Initiation à l'orchestration avec Docker Compose

Maxime Lambert

Février 2024

Cinquième TP de virtualisation pour le module R4.08, BUT Informatique deuxième année.

Travail en monôme ou binôme en salle de machines.

Savoirs de référence étudiés :

- Virtualisation d'applications
- Conteneurs

Pré-requis :

- Aisance sous un système Unix en ligne de commande
- CM du module R4.08 sur les conteneurs Linux
- Notions sur les réseaux TCP/IP
- Architecture client-serveur

Matériel ou logiciels utilisés :

- Accès au serveur Proxmox de l'Université depuis un navigateur web

1 Présentation du TP

Au cours de cette séance, vous allez prendre le temps de finaliser le TP3. Vous allez aussi vous appuyer sur les travaux précédemment effectués lors du TP4 où nous avons déployé une infrastructure CI/CD. Vous devrez obtenir un résultat similaire en utilisant cette fois-ci le client Docker Compose.

1.1 Évaluation

Un QCM de 20 minutes est à réaliser en fin de séance, votre enseignant vous indiquera le moment opportun. **Cette évaluation est strictement individuelle.**

L'intégralité des connaissances pratiques acquises lors des TP et des TD seront évaluées. Il n'est pas nécessaire de finaliser tous les déploiements dans la partie sur docker-compose.

1.2 Prérequis

→ Repartez de la VM utilisée lors du TP4.

→ Si cela n'a pas déjà été fait lors de ce précédent TP, exécutez sur votre machine hôte la commande

```
$ mv ~/.docker/config.json ~/.docker/config.json_ignore
```

De cette manière aucun proxy ne sera utilisé par vos conteneurs, mais vous pourrez facilement rétablir la configuration au besoin.

→ Faites le ménage sur votre machine

```
$ docker rm -f $(docker ps -aq)
$ docker system prune -f --all
```

→ Dans le fichier `/etc/hosts`, vérifiez la présence de la ligne :

```
$ 127.0.0.1       gitea drone-serveur drone-demon registre sonar sonarqube
```

→ Dans le fichier `/home/user/.profile`, vérifiez la présence de la ligne :

```
export no_proxy="localhost,127.0.0.1/8,172.17.0.0/16,prof,gitea,drone-
serveur,drone-demon,sonar,sonarqube,registre"
```

N.B : ne pas faire de retour à ligne, c'est une contrainte de formatage sur ce document.

→ En cas de modification d'un des deux précédents fichiers, redémarrez votre machine avec la commande `sudo reboot`.

2 Finalisation du TP3 : Exploitation d’un service web

Durée estimée : 30 minutes

Vous commencerez cette séance par terminer la section 4 « Exploitation d’un service web avec Docker » du TP3.

3 Découverte de Docker Compose

3.1 Motivation

Différents clients peuvent s'interfacer avec le moteur de Docker. Jusqu'à maintenant, nous avons utilisé exclusivement le client en ligne de commande. Ce client est idéal pour déployer des solutions reposant sur des conteneurs autonomes.

Le TP4 vous a conduit à déployer un ensemble de conteneurs devant collaborer afin de constituer notre chaîne d'intégration et de déploiement continu. Le déploiement des conteneurs était relativement fastidieux, notamment à cause de la complexité des commandes, et de la procédure à suivre : instancier un réseau, créer les répertoires pour persister la configuration, puis lancer le conteneur gitea, et ensuite drone.

3.2 Qu'est ce que Docker Compose ?

Docker Compose est un outil permettant de définir et d'exécuter une application multi-conteneurs. On utilise le terme d'orchestration lorsque l'on cherche à automatiser le déploiement et la gestion de plusieurs conteneurs.

Pour configurer une application multi-conteneurs, Compose interprète un fichier se nommant `docker-compose.yml`.

Dans ce fichier, vous allez définir la configuration, le réseau, les variables d'environnements, les volumes et les services.

Les services sont liés aux conteneurs mais c'est une notion différente. Il s'agit d'un concept abstrait permettant d'exécuter une ou plusieurs fois la même image d'un conteneur. Autrement dit, les services de Docker Compose permettent de réaliser la scalabilité horizontale[1] d'une application.

Par défaut avec Compose, un service comporte une instance de conteneur. Nous conserverons ce fonctionnement pendant ce TP.

3.3 Fonctionnalités de Compose

Voici les fonctionnalités clés de Docker Compose :

- Possibilité de définir de multiples environnements isolés sur le même hôte
- Conservation des données d'un volume entre les différentes versions d'un même conteneur
- Au redémarrage d'un service, uniquement les conteneurs dont la configuration a été modifiée sont recréés

1 Contrairement à la scalabilité verticale qui consiste à rajouter des ressources matérielles afin de gérer une charge de travail plus importante. La scalabilité horizontale consiste à ajouter plusieurs instances d'un même service réalisant le même type de tâches.

On utilise rarement Docker Compose dans un contexte de production bien que rien ne l'interdit. Généralement, son usage est réservé :

- Aux environnements de développement, par exemple pour l'*onboarding* de nouveaux arrivants, ou pour simplifier le développement local dans une architecture micro-services
- Aux environnements de tests automatisées. C'est un outil puissant permettant de monter toute une infrastructure avec une simple commande

3.4 Assistance sur la syntaxe du fichier docker-compose.yml

L'exemple de fichier `docker-compose.yml` du cours a été remis à l'annexe 1.

Un mémo contenant toute la syntaxe nécessaire pour réaliser ce TP est disponible à l'adresse suivante : https://www.programonaut.com/wp-content/uploads/2021/07/docker-compose-cheat-sheet.pdf

Plus verbeux, mais très complet la documentation officielle de Compose : https://docs.docker.com/compose/

3.5 Les commandes de Compose

Compose dispose d'une interface en ligne de commande appelable avec la commande linux docker compose.

Pour ce TP, il y a une commande à connaître : `docker-compose up`

Cette commande construit, (re)crée, démarre et attache à la sortie standard les conteneurs d'un service.

À moins qu'ils ne soient déjà en cours d'exécution, cette commande démarre également tous les services liés.

Pour arrêter Compose, il faut envoyer le signal SIGINT (`ctrl + c`)

4 Déploiement d’une solution multi-conteneurs avec Compose

→ Il vous est demandé de déployer à l’aide de Docker Compose les services « Gitea » et « Drone-serveur ».

Vous devrez obtenir le même résultat qu’à la fin de la section 3.3.3 « Instanciation du conteneur drone-serveur » du TP4. C’est-à-dire que votre serveur drone devra avoir accès à vos dépôts Git hébergés sur Gitea.

S’il vous reste du temps, vous pouvez continuer le TP4.

5 Annexes

Annexe 1 : Exemple de syntaxe d'un fichier docker-compose.yml

```
version: "3"

services:
  bdd:
    image: "postgres:15.1"
    environment:
      - POSTGRES_USER=my-user
      - POSTGRES_PASSWORD=pass
    ports:
      - 5432:5432
    volumes:
      - /tmp/data:/var/lib/postgresql/data
  frontend:
    build: ./webapp
    ports:
      - 80:80
    environment:
      - BACKEND_URL=http://backend:8080
  backend:
    image: mybackend:latest
    ports:
      - 8080:8080
```