Docker

Sans trop rentrer dans les détails mais pour savoir l’utiliser

(comment j’aurais aimé qu’on m’explique)

Pourquoi Docker ?

Pour éviter que le mec qui travail sous windows ai pas les mêmes configs que celui sur mac, linux etc…

Si on exécute tous la même images, aucune problème de compatibilités

Donc si quelqu'un à déjà tout setup, pas besoin de se prendre la tête avec des apt-get install ou autres connerie de Windows (qui compile sur windows ???)

C’est quoi docker ?

« Il s’agit d’une plateforme logicielle open source permettant **de créer, de déployer et de gérer des containers d’applications virtualisées sur un système d’exploitation**. »

C’est un logiciel qui vas gérer vos « containers »

C’est quoi une image

C’est un peu comme un .iso

Ca contient toutes les configurations d’un système

C’est quoi des containers ?

« Il s’agit d’un ensemble de processus logiciels léger et indépendant, regroupant tous les fichiers nécessaires à l’exécution des processus : code, runtime, outils système, bibliothèque et paramètres. Ils peuvent être utilisés pour exécuter des applications Linux ou Windows. »

C’est une sorte de processus qui est une machine avec toutes les libs chargées dessus.
Si je la lance sur n’importe quel machine, le comportement sera la même.

Lancer un container docker

LA commande :

docker run –it <image>

L’option –it lance l’image en mode interactif (un shell).

(run est équivalent a docker create + docker start)

Le réseau avec docker

C’est compliqué….

Le plus simple c’est de rediriger les ports de son container sur localhost.
Avec la commande run et l’option –p port_src:port_dst

Exemple

docker run –p 80:8080 <mon_image>

Le port 8080 du container est bindé sur 80 (donc acessible sur 127.0.0.1:80)

Nommer ses containers

Comme l'id des containers change, il est plus simple de le nommer. Il est donc possible de :

Nommer un container lors du start avec l'option –name

docker run --name le_nom –it <image>

Renomer un container

docker rename <old_name> <new_name>

Arrêter un container en cours

Pour stoper un container proprement :

docker stop <id_container>

Sinon : ctrl+c plusieurs fois

Sinon ctrl+p+q -> met le container en mode background

Gérer ses containers

Pour lister les container en cours d’éxecution :

```
docker ps
```

Pour lister tout les containers (même arrêté) :

```
docker ps -a
```

Redémarrer un container

Si un container est à l'arrêt il peut être redémarré simplement, sans se rappeler des différentes options :

docker start <id_container>

Docker exec

Executer un programme dans l'image (si on as plus l'accès interractif)

docker exec <id_container> <programme>

Exemple : docker exec –it 1317d2dbd3a8 bash

-> Lance un terminal sur le container 1317d2dbd3a8:

La mémoire avec docker

Si vous arrêtez un container, tout ce qui enregistré dessus seras perdu (sauf ce qui est en raccourcis ou mode persistent)

Si vous souhaitez utiliser des fichiers de votre machine locale deux solutions :

Créer une « raccourcis » sur le container vers mon système

```
docker run -v <path-of-src-directory>:<path-of-container-directory> <mon_image>
```

Copier des fichiers

```
docker cp <fichier_ou_dossier> <id_container>:<path_destination>
```

Gérer ses containers

Pour lister les container en cours d’éxecution :

```
docker ps
```

Pour lister tout les containers (même arrêté) :

```
docker ps -a
```

DockerHub

C’est un dépôt où plein d’images docker sont disponible.

Par exemple https://hub.docker.com/_/debian

docker pull debian:latest

Ou même pour un tp de software security

https://hub.docker.com/repository/docker/hureauo/m2cysec_physical_security

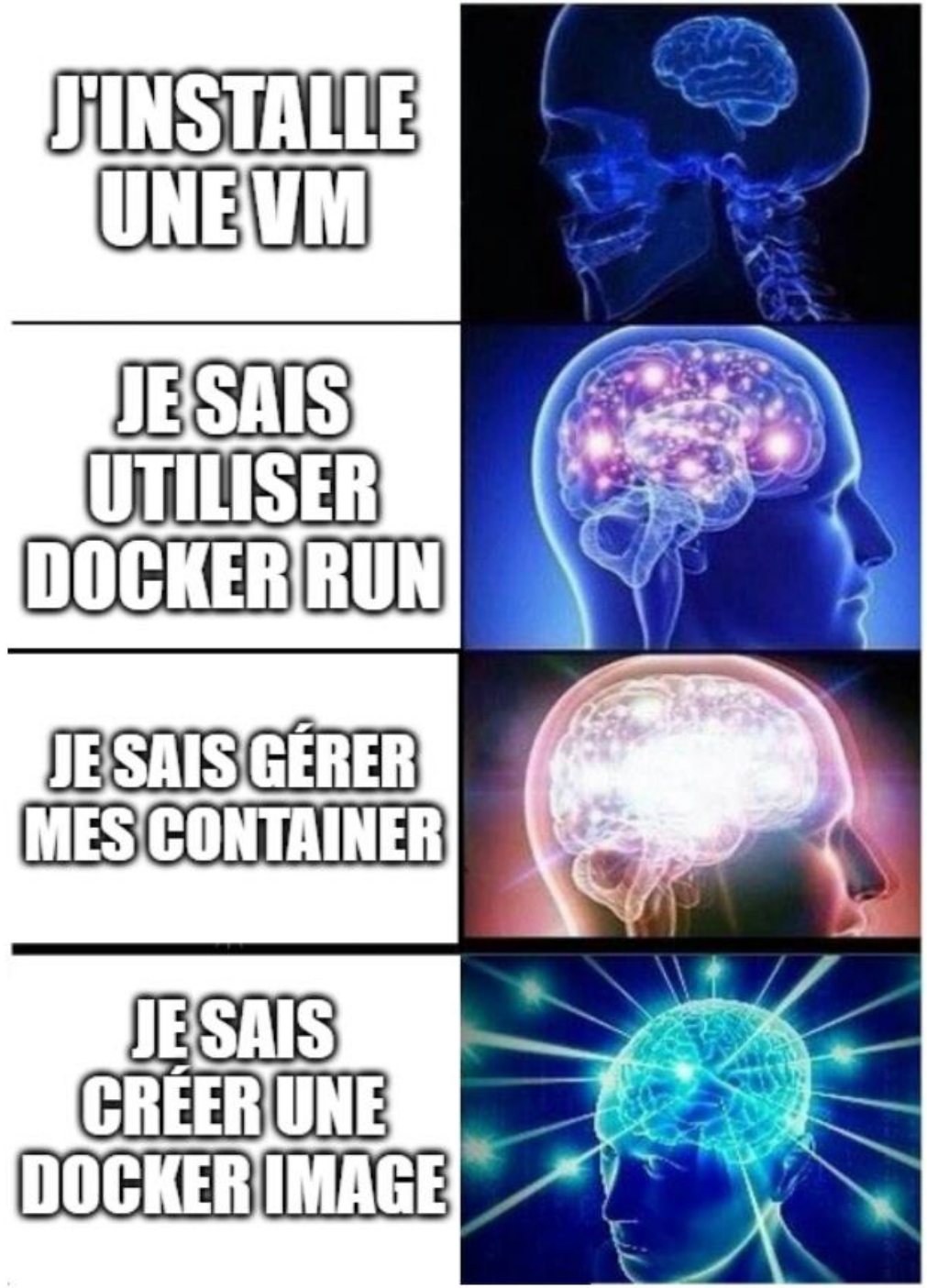
docker pull hureauo/m2cysec_physical_security

DockerHub

Je peux même directement créer un container avec une image du dockerhub :

```
docker run –it --name labPS –p 8888:8888 hureauo/m2cysec_physical_security
```

J’ai donc le notebook pour le lab de physicall security sur 127.0.0.1:8888 et le container s’appel labPS


J'INSTALLE UNE VM
JE SAIS UTILISER DOCKER RUN
JE SAIS GÉRER MES CONTAINER
JE SAIS CRÉER UNE DOCKER IMAGE

Construire une image

docker build –t <nom de l’image> <chemin_vers_dockerfile>

Docker file

Créer une image docker avec openssl 1.1.1.c pour éviter de mettre mon système en vrac mais coder la version que je souhaite.

```
FROM debian:jessie
LABEL maintainer="HUREAU Olivier <Olivier.Hureau@etu.univ-grenoble-alpes.fr>"
ENV \
        INITSYSTEM on \
        DEBIAN_FRONTEND=noninteractive
# Basic build-time metadata as defined at http://label-schema.org
ARG BUILD_DATE
ARG VCS_REF
LABEL org.label-schema.build-date=$BUILD_DATE \
        org.label-schema.docker.dockerfile="/Dockerfile" \
        org.label-schema.name="drakkar-cert-env-build"

#-------------Install basepackages-------------#
RUN apt-get update && DEBIAN_FRONTEND=noninteractive apt-get  -y install \
gcc \
make \
cmake \
build-essential \
gdb \
doxygen \
git \
curl \
nano \
valgrind \
wget \
&& apt-get clean  \
&& rm -rf /var/lib/apt/lists/ */tmp/* /var/tmp/*

#--- INSTALL OPENSSL 1.1.1c ---#
RUN cd /tmp && wget https://www.openssl.org/source/openssl-1.1.1c.tar.gz \
&& mkdir /opt/openssl && cd /opt/openssl \
&& tar xfzv /tmp/openssl-1.1.1c.tar.gz --directory /opt/openssl && cd openssl-1
&& ./config --prefix=/opt/openssl --openssldir=/opt/openssl/ssl -d shared no-as
&& make && make install \
&& rm /usr/bin/openssl && ln -s /opt/openssl/bin/openssl /usr/bin/openssl \
&& echo  /opt/openssl/lib >> /etc/ld.so.conf && ldconfig
```

Docker image, la magie

Une image est construite sur différentes « layer ». Une fois qu'une couche a était construites correctement, pas besoin de la refaire.

Exemple 1 : Précédemment, si la commande RUN d'openssl a planté pas besoin de refaire les apt-get install

Exemple 2 : Si j'ai déjà l'image de debian:latest, pas besoin de la re-télécharger.

Docker build

Avec la commande

docker build -t mon_image .

Si je me trouve dans un dossier où il y a un dockerfile cela vas créer l’image décrite et nomer l’image mon_image

Gestion des images

Lister les images

```
docker image ls
```

Supprimer une image

```
docker image rm <mon_image>
```

Comment j’ai construis l’image pour le tp3 en software security

Création du dockerfile

```
FROM debian:jessie
LABEL maintainer="HUREAU Olivier <Olivier.Hureau@etu.univ-grenoble-alpes.fr>"
ENV \
        INITSYSTEM on \
        DEBIAN_FRONTEND=noninteractive
# Basic build-time metadata as defined at http://label-schema.org
ARG BUILD_DATE
ARG VCS_REF
LABEL org.label-schema.build-date=$BUILD_DATE \
        org.label-schema.docker.dockerfile="/Dockerfile" \
        org.label-schema.name="drakkar-cert-env-build"

#--------------Install basepackages--------------#
RUN apt-get update && DEBIAN_FRONTEND=noninteractive apt-get  -y install \
gcc \
make \
build-essential \
gdb \
wget \
nano \
valgrind \
&& apt-get clean  \
&& rm -rf /var/lib/apt/lists/ */tmp/* /var/tmp/*

# -- INSTALL AFL -- #
RUN mkdir /SoftwareSecurity \
&& cd /SoftwareSecurity \
&& wget http://lcamtuf.coredump.cx/afl/releases/afl-latest.tgz \
&& tar zxvf afl-latest.tgz \
&& rm afl-latest.tgz
```

Comment j’ai construis l’image pour le tp3 en software security

Build de l’image :

```
docker build -t lab3-sofwtare-secu .
```

Création du container

```
docker run -it --name containerlab3 lab3-sofwtare-secu
```

Copie des fichiers

```
docker cp LabFuzzing containerlab3:/SoftwareSecurity/
```

Comment j’ai construis l’image pour le tp3 en software security

Commit du container en cours (sauvegarde l’état actuel du container en une image)

```
docker commit containerlab3 lab3topush
```

Tag du container vers l’image dockerhub

```
docker tag lab3topush hureauo/m2cysec_software_security
```

Push du container

```
docker push hureauo/m2cysec_software_security
```

Comment vous pouver utiliser l’image

```
docker –run –it hureauo/m2cysec_software_security
```

Cependant je conseil de faire un “raccourcis” pour vos fichier :

```
docker run -it -v chemin_absolu:/SoftwareSecurity/src
hureauo/m2cysec_software_security
```

Installation

Linux / mac : Google

Windows : Je recommande l'installation avec WSL2
https://docs.microsoft.com/fr-fr/windows/wsl/install-win10

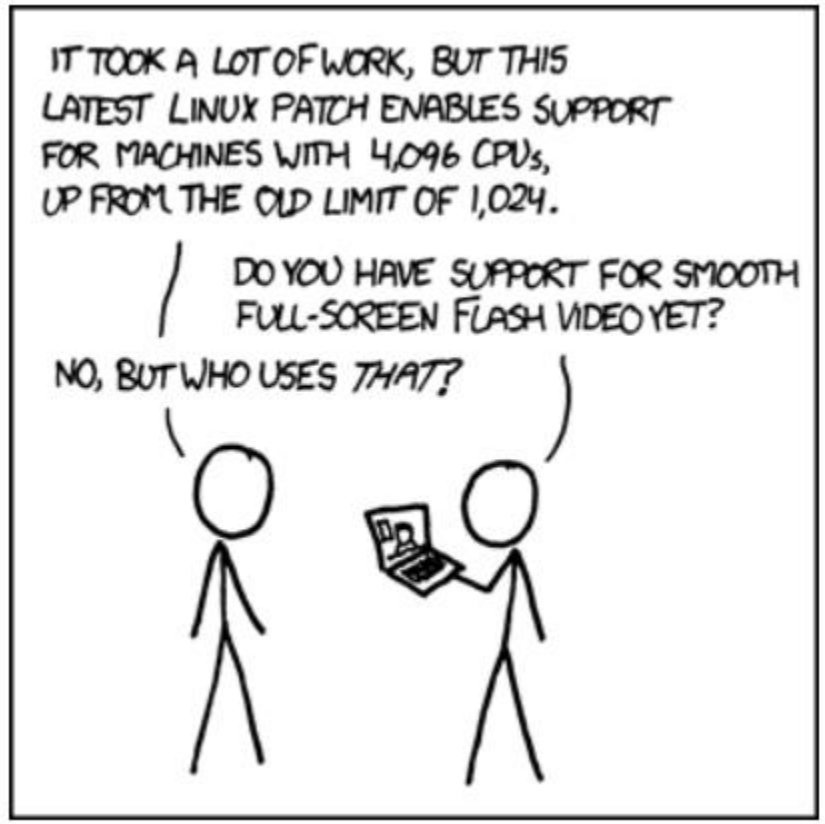
Si vous n'avez pas la bonne licence windows :
https://azureforeducation.microsoft.com/devtools
Login : loginUGA@azure.univ-grenoble-alpes.fr

Questions ?

1) RTFM

2) Google.fr

3) https://forums.docker.com/



Fichiers dispo sur https://github.com/egobiah/Decouvrir-Docker