Universidad de San Carlos de Guatemala
Facultad de Ingeniería
Escuela de Ciencias y Sistemas

Documentación Práctica 1

Erick Ivan Mayorga Rodríguez - 201901758

Proceso de instalación de Terraform

1. **Descargar** el ejecutable desde el sitio web de terraform.
2. **Extraer** el archivo en un directorio específico, de preferencia cercano de C:\ o un directorio común para programas como program files.
3. Realizar la **configuración de la variable de entorno**, se coloca la ruta del directorio utilizado en el paso anterior y dependiendo del sistema operativo se coloca la variable de entorno en el “PATH” del sistema, para luego de esto poder utilizar terraform en cualquier directorio mediante línea de comandos.

Proceso de instalación de Ansible

En un servidor de ubuntu, se procedió a utilizar los siguientes comando s

- “sudo apt update” : Para actualizar los paquetes del sistema.
- “sudo apt install software-properties-common -y”: actualiza scripts del sistema para gestión y configuración del software del ecosistema de papelería.
- “sudo add-apt-repository --yes --update ppa:ansible/ansible”: descarga desde un Personal package Archive ansible.
- “sudo apt install ansible -y”: se realiza la instalación de Ansible.

Cómo funciona Terraform

Se trata de una herramienta de infraestructura de código que permite a los usuarios definir y aprovisionar la infraestructura en diferentes servidores de nube, lo cual es útil para realizar configuraciones que definen el despliegue de un entorno o una topología en una infraestructura.

Para el aprovisionamiento en terraform se realizó lo siguiente:

Establecimiento de proveedor y uso específico de cuenta:

```
provider "google" {
    project = local.project_id
    region = "us-central1"
}

resource "google_service_account" "sa" {
    account_id   = "sa-201901758"
}
```

Establecimiento de reglas de firewall:

```
resource "google_compute_firewall" "web"{
    name = "web-access"
    network = local.network

    allow {
        protocol = "tcp"
        ports = ["80"]

    }

    source_ranges = ["0.0.0.0/0"]
    target_service_accounts = [google_service_account.sa.email]

}
```

Uso de IP estática:

```
resource "google_compute_address" "prod_vm_ip" {
  name   = "prod-vm-ip"
  region = "us-central1"
}
```

Creación de VM con uso de remote-exec para configuraciones adicionales:

```
resource "google_compute_instance" "prod-vm" {
    name         = "prod-vm"
    machine_type = "e2-micro"
    zone         = "us-central1-a"

    boot_disk {
        initialize_params {
            image = local.image
        }
    }

    network_interface {
        network = local.network
        access_config {
        nat_ip = google_compute_address.prod_vm_ip.address
        }
    }


    provisioner "remote-exec" {
    // Conectarse a través de SSH para configurar la VM de producción
    connection {
      type        = "ssh"
      host        = self.network_interface[0].access_config[0].nat_ip
      user        = "ansible"
      private_key = local.private_key
    }

    inline = [
      "sudo adduser ansible --gecos '' --disabled-password",
      "echo 'ansible:ansible' | sudo chpasswd",
      "echo 'ansible ALL=(ALL) NOPASSWD:ALL' | sudo tee /etc/sudoers.d/ansible",
      "sudo sed -i 's/^#PasswordAuthentication no/PasswordAuthentication yes/' /etc/ssh/sshd_config",
      "sudo systemctl restart sshd"
    ]
  }
}
```

Cómo funciona Ansible

Es una herramienta de gestión para la configuración y automatización, que tiene énfasis en la programación, red, operatividad e integración automática.

Ansible funciona ejecutando código en máquinas destino a las cuales denomina “nodos”, estos módulos pueden ser utilizados para cualquier cosa,desde copiar archivos a máquinas remotas hasta gestionar interfaces VLAN.

Los nodos se ejecutan desde un playbook de ansibleque a su vez se ejecuta en un nodo control. Los nodos control envían los módulos respectivos a todas las máquinas destino asignadas a la tarea.

en el presente proyecto se realizó lo siguiente:

“Become: yes” Utilizado para indicar que las tareas dentro del playbook se ejecutarán con privilegios elevados.
Instalar Nginx

```yaml
- hosts: produccion
  become: yes
  tasks:
  - name: Instalar Nginx
    apt:
      name: nginx
      state: present
      update_cache: yes
```

Eliminar la configuración predeterminada de Nginx

```yaml
- name: Eliminar la configuración predeterminada de Nginx
  file:
    path: /etc/nginx/sites-enabled/default
    state: absent
```

Copiar la configuración de Nginx para la aplicación

```yaml
- name: Copiar la configuración de Nginx para la aplicación
  template:
    src: nginx.conf.j2
    dest: /etc/nginx/sites-available/react_app
  notify: restart nginx
```

Habilitar la configuración de la aplicación

```yaml
- name: Habilitar la configuración de la aplicación
  file:
    src: /etc/nginx/sites-available/react_app
    dest: /etc/nginx/sites-enabled/react_app
    state: link
  notify: restart nginx
```

Copiar el build de React al servidor

```
- name: Copiar el build de React al servidor
  copy:
    src: /home/erick_mayorga_r/ansibleDespliegue/Practica1SA-201901758/practica1sa-201901758-app/dist
    dest: /var/www/react_app
    mode: '0755'
```

Define un handler llamado restart nginx, que utiliza el módulo service para reiniciar el servicio de Nginx cuando se notifique. Esto asegura que Nginx se reinicie solo si hay cambios en la configuración que requiera un reinicio.

```
handlers:
- name: restart nginx
  service:
    name: nginx
    state: restarted
```

Describir tecnologías a usadas

- **Terraform:** utilizado para provisionar y gestionar la infraestructura en proveedores de servicios de cloud como GCP, AWS, Azure, etc.
- **Ansible:** utilizado para la automatización de procesos en la configuración de servidores, despliegue de aplicaciones y orquestacion de otras tareas posteriores a un aprovisionamiento de un servicio en la nube.
- **Proveedor de servicios de nube**: En este caso se utilizó la nube de google, un proveedor de servicios de nube se trata de una compañía que cobra por servicios de máquinas virtuales para su uso como servidor, bases de datos, redes, almacenamiento y otros servicios. Se ofrece como un tercero para que las compañías tengan acceso a estos servicios.
- **Git**: utilizado como sistema de control de versiones.
- **React**: Es un framework de desarrollo web que utiliza NodeJs, en este caso solo se utilizaron las librerías específicas de react dado el uso de Vite, para generar el proyecto.