Virtualisation des systèmes

Florent Glück

April 07, 2025

Docker basics

Introduction

Le but de ce travail pratique est de vous familiariser avec les manipulations de base des containers Docker.

Préparation

Nous désirons **découpler** le client Docker du daemon `dockerd`, c'est à dire que chacun doit s'exécuter sur une **machine différente**.

Vous allez donc utiliser votre machine distante comme serveur Docker. C'est donc sur cette machine que s'exécutera le daemon `dockerd`.

Assurez-vous donc que vous avez bien créé, grâce à LVM, un espace de stockage unifié de tous les disques disponibles pour atteindre la capacité maximale disponible sur votre rootfs (60GB au total).

Procédez ensuite à l'installation de Docker. Pour rappel `apt-cache search` ou `apt search` permet de réaliser une recherche par mot-clés dans les repositories Ubuntu/Debian. Faites attention à installer le bon package (**lisez** les descriptions des paquets !).

Pour utiliser Docker, votre utilisateur doit impérativement faire partie du groupe `docker`[1]. **N'exécutez surtout pas Docker en tant que root/sudo !** Pour des raisons de sécurité, utilisez `sudo` **uniquement** lorsque réellement nécessaire et jamais autrement.

Côté serveur, vérifiez que le service `dockerd` est bien actif et assurez-vous que celui-ci soit toujours démarré au boot du système.

Sur le client (votre laptop personnel), configurez votre système afin que vous puissiez correctement exécuter le client Docker et créez un contexte utilisant votre serveur Docker distant.

Pour rappel, n'exécutez **jamais** le client Docker avec `sudo` ou en tant que `root`. Ce n'est pas nécessaire et c'est un trou de sécurité potentielle.

Configurez votre client Docker afin que le contexte courant utilise votre serveur distant. Enfin, vérifiez que tout fonctionne correctement.

[1] Voir https://linuxize.com/post/how-to-add-user-to-group-in-linux/ pour ajouter un utilisateur à un groupe.

Exercice 1

Une série d'exercices se trouvent sur Le Docker hub : https://hub.docker.com/. Commencez par les trouver à l'aide de la commande `docker search` (indice : hepia...) et localisez l'image correspondant au premier exercice (un vrai travail de Sherlock Holmes c'est sûre !).

Basé sur cette image et sans exécuter de shell interactif dans le container, affichez le contenu du fichier se trouvant à la racine du container de l'exercice 1. Procédez en deux étapes :

1) Listez les fichiers se trouvant à la racine
2) Affichez le contenu du fichier

A noter qu'au lancement d'un container vous pouvez spécifier une commande à executer (p.ex `cat` ou `ls -l`).

- Suite aux deux opérations ci-dessus, combien de containers ont été créés et quels sont leurs noms et IDs ?
- Quel est l'état des containers ?
- Quel est l'ID de l'image utilisée par les containers que vous venez d'exécuter ?
- Est-ce que l'ID de l'image sera identique chez vos collègues de classe ?
- Est-ce que les noms et IDs des containers seront identiques chez vos collègues ?
- Sur quelle machine se trouvent les images Docker ?

Exercice 2

Exécutez un container sur l'image `hepia/docker_ex03` avec un shell (`sh`) en mode interactif. Créez ensuite le fichier `fantasio` dans le répertoire `/ex03/`.

- Que se passe-t-il quand vous quittez le shell avec `ctrl+d` ?

Exécutez un nouveau container basé sur la même image que précédemment et nommez-le `zorglub`.

- Est-ce que le fichier `/ex03/fantasio` existe ou pas ? Pourquoi ?

Dans le container `zorglub`, exécutez à nouveau `touch /ex03/fantasio`, puis pressez la combinaison de touches `ctrl+p ctrl+q`.

- Quel est l'état (stoppé, en exécution, etc.) du container `zorglub` ?
- Quel est donc le rôle de la combinaison de touches `ctrl+p ctrl+q` ?

Attachez-vous (`docker attach`) alors au container `zorglub`.

- Est-ce que le fichier `/ex03/fantasio` existe encore ?

Dans un nouveau terminal, attachez-vous à nouveau au container `zorglub`. Placez côte à côte les deux terminaux dans lesquels vous êtes attaché au container `zorglub` et tapez des caractères au clavier.

- Qu'observez-vous et que pouvez-vous donc en déduire ?

Exécutez la commande `docker exec -it zorglub sh`.

- Est-ce que le fichier `/ex03/fantasio` existe toujours ? Combien de shells sont en cours d'execution (aide: `ps`) ?
- Quelle est la différence entre les commandes `attach` et `exec` ?

Terminez le shell (p.ex. avec `ctrl+d`) dans lequel vous aviez exécuté `docker exec ...` auparavant.

- Est-ce que le container `zorglub` est toujours en exécution ?

Exécutez un nouveau shell dans le container avec la commande `docker exec -it zorglub sh`, puis allez dans le terminal où vous aviez réalisé le `docker attach` plus haut. Terminez alors ce shell avec `ctrl+d`.

- Est-ce que le container `zorglub` est toujours en exécution ? Que pouvez-vous en conclure ?

Démarrez le container `zorglub` avec la commande `start -ai` (attach + interactive).

- Le fichier `/ex03/fantasio` existe-t-il toujours ?

Vous pouvez maintenant terminer et supprimer le container `zorglub` et également supprimer le container créé en début d'exercice.

Exercice 3

Exécutez un container basé sur l'image Alpine 3.17 en spécifiant de le détruire une fois terminé. Vérifiez que vous utilisez la bonne distribution en inspectant le contenu du fichier `/etc/os-release`.

Dans ce container, exécutez la commande `top`. Inspectez ensuite le processus `top` à l'intérieur du container et sur la machine hôte (via `ps -ef`).

- Selon votre observation, est-ce qu'un processus appartenant à root dans le container appartient également à root en dehors du container ?

Pour rappel, root possède le user ID (UID) 0.

- Pourquoi est-il dangereux qu'un processus UID 0 dans le container soit également UID 0 sur la machine hôte ?
- Quelle différence remarquez vous lorsque vous exécutez la commande `ps -ef` dans le container et sur la machine hôte ?
- A votre avis, quelle est la raison de cette différence ?

À l'intérieur du container, terminez `top` (touche `q`), puis installez et exécutez le programme `asciiquarium`. Pour info, Alpine utilise le package manager `apk`. Les équivalents Alpine de `apt-get update` et `apt-get install` sont `apk update` et `apk add`. Vous pouvez lister tous les paquets disponibles avec `apk list`.

- Quel est le PID de ce processus dans le container ? Quel est le PID de ce même processus sur la machine hôte et pourquoi est-il différent à votre avis ?

Vous allez maintenant investiguer empiriquement le comportement des containers.

- Quel est le processus portant le numéro de PID 1 dans le container ? Quel est le processus portant le même numéro de PID sur la machine hôte ?

- Pour généraliser, quel est le processus portant le PID 1 dans le container (càd quel que soit le container exécuté) ? Vous pouvez le déduire empiriquement en exécutant d'autres containers avec des programmes différents.

Comparer le résultat de la commande `uname -rv` dans le container et sur la machine hôte.

- Que remarquez-vous et que pouvez-vous en conclure ?

Similairement à ci-dessus, comparez le contenu du répertoire `/dev/`.

- Que remarquez-vous ? A votre avis quelle est la raison de cette différence ?
- Est-il possible de donner à un container accès à un ou plusieurs périphériques de la machine hôte ? Comment ? Trouver un moyen permettant de vérifier votre méthode.

Exécutez la commande `mount` sur la machine hôte et dans le container.

- Quelles différences remarquez vous, notamment en ce qui concerne le système de fichiers racine (`rootfs`) ?
- Retrouvez-vous l'entrée définissant le système de fichiers racine (`rootfs`) du container sur la machine hôte ? Que pouvez-vous donc conclure vis-à-vis du rootfs du container ?