Terraform & Pulumi

Jean-Marc Pouchoulon

octobre 2023

1 Avant de commencer

1.1 Pré-requis, recommandations et notation du TP.

Vous devez avoir Docker, KVM/libvirt installés sur votre machine afin de réaliser ce TP. Pour certaines parties une serveur proxmox et un serveur ESXi seront nécessaires. Terrafom doit être installé sur votre machine. (voir la procédure officielle)

Installez Terraform en suivant la procédure officielle

Vous ferez valider votre travail par l'enseignant à la fin de chaque partie.

2 Terraform & Docker

2.1 Commandes de bases Terraform

1. A l'aide de ces deux fichiers providers.tf main.tf générez un conteneur docker avec Terraform.

```
# fichier providers.tf

terraform {
  required_providers {
    docker = {
      source  = "kreuzwerker/docker"
      version = "3.0.2"
    }
  }
}

provider "docker" {
  host = "unix:///var/run/docker.sock"
}
```

```
# fichier main.tf
# Pulls the image
```

```
resource "docker_image" "ubuntu" {
  name = "ubuntu:latest"
}

# Create a container
resource "docker_container" "foo" {
  image = docker_image.ubuntu.image_id
  name  = "foo"
}
```

Utilisez les commandes suivantes:

```
terraform init
terraform validate
```

2. Que contient la directory .terraform ? le fichier .terraform.lock.hcl (voir `https://developer.hashicorp.com/terraform/tutorials/cli/init` pour vous aidez)
3. Rajoutez le provider suivante et refaite un validate. que se passe-t-il ?

```
random = {
  source  = "hashicorp/random"
  version = "3.5.1"
}
```

4. Générez un plan d'exécution et examinez-le avec les commandes suivantes:

```
terraform plan -out "tfplan"
terraform show "tfplan"
```

5. Appliquez le plan d'exécution avec la commande suivante:

```
terraform  apply -auto-approve "tfplan"
```

6. Utilisez l'instruction count=3 afin d'obtenir 3 conteneurs docker et modifiez le code en utilisant la variable ${count.index} .
7.

2.2 Un peu plus loin avec Docker et Terraform

1. Modifiez ce code afin d'obtenir une instance de container en état running en utilisant les instructions suivantes :

```
must_run = true
publish_all_ports = true
command = [....
```

2. Modifiez ce code afin d'obtenir une instance running de registry.iutbeziers.fr/debianiut:latest
   Vous devez avoir l'image sur votre machine avant de lancer le code terraform.
   Vous devez référencer l'image de cette façon:

```
data "docker_image" "debianiut" {
   name = "registry.iutbeziers.fr/debianiut:latest"
 }
# image registry.iutbeziers.fr/debianiut:latest
 resource "docker_image" "debianiut" {
 name = data.docker_image.debianiut.name
}
```

   et l'utilisez ensuite dans la ressource docker_container.

3. Supprimez votre environnement containérisés maintenant.

```
terraform destroy -auto-approve
```

4. Instanciez des containers sur un VM distante.
   Tout d'abord en utilisant SSH.

```
provider "docker" {
        host     = "ssh://user@remote-host:22"
        ssh_opts = ["-o", "StrictHostKeyChecking=no", "-o", "UserKnownHostsFile=/dev/null"]
}
```

   Puis rendez accessible Docker en remote sur le port tcp 2375 via une socket TCP en éditant le service Docker:

```
systemctl edit docker.service
```

   Rajouter les lignes suivantes:

```
[Service]
ExecStart=
ExecStart=/usr/bin/dockerd -H fd:// -H tcp://0.0.0.0:2375
```

   Redémarrez le service Docker.
   Modifiez votre providers.tf afin d'utiliser le port tcp 2375:

```
provider "docker" {
  host = "tcp://your-host-ip:2375/"
}
```

   nb: cette configuration ouverte à tout n'est pas sécurisée. Il aurait fallu utiliser TLS et des certificats clients.

2.3 Terraform wordpress

Utilisez l'article sous "Creative Common" de Julien Morot pour créer un wordpress avec Terraform.

3 Terraform et KVM/libvirt

3.1 pré-requis

Tout d'abord vous devez éditer le fichier /etc/libvirt/libvirtd.conf et remplir la valeur suivante:

```
security_driver = "none"
```

Puis redémarrer le service libvirt:

```
systemctl restart libvirtd
```

On va utiliser le provider libvirt pour terraform. Du à un bug il vous faut utiliser la version 1.5.7 de terraform. (voir `https://releases.hashicorp.com/terraform/1.5.7/`)

```
terraform {
  required_providers {
    libvirt = {
      source = "dmacvicar/libvirt"
    }
  }
  # instance the provider
```

```
  provider "libvirt" {
    uri = "qemu:///system"
  }
}
```

3.2 Créez une machine virtuelle ubuntu KVM via Terraform

Vous pouvez vous inspirer du code de Stéphane Robert. La configuration de la machine virtuelle utilisera cloud-init pour "customiser" la machine virtuelle.

3.3 Créez une machine virtuelle Ubuntu et Rocky via Terraform

Récupérez le projet :

```
git clone https://github.com/pushou/terraform-student.git
```

Récupérer les images cloud suivantes:

```
wget https://dl.rockylinux.org/pub/rocky/9/images/x86_64/Rocky-9-GenericCloud.latest.x86_64.qcow2
wget https://cloud-images.ubuntu.com/jammy/current/jammy-server-cloudimg-amd64.img
```

Resizer les images avec la commande suivante:

```
qemu-img resize jammy-server-cloudimg-amd64.img +20G
qemu-img resize focal-server-cloudimg-amd64.img +20G
mv Rocky-9-GenericCloud.latest.x86_64.qcow2 rocky.qcow2
mv Rocky-9-GenericCloud.latest.x86_64.qcow2 ubuntu.qcow2
export LIBGUESTFS_BACKEND=direct
sudo virt-customize -a ubuntu.qcow2 --root-password password:root
sudo virt-customize -a rocky.qcow2 --root-password password:root
```

Rajoutez à ce build une machine virtuelle Debian stable.

4 Utilisation de Pulumi