Service Docker

Cette page accueille certaines manipulations pour installer et se servir de Docker, ainsi que ses containers.

Contributeur Original

(SISR2-2020-2021) Aloïs Noël

Contributeurs aux modifications effectuées

(SISR2-2020-2021) Benjamin Hohn
(SISR2-2020-2021) Eric Boissonade

Sommaire

1. Installation :

Avant tout, le Docker sera sous os debian 10, il est nécessaire d’avoir une clé bootable avec debian

10 dessus.

Afin de définir l'ip de la machine debian 10 :

cd /etc/network nano interfaces

```
iface enp3s4f0 inet static

auto br0
iface br0 inet static
     address 172.20.168.211
     netmask 255.255.0.0
     gateway 172.20.0.254
     dns-nameservers 172.20.0.6
     bridge_ports enp3s4f0
     bridge_stp off
     bridge_fd 0
```

A adapter en fonction de l'ip souhaitée.

On installera les services avant d'installer Docker sur la machine et il est nécessaire d'être en super-utilisateur afin de réaliser les actions suivantes.

Faire su pour entrer en mode super-utilisateur, le mot de passe root sera demandé.

On vérifie l'adresse du DNS de votre serveur : nano /etc/resolv.conf Une fois dedans, remplacer par la bonne adresse si ce n'est pas le cas : Nameserver 172.20.0.6

1. Installation de Docker :

a. Installation des services avant Docker :

On va installer les paquets nécessaires pour Docker :

```
 apt install apt-transport-https ca-certificates curl gnupg2 software-
properties-common
```

```
 curl -fsSL https://download.docker.com/linux/debian/gpg | apt-key add -
```

```
 add-apt-repository "deb https://download.docker.com/linux/debian
$(lsb_release -cs) stable"
```

- Faites une update de votre système :

```
 apt update
```

- Upgrade après l'update :

```
 apt upgrade
```

b. Installation de Docker :

- On va installer les paquets Docker :

```
apt install docker-ce
```

- Ensuite vérifier la conformité du service Docker :

```
systemctl status docker
```

c. Configuration réseau

i. Changer l'IP des containers (IP fixe, en passant par Portainer)

Mettre en place un adressage IP fixe en créant un « network » spécifique, puis en assignant les machines à ce « network ». Nous sommes passés par portainer pour créer le réseau RLEK.

Ensuite, nous avons associé les machines (à leur création) sur le réseau :

```
 docker run -it --name nomconteneur  --network RLEK --ip 172.21.100.11
debian
```

L'IP et le nom du réseau est à adapter selon vos besoins.

Ou sur une machine existante :

```
docker connect --network RLEK --ip 172.21.100.12 conteneurLEK
```

L'IP et le nom du réseau est à adapter selon vos besoins.

Il a fallu ajouter la route pour le réseau 172.21 sur le routeur section.

Voir documentation ci dessous pour la route.

L'installation de Portainer est détaillée dans la section containers/applications de la documentation.

ii. Changer l’IP des containers (par plage IP)

- éditer ou créer un fichier /etc/docker/daemon.json de la manière suivante :

```
{
"bip": "x.x.x.x/x" (172.16.0.1 par exemple)
}
```

Une fois ce fichier modifié, lancez :

```
systemctl daemon-reload
systemctl restart docker
```

- Pour vérifier l'ip du docker :

```
ip a
```

- Pour supprimer le fichier si besoin. (dans /etc/docker)

```
rm -f daemon.json
```

iii. Mettre un routeur virtuel

Aller dans le fichier /etc/sysctl.conf"

```
nano /etc/sysctl.conf
```

```
# Décommenter la directive pour conserver le paramétrage à chaque
redémarrage
net.ipv4.ip_forward=1
```

Donc retirer le # de la ligne net.ipv4.ip_forward=1

Après configuration du fichier il faut reboot le serveur : systemctl reboot

iiii. Mettre une route sur le routeur section

Regarder fichier SIO2020-DonnesInfra pour voir le login, mdp et ip afin de se connecter, en telnet au routeur section.

```
en
conf t
```

```
ip route 172.16.0.0 255.255.0.0 172.20.168.211 (toujours un réseau et non
une machine pour la route sauf quand on définit le routeur/machine)
```

Cette route sera à adapter en fonction de la plage ip que l'on attribue au serveur Docker

Pour voir les routes :

```
do show ip route
```

2. Exploitation de Docker :

a. Utilisation de Docker :

Recherche d'une image :

```
docker search #nom_image
```

Après avoir exécuté cette commande une liste d'image disponible vous est affiché, ce sont toutes les images existant avec le $nom_image que vous avez choisi

Par exemple, si l'on veut une image d'ubuntu, on fera :

```
docker search ubuntu
```

Importation d'une image :

```
docker pull #nom_image
```

Par exemple, si l'on veut importer une image d'ubuntu, on fera :

```
docker pull ubuntu
```

Pour voir toutes les images étant sur votre serveur pour la conteneurisation il vous faut utiliser la commande :

```
docker images
```

Exécution d'un conteneur :

```
docker run -it #nom_image
```

Liste de tous les conteneur existant sur votre machine :

```
docker ps -a
```

Démarrer un conteneur :

```
docker start #id/nom du conteneur
```

Entrer dans un conteneur déjà démarré :

```
docker attach #id/nom du conteneur
```

Changer le nom d'un conteneur :

```
docker rename #id #nom
```

b. Création d'un container :

Avant tout voici une liste de commandes utiles avant de continuer :

```
docker images (permet de voir les images du docker)
docker ps -a (permet de voir tous les containers)
```

```
docker run -it (nom image) (lance/crée un container)
docker container start (lance un container)
docker attach (rentrer dans le container lancé)
```

```
docker ps (permet de voir les container actifs, en marche)
```

```
docker container rm -f (id du containeur) (supprime un container, le -f
force la suppression, utile lorsque le container est toujours actif)
```

```
docker rename (nom container) (nom “x”) (renommer un container)
```

Afin de créer, lancer, et rentrer dans le container faites la commande suivante :

```
docker run -it (nom image)
```

Par exemple :

```
docker run -it ubuntu
```

Pour quitter le container sans l’éteindre :

```
ctrl p + ctrl q
```

Pour quitter le container et l’éteindre :

```
exit
```

Pour connaitre version ubuntu (lorsque ubuntu est installé sur le container) :

```
cat /etc/lsb-release
```

Pour voir l’ip du container :

```
ifconfig
```

Pour voir le nom de la machine (container) :

```
hostname
```

Dans les détails on peut définir son nom et son ip dès sa création :

```
 docker run --it --name "nom du container" --net "nom du réseau" --ip "adresse ip du container" "image"
```

On peut aussi définir son adresse ip après sa création en le connectant à un réseau existant sur le Docker :

```
 docker network connect "réseau" --ip "adresse ip du container" "nom du container"
```

Pour voir les réseaux existants sur le Docker :

```
 docker network ls
```

Pour voir les informations d'un réseau en particulier :

```
 docker network inspect "nom du réseau"
```

Par exemple : docker network inspect rezodok

c. Créer un modèle a partir d'un container existant

On va créer un modèle (image) a partir d'un container existant :

```
 docker commit "nom du container que vous allez copier" "nom de l'image que vous allez créer"
```

Vérifier que l'image existe :

```
 docker images
```

Créer le container avec l'image créée : (–name, –net, –ip pas obligatoire)

```
 docker run -it --name "nom du container" --net "nom du réseau" --ip "adresse ip du container" "nom de l'image créee"
```

3. Applications dans les containers

a. Wordpress

Préparation à l’installation :

```
 apt update
```

```
apt upgrade
apt install nano
apt install wget
apt install net-tools
```

Installation des services :

```
apt install apache2 php libapache2-mod-php mariadb-server php-mysql
```

vérifier statut service apache2 :

```
service apache2 status
```

démarrer service apache2 :

```
service apache2 start
```

Pour vérifier le statut du service :

```
service mysql status
```

Pour démarrer le service :

```
service mysql start
```

Une fois la pile LAMP installée (avec les modules PHP les plus courants), on active le module mod_rewrite, dont a aussi besoin WordPress :

```
a2enmod rewrite
```

redémarrer le service :

```
service apache2 restart
```

Nous allons ensuite créer un hôte virtuel pour WordPress, dont la racine sera /var/www/wordpress :

```
nano /etc/apache2/sites-available/wordpress.conf
```

Ce qui ouvre avec l'éditeur nano un fichier dans lequel nous allons coller :

```
“wordpress.conf”
<VirtualHost *:80>
     ServerName wordpress.localhost
     DocumentRoot /var/www/wordpress
     <Directory /var/www/wordpress>
             AllowOverride all
             Require all granted
     </Directory>
     ErrorLog /var/log/apache2/error.wordpress.log
     CustomLog /var/log/apache2/access.wordpress.log combined
```

```
</VirtualHost>
```

Finalement, on active l'hôte virtuel et on recharge la configuration d'Apache :

```
sudo a2ensite wordpress
service apache2 reload
```

Installation de WordPress : Copie des fichiers Tout d'abord téléchargeons la dernière version de WordPress :

```
wget https://fr.wordpress.org/wordpress-latest-fr_FR.zip
```

Ensuite nous allons extraire le contenu du zip à la racine de notre hôte virtuel (/var/www/wordpress dans cet exemple) :

```
apt install unzip
sudo unzip wordpress-latest-fr_FR.zip -d /var/www
mv /var/www/wordpress /var/www/html/
```

Retirer le sudo si vous etes déja en super-utilisateur

On va renforcer légèrement la sécurité en attribuant des droits un peu restrictifs aux fichiers :

```
sudo chown www-data:www-data /var/www/wordpress -R
sudo chmod -R -wx,u+rwX,g+rX,o+rX /var/www/wordpress
```

Retirer le sudo si vous etes déja en super-utilisateur

Création de la base de données Pour créer la base de données qu'utilisera WordPress, le plus simple est de se connecter avec le client MySQL :

```
mysql
```

On arrive alors sur la console SQL sur laquelle nous allons entrer ces commandes (en remplaçant mot_de_passe) :

```
CREATE DATABASE wordpress;
CREATE USER 'wpuser'@'localhost' IDENTIFIED BY 'mot_de_passe';
GRANT ALL ON wordpress.* TO 'wpuser'@'localhost';
FLUSH PRIVILEGES;
QUIT;
```

Il faut remplacer mot_de_passe par un vrai mot de passe qu'on note pour la prochaine étape.

Login : wpuser MDP : mpuser

On vient de créer la base de données wordpress à laquelle l'utilisateur wpuser aura accès.

Installation via l'interface web WordPress devrait alors être accessible à l'adresse 172.16.0.4/wordpress/ , et cette adresse nous redirige sur une interface qui nous permet de finaliser l'installation. Cliquez sur C'est parti ! et renseignez les différents champs :

Nom de la base de données : wordpress

Identifiant : wpuser

Mot de passe : le mot de passe qu'on a noté à l'étape précédente (lors de la création de la base de donnée et de l'utilisateur wpuser) Donc mpuser

Adresse de la base de données : localhost

Préfixe des tables : wp_

Après avoir cliqué sur Lancer l'installation, on tombe sur un second formulaire. Il s'agit cette fois de définir un administrateur pour WordPress. Les champs sont assez clairs. Le titre du site sera visible par les internautes.

Login : wpuser MDP : mpuser

b. Portainer

On va créer le volume de données :

```
 docker volume create portainer_data
```

Ensuite créer le container qui va contenir portainer :

```
 docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -
v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data
portainer/portainer-ce
```

La commande “restart=always” permet au container de redémarrer à chaque démarrage du Docker

L’IP pour accéder au portainer (à adapter, l’ip présente ici est celle de notre serveur Docker) portainer : http://172.20.168.211:9000

Les logs pour s’y connecter : (ce sont des exemples, vous pouvez les modifier comme vous le souhaitez) admin mpadminSIO

Remarques : On remarque que le container n’a pas d’ip car on se connecte par le biais de l’ip du docker et de son port : 172.20.168.211:9000

Ouverture portainer vers l’extérieur :

Configuration du reverse proxy nginx sur le pm20 :

Création fichier portainer en copiant celui du si4 et en modifiant les lignes suivantes : server_name “nomdedomaine.net”; donc server_name “portainer.inforostand14.net”

location /{ proxy_pass http://ipdelamachine:port/; } donc location /{ proxy_pass http://portainer.inforostand14.net:9000/; }

Configuration dans la messagerie Orange : Création dans la messagerie Orange pour la zone DNS: portainer.inforostand14.net Type A Renseigner l'IP de la livebox

Redémarrer le service nginx sur le pm20 : service nginx restart