Mode opératoire : Docker en WSL2

Généralités

- Auteur initial : Jacques Buchaillot

Installer une distribution Linux sur WSL2

- Installer WSL2
- Installer Ubuntu ou Debian via le Windows Store
- Vérifier la version de WSL liée à Ubuntu

```
wsl -l -v
  NAME            STATE           VERSION
* Ubuntu-20.04    Running         2
```

> **Si vous n'êtes pas en version 2 suivez le paragraphe de mise à jour de version de WSL dans le lien précédent** :
>
> ### Upgrade version from WSL 1 to WSL 2
>
> New Linux installations, installed using the `wsl --install` command, will be set to WSL 2 by default.
>
> The `wsl --set-version` command can be used to downgrade from WSL 2 to WSL 1 or to update previously installed Linux distributions from WSL 1 to WSL 2.
>
> To see whether your Linux distribution is set to WSL 1 or WSL 2, use the command: `wsl -l -v`.
>
> To change versions, use the command: `wsl --set-version <distro name> 2` replacing `<distro name>` with the name of the Linux distribution that you want to update. For example, `wsl --set-version Ubuntu-20.04 2` will set your Ubuntu 20.04 distribution to use WSL 2.
>
> If you manually installed WSL prior to the `wsl --install` command being available, you may also need to enable the virtual machine optional component used by WSL 2 and install the kernel package if you haven't already done so.
>
> To learn more, see the Command reference for WSL for a list of WSL commands, Comparing WSL 1 and WSL 2 for guidance on which to use for your work scenario, or Best practices for setting up a WSL development environment for general guidance on setting up a good development workflow with WSL.

- Tester la connexion Internet :

```
sudo apt update
```

S'il n'y a pas de connexion, la cause peut être un problème de configuration automatique de DNS. Dans ce cas essayer la commande suivante :

```
sudo unlink /etc/resolv.conf
echo "nameserver 8.8.8.8" | sudo tee /etc/resolv.conf > /dev/null
echo -e "[network]\ngenerateResolvConf = false" | sudo tee /etc/wsl.conf >
/dev/null
```

Installations complémentaires :

```
sudo apt install net-tools
```

*Si vous avez choisi **Debian** et non Ubuntu,* configuration système complémentaire (explications) :

```
sudo update-alternatives --set iptables  /usr/sbin/iptables-legacy
sudo update-alternatives --set ip6tables /usr/sbin/ip6tables-legacy
```

Installer Docker Community Edition (`docker-ce`)

```
# Define variables used in following commands
source /etc/os-release

# Trust the officiel Docker repo
curl -fsSL https://download.docker.com/linux/${ID}/gpg | sudo apt-key add -

# Update repo info
echo "deb [arch=amd64] https://download.docker.com/linux/${ID} ${VERSION_CODENAME}
stable" | sudo tee /etc/apt/sources.list.d/docker.list
sudo apt update

# Actually upgrade packages
sudo apt upgrade

# Install official Docker release
sudo apt install docker-ce docker-ce-cli containerd.io docker-compose

# Add user to docker group and reconnect session to include new group
sudo usermod -aG docker $USER
newgrp docker
```

Configurer le service du daemon Docker (`service docker`)

Rappel : WSL2 fonctionne avec SysV Init et non SystemD.

Modifier le fichier de configuration du service Docker :

```
sudo vi /etc/default/docker
```

Rappels VI :

- *`i` pour passer en mode Insertion*
- *`Echap` puis `:wq (Entrée)` pour enregistrer et quitter*
- *`Echap` puis `:q! (Entrée)` pour quitter sans enregistrer*

Trouver la ligne 14, qui commence par `#DOCKER_OPTS=`. Ajouter une ligne en-dessous :

```
DOCKER_OPTS="-H tcp://`ifconfig eth0 | awk '/inet / { print $2 }' | cut -f2 -d:` -
H unix:///var/run/docker.sock --tls=false"
```

A date, il ne semble pas possible de spécifier le lancement d'un service dans une distribution Linux WSL2 (que ce soit au démarrage de Windows ou de la première connexion ligne de commande à WSL2). Tous les liens dans `/etc/init.d`, `/etc/rc*`, etc. échouent, même administrés en installant une commande comme `sysv-rc-conf`. TODO?

- **Exécuter dockerd au démarrage d'Ubuntu**

```
echo "sudo service docker start" >> ~/.profile
```

- Stopper wsl et relancer un Ubuntu

```
# Dans powershell ou cmd
wsl --shutdown
```

Le fichier de log du service Docker est disponible ici : `/var/log/docker.log`

Autres commandes disponibles :

```
# Pour savoir si le service Docker est démarré
sudo service docker status
# Pour arrêter le service Docker, au besoin
sudo service docker stop
```

Première utilisation du client ligne de commande (`docker`)

Premier test de Docker :

```
docker run --rm hello-world
```

(OPTIONNEL) Connexion au daemon Docker en TCP au lieu de la socket Unix standard

Ci-dessus, la configuration `/etc/default/docker`, le daemon Docker (par le biais du service) a été lancé en écoute sur une interface TCP, *en plus* de la socket Unix standard. A toutes fins utiles, il est possible de faire en sorte que le client Docker se connecte à l'API du daemon docker en TCP, en définissant une variable `DOCKER_HOST`.

Ceci est inutile a priori au sein de la ligne de commande WSL2. L'exposition de l'API du daemon Docker en TCP n'a de sens que pour des usages externes à partir de Windows.

Renseigner la variable `DOCKER_HOST` au démarrage de la session en ligne de commande :

```
vi ~/.bashrc
```

Ajouter la ligne suivante en fin de fichier :

```
export DOCKER_HOST=`ifconfig eth0 | awk '/inet / { print $2 }'`
```

Recharger l'environnement de session :

```
source ~/.bashrc
# Vérifier que l'adresse IP de l'interface WSL2 eth0 s'affiche bien :
echo $DOCKER_HOST
```

(OPTIONNEL) Utiliser docker depuis windows

Après avoir configuré la connexion au daemon Docker en TCP, il est possible d'utiliser le client docker depuis la ligne de commande windows.
Pour cela, il faut

- télécharger le docker-cli dans la version souhaité
  https://download.docker.com/win/static/stable/x86_64/
- positionner la variable d'environnement `DOCKER_HOST` dans son terminal windows
  les commandes suivantes sont un exemple avec une installation dans une distribution wsl nommée ubuntu-20.04
  - en powershell : `$Env:DOCKER_HOST='tcp://'+(wsl -d ubuntu-20.04 sh -c "/usr/sbin/ifconfig eth0 | awk '/inet / {print \`$2}'")+":2375"`
  - en cmd : `FOR /F "tokens=* USEBACKQ" %g IN (`wsl -d ubuntu-20.04 sh -c "/usr/sbin/ifconfig eth0 | awk '/inet / {print \$2}'"`) do (SET "DOCKER_HOST=tcp://%g:2375")`

L'utilisation de docker-cli utilise alors le daemon wsl.
Attention lors par exemple de montage disque, ceux-ci sont vu depuis wsl, l'accès au disque se fait

donc au format wsl `/c/Users/...` par exemple