POLYTECHNIQUE MONTRÉAL

INF8480 - Systèmes répartis et infonuagique

TP1 - Découverte des technologies de l'Infonuagique

Chargés de laboratoire :
Sébastien Darche
Sara Beddouch
Redacteur :
Pierre-Frederick Denys

Hiver 2024 - V7.0

1 Introduction

1.1 Prérequis

— **Introduction aux système répartis :** Historique, concepts de base, modèles et caractéristiques des systèmes.
— **Infonuagique :** clients et serveurs de l'infonuagique, services pour les machines virtuelles et les containers.

1.2 But du TP

Dans ce TP, vous allez vous entraîner à installer une machine virtuelle linux, découvrir les containers Docker et apprendre à les manipuler.

1.3 Choix de l'environnement

3 possibilités pour réaliser le TP :

1. Choix 1 (préféré) : **sur votre propre ordinateur portable ou fixe** réaliser le TP selon les instructions.
2. Choix 2 : **sur les postes du lab L4712 en présentiel** Virtualbox est installé sur les PC, passez à la section 2.2, n'ignorez surtout pas les parties encadrées en rouge.
3. Choix 3 : **sur les postes du lab L4712 à distance** uniquement si vous ne pouvez pas venir à polytechnique, et si votre propre ordinateur ne dispose pas d'au moins 8gb de RAM ou qu'il est trop lent. La procédure d'accès à distance est `https://tinyurl.com/yeysk2wn`

2 Partie 1 : Installation d'une machine virtuelle et d'une distribution linux

L'installation de cette machine virtuelle va vous permettre de pratiquer l'installation d'une distribution linux dans une machine virtuelle Virtualbox.

2.1 Installation de Virtualbox

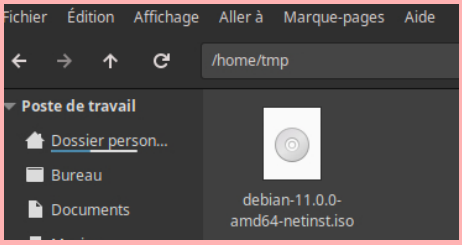
Téléchargez et installer Virtualbox sur votre PC selon votre OS : https ://www.virtualbox.org/.

Info
Si vous n'avez jamais utilisé VMware ou virtualbox, il se peut que vous ayez à activer la virtualisation dans le BIOS de votre PC.

2.2 Installation de la machine virtuelle

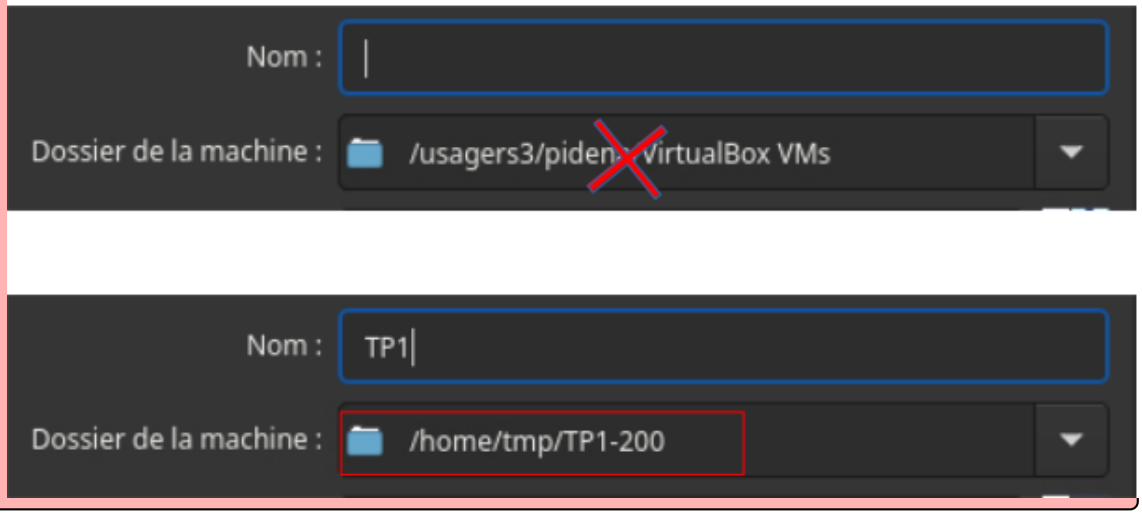
1. Téléchargez l'image ISO du système d'exploitation : `https://cdimage.debian.org/cdimage/archive/11.0.0/amd64/iso-cd/debian-11.0.0-amd64-netinst.iso`

> **Sur les PC du L4712**
> Télécharger l'ISO et déplacez le dans le dossier `/home/tmp/`



2. Créer une nouvelle machine virtuelle (`Machine > Nouvelle`) nommez la : `TP1 INF8480`
   — Choisissiez le type `linux` et la version `Debian (64-bit)`

> **Sur les PC du L4712**
> Créer un dossier `TP-XXX` avec XXX votre numéro de groupe et choisir ce dossier pour créer la VM. Si la VM est dans le dossier par défaut, l'espace est insuffisant et la VM ne se déploiera pas.

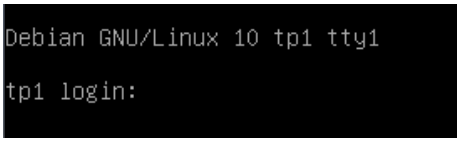


   — Choisir la quantité de mémoire selon votre PC, (min 4Go)
   — Créer un nouveau disque virtuel avec les options par défaut et de taille limite 100 Go

3. Lancez la VM et choisir le disque de démarrage `debian-11.0.0-amd64-netinst.iso` à partir de l'ISO téléchargée.
   — Une fois l'ISO lancé, choisir `Install` ET PAS `Graphical install`

— Nommez la `tp1`
— Pas de nom de domaine
— Mot de passe root : `inf8480`
— Nouvel utilisateur `inf8480` avec mdp `inf8480`
— Disque entier avec tout dans la même partition
— **Bien lire les questions posées par l'installateur.**
— **IMPORTANT :** Lors du choix de logiciels, cocher uniquement les deux derniers et décocher les autres (on ne veut PAS d'environnement graphique)

4. Si tout est OK, vous devriez avoir :



5. A partir de ce moment, **vous n'avez plus besoin** de l'interface de la VM, laissez la en arrière plan.
6. Configurer la redirection de port Virtualbox (on souhaite que le port 2223 de notre PC (hôte) soit redirigé sur le port 22 de la VM (invité).

2.3 Connexion et configuration de la VM

1. Encore une fois, laissez tomber l'interface virtualbox de la VM, et lancez une autre console (powershell, terminator...)
2. Lancer la commande :

```
[MONPC]$ ssh  -p 2223 inf8480@localhost
```

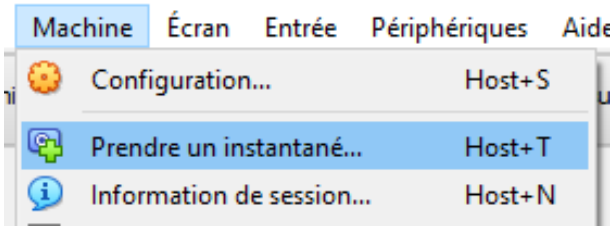
Vous devriez alors avoir un prompt dans la VM :

```
C:\Users\PIERRE>ssh -p2223 inf8480@localhost
The authenticity of host '[localhost]:2223 ([127.0.0.1]:2223)' can't be established.
ECDSA key fingerprint is SHA256:RmYpO/G6rKeB3W6xtR0wx/jHCoxkqJPpbBr+cjIOYas.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[localhost]:2223' (ECDSA) to the list of known hosts.
inf8480@localhost's password:
Linux tp1 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2 (2020-11-28) x86_64Linux tp1 4.19.0-13-amd64 #1 SMP Debian 4.19.160-2
 (2020-11-28) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
inf8480@tp1:~$
```

3. Passer en root
4. Mettre à jour le système et la liste des paquets
5. Ajouter l'utilisateur `inf8480` au groupe sudo

```
[VM]$ apt-get install sudo -y
[VM]$ usermod -aG sudo inf8480
```

6. Taper `exit` 2 fois pour vous déconnecter de la VM puis relancer la commande SSH. Vous devriez être capable de faire `sudo` avec l'utilisateur `inf8480`
7. Effectuer un premier snapshot de la VM pour y revenir en cas de soucis.



La VM est maintenant prête !

3 Prise en main de Docker

Docker est une plate-forme permettant de créer et administrer des applications dans des environnements isolés et indépendant de l'infrastructure appelés containers. Les containers partagent donc des similarités avec les machines virtuelles mais aussi quelques différences :

Les containers partagent le noyau du système d'exploitation de l'hôte contrairement aux VM. L'isolation des processus est donc au niveau du système d'exploitation pour les containers alors qu'elle est au niveau matériel pour les VM. Ainsi, Un container fait en général 150Mo alors qu'une VM fait plusieurs Go et la création des container ne prends que quelques secondes contrairement à quelques minutes pour une VM. Docker est utilisé pour :

— Partager le code d'applications avec la configuration et les dépendances (vous réalisez une application python, vous partagez l'image Docker, et vous êtes sur que l'utilisateur pourra utiliser l'application avec une simple commande peu importe son matériel ou son système d'exploitation)

— Déployer des applications en faisant abstraction du matériel et en séparant les fonctionnalités. (Paypal orchestre plus de 200 000 containers dans son architecture)
— Déployer des applications et répondre aux variations de charge (lors du black Friday, un site web de vente en ligne déploie 1500 containers au lieu de 110 habituellement).

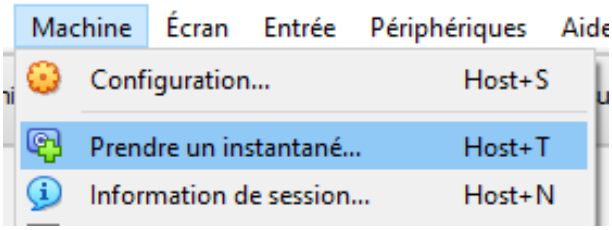
En savoir plus : `https://docs.docker.com/get-started/overview/`.

3.1 Installer Docker

1. Installer docker à partir de la documentation officielle : `https://docs.docker.com/engine/install/debian/`. Utiliser seulement la section "Install using the repository".
2. Si tout est OK, vous devriez pouvoir lancer un premier container :

```
inf8480@tp1:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
0e03bdcc26d7: Pull complete
Digest: sha256:31b9c7d48790f0d8c50ab433d9c3b7e17666d6993084c002c2ff1ca09b96391d
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
```

3. **Effectuer un second snapshot de la VM**



3.2 Administration de base

Attention

Vous devez impérativement à partir de ce point respecter les noms des éléments (containers, numéros de ports) pour le bon fonctionnement du script de correction automatique.

La commande `docker ps` est celle que vous allez le plus utiliser. Elle vous retourne la liste des containers en fonctionnement, et l'option `-a` permet de lister tous les containers qui sont en marche ou stoppés.

```
[VM]$ sudo docker ps -a
```

La commande `docker run` vous permet de démarrer un container à partir d'une image.

1. **Vous allez lancer un serveur web Apache qui réponds sur le port 80 :**

```
[VM]$ sudo docker run -dit --name webservertp1  -p 8086:80 httpd
```

2. Vérifiez son statut avec `docker ps`
3. Un redirection de port a été mise en place par Docker, du port 80 de l'intérieur du container vers le port 8086 de la VM. Exposer le port 8086 de la VM dans Virtualbox.
4. Avec le navigateur, allez sur : `http://localhost:8086/`. Vous devriez voir "It works!"
5. Connectez vous maintenant dans le container (on lance un processus bash dans le container et on le pipe dans la console de la VM) :

```
[VM]$ sudo docker exec -it webservertp1 /bin/bash
```

6. Installer vim ou nano avec apt-get pour avoir un éditeur de fichier
7. Editer le fichier `htdocs/index.html` et remplacer le « It works » par un autre texte.
8. Actualisez totalement la page web avec `ctrl+F5` ou réouvrez un onglet `http://localhost:8086/`

Maintenant que le serveur web est en place, vous allez transférer une page web de votre PC vers le container, en passant par la VM

3.3 Transfert de fichiers

Transférer des fichiers sur le container de votre PC à la VM et de la VM au container.

1. Sur **votre PC**, créer un fichier `page2.html` avec comme contenu :

```
<html>
    <body>
        <p>Voici la page 2 du site web INF8480</p>
    </body>
</html>
```

2. Ouvrez une nouvelle console SUR VOTRE PC, PAS SUR LA VM NI DANS LE CONTAINER dans le dossier où se trouve le fichier, et utiliser scp pour transférer le fichier de votre PC à la VM :

```
[MON_PC]$ scp -P 2223 page2.html inf8480@localhost:.
```

3. Utiliser ensuite `docker cp` pour copier le fichier de la VM vers le dossier web du serveur apache dans le container : COMMANDE A EXECUTER DANS LA VM

```
[VM]$ sudo docker cp page2.html webservertp1:/usr/local/apache2/
   htdocs/
```

4. Accéder avec votre navigateur à `http://localhost:8086/page2.html` vous devriez voir la nouvelle page web transférée!

3.4 Créer des images

Le serveur web contient maintenant 2 pages web personnalisées, et vous souhaitez partager une nouvelle image de ce serveur web avec les pages personnalisées.

1. Créez une image du container avec la commande `commit`. La nomenclature des images est normalisée : `<nomDeLAuteur>/<nomImage>:<version>`.

```
[VM]$ sudo docker commit webservertp1 inf8480/webimg1:v1
```

2. Cette commande permet de « figer » la configuration et les fichiers du container dans une image, et de l'utiliser ensuite pour lancer d'autres containers. Vérifier la création de l'image :

```
[VM]$ sudo docker images
```

3. Comprendre le fonctionnement des images avec `docker history` qui permet de voir les modificiations apportées à l'image de base.

```
[VM]$ sudo docker history inf8480/webimg1:v1
```

```
inf8480@tp1:~$ sudo docker history inf8480/webimg1:v1
IMAGE          CREATED         CREATED BY                                      SIZE      COMMENT
1511c6f9bfe7   5 minutes ago   httpd-foreground                                19MB
683a7aad17d3   35 hours ago    /bin/sh -c #(nop)  CMD ["httpd-foreground"]     0B
<missing>      35 hours ago    /bin/sh -c #(nop)  EXPOSE 80                    0B
<missing>      35 hours ago    /bin/sh -c #(nop) COPY file:c432ff61c4993ecd…   138B
<missing>      35 hours ago    /bin/sh -c #(nop)  STOPSIGNAL SIGWINCH          0B
<missing>      35 hours ago    /bin/sh -c set -eux;   savedAptMark="$(apt-m…   60.9MB
<missing>      35 hours ago    /bin/sh -c #(nop)  ENV HTTPD_PATCHES=           0B
<missing>      35 hours ago    /bin/sh -c #(nop)  ENV HTTPD_SHA256=740eddf6…   0B
<missing>      35 hours ago    /bin/sh -c #(nop)  ENV HTTPD_VERSION=2.4.46     0B
<missing>      35 hours ago    /bin/sh -c set -eux;  apt-get update;  apt-g…   7.38MB
<missing>      35 hours ago    /bin/sh -c #(nop) WORKDIR /usr/local/apache2    0B
<missing>      35 hours ago    /bin/sh -c mkdir -p "$HTTPD_PREFIX"  && chow…   0B
<missing>      35 hours ago    /bin/sh -c #(nop)  ENV PATH=/usr/local/apach…   0B
<missing>      35 hours ago    /bin/sh -c #(nop)  ENV HTTPD_PREFIX=/usr/loc…   0B
<missing>      44 hours ago    /bin/sh -c #(nop)  CMD ["bash"]                 0B
<missing>      44 hours ago    /bin/sh -c #(nop) ADD file:422aca8901ae3d869…   69.2MB
```

L'id 683a7aad17d3 correspond à l'image httpd (serveur Apache), et chaque ligne correspond à un changement de l'image de base (mise à jour des paquets, installation de nano et modificiation de la page web) **Docker ne va pas stocker une copie complète de l'image modifée, mais les changements subséquents à cette image appelés « layers ».**

Le partage d'une nouvelle image est donc très léger.

Vous verrez dans un prochain TP, une méthode pour construire une image à partir d'un script appelé « Dockerfile ».

Maintenant que l'image est créée, la section suivante va l'utiliser pour lancer un nouveau container.

3.5 Utiliser des images

1. Lancer un nouveau container à partir de l'image créée précédemment :

```
[VM]$ sudo docker run -dit --name webservertp2  -p 8087:80
   inf8480/webimg1:v1
```

2. Ouvrir le port 8087 sur la VM
3. Connectez vous à `http://localhost:8087/` et à `http://localhost:8087/page2.html`. Vous devriez voir les pages personnalisées.

Note

Cette méthode permet de partir d'une image de base (serveur web) d'ajouter de la configuration ou des fichiers puis d'en faire une image pour l'utiliser dans d'autres containers.

Il est ensuite possible de partager cette image sur dockerhub avec `docker push`, pour faciliter le déploiement d'une application. Vous installez votre application dans le container, vous déposez l'image sur docker hub, l'utilisateur n'a plus qu'à cloner l'image pour obtenir une instance de votre application.

3.6 Monter un dossier distant

Dans une infrastructure d'entreprise les containers sont lancés en très grand nombre sur plusieurs serveurs pour répondre à la demande. Paypal gère par exemple 200 000 containers. Pour ne pas dupliquer le contenu à servir dans chacun des containers, les fichiers sont stockés sur des serveurs de fichiers, et partagés sur le réseau sous forme de « volume » accessible par tous les containers.

1. Nous allons donc lancer un nouveau container et y attacher un dossier distant qui se trouve sur la VM avec l'option `-v`. Ce dossier distant est monté sur le répertoire public su serveur web.

```
[VM]$ mkdir web
[VM]$ sudo docker run -dit --name webservertp3  -p 8088:80 -v "$
   (pwd)"/web:/usr/local/apache2/htdocs/   inf8480/webimg1:v1
```

2. Ouvrir le port 8088 sur la VM
3. Accéder à `http://localhost:8088/`, vous devriez voir une liste vide qui correspond à la liste des fichiers disponibles sur le dossier distant monté.
4. créer maintenant des fichiers dans le dossier distant sur la VM :

```
[VM]$ touch web/p1.html
[VM]$ touch web/p2.html
```

5. Actualisez la page web, vous devriez alors voir la liste des nouvelles page ajoutées.

Cette manipulation montre qu'il est possible de monter des répertoires distants sur un container, cela peut être un dossier sur le serveur, ou sur le réseau (un stockage bucket Amazon s3...).

Cela permet de modifier facilement les fichiers du container, et de pouvoir partager le stockage entre plusieurs containers.

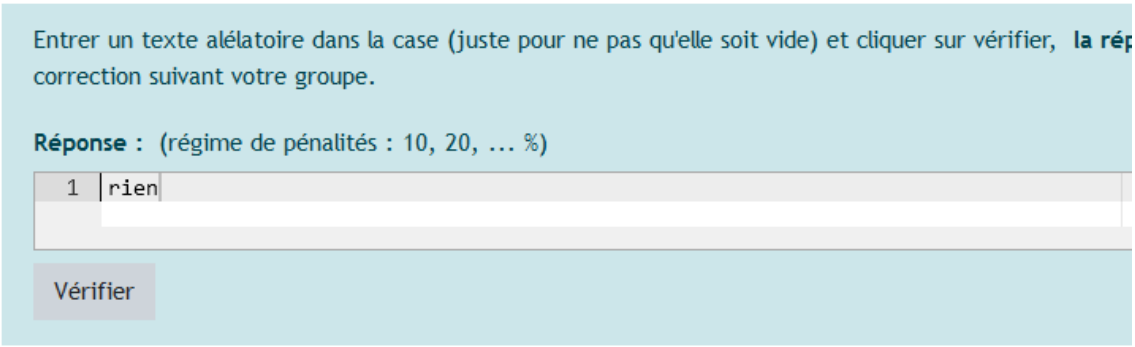
4 Conclusion et remise

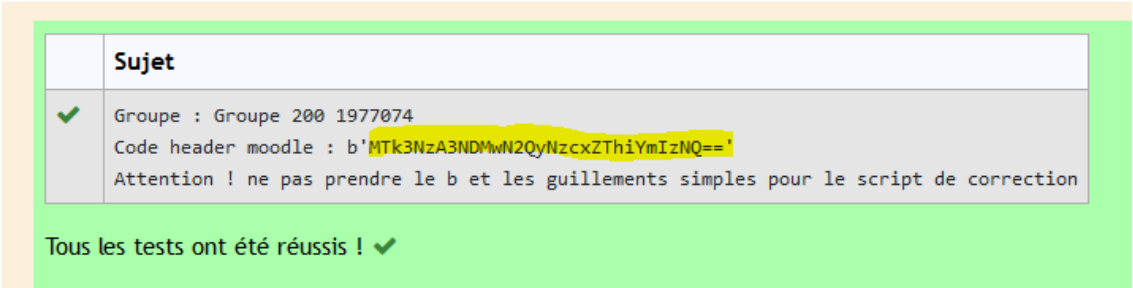
Attention

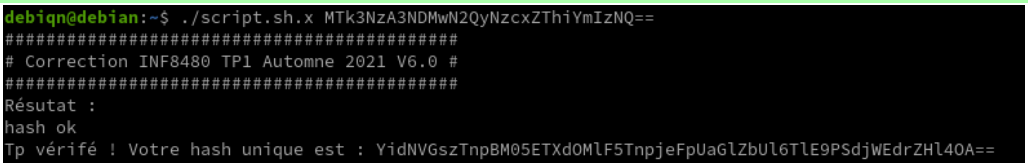
Pour les TP suivants, vous devez avoir compris le fonctionnement des containers docker, et la différence avec une machine virtuelle. Vous devez maitriser les commandes de base pour lancer les containers et manipuler les images. Vous devez être capable de transférer des fichiers de votre PC vers la VM et de la VM vers le container.

Remise :

1. Utilisez la commande `scp` tel que vous l'avez vu précédemmen pour transférer le script de correction `correction_tp1.sh.x` SUR LA VM.
2. Se connecter en SSH sur la VM, rendre le script exécutable (`chmod`).
3. exécuter le script de correction avec en paramètre le hash obtenu dans la première question du quiz moodle :







4. Le hash obtenu est à copier coller dans la seconde question du qui moodle. **N'oubliez pas de cliquer sur vérifier avant de cliquer sur terminer le test !**

♢