

Jonathan Morgado-Samagaio

TP ASR 6

Introduction.....	3
Installation et démarrage de Docker.....	3
Utilisation de Docker.....	4
Création d'un serveur de développement.....	5
Pour aller plus loin.....	6

Introduction

Dans ce TP, nous allons voir comment utiliser Docker. Nous allons télécharger, créer, lancer et modifier des images et accéder aux conteneurs.

Installation et démarrage de Docker

On commence par créer une machine virtuelle CentOS. On la lance et on ouvre un terminal en super utilisateur.

On ajoute le repo de docker dans Yum :

```
[root@localhost iutbuser]# yum-config-manager \ --add-repo \ https://download.docker.com/linux/centos/docker-ce.repo
Modules complémentaires chargés : fastestmirror, langpacks
```

Et on installe Docker :

```
[root@localhost yum.repos.d]# yum install docker-ce
```

On démarre Docker :

```
[root@localhost yum.repos.d]# systemctl start docker
```

```
[root@localhost yum.repos.d]# systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/usr/lib/systemd/system/docker.service; disabled; vendor preset: disabled)
   Active: active (running) since ven. 2022-03-25 16:21:49 CET; 54s ago
     Docs: https://docs.docker.com
    Main PID: 16968 (dockerd)
      Tasks: 9
     Memory: 38.5M
    CGroup: /system.slice/docker.service
            └─16968 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd...

mars 25 16:21:46 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:46.94...c
mars 25 16:21:46 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:46.94...c
mars 25 16:21:47 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:47.00...c
mars 25 16:21:48 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:48.92...c
mars 25 16:21:49 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:49.19...c
mars 25 16:21:49 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:49.53...c
mars 25 16:21:49 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:49.59...4
mars 25 16:21:49 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:49.59...c
mars 25 16:21:49 localhost.localdomain systemd[1]: Started Docker Application Conta...
mars 25 16:21:49 localhost.localdomain dockerd[16968]: time="2022-03-25T16:21:49.68...c
Hint: Some lines were ellipsized, use -l to show in full.
[root@localhost yum.repos.d]# docker images
REPOSITORY      TAG              IMAGE ID         CREATED          SIZE
[root@localhost yum.repos.d]#
```

root@localhost:/etc/yum.repos.d | Index of linux/centos/7/x86_64/sta... | 1 / 4

Nous pouvons voir que le processus est en cours et nous pouvons aussi voir les images présentes (il n'y en a pas).

Utilisation de Docker

On installe Debian:stable avec Docker :

```
[root@localhost yum.repos.d]# docker pull debian:stable
stable: Pulling from library/debian
18409aafdb45: Pull complete
```

```
[root@localhost yum.repos.d]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
debian        stable    869b395b26d8   8 days ago    124MB
```

La version stable de Debian à bien été téléchargé.

On démarre maintenant le Docker avec le bash :

```
[root@localhost yum.repos.d]# docker run -t -i debian:stable /bin/bash
root@2565c745bf67:/#
```

On crée les deux utilisateurs sur l'image :

```
asr1:x:1000:1000::/home/asr1:/bin/sh
asr2:x:1001:1001::/home/asr2:/bin/sh
```

On crée ensuite le dossier demandé:

```
root@2565c745bf67:/# mkdir -p /var/www/site/dev
```

Lorsque l'on quitte et revient dans le conteneur, les deux utilisateurs créés précédemment n'existent plus. En effet, ils ne sont pas présents dans l'image de base et ne sont donc pas sauvegardés.

Pour pouvoir la sauvegarder, nous devons la commit en une nouvelle image :

```
[root@localhost yum.repos.d]# docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED        STATUS
PORTS         NAMES
978c3e4adee4   debian:stable  "/bin/bash"             2 minutes ago Exited (0) 24 seconds ago
hopeful_kapitsa
2565c745bf67   debian:stable  "/bin/bash"             7 minutes ago Exited (0) 2 minutes ago
exciting_tu
[root@localhost yum.repos.d]# docker commit -m "Ajout utilisateurs" 978c3e4adee4 jonathan/debian
sha256:17689eabc9fec63c7d15ca2024b0e99da0fa4d0a96ed329c76603847c9796d0e
```

On liste les conteneurs en cours avec docker ps -a et on docker commit.

```
[root@localhost yum.repos.d]# docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
jonathan/debian  latest    17689eabc9fe   57 seconds ago 124MB
debian        stable    869b395b26d8   8 days ago    124MB
```

Docker a donc créé une nouvelle image.

On run la nouvelle image et on regarde les utilisateurs :

```
asr1:x:1000:1000:~/home/asr1:/bin/sh
asr2:x:1001:1001:~/home/asr2:/bin/sh
```

Ils sont bien présent dès le lancement du conteneur.

Création d'un serveur de développement

On vient créer un fichier Dockerfile avec ces commandes :

```
FROM centos:7
MAINTAINER Jonathan Samagaio
RUN yum install -y httpd php wget
RUN mkdir -p /var/www/site1
ADD aliases.conf /etc/httpd/conf.d
```

On vient donc installer les librairies suivantes httpd, php et wget. On vient ensuite créer un dossier et remplacer le fichier de config par un fichier sur notre machine.

On créer ensuite notre image avec la commande suivant :

```
[root@localhost ~]# docker build . -t jonathan/devweb
```

```
[root@localhost ~]# docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
jonathan/devweb	latest	55ced08fc5e	3 minutes ago	415MB
jonathan/debian	latest	17689eabc9fe	14 minutes ago	124MB
debian	stable	869b395b26d8	8 days ago	124MB
centos	7	eeb6ee3f44bd	6 months ago	204MB

Notre image à donc bien été créée.

Nous pouvons maintenant lancer le conteneur :

```
root@localhost ~]# docker run -t -i jonathan/devweb /bin/bash
root@4247319fc96e /]# /usr/sbin/httpd
```

On tente d'exécuter la commande wget :

```
[root@4247319fc96e /]# wget http://localhost/.noindex.html
--2022-03-25 15:52:31-- http://localhost/.noindex.html
Resolving localhost (localhost)... 127.0.0.1, ::1
Connecting to localhost (localhost)|127.0.0.1|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 4897 (4.8K) [text/html]
Saving to: '.noindex.html'

100%[=====>] 4,897 --.-K/s in 0s

2022-03-25 15:52:31 (254 MB/s) - '.noindex.html' saved [4897/4897]

[root@4247319fc96e /]# █
```

Nous venons maintenant modifier notre Dockerfile pour ajouter le lancement de httpd au démarrage :

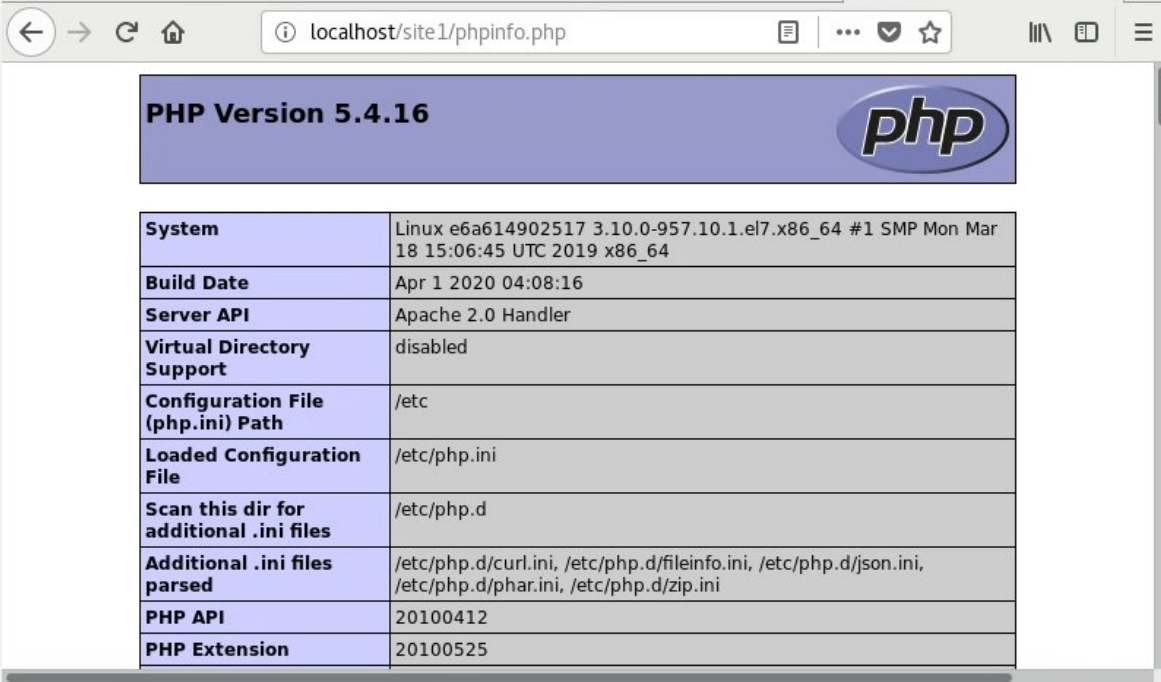
```
FROM centos:7
MAINTAINER Jonathan Samagaio
RUN yum install -y httpd php wget
RUN mkdir -p /var/www/site1
ADD aliases.conf /etc/httpd/conf.d
CMD ["/usr/sbin/httpd", "-DFOREGROUND"]
```

Nous devons re build notre image et la relancer. Nous allons maintenant la lancer en mode daemon :

```
[root@localhost ~]# docker run -d -p 80:80 -v /site:/var/www/site1 jonathan/devweb
```

Nous pouvons voir le `-p` qui fait une redirection de port et le `-v` qui vient monter un dossier de notre machine sur le conteneur.

Après le lancement, on peut accéder au fichier que nous avons créer dans `/site` de notre machine virtuel :



System	Linux e6a614902517 3.10.0-957.10.1.el7.x86_64 #1 SMP Mon Mar 18 15:06:45 UTC 2019 x86_64
Build Date	Apr 1 2020 04:08:16
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/curl.ini, /etc/php.d/fileinfo.ini, /etc/php.d/json.ini, /etc/php.d/phar.ini, /etc/php.d/zip.ini
PHP API	20100412
PHP Extension	20100525

Nous pouvons voir dans la partie system que nous retrouvons le hash de notre image.

Pour aller plus loin

Après création du compte sur le site, on créer notre repository en public. On se connecte après sur notre compte depuis la machine virtuel :


```
[root@localhost ~]# docker login
Login with your Docker ID to push and pull images from Docker Hub. If you don't have a
Docker ID, head over to https://hub.docker.com to create one.
Username: geikenpacku
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```


On renomme notre image :

```
[root@localhost ~]# docker tag jonathan/devweb jonathan/devweb:1.0
```

Et on push :

```
[root@localhost ~]# docker tag jonathan/devweb geikenpacku/devweb:1.0
[root@localhost ~]# docker push geikenpacku/devweb:1.0
The push refers to repository [docker.io/geikenpacku/devweb]
b6e29193d589: Pushed
fa9892d6a3a3: Pushed
4549b87d501d: Pushed
174f56854903: Pushed
1.0: digest: sha256:bb281ea05d0578643ca41da418f009d5e6a54f500c3c03138336f778285db299 si
ze: 1155
```

Nous retrouvons notre image sur le repo :

TAG	OS	PULLED	PUSHED
1.0		---	3 minutes ago

Nous pouvons maintenant installer des les images que l'on veut :

```
Status: Downloaded newer image for mysql:latest
docker.io/library/mysql:latest
[root@localhost ~]# docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
geikenpacku/devweb  1.0          73d6cd32eb67     24 minutes ago   415MB
jonathan/devweb     1.0          73d6cd32eb67     24 minutes ago   415MB
jonathan/devweb     latest       73d6cd32eb67     24 minutes ago   415MB
<none>              <none>       1df8dc0b2565     31 minutes ago   415MB
jonathan/debian     latest       17689eabc9fe     53 minutes ago   124MB
mysql               latest       562c9bc24a08     7 days ago       521MB
debian              stable       869b395b26d8     8 days ago       124MB
centos              7           eeb6ee3f44bd     6 months ago     204MB
[root@localhost ~]# docker run -t -i mysql
2022-03-25 16:29:10+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.2
8-ldeb1an10 started.
2022-03-25 16:29:10+00:00 [Note] [Entrypoint]: Switching to dedicated user 'mysql'
2022-03-25 16:29:10+00:00 [Note] [Entrypoint]: Entrypoint script for MySQL Server 8.0.2
8-ldeb1an10 started.
2022-03-25 16:29:10+00:00 [ERROR] [Entrypoint]: Database is uninitialized and password
option is not specified
You need to specify one of the following:
- MYSQL_ROOT_PASSWORD
- MYSQL_ALLOW_EMPTY_PASSWORD
- MYSQL_RANDOM_ROOT_PASSWORD
[root@localhost ~]# docker run -t -i mysql /bin/bash
root@df32961a0762:/#
```