DO THUAN Céline - JEONG Seo-Jin - QIU Ruizhe

Lien du git

Voici le lien du git avec tous les fichiers de configuration : https://github.com/celdot/gin208_project.git

A chaque fois qu'il y a une commande `terraform destroy` suivi de ````terraform planetterraform apply```, de nouvelles adresses IP sont allouées aux machines. Il faut donc remplacer les adresses IP adaptées dans les fichiers de configuration à chaque fois.

Architecture du réseau

Avec AWS, nous mettons en place 2 réseaux : un réseau public et un réseau privé. Les deux réseaux sont connectés à une internet gateaway, en passant par une table de routage.

Nous allons ensuite instancier une VM dans chacun des réseaux. La VM instanciée sur le réseau public va servir d'interface web pour les utilisateurs qui regardent le live stream. LA VM instanciée sur le réseau privé stocke les vidéos à envoyer en live et va les transmettre à la VM publique.

Modules utilisés

Les deux VMs sont configurées avec Ansible. On installe `Nginx` et `rtmp` dans la VM publique et ````ffmpeg``` dans la VM privée.

VM publique

Groupe de sécurités

Nous mettons en place un groupe de sécurité sur le fichier terraform `main.tf` pour la VM publique, qui autorise tout le traffic sortant et qui autorise tout le trafic entrant sur le port 22 (SSH), le port 80 (HTTP), le port 443 (HTTPS). On autorise également le trafic entrant sur le port 1943 (le port choisi que nous avons choisi pour le trafic rtmp) pour le trafic provenant des adresses IP du masque d'adresses IP privées que nous avons alloué.

Installation et configuration de RTMP

Nous utilisons Ansible pour installer `Nginx` et le package `rtmp`. Nous devons ensuite configurer les deux, et ce dans le fichier `nginx-install.yaml`.

Modification de la configuration du site Nginx

Pour la configuration de l'hôte virtuel, nous devons modifier le fichier de configuration du site Nginx `devops.conf.j2` pour gérer les requêtes HTTP et fournir des services vidéo.

Déploiement du site web

Nous avons modifié `index.html` pour permettre à la page Web de lire les flux vidéo HLS.

Utilisation d'Ansible pour déployer le site

Nous mettons en place un fichier `inventory_frontend.ini`pour initialiser Ansible et lui permettre d'utiliser SSH. Il faut remplacer l'adresse IP par l'adresse IP publique de la VM publique.

Il faut aussi remplacer l'adresse IP par cette adresse dans la configuration DNS.

On exécute ensuite les commandes suivantes pour permettre à Ansible de configurer notre server public.

```
$ ansible-playbook -i inventory.ini nginx-install.yaml
$ ansible-playbook -i inventory.ini create-vhost.yaml
$ ansible-playbook -i inventory.ini deploy-website.yml
```

Video Streamer

Dans cette partie, nous configurons le serveur de streaming pour stocker des vidéo et la transmettre au serveur public via ffmpeg.

Nous avons choisi de faire l'exercice avec uniquement une vidéo.

Connexion SSH

Ici, le serveur est censé être un serveur privé, avec aucun accès à internet et ne devrait communiquer qu'avec le réseau public.

Néanmoins, nous devons le configurer avec Ansible. Il est donc nécessaire de pouvoir s'y connecter avec SSH.

Nous utilisons donc un bastion SSH pour se connecter en SSH au serveur vidéo, depuis le containeur ilab, en passant par le serveur front et on autorise le traffic SSH depuis notre réseau public et non pas depuis n'importe quelle adresse IP. Ainsi, le réseau privé est également connecté au gateway internet.

On précise cette configuration dans le fichier `inventory_backend.ini`. Il faut remplacer la première adresse IP par l'adresse IP privée de la VM privée et la deuxième adresse IP par l'adresse IP publique de la VM publique.

Groupe de sécurité

Nous mettons en place un groupe de sécurité sur le fichier terraform `main.tf` pour la VM privée, qui autorise tout le traffic sortant et qui autorise le trafic entrant sur le port 22 (SSH) et le port 1943 (rtmp) depuis les adresses IP de notre réseau public.

Installation de FFMPEG

Nous utilisons Ansible pour installer et vérifier l'installation de ffmpeg sur le serveur dans le fichier `ffmpeg-install.yaml`.

Configuration de FFMPEG

Dans le playbook `ffmpeg-config.yml`, nous organisons les tâches de création des répertoires nécessaires, de téléchargement des vidéos et des scripts de streaming ffmpeg sur le serveur, ainsi que les tâches de démarrage et de gestion automatiques du processus de streaming ffmpeg.

Nous avons d'abord créé le répertoire de configuration de ffmpeg et le répertoire de stockage vidéo. Ensuite, nous vérifions si la vidéo existe et la téléchargeons sur le serveur si elle n'existe pas.

Après cela, nous téléchargeons le script de streaming ffmpeg `ffmpeg -re -i /videos/BigBuckBunny_320x180.mp4 -loop -1 -c:v copy -c:a copy -f flv rtmp://192.168.3.75:1935/live/ stream` pour transmettre la vidéo au serveur web frontal. Enfin, nous créons un service systemd pour gérer le processus de streaming et démarrer le script.

Il faut utiliser l'adresse IP privée de la VM publique.

On utilise ici l'adresse IP privée du réseau public pour le trafic vidéo.

On exécute ensuite les commandes suivantes pour configurer le serveur.

```
$ ansible-playbook -i inventory.ini ffmpeg-install.yaml
$ ansible-playbook -i inventory.ini ffmpeg-config.yml
```

Maintenant, dans le dossier `/tmp/hls` du serveur web frontal, nous pouvons trouver les fichiers stream.m3u8`et`stream-0.ts` et accéder à l'URL http://192.168.3.75/ pour regarder la vidéo.