws_instance.example: Creation complete after 34s [id=i-0ccaeda13255269dc]
```

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



```
1 ## ec2.tf
2 provider "aws" {
3     access_key = "<aws_access_key>"
4     secret_key = "<secret_key>"
5     region = "<aws_region>"
6 }
7 resource "aws_instance" "example" {
8     count = 5
9     ami = "ami-v1"
10    instance_type = "t2.micro"
```

```
alakazam@vaibhavs-MacBook-Air auftr $ terraform apply
```

An execution plan has been generated and is shown below.
Resource actions are indicated with the following symbols:
+ create

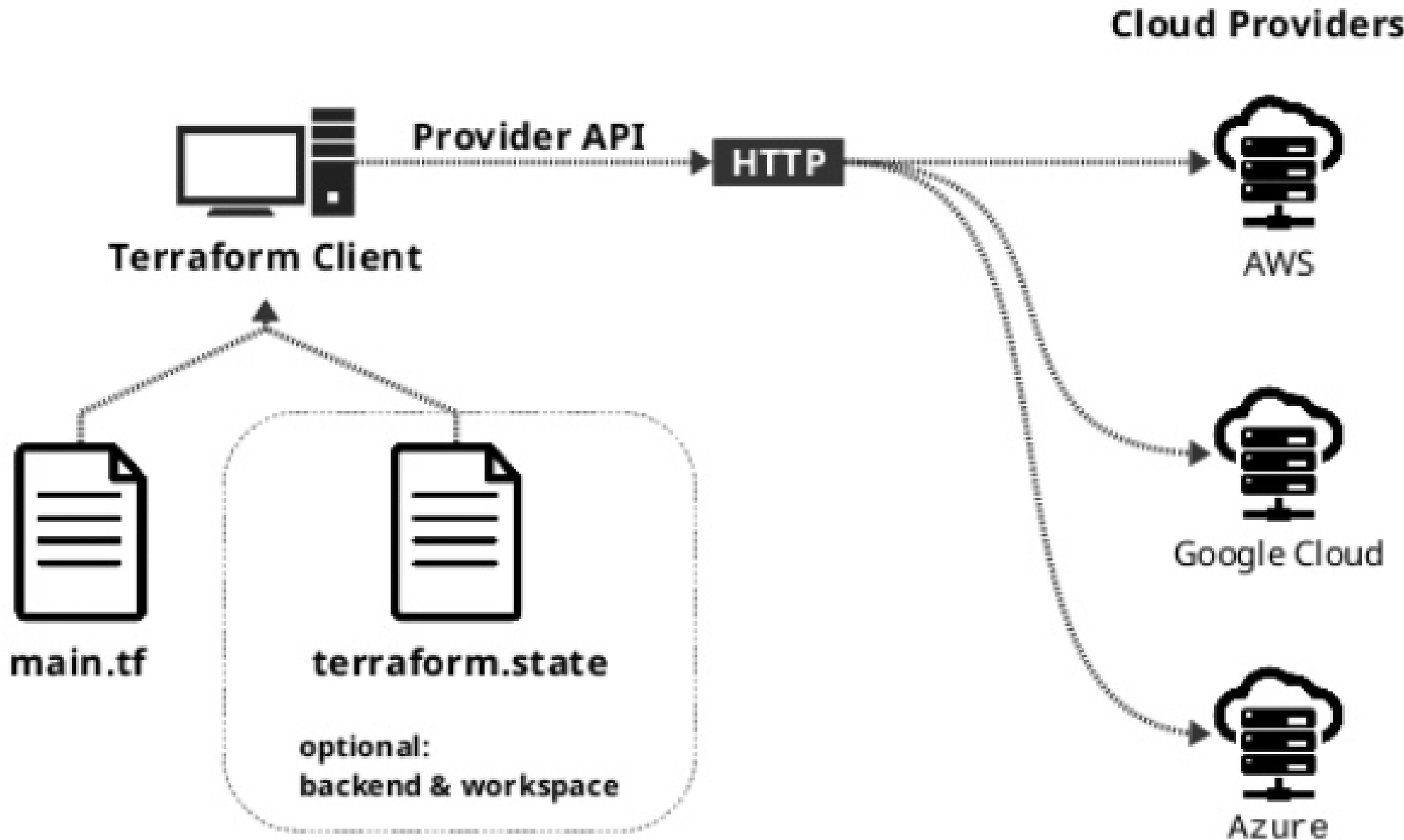
Terraform will perform the following actions:

```
# aws_instance.example will be created
+ resource "aws_instance" "example" {
  + ami               = "ami-13be557e"
  + arn               = (known after apply)
  + associate_public_ip_address = (known after apply)
  + availability_zone = (known after apply)
  + cpu_core_count    = (known after apply)
```

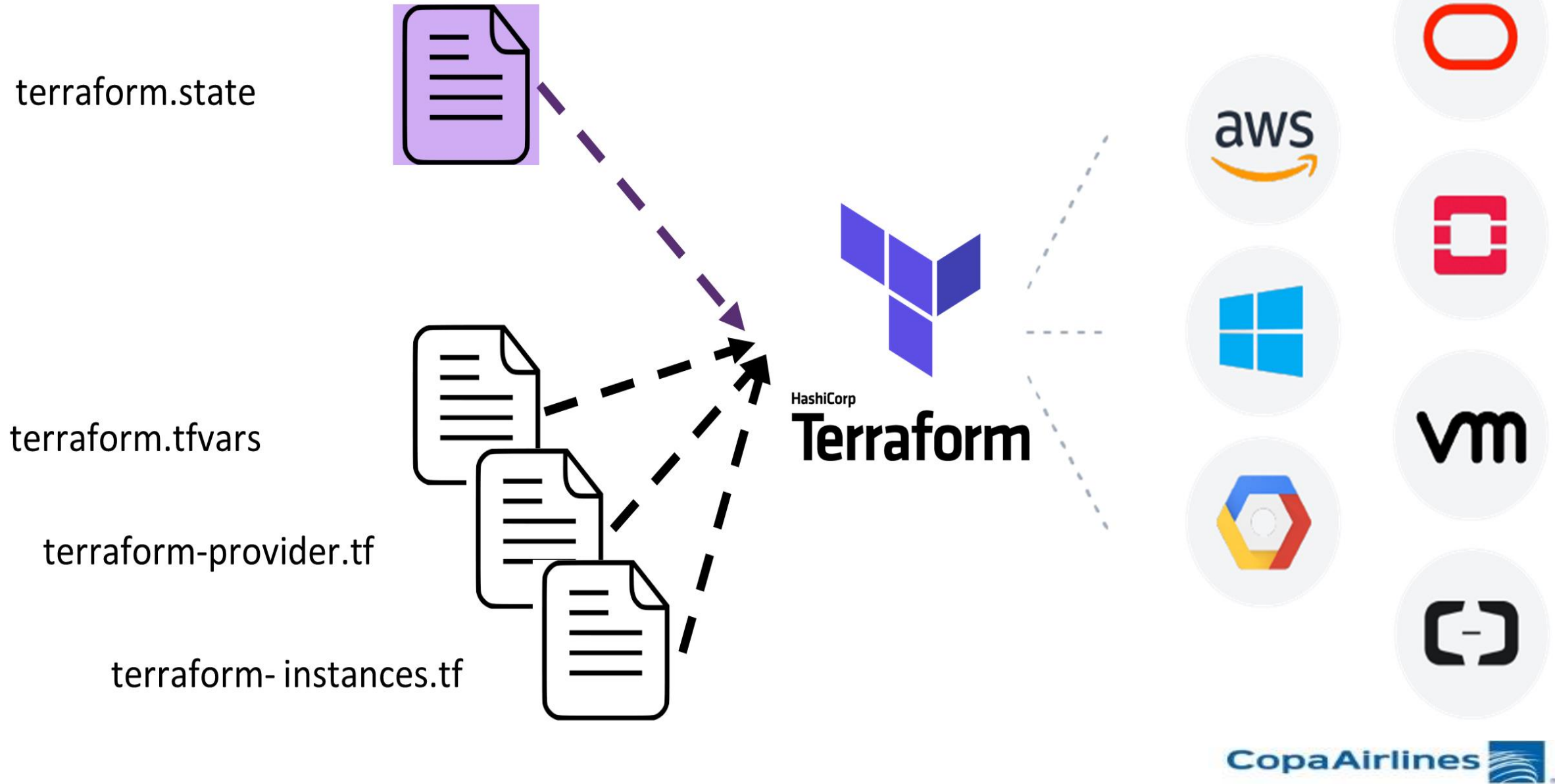

¿Cómo aprender Terraform?

- En el sitio de Terraform (<https://www.terraform.io/>)
- Sobre los proveedores:
 - <https://registry.terraform.io/browse/providers>
 - Azure
 - <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs>
 - VMware
 - <https://registry.terraform.io/providers/hashicorp/vsphere/latest>
- **Buscando** en foros (<https://stackoverflow.com/>) o sitios (<https://medium.com/>).
- **Aplicarlo** a una **situación** en **2021** que consideras puedas resolver a través de **scripting**.

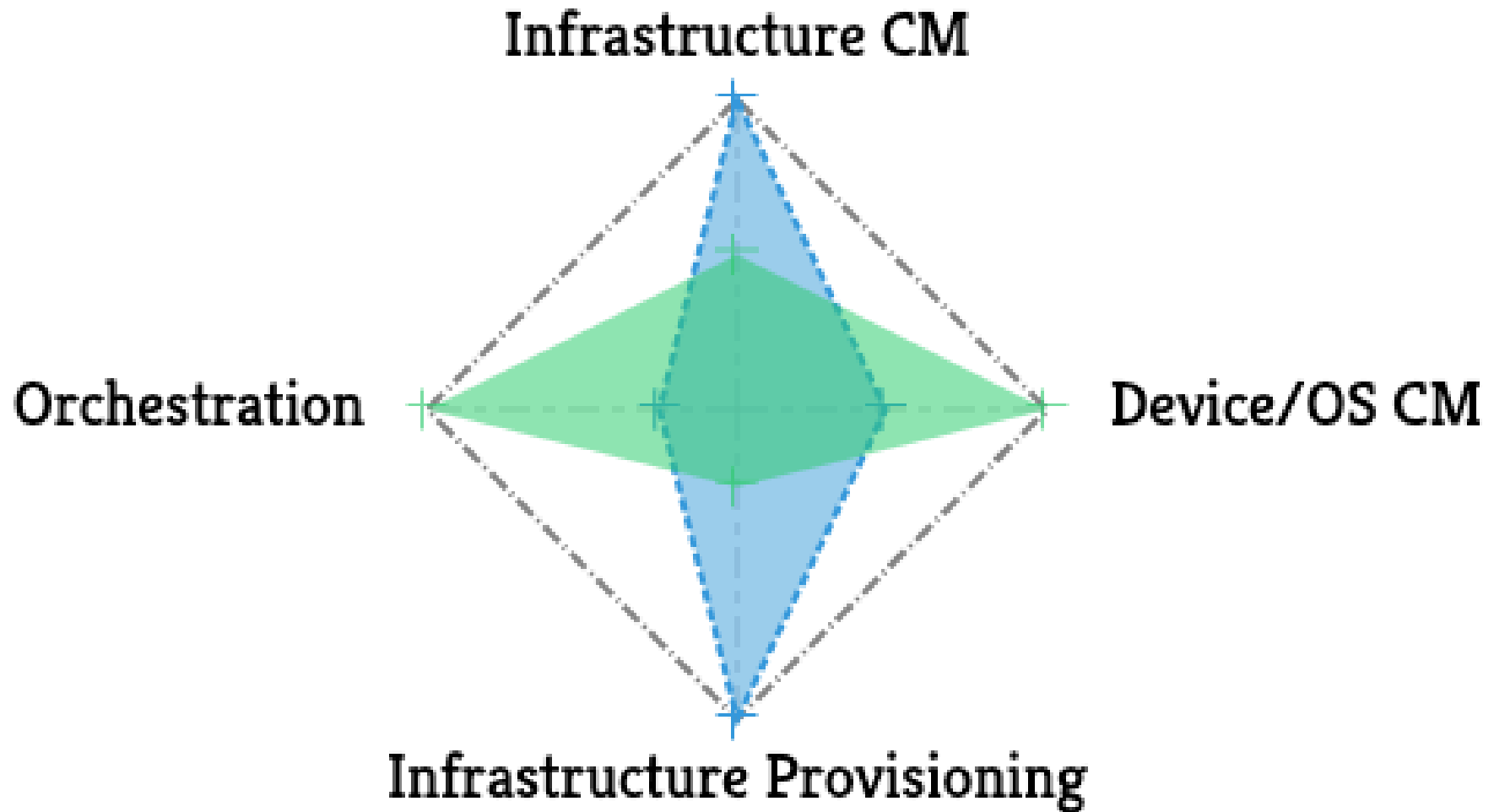
¿Cómo funciona Terraform?



¿Cómo funciona Terraform?



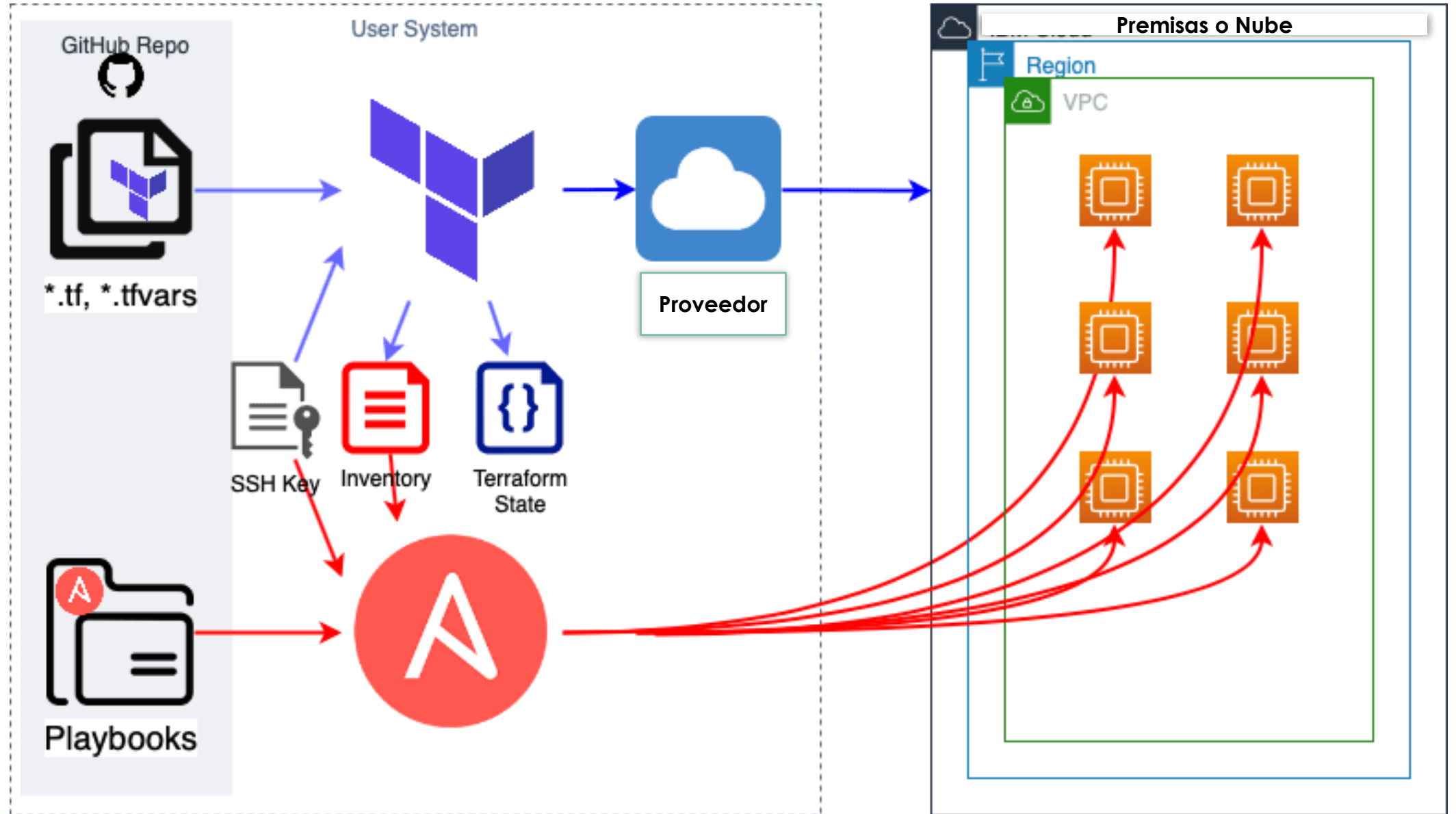
 Terraform  Ansible



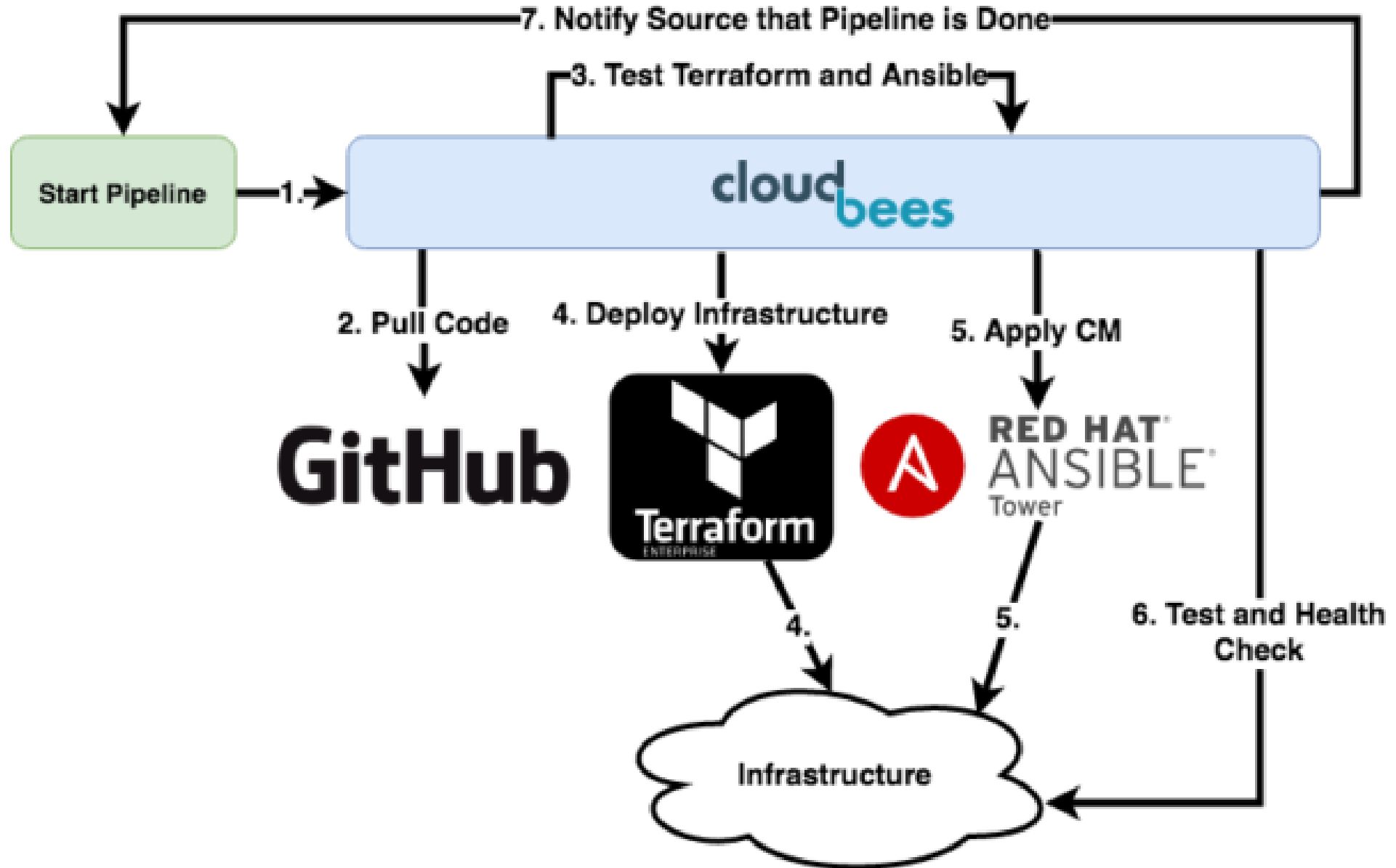
Terraform vs. Ansible

Factor	Ansible	Terraform
Type	Ansible is a configuration management tool	Terraform is an orchestration tool
Infrastructure	Ansible provides support for mutable infrastructure	Terraform provides support for immutable infrastructure
Language	Ansible follows a procedural language	Terraform follows a declarative language
VM provisioning, networking and storage management	Ansible provides partial VM provisioning, networking and storage management	Terraform provide comprehensive VM provisioning, networking and storage management
Packaging and templating	Ansible provides complete support for packaging and templating	Terraform provides partial support for packaging and templating
Lifecycle (State) Management	Ansible does not have lifecycle management	Terraform is heavily dependent on lifecycle or state management

¿Cómo se integra con Ansible?



¿Cómo se integra con Ansible y Jenkins?

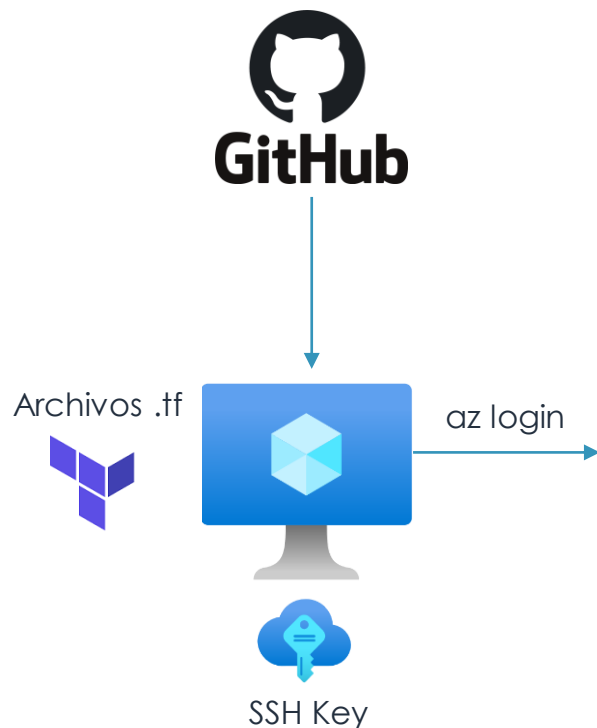




- En estos laboratorios vamos a aprender a:
 - Instalar **Terraform**
 - Instalar **Azure CLI**
 - Usar el provider **azurerm** y estos recursos:
 - azurerm_resource_group
 - azurerm_virtual_network
 - azurerm_subnet
 - azurerm_network_interface
 - azurerm_network_security_group
 - azurerm_network_interface_security_group_association
 - azurerm_storage_account
 - azurerm_linux_virtual_machine
- Usar y definir variables
- Desplegar y eliminar infraestructura



Laboratorios – Comienza la diversión



Azure Cloud



Resource group

terraform-azure-<usuario>



Tag

terraform-azure-<usuario>



Public IP

ip_publica

Asociada a VM: AZR-SANDBOX-<usuario>-01



Security group

terraform-azure-<usuario>-sg

Puertos permitidos:

- TCP: 22 (SSH), 80 (HTTP), 443 (HTTPS)
- ICMP: ping



Storage account

almacenamiento

- Tier: Standard
- Replicación: LRS

Región
"eastus"



Virtual network

terraform-scripting-azure-<usuario>

10.<id_participante>.0.0/16

Subnetwork

<usuario>-1

10.<id_participante>.1.0/24



Network Interface Card

<usuario>-1-interfaz-red



Network Interface Card

AZR-SANDBOX-<usuario>-01-interfaz-red



Network Interface Card

AZR-SANDBOX-<usuario>-02-interfaz-red



VM

Nombre: AZR-SANDBOX-<usuario>-01

Tamaño: Standard_DS1_v2

Publisher: Canonical

Offer: Ubuntu Server

SKU: 20_04-LTS-Gen2



VM

Nombre: AZR-SANDBOX-<usuario>-02

Tamaño: Standard_DS1_v2

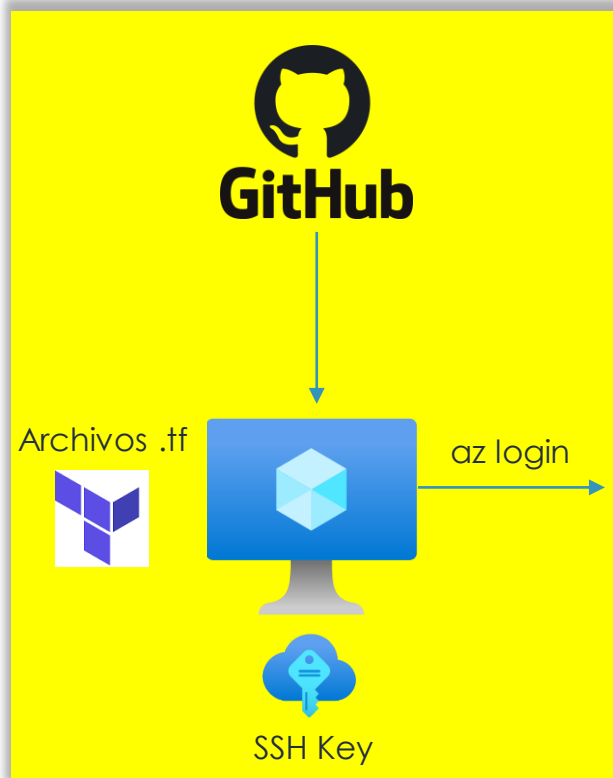
Publisher: Canonical

Offer: Ubuntu Server

SKU: 18.04-LTS



Laboratorios – Comienza la diversión



 **Azure Cloud**



- En **Sharepoint** se encuentra el archivo con todo el inventario de VMs creadas, selecciona **1 VM** (registra en el Excel tu **nombre** y apunta el número de la **subred**).
- Ve al siguiente repositorio y **clónalo** en la VM:
 - `git clone https://github.com/HugoAquinoNavarrete/terraform_scripting_azure.git`
- Instala **unzip** y **Terraform**:
 - `cd terraform_scripting_azure`
 - `wget https://releases.hashicorp.com/terraform/0.13.5/terraform_0.13.5_linux_amd64.zip`
 - `sudo apt install unzip`
 - `unzip terraform_0.13.5_linux_amd64.zip`
 - `sudo mv terraform /usr/local/bin/`
 - `terraform --version`



- Instala **Azure CLI**
 - `curl -sL https://aka.ms/InstallAzureCLIDeb | sudo bash`
- **Conéctate** a Azure ejecutando:
 - `az login`
- Lista las **suscripciones**:
 - `az account list --output table`
- **Selecciona la suscripción** que usaremos en el curso:
 - `az account set --subscription "<id-suscripción>"`



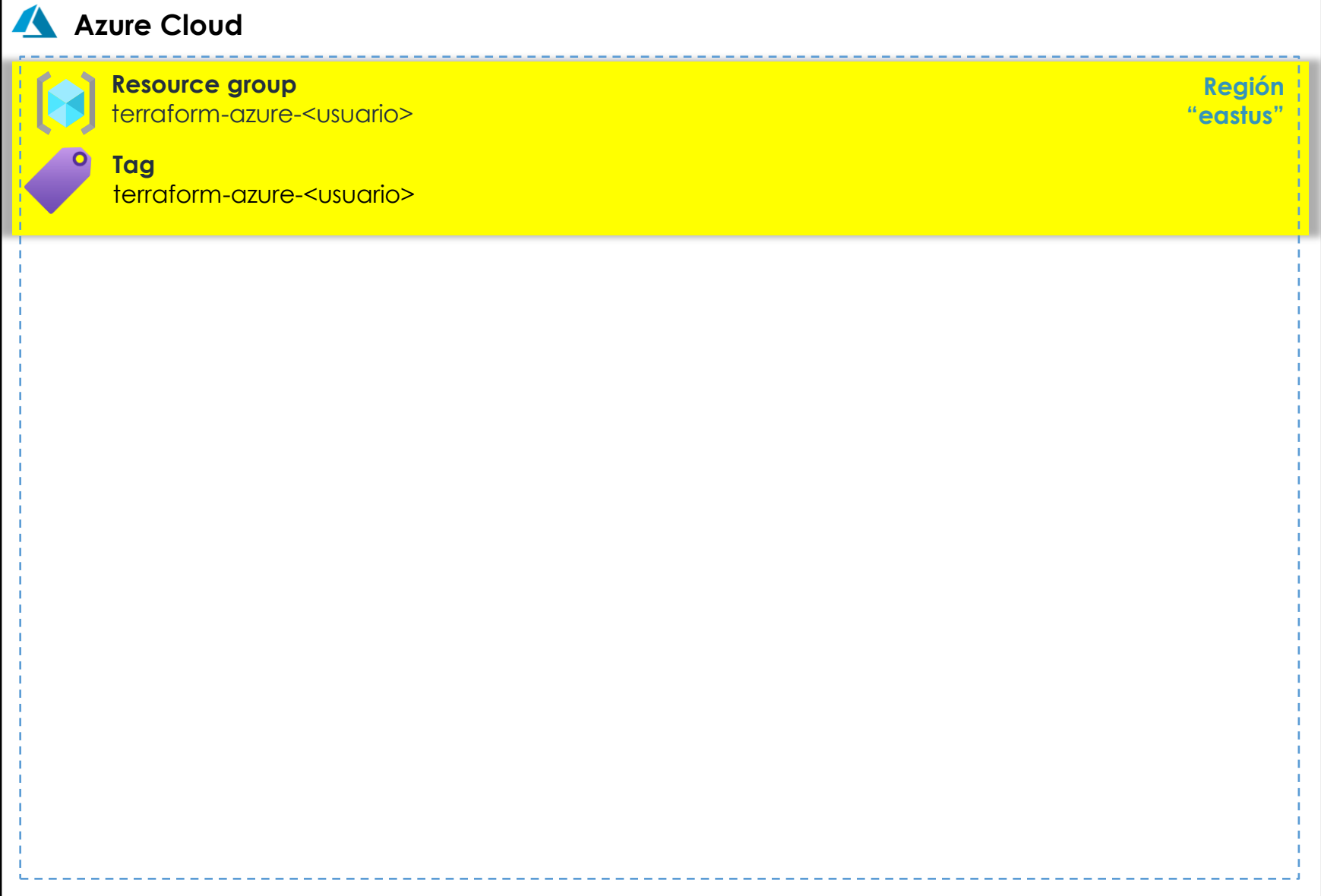
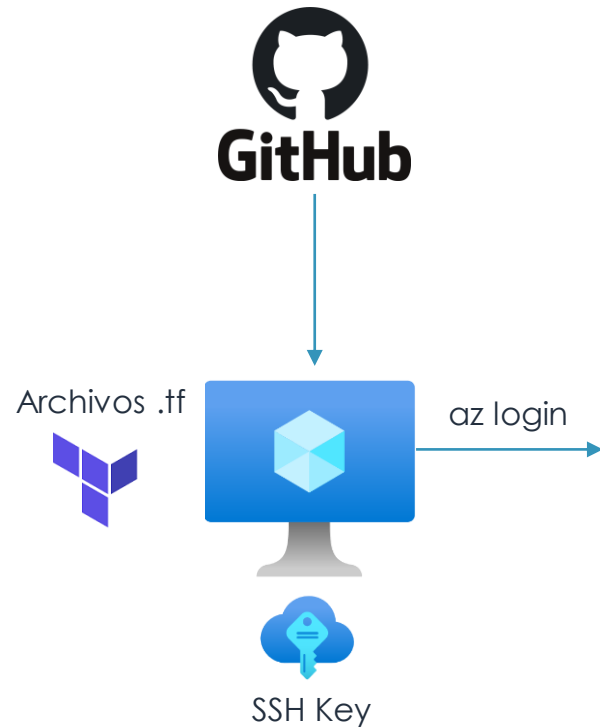
- Abre el siguiente link:
 - <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs>



- Ve al directorio “**terraform_scripting_azure**” e intenta iniciar Terraform
 - `cd terraform_scripting_azure`
 - `terraform init`
- **¿Se pudo iniciar Terraform?**
- Abre el siguiente link:
 - <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs>
- Copia el archivo “**azure.txt**” a “**azure.tf**”, intenta inicializar nuevamente Terraform, planifica la infraestructura e intenta crear la infraestructura:
 - `cp azure.txt azure.tf`
 - `terraform init`
 - `terraform plan`
 - `terraform deploy`



Laboratorios – Comienza la diversión





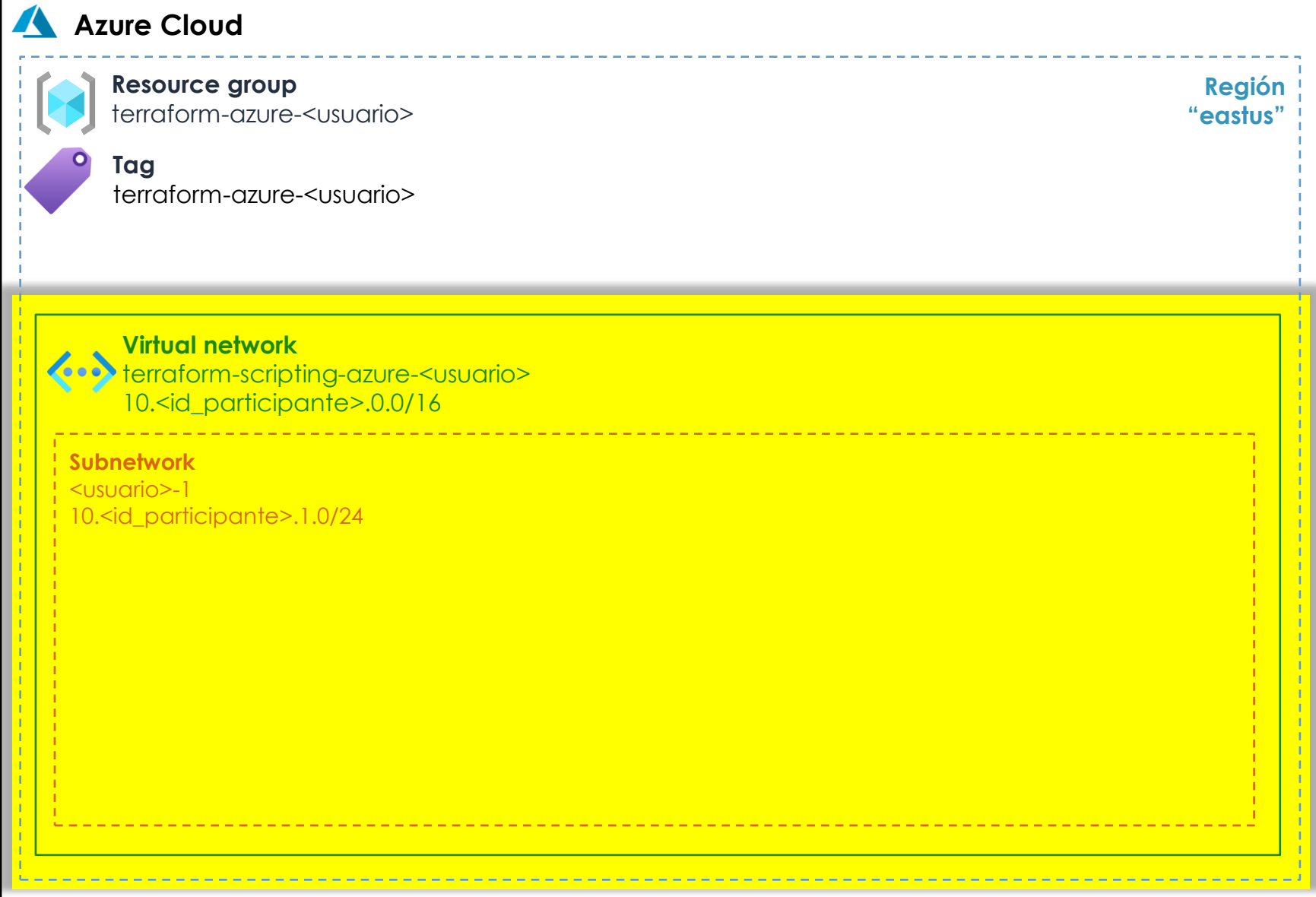
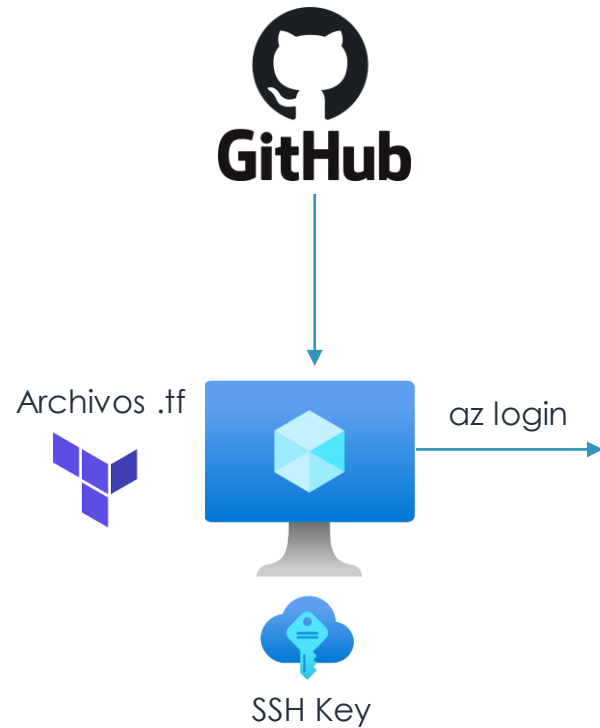
- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/resource_group
- Copia el archivo “**main.txt**” a “**main.tf**”, planifica la infraestructura:
 - `cp main.txt main.tf`
 - `terraform plan`
- ¿Se pudo desplegar la infraestructura?



- Copia el archivo “**variables.txt**” a “**variables.tf**”, e intenta hacer el despliegue:
 - `cp variables.txt variables.tf`
 - `terraform plan`
- ¿Se pudo desplegar la infraestructura?
- Ajusta las variables dentro de “**variables.tf**” que marcaron error e intenta hacer el despliegue:
 - `nano variables.tf`
 - `terraform plan`
 - `terraform apply`
- Verifica el “**resource_group**” creado:
 - `az group list --output table`



Laboratorios – Comienza la diversión





- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/virtual_network
- Instala **ipcalc** en la VM:
 - `sudo apt install ipcalc`
- Copia el archivo “**network.txt**” a “**network.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp network.txt network.tf`
 - `terraform plan`
 - `terraform apply`
- ¿Se pudo desplegar la infraestructura?



- Usa **ipcalc** en la VM con el rango de direccionamiento asignado y analicemos la salida del comando:
 - `ipcalc <segmento_red>/<máscara>`
- Ajusta las variables dentro de “**variables.tf**” que marcaron error e intenta hacer el despliegue:
 - `nano variables.tf`
 - `terraform plan`
 - `terraform apply`
- Verifica la “**azurerm_virtual_network**” creada:
 - `az network vnet list --output table`



- Abre el siguiente link:
 - <https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/subnet>
- Edita el archivo “**network.tf**”, elimina el carácter “#” de las líneas donde aparezca, planifica la infraestructura e intenta desplegarla:
 - `nano network.tf`
 - `terraform plan`
 - `terraform apply`
- **¿Se pudo desplegar la infraestructura?**



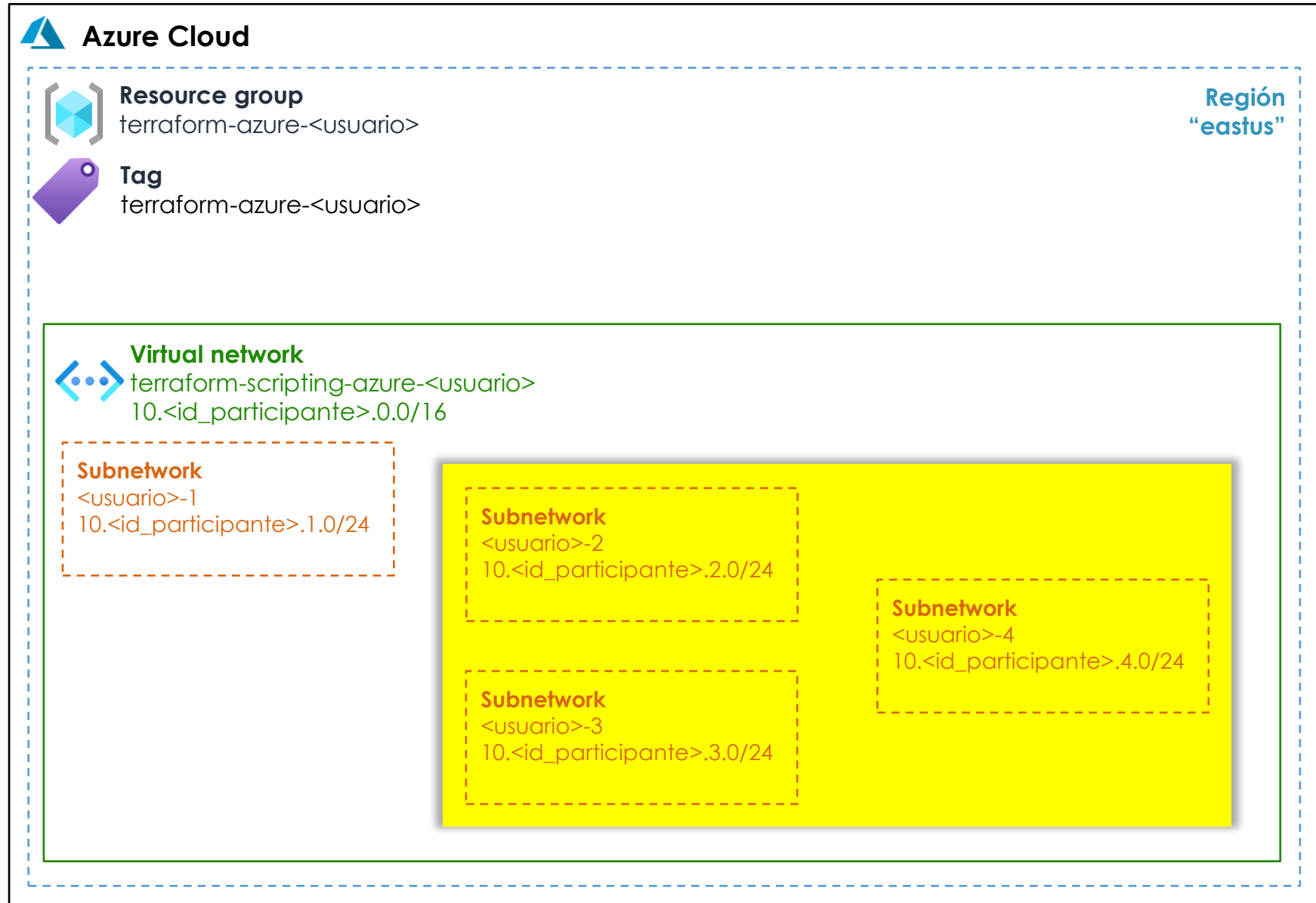
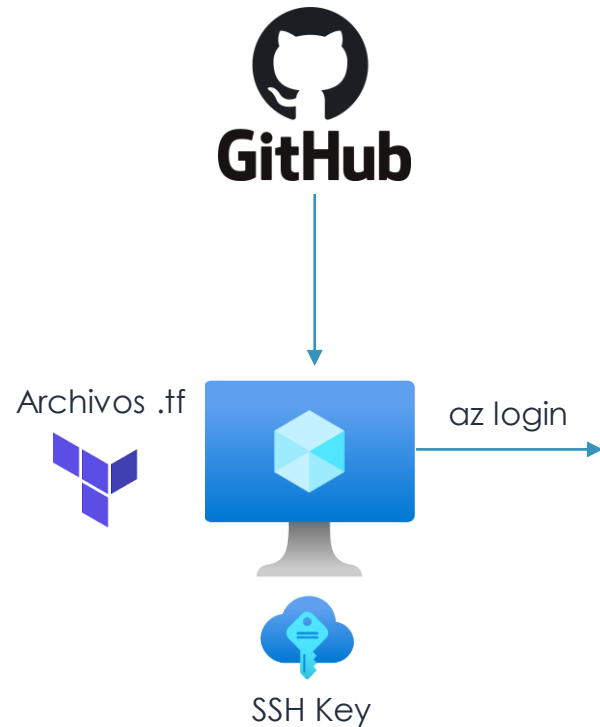
- Ajusta las variables dentro de “**variables.tf**” que marcaron error e intenta hacer el despliegue:
 - nano variables.tf
 - terraform plan
 - terraform apply
- Verifica la “**azurerm_virtual_network**” creada:
 - az network vnet subnet list -g terraform-azure-<usuario>-rg --vnet-name terraform-scripting-azure-<usuario> --output table



- **Reto 1 – Agrega 3 subredes y muéstralas**
 - Ajusta el archivo “**network.tf**” y “**variables.tf**” para agregar **3 subredes adicionales**.
 - Despliega y muestra las subredes creadas.

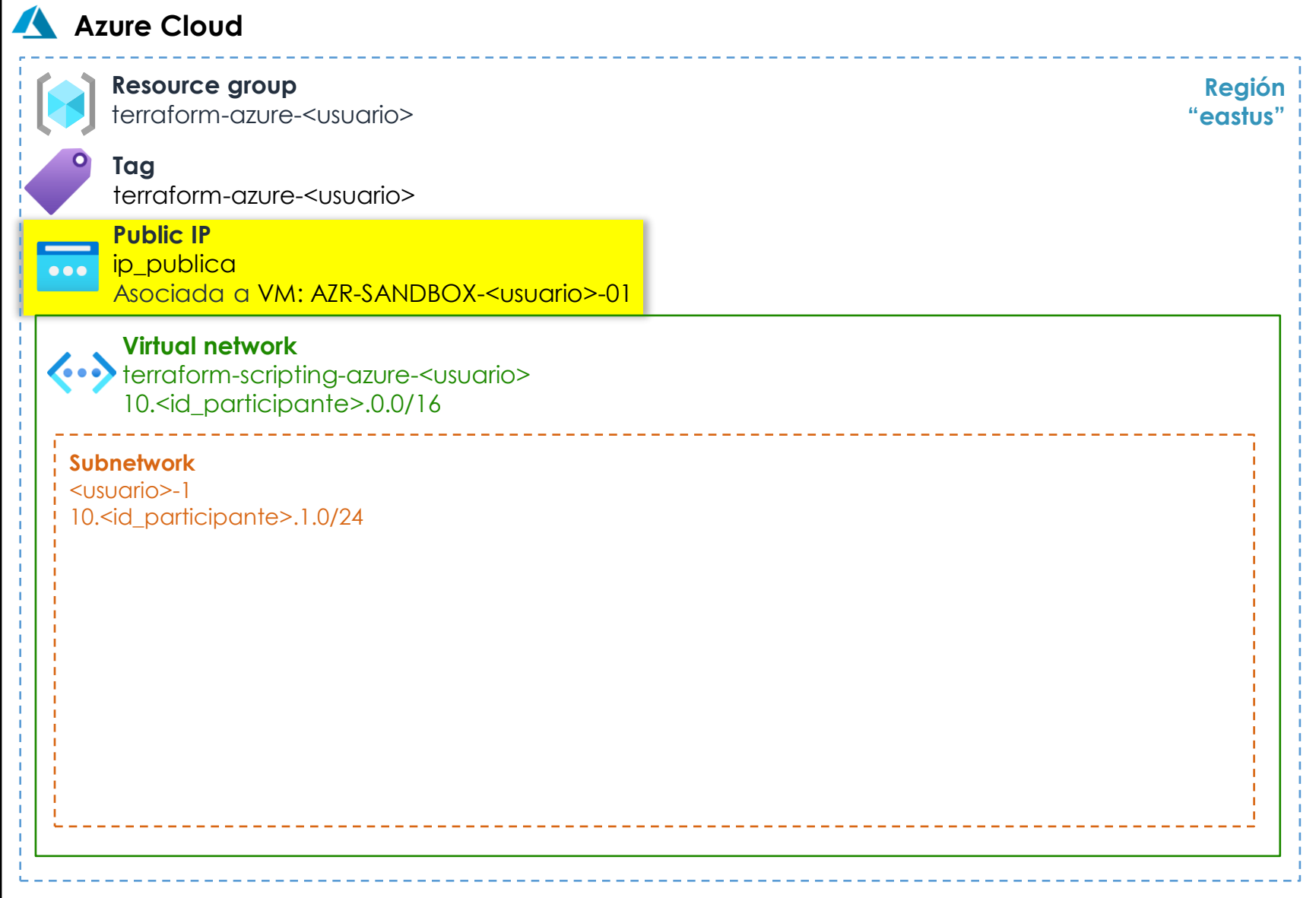
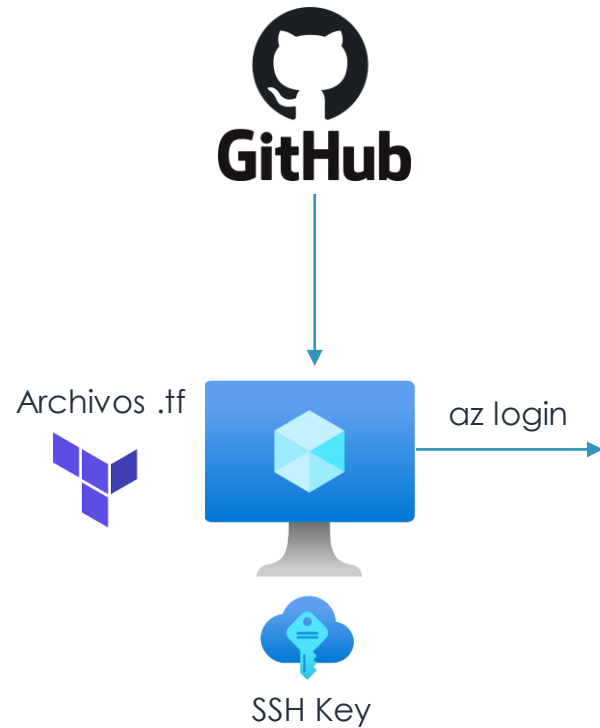


Laboratorios – Comienza la diversión





Laboratorios – Comienza la diversión

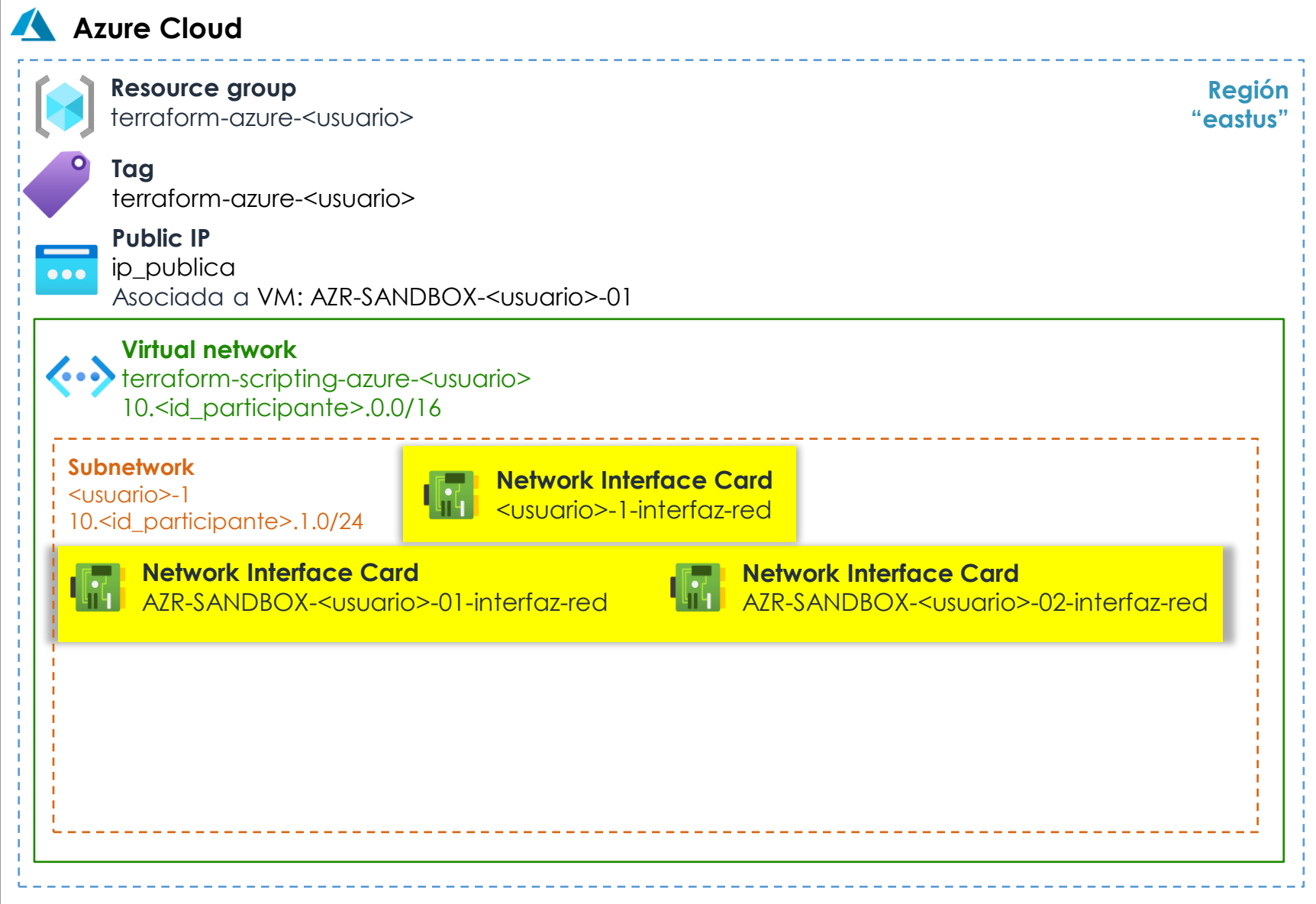
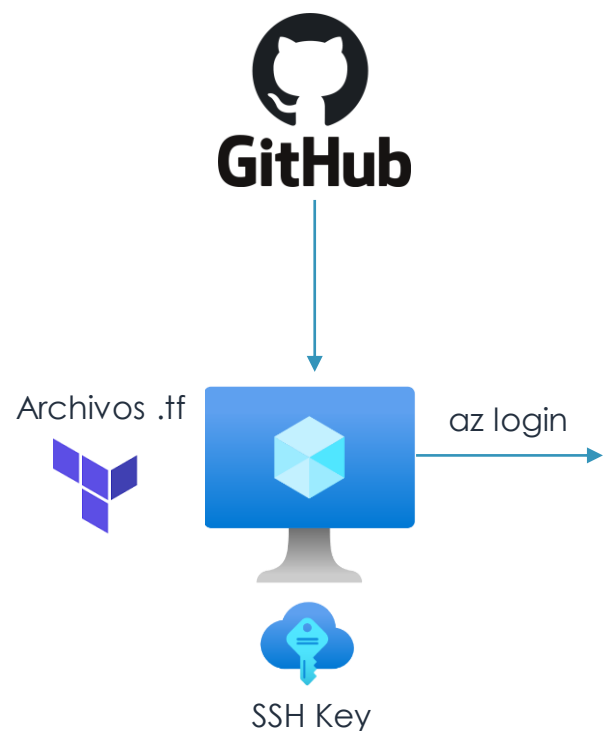




- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/public_ip
- Copia el archivo “**ip_publica.txt**” a “**ip_publica.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp ip_publica.txt ip_publica.tf`
 - `terraform plan`
 - `terraform apply`
- ¿Se pudo desplegar la infraestructura?
- Verifica la “**azurerm_public_ip**” creada:
 - `az network public-ip list -g terraform-azure-<usuario>-rg --output table`



Laboratorios – Comienza la diversión





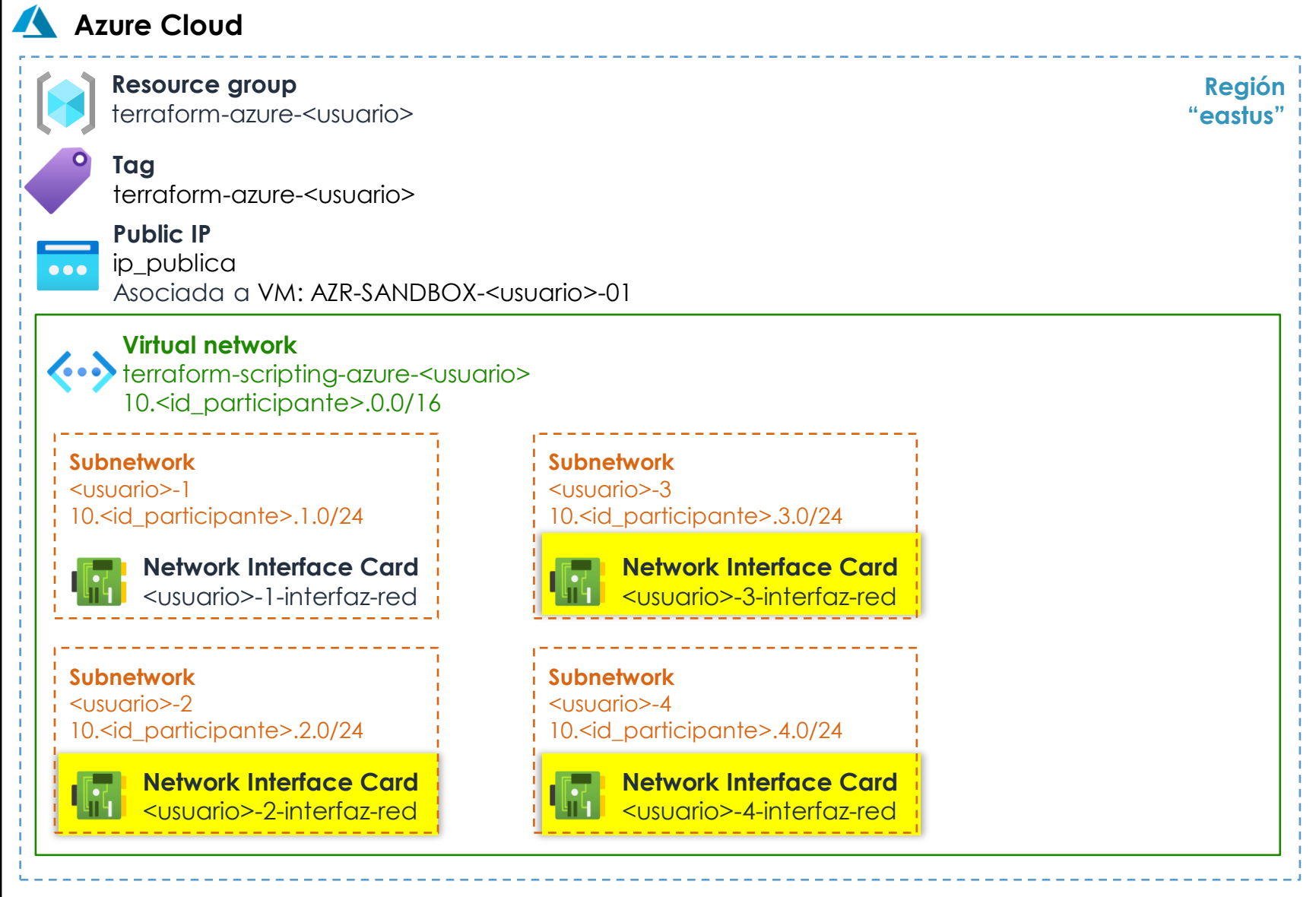
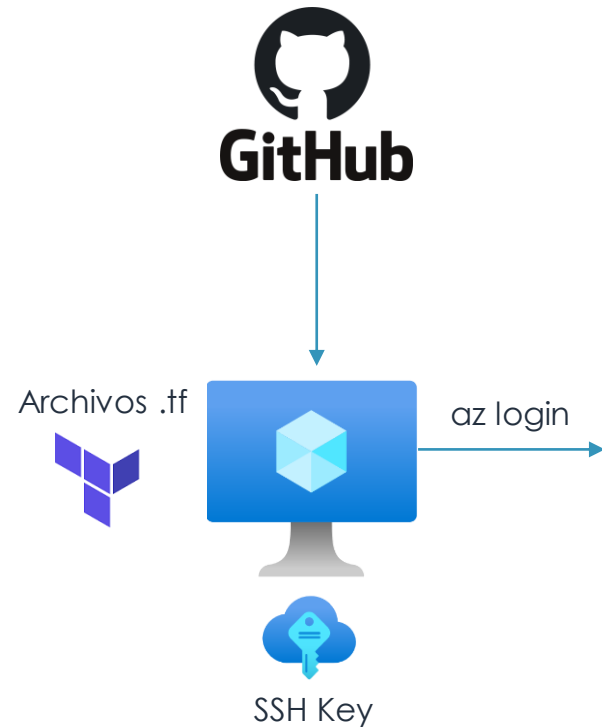
- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/network_interface
- Copia el archivo “**network_interfaces.txt**” a “**network_interfaces.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp network_interfaces.txt network_interfaces.tf`
 - `terraform plan`
 - `terraform apply`
- **¿Se pudo desplegar la infraestructura?**
- Verifica la “**azurerm_network_interface**” creada:
 - `az network nic list -g terraform-azure-<usuario>-rg --output table`



- **Reto 2 – Agrega 3 NICs y muéstralas**
 - Ajusta el archivo “**network_interfaces.tf**” para agregar **3 NICs (1 por cada subred)**.
 - Despliega y muestra las NICs creadas.

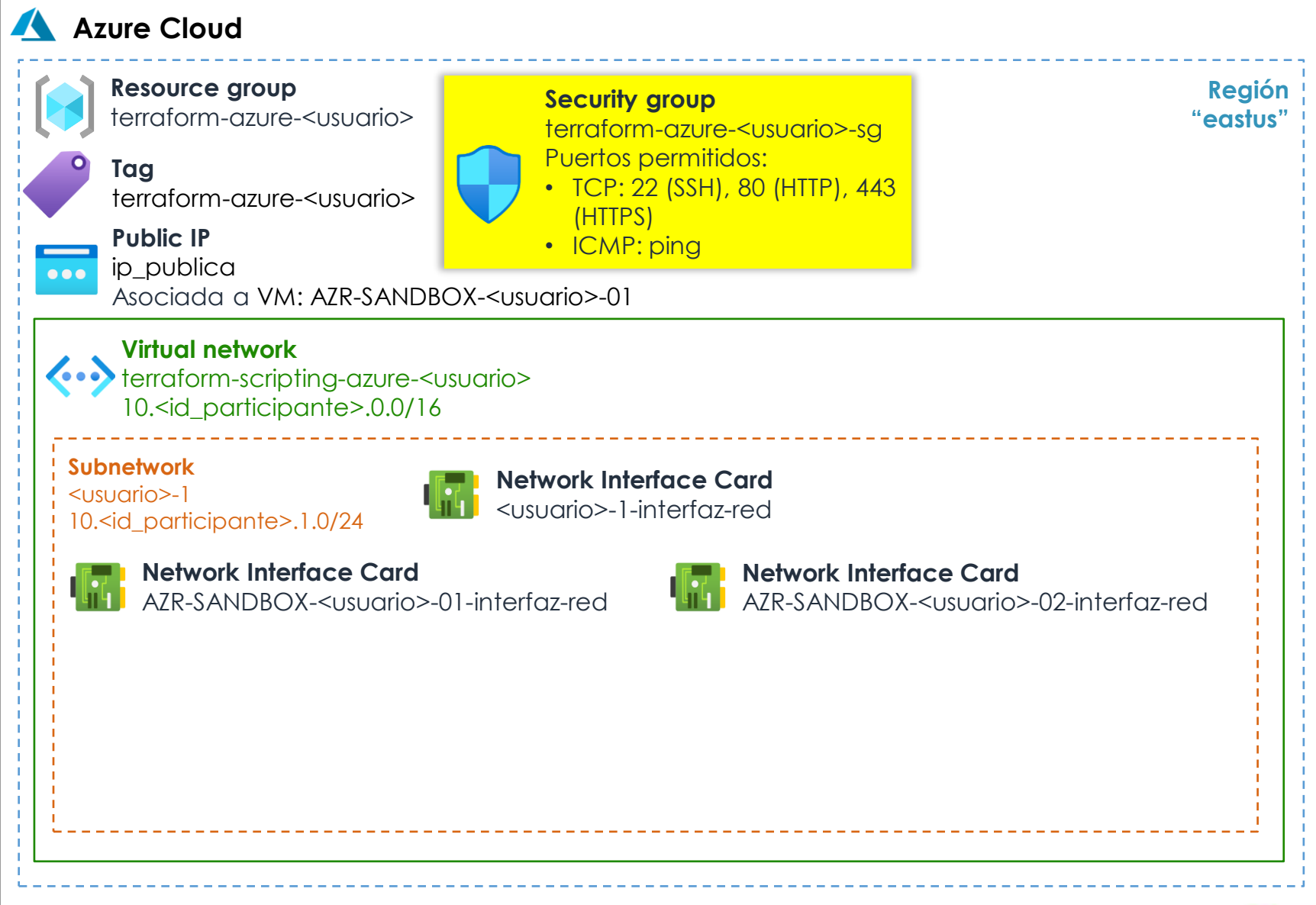
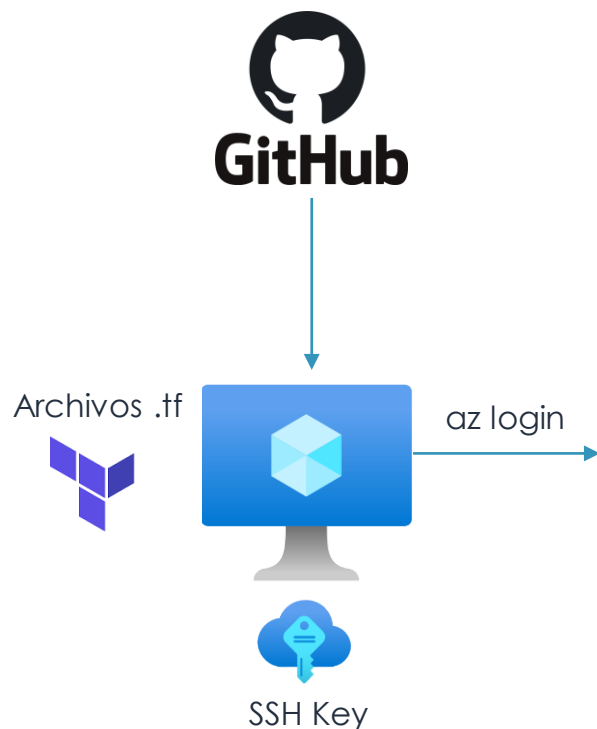


Laboratorios – Comienza la diversión





Laboratorios – Comienza la diversión





- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/network_security_group
- Copia el archivo “**security.txt**” a “**security.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp security.txt security.tf`
 - `terraform plan`
 - `terraform apply`
- **¿Se pudo desplegar la infraestructura?**
- Verifica el “**azurerm_network_security_group**” creado:
 - `az network nsg list -g terraform-azure-<usuario>-rg --output table`



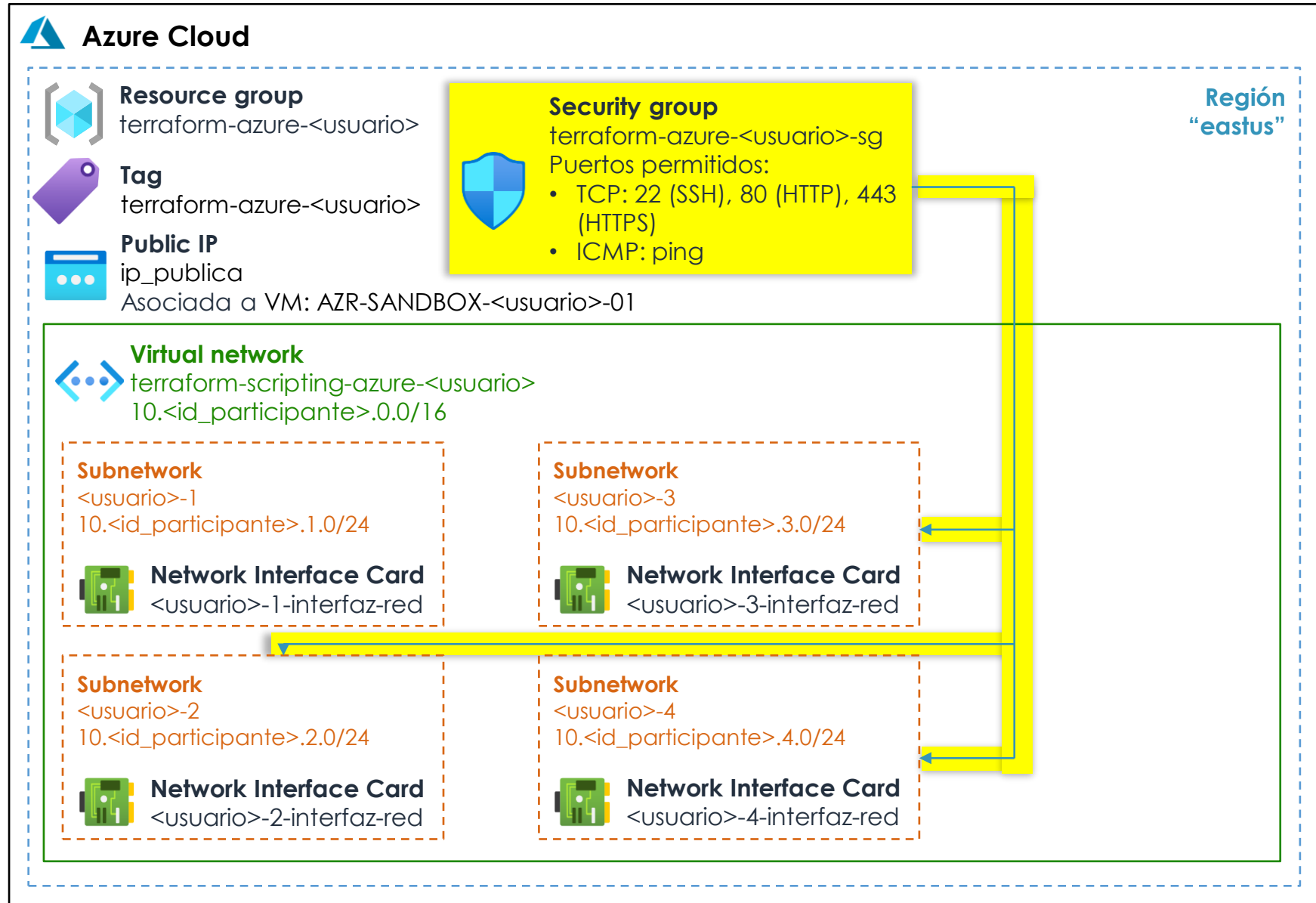
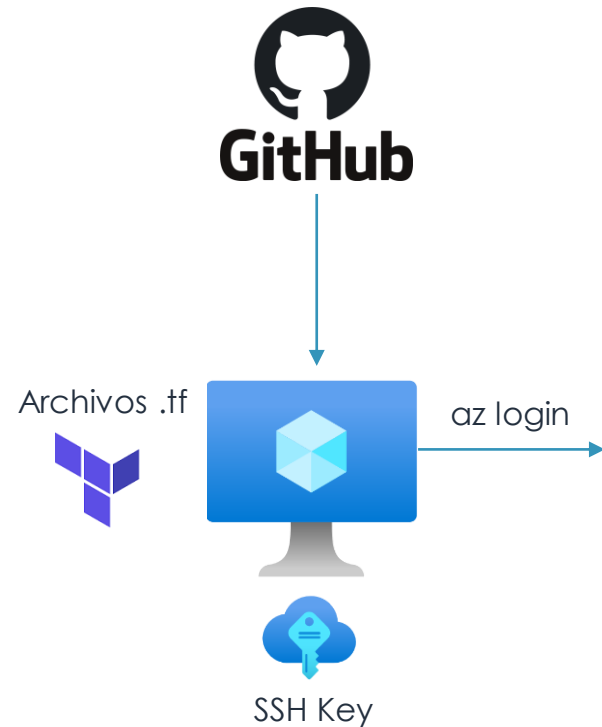
- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/network_interface_security_group_association
- Edita el archivo “**security.txt**”, elimina el carácter “#” de las líneas donde aparezca, planifica la infraestructura e intenta desplegarla:
 - nano security.tf
 - terraform plan
 - terraform apply
- ¿Se pudo desplegar la infraestructura?



- **Reto 3 – Asocia 3 subredes al “network_security_group”**
 - Ajusta el archivo “**security.tf**” para agregar **3 subredes** al “**network_security_group**”.

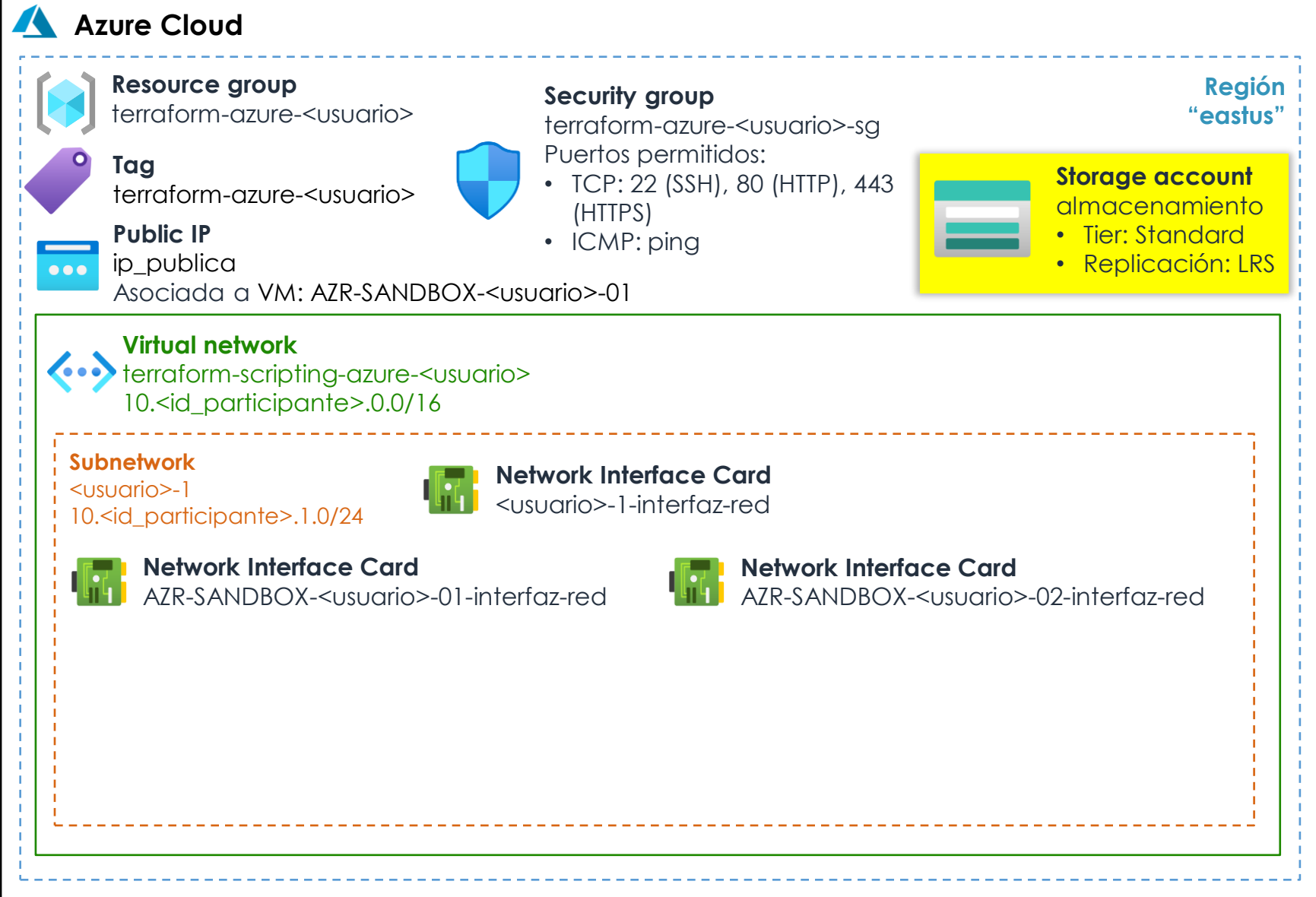
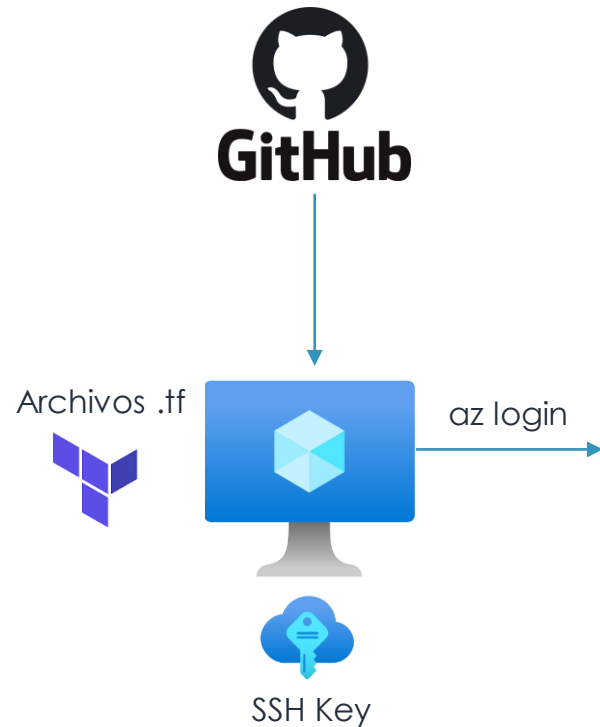


Laboratorios – Comienza la diversión





Laboratorios – Comienza la diversión





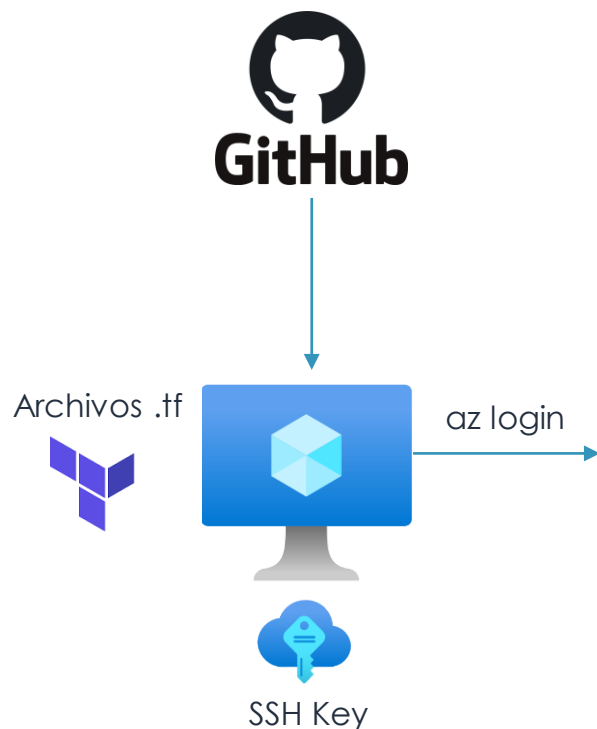
- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/storage_account
- Copia el archivo “**storage.txt**” a “**storage.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp storage.txt storage.tf`
 - `terraform plan`
 - `terraform apply`
- ¿Se pudo desplegar la infraestructura?



- Ajusta las variables dentro de “**variables.tf**” que marcaron error e intenta hacer el despliegue:
 - nano variables.tf
 - terraform plan
 - terraform apply
- Verifica el “**azurerm_storage_account**” creado:
 - az storage account list -g terraform-azure-<usuario>-rg --output table



Laboratorios – Comienza la diversión



Azure Cloud



Resource group

terraform-azure-<usuario>



Tag

terraform-azure-<usuario>



Public IP

ip_publica

Asociada a VM: AZR-SANDBOX-<usuario>-01



Security group

terraform-azure-<usuario>-sg

Puertos permitidos:

- TCP: 22 (SSH), 80 (HTTP), 443 (HTTPS)
- ICMP: ping



Storage account

almacenamiento

- Tier: Standard
- Replicación: LRS

Región
"eastus"



Virtual network

terraform-scripting-azure-<usuario>

10.<id_participante>.0.0/16

Subnetwork

<usuario>-1

10.<id_participante>.1.0/24



Network Interface Card

<usuario>-1-interfaz-red



Network Interface Card

AZR-SANDBOX-<usuario>-01-interfaz-red



Network Interface Card

AZR-SANDBOX-<usuario>-02-interfaz-red



VM

Nombre: AZR-SANDBOX-<usuario>-01

Tamaño: Standard_DS1_v2

Publisher: Canonical

Offer: Ubuntu Server

SKU: 20_04-LTS-Gen2



VM

Nombre: AZR-SANDBOX-<usuario>-02

Tamaño: Standard_DS1_v2

Publisher: Canonical

Offer: Ubuntu Server

SKU: 18.04-LTS



- Abre el siguiente link:
 - https://registry.terraform.io/providers/hashicorp/azurerm/latest/docs/resources/linux_virtual_machine
- Copia el archivo “**vm_subred_1.txt**” a “**vm_subred_1.tf**”, planifica la infraestructura e intenta desplegarla:
 - `cp vm_subred_1.txt vm_subred_1.tf`
 - `terraform plan`
 - `terraform apply`
- ¿Se pudo desplegar la infraestructura?



- Edita el archivo “**network_interfaces.tf**”, elimina el carácter “#” de las líneas donde aparezca, planifica la infraestructura e intenta desplegarla:
 - nano network_interfaces.tf
 - terraform plan
- ¿Se pudo desplegar la infraestructura?
- Ajusta las variables dentro de “**variables.tf**” que marcaron error e intenta hacer el despliegue:
 - nano variables.tf
 - terraform plan
 - terraform apply
- ¿Se pudo desplegar la infraestructura?



- Verifica la “**azurerm_linux_virtual_machine**” creada así como el direccionamiento asociado:
 - `az vm list -g terraform-azure-<usuario>-rg --output table`
 - `az vm list-ip-addresses -g terraform-azure-<usuario>-rg --output table`
- Edita el archivo “**vm_subred_1.tf**”, elimina el carácter “**#**” de las líneas donde aparezca, planifica la infraestructura e intenta desplegarla:
 - `nano vm_subred_1.tf`
 - `terraform plan`
 - `terraform apply`
- Verifica la otra “**azurerm_linux_virtual_machine**” creada así como el direccionamiento asociado:
 - `az vm list -g terraform-azure-<usuario>-rg --output table`
 - `az vm list-ip-addresses -g terraform-azure-<usuario>-rg --output table`



- Valida conectividad ICMP a la VM con IP pública:
 - `ping <ip_publica_instancia_creada>`
- Conéctate por SSH (copia la llave key a un archivo que se llame “key” dentro del directorio “keys” y cámbiale el permiso a “400”) a la VM con IP pública:
 - `ssh -v -l azureuser -i keys/key <ip_publica_instancia_creada>`
- Verifica el nombre de ese host:
 - `hostname`



- Valida conectividad ICMP a la otra VM con IP privada:
 - `ping <ip_privada_instancia_creada>`
- Intenta conectarse por SSH (copia la llave key a un archivo que se llame “key” y cámbiale el permiso a “400”) a la otra VM con IP privada:
 - `ssh -v -l azureuser -i key <ip_privado_instancia_creada>`
- Verifica el nombre de ese host:
 - `hostname`

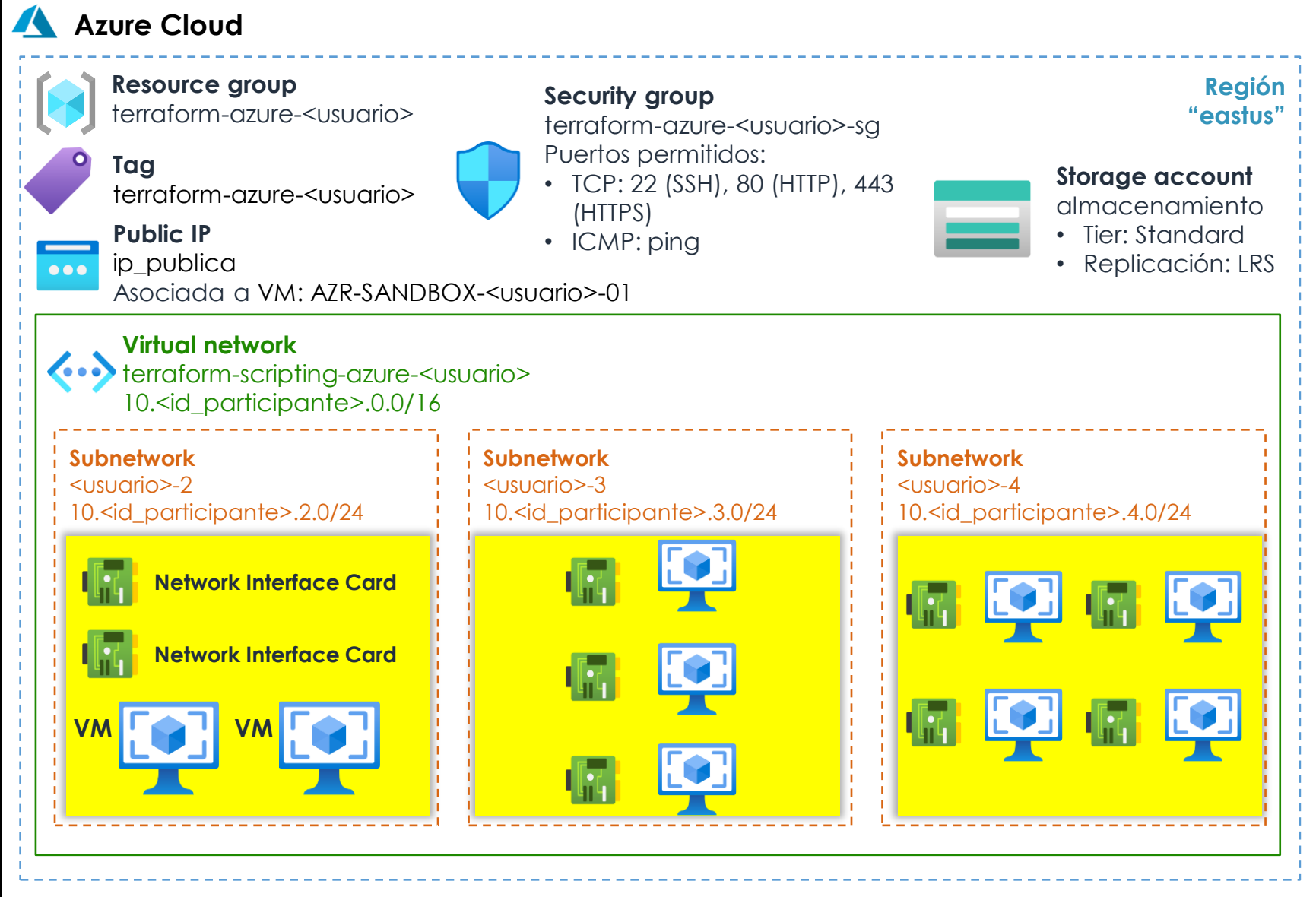
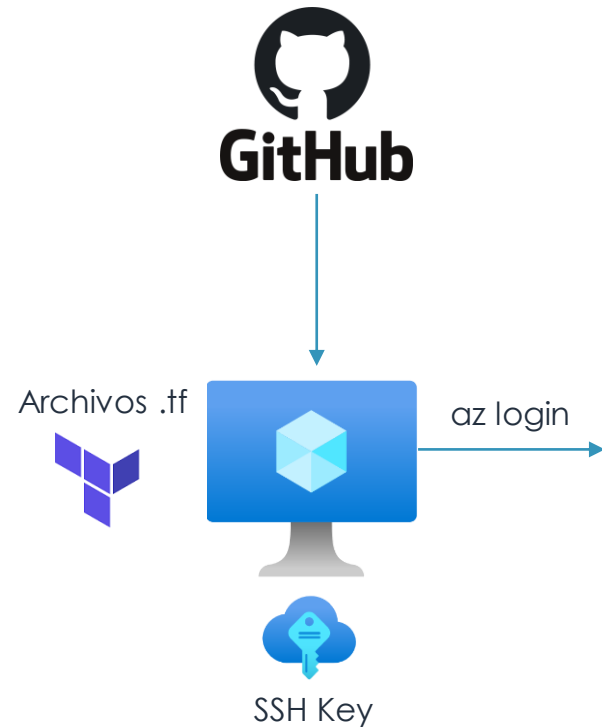


- **Reto 4 – Crea VMs**

- Crea el archivo **“vm_subred_2.tf”** para agregar **2 VMs “Ubuntu 18.04”** de tamaño **“micro”** con direccionamiento privado.
- Crea el archivo **“vm_subred_3.tf”** para agregar **2 VMs “Ubuntu 20.04”** de tamaño **“micro”** y **1 VM “Ubuntu 18.04”** tamaño **“small”**, todas con direccionamiento privado.
- Crea el archivo **“vm_subred_4.tf”** para agregar **2 VMs “Ubuntu 20.04”** de tamaño **“micro”** y **2 VMs “Ubuntu 18.04”** tamaño **“small”**, todas con direccionamiento privado.
- Verifica conectividad desde la VM que tiene IP pública.



Laboratorios – Comienza la diversión



- Tiempo para repasar jugando



Gracias

