M2102 - TP 6: Containers Docker

V0.1 – Guillaume Urvoy-Keller / V1.0 – Amaury Boller

March 27, 2025

1 Configuration hôte Docker

1. Créez une machine virtuelle Debian 64 bits avec le script createvm et démarrez-là au travers de l'interface Virtualbox.

2. Connectez-vous sur la VM en **root**. Pour installer docker, il faut ajouter des dépôts spécifiques de paquets pour Debian. Pour cela, exécutez les commandes suivantes:

```
# Mise à jour de la liste des paquets disponible
apt-get update
# Installation du paquet ca-certificates et curl
apt-get install ca-certificates curl
# Creation d'un repertoire avec des droits spécifiques
install -m 0755 -d /etc/apt/keyrings
# Ajout des clés GPG de docker au système
curl -fsSL https://download.docker.com/linux/debian/gpg -o /etc/apt/keyrings/docker.asc
#Affectation des droits de lectures pour tous sur le fichier
chmod a+r /etc/apt/keyrings/docker.asc

# Ajout des dépots docker:
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] \
https://download.docker.com/linux/debian \
$(. /etc/os-release && echo "$VERSION_CODENAME") stable"  \
> /etc/apt/sources.list.d/docker.list
```

   Lisez bien les commentaires avant chaque commande, certaines pourraient vous être nécessaires pour la suite ...
   Note: La procédure peut varier, si vous avez des erreurs, vérifiez que les commandes ci-dessus sont à jour sur le site officiel de docker: `https://docs.docker.com/engine/installation/linux/debian/`

3. Vérifiez que le fichier /etc/apt/sources.list.d/docker.list contient bien le bon dépôt pour **votre** distribution debian.

4. Maintenant que vous avez ajouté un nouveau dépôt, il faut à nouveau mettre a jour la liste des paquets disponibles.
   Quelle est la commande ? Exécutez-là.

5. Installer le paquet lsb-release qui vous sera nécessaire pour la suite
   Quelle commande avez-vous utilisée ? *Note: Une fois installé, la commande qui correspond au paquet lsb-release est lsb_release.*
   Cette commande permet de connaître simplement la version de votre distribution linux. Lorsque vous en aurez besoin dans la suite, il faudra l'installer de la même manière dans votre container docker

6. On peut maintenant installer docker, pour cela installez les paquets suivants:
   docker-ce docker-ce-cli containerd.io docker-buildx-plugin
   Quelle commande avez vous utilisée ?

7. On va maintenant vérifier que Docker est bien installé en lançant notre premier container **hello-world** avec la commande:

   ```
   docker run hello-world
   ```

   Le hello-world est généralement la première chose que l'on apprend pour tout nouveau langage de programmation. Les mainteneurs de docker ont créé un container équivalent à cela Si tout a bien été configuré, vous devriez avoir quelque chose comme cela:

   ```
   sudo docker run hello-world
   Unable to find image 'hello-world:latest' locally
   latest: Pulling from library/hello-world
   b901d36b6f2f: Pull complete
   0a6ba66e537a: Pull complete
   Digest: sha256:....
   Status: Downloaded newer image for hello-world:latest

   Hello from Docker.
   This message shows that your installation appears to be working correctly.
   [...]
   ```

   Les premières lignes avec les mots *Unable to find image..., Pulling..., ...Pull...* signifient que l'image hello-world n'a pas été trouvée dans votre VM, docker va donc aller récupérer l'image demandée directement depuis internet, de la même manière que lorsque vous installez un nouveau paquet avec la commande apt.
   L'image est ensuite exécutée automatiquement et le résultat est affiché à partir de la ligne *Hello from Docker* jusqu'à là fin

2 Premiers containers

1. Nous allons maintenant utiliser un container plus intéressant que hello-world : le container officiel ubuntu. Ce container est **officiel** car il est distribué et maintenu par l'équipe officiel de la distribution linux Ubuntu.

   (a) Naviguez sur le hub docker `https://hub.docker.com`, en utilisant son moteur de recherche trouvez les images officielles d'ubuntu, debian, mysql et httpd (apache). Sur la page de chaque image, la commande permettant de l'ajouter à votre système est affichée en haut à droite, il s'agit de la commande:
   docker pull IMAGE_NAME
   Comment savoir si une image est officielle ?
   Vous pouvez filtrer les résultat pour n'afficher que les images officielles en utilisant le menu de gauche dans la section *Trusted content* .

   (b) Dans votre VM, ajoutez l'image de la dernière version ubuntu
   Quelle est la commande ? Exécutez-là.

   (c) Vérifiez que l'image a correctement été téléchargée et ajoutez à docker en tapant **docker image list** ou simplement **docker images**
   Quand l'image a-t- elle été créée et quelle est la taille qu'elle occupe sur votre VM ?

2. Démarrons notre premier containers ubuntu. **docker run ubuntu**. Que se passe-t-il?

3. Un container ne reste en vie que si un processus est actif. On peut lister les containers actifs avec la commande **docker ps**.
   On peut aussi lister tous les containers, actifs ou inactifs avec **docker ps -a**.
   Comme vous n'avez pas donné de nom à votre container, docker lui en génère un aléatoire, généralement en 2 mots séparés par un underscore.
   Que vous retourne la commande **docker ps -a** et pourquoi?

4. Nous allons maintenant relancer cette image, en lui donnant notre propre nom avec l'option --name, mais cette fois ci en ouvrant un terminal dans le container. Pour cela il faut rediriger l'entrée standard du container avec l'option -i et ouvrir un pseudo-terminal avec -t, le tout en exécutant le processus /bin/bash :
   docker run -ti --name=ubuntu ubuntu /bin/bash
   Quelle est la version d'ubuntu du container (utilisez la commande **lsb_release avec l'option -a**)?

5. Ouvrez un second terminal. Listez les containers actifs, combien y en a-t-il?

6. Le principe d'un container est qu'il ressemble à une VM vu de l'intérieur, mais qu'il est un ensemble de processus vu de l'extérieur. L'extérieur ici, c'est la machine hôte Debian. Placez vous dans votre second terminal (celui de l'hôte).

   (a) Listez les processus avec top

   (b) Ajoutez le namespace des processus afin de voir le namespace de votre container en tapant **f** dans le top puis en allant sélectionner le champ **nsPID**. Une fois ce champ sélectionné avec les flèches et la barre d'espace, retournez à l'affichage général en tapant q. Si besoin, agrandissez la fenêtre de votre terminal pour qu'il y ait suffisamment de place pour que la colonne nsPID puisse apparaître à l'écran.

   (c) top liste les processus par défaut par leur activité CPU. On va donc "faire du bruit" dans le container avec la commande **yes**.

   (d) Alternativement, on peut classer les processus suivant le critère désiré. Nous allons les classer de façon décroissante en tapant à nouveau **f** puis en se plaçant sur la colonne nsPID et en tapant *s*.

   Quel est le namespace id de votre container? Vérifiez que vous voyez tous les processus de votre container en listant également dans le container avec la commande top.

7. De la même manière, un container obtient des interfaces réseaux qui sont privées, et séparées de la machine hôte.

   (a) En Listant les interfaces réseaux de votre container, trouvez quelle est l'adresse IP de votre container.

   (b) Docker assigne une adresse privée à chaque container et lui donne accès au reste d'internet au travers d'une passerelle qui est l'interface virtuelle docker de l'hôte. Pour bien le comprendre

      i. Trouvez la passerelle par défaut de votre container avec la commande **ip route** ou **netstat -r**.
      pour utiliser les commandes ifconfig, netstat... il faut installer le paquet net-tools

      ii. L'ip de la passerelle par défaut du conteneur doit correspondre à une IP de la machine hôte debian, quel est son nom?.

8. Nous allons maintenant arrêter notre container.

   (a) Pour rappel, lorsqu'il n'y a plus de processus, le container se termine. Il suffit donc de fermer le bash en utilisant la commande **exit** dans le container. Vous pouvez également faire un **docker stop ubuntu** depuis l'hôte debian.
   Listez les container docker (sans oublier l'option -a), dans quel état se trouve votre container ?

(b) Vous pouvez le redémarrer en tapant : **docker start ubuntu**. Vérifiez à nouveau son état, quel est son statut ?

(c) Le container est démarré mais cette fois ci, vous n'avez plus la main dessus. Pour récupérer un terminal dans votre container, lancez la commande **docker attach ubuntu** (parfois il faut appuyer sur une touche du clavier pour avoir le prompt). Testez avec votre container puis arrêtez-le à nouveau.

(d) Relancer la commande
docker run -ti --name=ubuntu ubuntu /bin/bash
Quelle erreur se produit et pourquoi ?

9. Pour pouvoir lancer un nouveau container avec le même nom, il faut tuer celui dont on a plus besoin. Cela se fait avec la commande **docker rm ubuntu**. Cela n'est possible que si le container est arrêté!
Listez les processus docker avec l'option -a, que voyez vous ?

3 Gestion des containers

1. Nous allons redémarrer un autre container basé sur l'image ubuntu avec un nom différent pour changer :

   docker run -ti --name=bob ubuntu /bin/bash

2. On peut inspecter ce qui se passe dans le container depuis la machine hôte avec la commande **docker logs bob** ou mieux encore **docker logs -f bob** qui est (un peu) l'équivalent du **tail -f /var/log/syslog** que l'on ferait sur la machine hôte.
On peut d'ailleurs exécuter cette commande pour voir le démon docker qui reporte les manipulations que vous effectuées (démarrage, arrêt de containers, etc.). Faites le test en faisant un **ps** dans le container et en ayant le logs ouvert en continue depuis l'hôte. Puis faites un **top** dans le container.

3. On peut également ajouter le timestamp pour chaque ligne de log avec l'option t : **docker logs -ft bob**

4. Docker fournit aussi des statistiques sur l'usage des ressources CPU/mémoire/réseau des containers: **docker stats bob**. Pour que cela soit intéressant, il faut que le container soit actif. Lancez la commande **yes** et vérifiez les statistiques.

4 Gestion des images

1. Commençons par lister les images disponibles. Quelle est la commande ?

2. Pour ajouter une image localement sans pour autant la lancer directement, il suffit de faire un simple **docker pull IMAGE_NAME**. Allez sur le hub docker et cherchez une image du serveur web Apache officielle (aussi connu sous le nom **httpd**) puis téléchargez-la. Quelle est la commande que vous avez utilisée ?

3. Nous allons maintenant créer notre propre image qui va nous permettre de lancer un serveur Web.
Pour cela nous allons créer un fichier Dockerfile qui contiendra toutes les étapes de construction de votre image.

   (a) Créez un répertoire

   `~/Docker/Apache`

   (b) Dans ce répertoire, créez un fichier Dockerfile. Dans ce fichier, mettez les commandes :

```
FROM ubuntu:latest
   RUN apt-get update && apt-get install -y tzdata && apt-get install -y apache2

   WORKDIR /var/www/html

   ENV APACHE_RUN_USER=www-data
   ENV APACHE_RUN_GROUP=www-data
   ENV APACHE_LOG_DIR=/var/log/apache2
   ENV APACHE_PID_FILE=/var/run/apache2.pid
   ENV APACHE_RUN_DIR=/var/run/apache2
   ENV APACHE_LOCK_DIR=/var/lock/apache2

   RUN mkdir -p $APACHE_RUN_DIR $APACHE_LOCK_DIR $APACHE_LOG_DIR

   ENTRYPOINT [ "/usr/sbin/apache2" ]
   CMD ["-D", "FOREGROUND"]
   EXPOSE 80
```

(c) Créons l'image **docker build -t="votre_nom/apache" .** Attention à ne pas oublier le point '.' à la fin de la commande (qui indique que le fichier Dockerfile est dans le répertoire local. Pour éviter tout conflit avec une image officielle, nommez votre image sous la forme votre_nom/apache et non apache uniquement.

(d) Listez les images, que voyez vous ?

5 Faire tourner des applications avec Docker

1. Jusqu'à présent, nous avons fait tourner des containers ubuntu en mode interactif. C'est intéressant, mais dans un contexte de production où un opérateur veut démarrer beaucoup de containers sur une machine, on va préférer les démarrer en mode démon, c'est-à-dire en tâche de fond. Cela se fait simplement avec l'option -d.

   (a) Commençons par un simple **docker run -d --name=demon ubuntu /bin/bash**. Que se passe-t-il? Listez les processus docker (sans oublier l'option -a) pour comprendre.

   (b) Détruisez le container précédent. Nous allons corriger le problème précédent en faisant quelque chose dans le container :
   docker run -d --name=demon ubuntu /bin/sh -c "while true; do echo hello world; sleep 1; done"

   (c) Montrez ce qui se passe dans le container depuis la machine hôte puis détruisez le container. Quelle commande avez vous utilisée ?

2. Nous allons maintenant consulter le serveur web du container que vous avez créé directement depuis la machine hote. Les dernières versions d'ubuntu embarquent déjà un serveur apache, vous aurez donc un conflit sur le port 80 dans la suite.
   Vérifiez avec la commande *ps -ef | grep -E 'http|apache'* et stoppez le si nécessaire.

3. Démarrez un container votre_nom/apache en mode démon en exposant le port 80 : **docker run -d -p 80 --name=apache votre_nom/apache**

4. L'option -p vous permet d'exposer le port interne du container (le port 80 du serveur apache) depuis le port 80 de l'hôte debian.
   Pour trouver quel est le port choisi par Docker, il suffit de taper **docker port apache 80**. Faites un test en ouvrant votre navigateur et vous connectant sur la machine locale (l'hôte debian) sur le port docker.

5. On peut mieux contrôler le port choisi. Si par exemple on veut que cela soit le port 80, il suffit de modifier la commande précédente :

```
docker run -d -p 80:80 --name=apache votre_nom/apache
```

le premier 80 correspond au port mappé sur l'hote, tandis que celui après les ':' correspond au port à écouter dans votre container Stoppez, détruisez et recréez votre container pour vérifier que cela fonctionne.

6. On veut maintenant en plus contrôler le contenu du serveur Web depuis l'hôte debian. Pour cela on va :

   (a) créer un répertoire **website** dans le répertoire Apache créé précédemment. Quelle est la commande pour créer ce répertoire ?

   (b) mettre dans ce répertoire un fichier index.html avec le contenu suivant :

   ```
   <html>
   Hello world
   </html>
   ```

   Quelle commande avez vous utilisée pour créer le fichier ?

   (c) Démarrer votre container en montant le répertoire apache là où le serveur Apache dans le container va chercher ses données :

   ```
   docker  run -d -p 80:80  -v ~/Docker/Apache/website:/var/www/html
   --name=apache votre_nom/apache
   ```

   Depuis un navigateur ouvert sur la VM, ouvrez l'ip de votre VM (http://XX.XX.XX.XX:80). A quoi correspond l'option -v de la commande précédente ?

6 Pour aller plus loin (optionnel)

1. Le projet Kasm permet d'utiliser des applications directement depuis votre navigateur. Il est par exemple possible d'utiliser un terminal, un éditeur de code Visual Studio Code ou même le lecteur video VLC directement depuis votre firefox. Il est même possible de lancer un firefox dans votre firefox ...
   Lancer la commande suivante:f

   ```
   docker  run -d -p 443:6901 -e VNC_PW=tititoto --name=firefox kasmweb/firefox:1.16.1
   ```

   Puis ouvrez une page web en https, sur l'ip de l'hote. Connectez-vous avec les identifiants suivant:
   login: kasm_user
   password: tititoto
   Que voyez vous ?

2. Il existe des milliers d'images docker sur dockerhub qui est l'un des principaux dépôts d'images docker mais il en existe d'autres tel que https://quay.io/search ou encore https://gallery.ecr.aws/. Vous pouvez également créer vos propres registry docker.
   Depuis la registry d'amazon (aws), chercher l'image d'ubuntu et récupérez-là.
   Quelle commande avez vous utilisée ?