

**Déploiement manuel des conteneurs Apache + MariaDB**

Effectué par

**Théo CANOUEL, Clément MAROY et Adrien SAINSARDOS**

1. **Vérification des conteneurs existants**

Il faut en premier lieu utiliser la commande « docker ps -a --format '{{.Names}}' » afin d’obtenir une liste des noms de tous les conteneurs existants.

Il faut ensuite vérifier les numéros des conteneurs afin de savoir lequel associer à nos nouveaux conteneurs.

Exemple :

$ docker ps -a --format '{{.Names}}'  
reverse-proxy  
webserver3  
webserver2  
webserver1  
db3  
db2  
db1

Ici, il faudra créer webserver4 et db4.

1. **Création des conteneurs**

**Création du nouveau réseau :**

Afin d’isoler notre paire de conteneurs des conteneurs préexistants, il faut créer un nouveau réseau à l’aide de la commande « docker network create webserver??-network ».

Pour reprendre notre exemple, cela donnerait : « docker network create webserver4-network »

**Création des conteneurs :**

Il faudra ensuite créer nos conteneurs dans ce nouveau réseau avec les commandes : « docker run -d --name db?? --network webserver??-network -e MARIADB\_ROOT\_PASSWORD=michel -e DB\_INSTANCE=db?? -v db??-data:/var/lib/mysql my-mariadb »

« docker run -d --name webserver?? --network webserver??-network -e DB\_HOST=db?? -e DB\_NAME=webserver??db -p 8083:80 my-webserver”

**Modification du reverse-proxy :**

Il y a plusieurs modifications à faire sur le reverse-proxy afin de faire fonctionne le nouveau serveur web. La première est d’ajouter le nouveau réseau créé au reverse-proxy :

“docker network connect webserver??-network reverse-proxy”

Ensuite, il faut modifier sa configuration.

Il faut donc se connecter au reverse-proxy :

“docker exec -it reverse-proxy sh”

La première commande à effectuer est :

“sed -i '/server webserver??:80;/a\ server webserver??:80;' /etc/nginx/conf.d/default.conf”

Pour notre exemple :

“sed -i '/server webserver3:80;/a\ server webserver4:80;' /etc/nginx/conf.d/default.conf”

Puis la seconde est :

“sed -i '$i\\n\ # Specific route for webserver??\n\ location /server??/ {\n\ proxy\_pass http://webserver??:80/;\n\ proxy\_set\_header Host $host;\n\ proxy\_set\_header X-Real-IP $remote\_addr;\n\ proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;\n\ proxy\_set\_header X-Forwarded-Proto $scheme;\n\ }' /etc/nginx/conf.d/default.conf”

Pour notre exemple :

“sed -i '$i\\n\ # Specific route for webserver4\n\ location /server4/ {\n\ proxy\_pass http://webserver4:80/;\n\ proxy\_set\_header Host $host;\n\ proxy\_set\_header X-Real-IP $remote\_addr;\n\ proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;\n\ proxy\_set\_header X-Forwarded-Proto $scheme;\n\ }' /etc/nginx/conf.d/default.conf”

Enfin, pour valider la configuration, il faut effectuer la commande suivante, toujours dans le conteneur reverse-proxy :

« nginx -s reload »