INSTALLATION D'UN SERVEUR DOCKER ROOTLESS SOUS RASPBERRY PI

Raspberry Pi - Debian Bullseye

Configuration de base

Tutoriel **DOCKER** - RASPBERRY PI

David GOÏTRÉ

Table des matières

Introduction

DOCKER a toujours exigé des privilèges root pour s'exécuter. En effet, certaines fonctionnalités telles que les espaces de noms ou les points de montage qui constituent la base de Docker ont toujours requis des privilèges élevés.

Étant donné que Docker est composé d'une pile entière de processus sous-jacent (runc, containerd, dockerd, etc.). L'exécution de docker mode Rootless signifie donc que l'intégralité de la stack docker sera exécutée en mode Rootless. De plus, étant donné que dockerd lui-même s'exécute en tant qu'utilisateur non root, les conteneurs lancés n'auront également aucun privilège root associé.

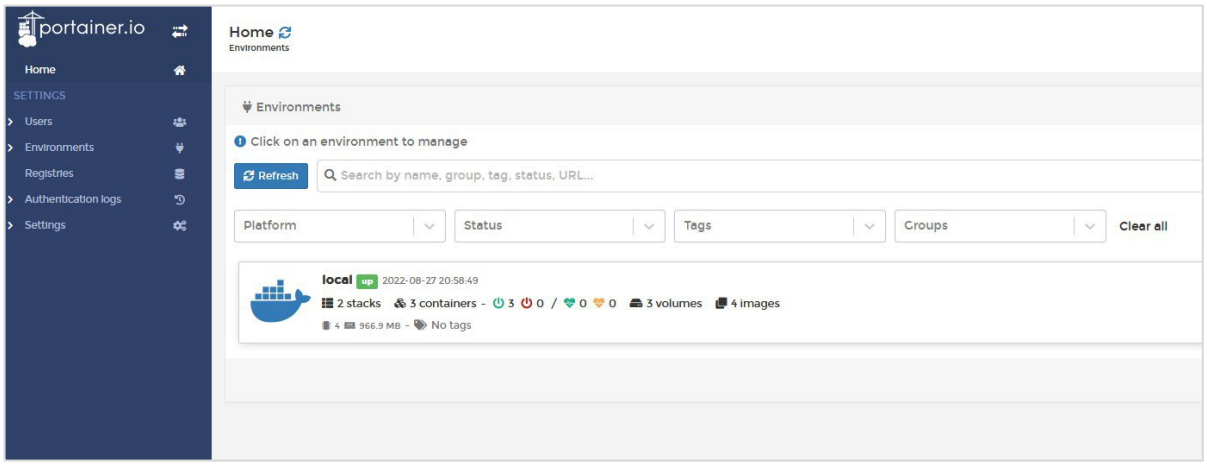
Ce mode, qui constitue une bonne pratique recommandée par l'ANSI protège le système hôte contre des attaques qui pourraient exploiter des vulnérabilités potentielles du code de l'application ou d'erreurs de configuration.

1. Pré requis

On a besoin des différents matériels et logiciels pour la création d'un Serveur DOCKER ROOTLESS avec un RaspberryPi.

- Un ou des PC client sous Windows
- Une Box (Free, Orange, Sfr…)
- Un Raspberry 3B+ avec l'OS Raspian Bullseye installé avec Etcher
- Le logiciel Putty pour se connecter en SSH au serveur
- Connaitre l'interface réseau (eth0, br0, ens3…) via la commande : **ip a**
  Pour notre test c'est **l'interface eth0** qui sera utilisée

Voici l'interface que l'on doit obtenir une fois connecter au serveur **PORTAINER avec Docker Rootless** mise en place.



2. Connexion au serveur

a) Activer le **SSH** sur le serveur. Pour ce faire, ouvrir la carte SD du RaspberryPi via l'explorateur de Windows et créer un fichier **ssh** (sans extension) à sa racine.

b) Ouvrir **Putty** et se connecter au serveur DOCKER avec les identifiants (par défaut **pi/raspberry**)

c) Mettre à jour les packages du système vers la dernière version. Exécuter la commande suivante pour mettre à jour et mettre à niveau les packages de votre système :

```
# apt-get update -y
# apt-get upgrade -y
```

3a. Paramétrage Ethernet du serveur

Avant d'aller plus loin, il nous faut connaître l'interface réseau de notre serveur **RaspberryPI** et lui attribuer une adresse IP fixe.

a) Lister les interfaces

```
$ ip link | awk '{ print $2}'   # liste les interfaces
# ethtool <interface> | grep detected   # détecte l'interface connectée
```

b) Définir une adresse IP fixe

```
# nano /etc/dhcpcd.conf  # ouvre le fichier de configuration réseau
```

c) Copier le texte ci-dessous à la fin du fichier **dhcpcd.conf**

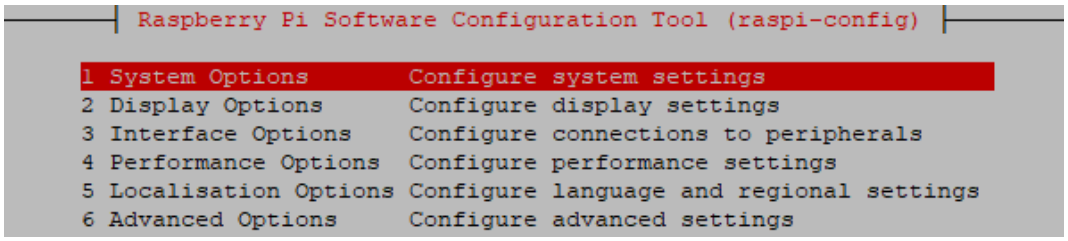
```
interface nom de l'interface réseau
static ip_address=192.xxx.xxx.xxx/24
static routers=192.xxx.xxx.xxx
```

d) Rebooter le serveur

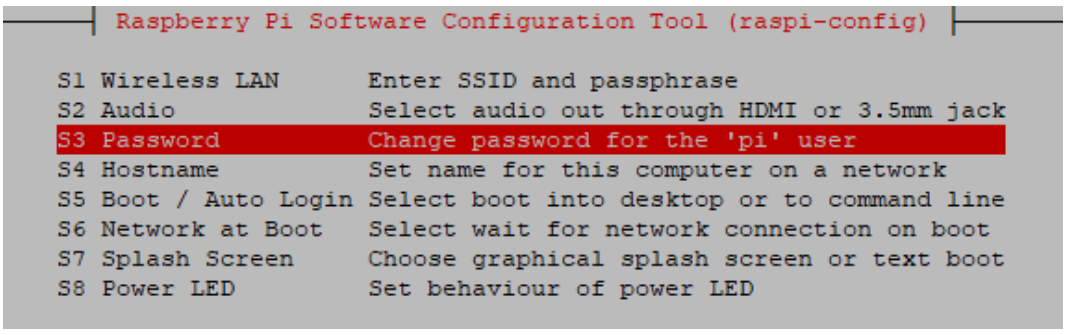
```
# sudo reboot
```

e) Paramétrer le serveur

```
$ raspi-config   # ouvre l'utilitaire, sélectionner le menu System Options
```



Sélectionner le menu **S3 Password** pour modifier le mot de passe et **S4 Hostname** pour modifier le nom du serveur.



3b. Paramétrage Wifi du serveur

Par défaut le Wifi est désactivé. Il faut créer un fichier **wpa_supplicant.conf** et le copier à la racine de la carte SD, permettant à Raspberry Pi OS de lire le fichier au prochain démarrage et d'appliquer la configuration directement.

a) ouvrir un éditeur de texte et copier le texte suivant

```
country=FR
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
        ssid="NOM_RESEAU"
        scan_ssid=1  #nécessaire quand le ssid n'est pas diffuser
        psk="MOTDEPASSE"
        key_mgmt=WPA-PSK
}
```

b) Modifier les champs du **SSID** et **PSK**

c) Enregistrer le fichier sous le nom **wpa_supplicant.conf** et copier-le à la racine de la carte SD

4. Limitation de Docker Rootless

DOCKER ROOTLESS possède des limitations sur des fonctionnalités qui sont importantes à connaitre, avant de l'installer.

a) Seuls les pilotes de stockage suivants sont pris en charge :
- overlay2 (uniquement en cas d'exécution avec le noyau 5.11 ou ultérieur, ou le noyau Ubuntu)
- fuse-overlayfs (uniquement s'il fonctionne avec le noyau 4.18 ou ultérieur et fuse-overlayfsest installé)
- btrfs (uniquement s'il fonctionne avec le noyau 4.18 ou ultérieur, ou ~/.local/share/dockerest monté avec user_subvol_rm_allowedl'option de montage)
- vfs

b) Cgroup est pris en charge uniquement lors de l'exécution avec cgroup v2 et systemd

c) Les fonctionnalités suivantes ne sont pas prises en charge :
- AppArmor
- Point de contrôle
- Réseau superposé
- Macvlan
- Exposer les ports SCTP
- Pour utiliser la pingcommande, voir Routage des paquets ping .
- Pour exposer les ports TCP/UDP privilégiés (< 1024)

d) IPAddress montré dans **docker inspect** est un espace de noms à l'intérieur de l'espace de noms réseau de RootlessKit. Cela signifie que l'adresse IP n'est pas accessible depuis l'hôte sans nsenter-ing dans l'espace de noms du réseau.

e) Le réseau hôte (docker run --net=host) est également espace de noms à l'intérieur de RootlessKit. Les montages NFS car le docker "data-root" n'est pas pris en charge. Cette limitation n'est pas spécifique au mode sans racine

5. Création d'un utilisateur

Par défaut, l'utilisateur **DOCKER** est le **root** de la machine hôte. Donc possède tous les privilèges pour docker aussi.

a) Créer l'utilisateur et l'ajouter au groupe Docker

```
# sudo apt install uuidmap
# sudo adduser <nomuser>
# sudo usermod -aG sudo docker
```

6. Configurer le référentiel

Avant de commencer l'installation de docker, il faut récupérer toutes les sources

a) Télécharger les sources

```
# sudo apt-get update
# sudo apt-get install ca-certificates
# curl gnupg lsb-release uidmap iptables dbus-user-session && \ curl -fsSL
https://download.docker.com/linux/debian/gpg
# sudo gpg --dearmor -o  /usr/share/keyrings/docker-archive-keyring.gpg && \ echo "deb
[arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-
keyring.gpg] https://download.docker.com/linux/debian $(lsb_release -cs) stable"
# sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

7. Installation le moteur Docker

a) Installer le **Docker Engine**

```
# sudo apt-get install docker-ce docker-ce-cli containerd.io
# sudo systemctl disable --now docker.service docker.socket
# sudo reboot
```

8. Récupération du script d'installation

a) Désinstaller docker

```
# sudo apt-get install -y docker-ce-rootless-extras
```

9. Publication des conteneurs en utilisant les ports inférieurs à 1024

a) Publier les ports inférieurs à 1024

```
# sudo setcap cap_net_bind_service=ep $HOME/bin/rootlesskiE/bin/rootlesskit
```

10. Enregistrer les variables d'environnements

a) Editer le fichier **bashrc**

```
# sudo nano ~/.bashrc  # Edite le fichier bashrc
```

b) Copier les lignes ci-dessous en bas du fichier

```
# export PATH=/home/docker/bin:$PATH
# export DOCKER_HOST=unix:///run/1000/docker.sock
```

11. Installation des packages pour Linux Debian

a) Installer les packages

```
# sudo apt install -y dbus-user-session
# sudo apt install -y fuse-overlayfs
# sudo apt install -y slirp4netns
# sudo slirp4netns --version doit etre supérieur à la v0.4.0
```

12. Installation de Docker Rootless avec l'utilisateur créer

Avant de pouvoir installer **Docker Rootless**, il faut se connecter avec l'utilisateur créer précédemment

a) Installer **Docker Rootless**

```
$ dockerd-rootless-setuptool.sh install
$ systemctl --user start docker
$ systemctl --user enable docker
$ sudo loginctl enable-linger $(whoami)
```

13. Installation de Portainer

Toujours avec l'utilisateur **non root**, installer Portainer. La différence de cette commande par rapport à celle utilisée par Docker réside dans le changement du socket **/$XDG_RUNTIME_DIR**.

a) Installer Portainer en étant connecté en non root

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

14. Opérations divers

Ces commandes sont optionnelles, mais peuvent être utiles dans certains cas.

a) Désactiver le démon docker à l'échelle du système est déjà en cours d'exécution

```
# sudo systemctl disable --now docker.service docker.socket
```

b) Supprimer le service **systemd** du démon Docker

```
# dockerd-rootless-setuptool.sh uninstall
```

15. Conclusion

DOCKER ROOTLESS est installé et configuré avec succès sur le serveur **RaspberryPi Debian 11**. On peut désormais créer des containers Docker Rootless.

Destiné au RaspberryPi (Raspbian), **DOCKER ROOTLESS** fonctionne aussi parfaitement sur une distribution Ubuntu, Debian...

Pour **Debian** : Installation docker rootless sur Debian
Pour **Information** : Le mode rootless