FONCTIONNALITES AVANCEES DE DOCKER ET PORTAINER

Raspberry Pi - Debian Bullseye
Configuration avancée

Tutoriel **DOCKER** - RASPBERRY PI

David GOÏTRÉ

Table des matières

Introduction 1

1. Création du réseau de configuration Macvlan via Portainer 1

2. Création du réseau de création Macvlan via Portainer 2

3. Association du MacVlan au conteneur nginx via Portainer 2

4. Création du réseau Macvlan avec Docker 3

5. Association du MacVlan au conteneur nginx via Docker 3

6. Création d'un conteneur persistant 4

7. Sécurisation de Docker avec les User Namespace 4

8. Installation de Grafana 5

9. Installation de Duplicati 5

10. Installation de watchtower 6

11. Installation de Adguard Home 6

12. Liens annexes 7

13. Conclusion 7

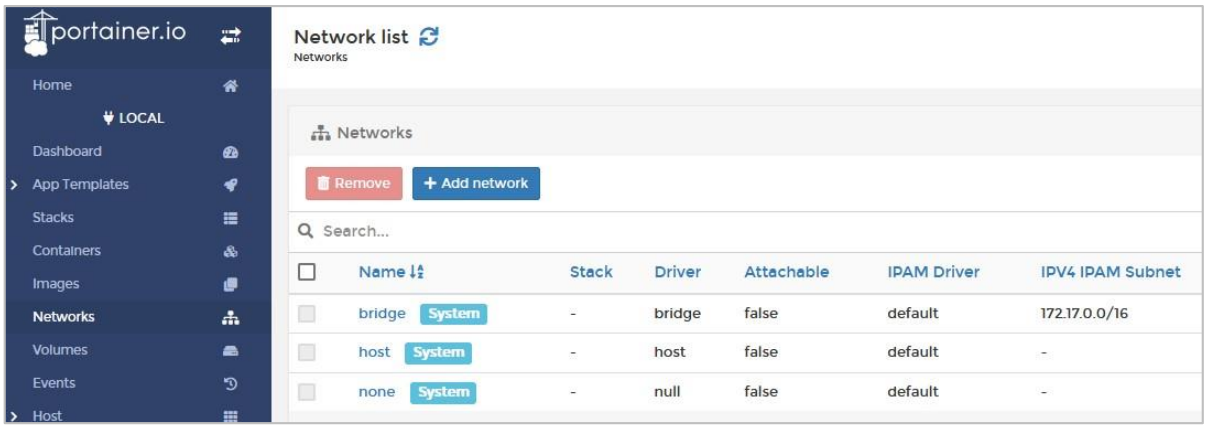
Introduction

DOCKER est une plateforme permettant de gérer plusieurs conteneurs sur une même machine. **PORTAINER** est une interface utilisateur de gestion légère qui nous permet de gérer facilement vos différents **environnements Docker** (hôtes Docker ou clusters Swarm). Nous allons voir dans ce tutoriel les fonctions avancées pour aller plus loin dans le développement et la mise en place d'applications.

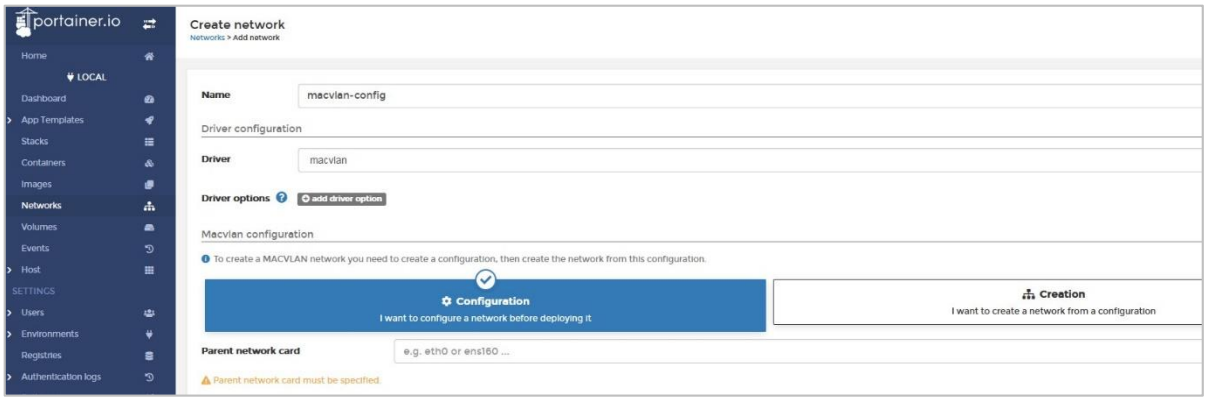
1. Création du réseau de configuration Macvlan via Portainer

L'utilisation du driver **macvlan** est parfois le meilleur choix lorsque vous utilisez des applications qui s'attendent à être directement connectées au réseau physique, car le driver Macvlan nous permet d'attribuer une adresse MAC à un conteneur, le faisant apparaître comme un périphérique physique sur notre réseau. Le moteur Docker route le trafic vers les conteneurs en fonction de leurs adresses MAC.

a) Cliquer sur le bouton **Networks,** puis sur le bouton **Add network**



b) Créer un réseau de **configuration**



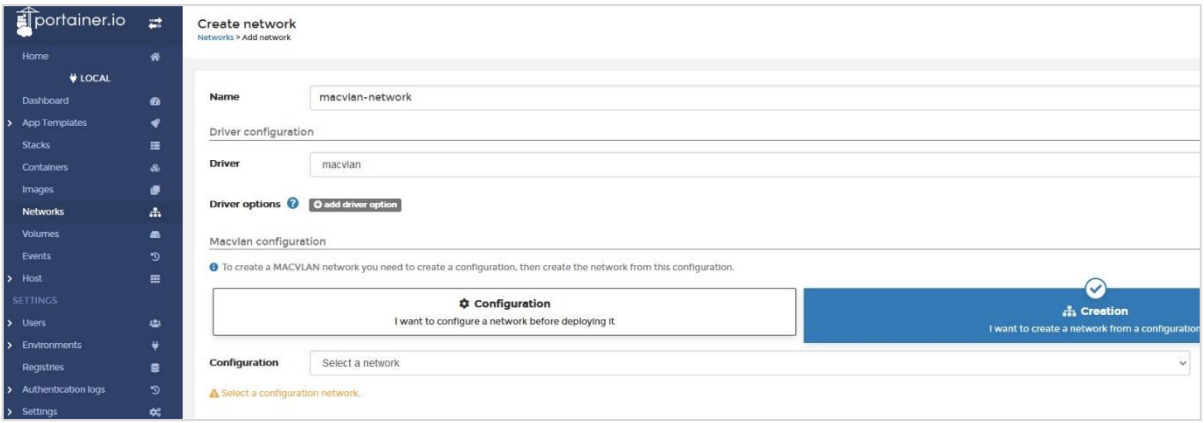
- Choisir un nom pour notre réseau de configuration : **macvlan-config**
- Choisir le driver **macvlan** (le bouton **Configuration** est sélectionner par défaut)
- Saisir le **subnet** (sous réseau) : 192.168.x.x/24
- Saisir la **gateway** (passerelle de notre routeur) : 192.168.0.254
- Saisir l'IP range (plage d'@IP) : 192.168.x.50/28 permettra d'avoir 14 @ip utilisables
- Saisir le **Parent network card** (nom de la carte réseau du serveur Docker)

c) Cliquer sur le bouton **Create the network**

2. Création du réseau de création Macvlan via Portainer

C'est ce réseau de **création** qui sera utiliser par le conteneur que l'on veut y associer

a) Revenir sur la création de réseau et cliquer de nouveau sur le bouton **Add network**
- Choisir le driver **macvlan** (le bouton **Configuration** est sélectionner par défaut)
- Saisir un nom pour le réseau de création : **macvlan-network**
- Cliquer sur le bouton **Création**
- Saisir le nom de la **carte réseau de l'hôte** (ex : eth0)
- Dans la **liste configuration**, sélectionner le réseau créer précédemment **macvlan-config**

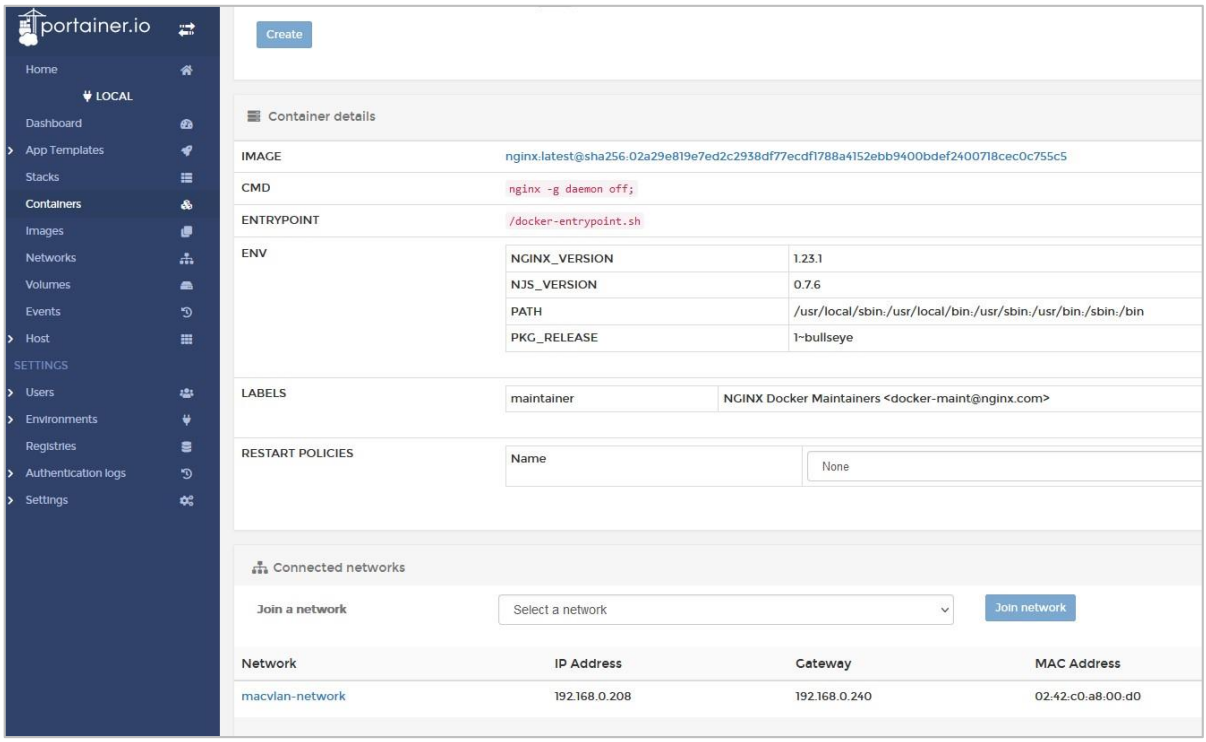


b) Cliquer sur le bouton **Create the network**

3. Association du MacVlan au conteneur nginx via Portainer

On a installé un conteneur Docker via le **dockerhub** de **Portainer**. Il faut maintenant l'attacher au réseau **macvlan**.

a) Editer le conteneur nginx et sélectionner dans la liste **macvlan-network**



b) cliquer sur le bouton **Join network**

4. Création du réseau Macvlan avec Docker

Docker propose également un moyen simple et efficace pour créer et affecter un sous réseau à un conteneur

a) Création des réseaux de **configuration** et **création**

```
# sudo docker network create  --config-only --subnet 192.168.0.0/24 --gateway 192.168.0.254 --ip-range 192.168.0.220/28 -o parent=eth0 macvlan-config
# sudo docker network create -d macvlan --scope local --config-from macvlan-config macvlan-network
```

Maintenant il nous reste plus qu'à déployer un docker-compose.yml avec les réseaux que l'on vient de créer. **Attention dans le macvlan, les IP du conteneur sont directement accessibles depuis l'extérieur de l'hôte (sans avoir à passer par le NAT). La publication de ports dans macvlan n'est donc pas nécessaire et peut provoquer une erreur lorsque l'on essaye de le publier.**

5. Association du MacVlan au conteneur nginx via Docker

Maintenant il n'y a plus qu'à déployer le **docker-nginx.yml** ci-après : prendre soin de changer correctement le nom du réseau macvlan-network si l'on n'a pas choisi le même et choisir l'@IP qui nous convient pour le conteneur.

a) Vérifier la présence de **Docker-compose**. Si le terminal affiche un numéro de version, c'est que tout s'est bien passé, sinon lancer la commande pour l'installer.

```
# sudo docker-compose -v  # affiche la version
# sudo apt install docker-compose # installation de docker-compose
```

b) Créer le fichier **docker-nginx.yml** et copier/coller le script ci-dessous dedans

```
version: '3'
services:
   web:
     image: nginx:latest
     ports:
       - "8080:80"
     volumes:
       - ./src:/usr/share/nginx/html
     networks:
       macvlan-network:
          ipv4_address: 192.168.xxx.xxx  #Adresse IP de notre conteneur
      links:
       - php
   php:
     image: php:7-fpm
```

c) Pousser le fichier **docker-nginx.yml** via **docker-compose**

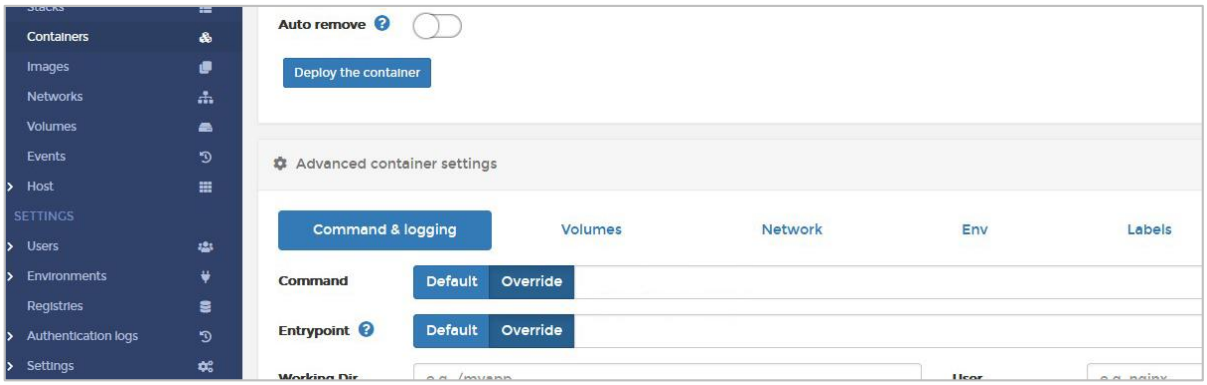
```
# sudo docker-compose -f --net macvlan-network docker-nginx.yml up --detach
```

d) Se connecter à **Portainer** pour vérifier que tout s'est bien passé

6. Création d'un conteneur persistant

Certains conteneurs créés ne sont plus accessibles via la console, car ils se déconnectent. Pour pallier à ce problème, on ajouter une commande dans le conteneur.

a) Editer le conteneur
b) Descendre jusqu'à la section **Advanced container settings**
c) Dans le champ **command**, saisir le texte **'sleep' 'infinity'**



d) Cliquer sur le bouton **Deploy the container**

7. Sécurisation de Docker avec les User Namespace

Par défaut Les conteneurs Docker s'exécutent avec le **compte root**, de même que les programmes qui s'exécutent à l'intérieur du conteneur. Pour supprimer ce trou de sécurité, on doit créer un **utilisateur non root** pour gérer ces conteneurs. **Attention, cet utilisateur n'aura plus les privilèges nécessaires pour gérer Portainer**.

a) Créer un fichier **create-username-dockremap.sh** contenant le script ci-dessous

```
#!/bin/bash

#####################################################################
#  DESCRIPTION:  création d'un user spécifique pour docker
#####################################################################

groupadd -g 500000 dockremap &&
groupadd -g 501000 dockremap-user &&
useradd -u 500000 -g dockremap -s /bin/false dockremap &&
useradd -u 501000 -g dockremap-user -s /bin/false dockremap-user

echo "dockremap:500000:65536" >> /etc/subuid &&
echo "dockremap:500000:65536" >>/etc/subgid
echo "
{
  \"userns-remap\": \"default\"
}
  " > /etc/docker/daemon.json

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

b) Exécuter le script

```
# sudo chmod +x create-username-dockermap.sh
# sudo ./create-username-dockremap.sh
```

c) Vérifier le changement d'utilisateur sur le service

```
# sudo ps aux | grep docker
# sudo cat /etc/subuid |grep dockremap  # affiche l'utilisateur non root
```

d) Le fichier **/etc/docker/daemon.json** a été créé et contient l'utilisateur non root par default de docker. Pour revenir à l'utilisateur **root**, réouvrir le fichier **daemon.json** renommer la ligne **userns-remap : "default"** en **userns-remap : "user:group"** ou supprimer-là.

e) Restreindre les communications entre les containers sur l'interface bridge créé par Docker

- Ouvrir le fichier **daemon.json**
- Ajouter cette ligne **{ "icc": false }**

8. Installation de Grafana

Grafana est une plateforme open source taillée pour la surveillance, l'analyse et la visualisation des métriques IT. Elle est livrée avec un serveur web (écrit en Go) permettant d'y accéder via une API HTTP. Sous licence Apache 2.02, Grafana génère ses graphiques et tableaux de bord à partir de bases de données de séries temporelles (time series database) telles que Graphite, InfluxDB ou OpenTSDB. Cette plateforme est aussi un outil indispensable pour créer des alertes.

Véritable éditeur de dashboards informatiques, Grafana permet également de les partager sous forme de snapshot (ou instantanés) avec d'autres utilisateurs.

a) Installer **Grafana** via Docker

```
$ sudo docker pull grafana/grafana
$ sudo docker run -d --name=grafana -p 3000:3000 grafana/grafana
```

b) Attribuer un réseau **macvlan** à Grafana

c) Se connecter avec les identifiants admin/admin

d) Changer le mot de passe

9. Installation de Duplicati

Duplicati est un logiciel de sauvegarde gratuit permettant de faire des sauvegardes. Les sauvegardes de fichiers et dossiers peuvent être locales ou distantes. Bien entendu, il est possible de chiffrer les données avant de les transmettre (AES-256 ou GPG au choix). Il permet de faire des sauvegardes incrémentielles et propose également la déduplication des données afin de gagner en espace de stockage. Le logiciel dispose d'une fonction de vérification des fichiers de sauvegarde, il est à même de réaliser des instantanés (VSS sous Windows / LVM sous Linux). Il gère également la sauvegarde de fichiers ouverts. Enfin, Duplicati dispose de son propre planificateur de tâches.

a) Installer **Duplicati** via Docker

```
$ sudo docker pull duplicati/duplicati
$ sudo docker run -p 8200:8200 -v /some/path:/some/path duplicati/duplicati
```

b) Se connecter à Duplicati avec l'url : **@IPServeur:8200**

10. Installation de watchtower

Watchtower est un conteneur qui va surveiller vos conteneurs à la recherche de mise à jour disponible. Si une mise à jour est disponible, alors Watchtower, arrête le conteneur, récupère la nouvelle image, et recréé le conteneur avec la nouvelle image.

a) Installer watchtower via **Docker**

```
docker run -d \
    --name watchtower \
    --restart=unless-stopped \
    -e WATCHTOWER_SCHEDULE="0 0 0 * * *" \
    -e WATCHTOWER_CLEANUP="true" \
    -e TZ="Europe/paris" \
    -v /var/run/docker.sock:/var/run/docker.sock \
    containrrr/watchtower \
```

b) Pour une **mise à jour sélective**
- ajouter une ligne à la fin du script avec : **containername1 contenairname2 containername3**….

c) Pour une **mise à jour par label**, ajouter la ligne **-e WATCHTOWER_LABEL_ENABLE="true" ** avant la ligne TimeZone. Ensuite ajouter le label suivant au conteneur via Portainer : **com.centurylinklabs.watchtower.enable=true**

Attention le paramètre Watchtower_Schedule est au format : seconde | minutes | heures | jour du mois | mois | jour de la semaine

11. Installation de Adguard Home

Adguard Home est une est solution permettant le blocage de publicités sur réseau. Il fonctionne tel un serveur DNS primaire capable de filtrer et bloquer des publicités. Inutile d'avoir un bloqueur de publicité sur notre ordinateur, smartphone, console, TV… Adguard Home s'occupe de tout, car il fonctionne avec les appareils connectés sur votre réseau.

a) Copier le script ci-dessous dans un nouveau stack

```
version: "2"
services:
  adguardhome:
    image: adguard/adguardhome
    container_name: adguardhome
    ports:
      - 53:53/tcp
      - 53:53/udp
      - 784:784/udp
      - 853:853/tcp
      - 3000:3000/tcp
      - 80:80/tcp
      - 443:443/tcp
    volumes:
      - ./work:/opt/adguardhome/work
      - ./conf:/opt/adguardhome/conf
  restart: unless-stopped
```

b) Attribuer un réseau **macvlan** à Adguard Home
c) Configurer l'interface d'écoute, comme sur l'image ci-dessous

12. Liens annexes

Liste de contenu à télécharger pour Docker et Portainer
- Calculateur d'@IP/Sous réseau
- Tableau Ipv4 CDR
- Commandes docker
- Docker et la sécurité

13. Conclusion

DOCKET et **PORTAINER** ont chacun leurs fonctionnalités. On peut choisir de travailler en ligne de commande ou en interface web pour obtenir le même résultat.

Destiné au RaspberryPi (Raspbian) avec Docker et Portainer, on peut parfaitement installer plusieurs conteneurs qui fonctionneront parfaitement. Attention tout de même à ne pas surcharger les performances du serveur.