1. Activité : Installation de Docker sur un OS Debian/Ubuntu

1.1. Installation de Docker

Lancer le terminal et tapez les commandes suivantes :

```
$ curl -fsSL https://get.docker.com -o get-docker.sh
$ sudo sh get-docker.sh
```

1.1.1. Autre démarche

- Installer les pré-requis

```
# apt -y update && apt -y upgrade
# apt -y install apt-transport-https ca-certificates curl gnupg2 software-properties-common
```

- Installation de Docker :
  - importation de la signature du dépôt
  - Ajout du dépôt
  - Installation de Docker

```
# curl -fsSL https://download.docker.com/linux/debian/gpg | apt-key add -
# add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/debian $(lsb_release -cs)
stable"
# apt update
# apt install -y docker-ce docker-ce-cli containerd.io
```

1.1.2. Autoriser un compte à utiliser Docker

Pour permettre à votre compte d'**utiliser Docker comme un utilisateur non-root** il faut ajouter **l'ajouter** au groupe **docker** avec la commande suivante (pour le compte btssio) :

```
$ sudo usermod -aG docker btssio
```

Il est ensuite nécessaire de **se déconnecter puis de se reconnecter** pour que l'appartenance à ce nouveau groupe soit effective.

1.2. Vérification de Docker

- visualisez la version de Docker avec la commande suivante :

```
btssio@ubuntdocker:/$ docker -v
Docker version 18.09.5-ce, build e8ff056
```

- Lancez un premier conteneur avec la commande suivante :

```
btssio@ubuntudocker:~$ docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
1b930d010525: Pull complete
Digest: sha256:92695bc579f31df7a63da6922075d0666e565ceccad16b59c3374d2cf4e8e50e
Status: Downloaded newer image for hello-world:latest

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.
```

```
To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/

btssio@ubuntudocker:~$
```

1.3. Vérification de Python

Depuis le terminal, vérifiez que Python 2.7 est opérationnel :

- visualisez la version de Python avec la commande suivante :

```
$ python
Python 2.7.15rc1 (default, Nov 12 2018, 14:31:15)
[GCC 7.3.0] on linux2
Type "help", "copyright", "credits" or "license" for more information.
>>>
```

Si Python 2.7 n'est pas installé faites-le :

```
$ sudo apt -y install python
```

- vérifiez que pip est installé :

```
$ pip -V
pip 9.0.1 from /usr/lib/python2.7/dist-packages (python 2.7)
```

Si pip n'est pas installé faites-le, puis installez ipaddress:

```
$ sudo apt -y install python-pip
```

Installez ipaddress:

```
$ sudo -H pip install ipaddress
```

1.4. Gestion des interfaces réseaux avec Docker

Votre serveur possède maintenant plusieurs types d'interface réseau :

```
btssio@ubuntudocker:~$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
    valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
    valid_lft forever preferred_lft forever
2: enp0s3: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UP group default qlen 1000
    link/ether 08:00:27:af:88:bf brd ff:ff:ff:ff:ff:ff
    inet 192.168.1.193/24 brd 192.168.1.255 scope global dynamic enp0s3
    valid_lft 86273sec preferred_lft 86273sec
    inet6 fe80::a00:27ff:feaf:88bf/64 scope link
    valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:5a:da:db:4a brd ff:ff:ff:ff:ff:ff
    inet 172.17.0.1/16 brd 172.17.255.255 scope global docker0
    valid_lft forever preferred_lft forever
btssio@ubuntudocker:~$
```

Les interfaces réseaux créées par défaut :

- **lo** : interface de **bouclage** (loopback) avec l'adresse IP 127.0.0.1/8,
- **enps03** : interface **physique** avec l'adresse IP obtenue par votre box Internet compte tenu du mode d'accès au réseau par pont (bridge) de VirtualBox. Ici il s'agit de 192.168.1.193/24 mais cela doit être différent sur votre propre réseau local.

L'interface créée par Docker :

- **docker0** : interface qui est un pont (bridge) et dont l'adresse IP est 172.17.0.1/16.

Docker gère trois types de réseaux :

```
btssio@ubuntudocker:~$ docker network ls
NETWORK ID          NAME                DRIVER              SCOPE
e7596d1a2552        bridge              bridge              local
c28efeb7848e        host                host                local
708458e85954        none                null                local
btssio@ubuntudocker:~$
```

Les identifiants NETWORK ID sont générés à l'installation de Docker sont différents d'un ordinateur à l'autre. Les autres informations sont identiques : NAME, DRIVER et SCOPE.

Le réseau **BRIDGE** et le réseau par **défaut**, réseau sur lequel sont connectés les conteneurs créés.

Pour **connecter** un conteneur sur un autre réseau, il faut alors utiliser le paramètre **–network**.

2. Retour Accueil Docker

- Docker

From:
https://siocours.lycees.nouvelle-aquitaine.pro/ - **Les cours du BTS SIO**

Permanent link:
https://siocours.lycees.nouvelle-aquitaine.pro/doku.php/reseau/docker/installationdockerlinux

Last update: **2026/05/15 16:03**