C09-docker - TP LINUX - MTN

Guillaume ASTIER

26/02/16

Table des matières

Introduction

Docker est un logiciel libre qui automatise le déploiement d'applications dans des conteneurs logiciels

Docker est un outil qui peut empaqueter une application et ses dépendances dans un conteneur isolé, qui pourra être exécuté sur n'importe quel serveur ». Ceci permet d'étendre la flexibilité et la portabilité d'exécution d'une application, que ce soit sur la machine locale, un cloud privé ou public, une machine nue, etc.

INSTALL

Depuis les binnaires

Bien que docker / podman existe sur les dépôts, (apt/dnf …) nous allons utiliser la version “officiel” de docker.

https://get.docker.com/

Ce script permet d’analyser la distributions sur la quelle vous êtes et faire le travail a votre place.

- Faite un curl de l’url pour voire le code s’afficher sur votre TTY (ou depuis votre poste hôte depuis un navigateur)
- Avec la redirection pipe (|) envoyez la sortie standard de votre commande curl au binaire bash.
- Indiquez ce qu’il se passe.

USER

Pour pouvoir utiliser docker il faut que votre utilisateur soit dans le groupe du même nom. Faite l’opération nécessaire et n’oubliez pas de vous déconnecter puis de vous reconnecter.

Lancement d’un instance docker de base

Help me ...

Pour lancer une instance avec docker il faut utiliser les options disponibles dans le binnaire :

```
1 docker --help
```

Lancement d’un Hello World

```
1 $ docker run hello-world
2 Unable to find image 'hello-world:latest' locally
3 latest: Pulling from library/hello-world
```

```
4 9bb5a5d4561a: Pull complete
5 Digest: sha256:
     f5233545e43561214ca4891fd1157e1c3c563316ed8e237750d59bde73361e77
6 Status: Downloaded newer image for hello-world:latest
7
8 Hello from Docker!
9 This message shows that your installation appears to be working
     correctly.
```

Comme vous pouvez le remarquer l'image docker n'étant pas disponible le binaire est aller chercher dans ses *dépôts* l'image correspondante.

Une fois téléchargé, docker lance le conteneur *Hello*

Le seul travail du conteneur est d'afficher *Hello from Docker!* sur la sortie standard et de se terminer.

Relancez l'instance

Lancement d'une instance docker complète

Récupération de l'image ubuntu

Dans un premier temps nous allons d'abord récupérer l'image en question :

```
1  docker pull ubuntu:trusty
2 trusty: Pulling from library/ubuntu
3 324d088ce065: Pull complete
4 2ab951b6c615: Pull complete
5 9b01635313e2: Pull complete
6 04510b914a6c: Pull complete
7 83ab617df7b4: Pull complete
8 Digest: sha256:
     b8855dc848e2622653ab557d1ce2f4c34218a9380cceaa51ced85c5f3c8eb201
9 Status: Downloaded newer image for ubuntu:trusty
```

Cette commande va télécharger depuis le Docker Hub l'image de la version 14.04 (trusty) d'Ubuntu.

Il existe bien d'autres images que vous pourrez trouver sur le registry Docker.

Le registry (dépôt) est accessible depuis l'url : https://hub.docker.com/

Il faut se créer un compte pour pouvoir visualiser les dépôts.

Visualisation des images

Pour visualiser les images disponibles vous devez utiliser :

```
1 $  docker images
2 ubuntu                                                              14.04
                    3b853789146f        5 weeks ago         223MB
3 hello-world                                                         latest
                   e38bc07ac18e        5 weeks ago         1.85kB
```

Pour info

Pour information, une machine virtuelle complète peut atteindre des tailles conséquentes.

Si on prend le filer + le systèmes + les disques Virtuelle on atteint au minum 4 a 6 Go

Ici notre conteneur docker ne fait que 223 MB

Lancement du conteneur

Lancement avec les options

```
1 $ docker run -ti ubuntu:trusty bash
```

Ici nous laçons donc le conteneur avec pour process bash

- run : Lance le conteneur
- -ti : Mode interactif dans terminal
- ubuntu:trusty : l'image à utiliser
- bash : le process qui va être lancé sur le conteneur

Analyser les conteneurs docker lancés :

```
$  docker ps
CONTAINER ID        IMAGE               COMMAND
    CREATED             STATUS              PORTS
                                          NAMES
baecdd9bc67d        ubuntu:trusty       "bash"                  About
     a minute ago   Up About a minute
                                                  admiring_yalow
```

Comment se connecter au conteneur

```
 docker exec -it baecdd9bc67d bash
```

Nous lançons docker avec comme options :

- exec : execute sur la machine distante
- -ti : terminal interactif
- baecdd9bc67d : Id de l'insance
- bash : Le process à éxecture sur le conteneur

```
$ docker exec -ti fe4bf20b38bc bash
root@fe4bf20b38bc:/# ps axf
  PID TTY      STAT   TIME COMMAND
   25 pts/0    Ss     0:00 bash
   40 pts/0    R+     0:00  \_ ps axf
    1 ?        Ss     0:00 bash
```

On peut voir ici le nombre TOTAL de process sur le conteneur.

- PID : *1* le process bash lancé en daemon
- PID : *25* le process bash lancé lors de la connexion avec *exec*

Création d'un conteneur docker

Pour créeer une image docker nous devons créer un fichier Dockerfile qui servira de base à la creation de l'image:

Le fichier ne comprend pas toutes les options mais sert de base pour la génération.

Certaines options lors du lancement de l'image complète les possibilitées infini de docker

---

```
# Depuis l'image :
FROM debian:stable
# Qui s occupe de cette image:
MAINTAINER guillaume@gastier.net
# Lancement de commande sur le conteneur pour la création : (RUN)
RUN apt-get update && apt-get upgrade -y && apt-get install -y nginx
    procps
# Ajout une variable d'environement au sein de l'image
ENV ISEN_VERSION 1.3
# Ouvre le port 80
EXPOSE  80
# Ajout les volumes local vers le conteneur:
ADD ./home/www /home/www
#Copie les données :
COPY ./etc/nginx/sites-enabled/default /etc/nginx/sites-enabled/default
# Exectute la commande au démarrage du conteneur
CMD ["nginx", "-g", "daemon off;"]
```

Les commandes :

Pour compiler votre image :

```
docker build -t isen/mtn:1.1 .
```

Pour lancer votre conteneur :

```
docker run -d isen/mtn:1.1
```

Pour lancer votre conteneur avec redirection de port :

```
docker run -p 8080:80 -d isen/mtn:1.1
```

Pour lancer votre conteneur avec le partage de repertoire :

```
docker run -v $(pwd)/home/www /home/www -d isen/mtn:1.1
```

Pour lancer votre conteneur avec rep + port

```
docker run -v $(pwd)/home/www:/home/www -p 8080:80 -d isen/mtn:1.1
```

Pour vous connecter à votre conteneur en shell :

```
docker exec -ti 33f34112c812 bash
```

Mise en pratique

Récupération d'un dépôt git:

Nous allons prendre le dépôt git ci dessous :

```
git clone https://git.gastier.net/guillaume/docker_mtn_tp
cd docker/
```

Génération et lancemnt de l'intance

Explorez le contenu du dépôt et expliquer les étapes de fabrication

---

Analyse des données

- Que fait cet instances et comment
- Réalisez à l'aide de l'outil un nouveau document à l'aide du langage markdown (https://docs.framasoft.org/fr/gra