Projet Cloud Documentation

Réalisé par
Triouleyre Thomas
Martin Mathieu

Date de création : May 20, 2023

Contents

Chapter 1

Introduction

1.1 Présentation du projet

Ce projet a pour but de déployer une application fournie de la manière la plus automatisée et résiliante possible. L'application permet à ses utilisateurs d'héberger des images, en les redimensionnant au passage dans différentes tailles. L'application comporte de trois composants :

un backend se chargant de sauvegarder les images téléversées, ainsi que de servir les images hébergées un worker se chargant de redimensionner les images téléversées dans différentes tailles un frontend, sous la forme d'une interface web, pour permettre aux utilisateurs d'envoyer les images Les images sont stockées dans un espace de stockage compatible S3.

Le backend utilise la bibliothèque Python Celery pour envoyer des tâches à exécuter au worker via une file de message.

1.2 Ce que nous avons réalisé

Vous trouverez dans ce rapport un résumé de toutes les commandes et déploiments que nous avons réalisé. Les chapitres sont plus ou moins succint en fonction du temps et de la réussite de chacun. Nous avons déployer l'application en local chacun des composants marchaient sans problème mais sans déploiement nomad et consul. Nous ne savons pas si le projet fonctionne avec Nomad et Consul car nous avons des conflits avec un autre groupe qui a fait buguer Nomad et nous n'avons pas réussi à retravailler dessus depuis dans des délais raisonable. Nous avons donc fait en sorte que le projet fonctionne en local, chaque docker indépendamment fonctionnait, mais nous n'avons pas essayé de les faire marcher ensemble. Vous trouverez donc dans ce fichier la documentation du projet et les explications de ce que nous avons fait. Dans le dépôt Git, les fichiers que nous avons ajoutés sont dans les dossiers "src", "conf_nomad_consul", "web" et "api".

Chapter 2

Dockerfile

Les fichiers Dockerfile sont écrits dans le dossier "web" pour le frontend et dans le dossier "api" pour le worker et le backend. Nous avons eu du mal à réaliser le Dockerfile pour le backend et le worker car Poetry nous a donné du fil à retordre ainsi que les variables d'environnement. Nous avons également eu beaucoup de mal à cadrer le projet pour comprendre le nombre de Dockerfile à réaliser et quoi déployer sur chaque conteneur. Exemple pour le web :

```
FROM node:latest

WORKDIR /usr/src/app

COPY . .
EXPOSE 3000

RUN npm install
RUN npm run build .

CMD ["npm" , "run", "start"]
```

Ici on récupère l'image de node, on se place dans le dossier de l'application, on copie le contenu du dossier courant dans le conteneur, on expose le port 3000, on installe les dépendances, on build l'application et on lance l'application. Pour le worker et le backend le fichier est un peu plus compliqué mais le principe reste le même.

Chapter 3

Commandes

Pour lancer en local les conteneurs, il faut se connecter en SSH sur la machine virtuelle avec les commandes suivantes :

```
ssh -L 3000:localhost:3000 <vm>
```

```
docker build -t web web/
docker run -p 3000:3000 web
```

Pour le frontend par exemple.

Chapter 4

Script Bash

Dans le dossier "src", 4 scripts Bash sont présents :

- `launch_borie.sh` : permet de reconfigurer entièrement la machine virtuelle "borie" en clonant le dépôt Git, en passant sur la bonne branche, en copiant les fichiers de configuration Nomad et Consul aux bons endroits ("/etc/consul.d" et "/etc/nomad.d"), en lançant les services Consul et Nomad, et en rejoignant le cluster lié au bon datacenter.
- `launch_republique.sh` : identique au script précédent, mais pour la machine virtuelle "republique".
- `launch_wacken.sh` : identique au script précédent, mais pour la machine virtuelle "wacken".
- `launch_all.sh` : se connecte sur la machine virtuelle "borie" et lance la commande `nomad job run projet-cloud-virt/conf_nomad_consul/conf-nomad.hcl`, qui permet de lancer les jobs Nomad sur les machines virtuelles. Il est possible, en enlevant les commentaires, de lancer les 3 scripts précédents.

Chapter 5

Configuration Nomad et Consul

5.1 Consul

Nous avons récupéré la configuration Consul déjà présente dans le dossier `consul.d` de la machine virtuelle partagée "borie" et nous l'avons modifiée pour la mettre sur les machines virtuelles "wacken" et "republique" de la manière suivante :

Pour "republique" :

- `advertise_addr = "172.16.1.25"`
- `dns = "172.16.1.25"`
- `bootstrap_expect = 3`

Pour "wacken" :

- `advertise_addr = "172.16.1.32"`
- `dns = "172.16.1.32"`
- `bootstrap_expect = 3`

Pour "borie", aucune modification n'a été faite, à part le `bootstrap_expect` qui est passé à 3.

5.2 Nomad

Nous avons récupéré la configuration Nomad déjà présente dans le dossier `nomad.d` de la machine virtuelle partagée "borie" et nous l'avons modifiée pour la mettre sur les machines virtuelles "wacken" et "republique" de la manière suivante :

Pour "republique" :

```
advertise {
  http = "172.16.1.25"
  rpc = "172.16.1.25"
  serf = "172.16.1.25"
}

server {
  enabled = true
  bootstrap_expect = 3
}

client {
  enabled = true
}
```

Pour "wacken" :

```
advertise {
  http = "172.16.1.32"
  rpc = "172.16.1.32"
  serf = "172.16.1.32"
}

server {
  enabled = true
  bootstrap_expect = 3
}

client {
  enabled = true
}
```

Nous avons eu un problème de regroupement des clusters avec un autre groupe, ce qui a fait buguer Nomad, et nous n'avons pas réussi à retravailler dessus depuis. Mais avant cela, nous arrivions à déployer le bon nombre de machines virtuelles et de conteneurs.

Chapter 6

HAProxy

Le HAProxy est là pour gérer le load balancing. D'après ce que nous avons vu, nous pouvons le déployer sur le frontend ou le backend en fonction de nos besoins.

L'exécution de HAProxy sur Nomad permet de le gérer en tant que travail Nomad, ce qui facilite la gestion des tâches et des mises à l'échelle dans le cluster Nomad. Cela peut être utile si nous avons déjà une infrastructure basée sur Nomad et que nous souhaitons centraliser la gestion de tous les services.

L'exécution de HAProxy sur Consul tire parti des fonctionnalités de service discovery de Consul, ce qui simplifie la configuration du load balancing en utilisant les informations de service enregistrées dans Consul. Cela peut être utile si nous utilisons Consul pour la découverte des services.

Nous avons donc décidé, en fonction de l'état d'avancement du projet et du temps qu'il nous reste, de le déployer sur Consul.

La configuration se trouve dans le fichier `haproxy.cfg`.

Chapter 7

Keepalived

Keepalived est installé sur chacune des machines afin d'assurer une redondance des services en cas de panne d'une des machines. La configuration est quasiment la même pour chacune des machines et se trouve dans les fichiers nommés `keepalived_vm.conf`.

La différence qui se trouve dans ces configurations réside au niveau de l'état, où nous trouvons un MASTER (la machine virtuelle partagée "borie") et deux BACKUP (les machines virtuelles "republique" et "wacken").

Nous utilisons l'adresse virtuelle donnée par le professeur : 172.16.3.0/16

Chapter 8

Conclusion

En conclusion, malgré les difficultés rencontrées avec les conflits et les problèmes de configuration, nous avons réussi à faire fonctionner les différentes parties du projet en local. Cependant, nous n'avons pas pu les faire marcher ensemble en raison des problèmes avec Nomad. Malgré cela, nous avons documenté toutes les étapes que nous avons suivies et les configurations que nous avons réalisées. Nous espérons que ces informations seront utiles pour la suite du projet.