Docker

Sommaire :

I. Objectifs :

A. Installer Docker

- Mettre à jour la liste des paquets disponibles pour Debian Buster.
- Procéder à l'installation de Docker sur le système Debian.

B. Configurer Docker

- Inclure l'utilisateur dans le groupe Docker pour lui permettre d'exécuter des commandes Docker sans utiliser sudo.

C. Tester Docker en créant des conteneurs

- Télécharger une image Docker depuis le registre Docker Hub.
- Créer un conteneur à partir de l'image téléchargée pour vérifier le bon fonctionnement de Docker.

D. Installer Apache

- Installer le serveur web Apache sur Debian Buster à l'aide de la gestion des paquets.

E. Installer Nginx en le personnalisant

- Installer le serveur web Nginx sur Debian Buster en utilisant la gestion des paquets.
- Personnaliser la configuration de Nginx selon les besoins spécifiques du projet ou de l'application.

II. Qu’est-ce que Docker ?

Docker est une plateforme open-source qui permet de développer, déployer et exécuter des applications dans des conteneurs. Un conteneur **Docker** encapsule une application avec tous ses dépendances logicielles, ce qui permet à l'application de s'exécuter de manière cohérente et isolée sur n'importe quel environnement compatible avec **Docker**, qu'il s'agisse d'un ordinateur portable, d'un serveur physique ou d'un cloud public.

En utilisant **Docker**, les développeurs peuvent packager leurs applications avec toutes leurs dépendances dans des conteneurs légers et portables. Cela facilite le déploiement et la gestion des applications, car les conteneurs **Docker** garantissent la cohérence entre les environnements de développement, de test et de production.

III. Installation de Docker

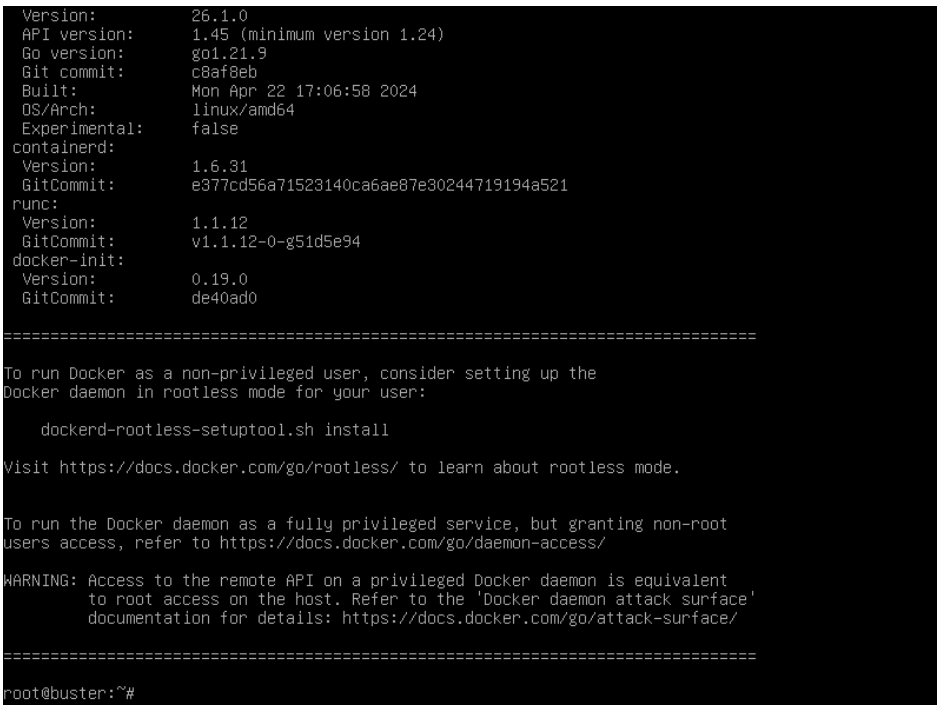
apt update

wget https://get.docker.com/

```
root@buster:~# wget https://get.docker.com/
--2024-04-23 11:58:35--  https://get.docker.com/
Résolution de get.docker.com (get.docker.com)… 52.84.174.2, 52.84.174.92, 52.84.174.106, ...
Connexion à get.docker.com (get.docker.com)|52.84.174.2|:443… connecté.
requête HTTP transmise, en attente de la réponse… 200 OK
Taille : 21927 (21K) [text/plain]
Sauvegarde en : « index.html »

index.html              100%[===================================>]  21,41K  --.-KB/s    ds 0,005s

2024-04-23 11:58:35 (3,98 MB/s) - « index.html » sauvegardé [21927/21927]

root@buster:~#
```

Bash Index.html

(attendre quelques minutes)



Bien désactiver le proxy avant d'effectuer la commande : **nano /etc/apt/apt.conf**

systemctl status docker.service

Cette commande est utile pour **vérifier** rapidement l'état du service **Docker** et déterminer s'il fonctionne correctement sur votre système Debian Buster.

IV. Installation d’un conteneur

Maintenant que Docker est installé et opérationnel, vous pouvez créer un ou plusieurs conteneurs à partir d'une image spécifique. Pour ce faire, commencez par télécharger l'image souhaitée à partir du site **Docker Hub**.

Par exemple, si vous souhaitez télécharger l'image de **DokuWiki**, un site web collaboratif, vous pouvez trouver cette image sur le site **Docker Hub** en suivant ce lien: **https://hub.docker.com/**. Une fois que vous avez trouvé l'image recherchée, utilisez la commande '**docker pull**' suivie du nom de l'image pour télécharger l'image sur votre système local.

docker pull mprasil/dokuwiki

```
root@buster:~# docker pull mprasil/dokuwiki
Using default tag: latest
latest: Pulling from mprasil/dokuwiki
2b55860d4c66: Pull complete
2d785b3b2b4c: Pull complete
e8c2ce754318: Pull complete
a016a328fe36: Pull complete
74d3af8881ec: Pull complete
f7e015da3e46: Pull complete
1b00e5cae740: Pull complete
c7e2977ff067: Pull complete
Digest: sha256:507db13c7bda01572e86fc829eb32b447328b28a0dc4314b908a6987eb3d9efe
Status: Downloaded newer image for mprasil/dokuwiki:latest
docker.io/mprasil/dokuwiki:latest
root@buster:~# _
```

Une fois l'image téléchargée et installée théoriquement, vous pouvez vérifier sa présence dans le dépôt d'images locales à l'aide de la commande 'docker images'. Cette commande vous affichera la liste de toutes les images **Docker** installées sur votre système, y compris l'image '**mprasil/dokuwiki**' si elle a été téléchargée avec succès.

docker images

```
root@buster:~# docker images
*REPOSITORY          TAG       IMAGE ID       CREATED         SIZE
mprasil/dokuwiki    latest    78c138e0b03c   16 months ago   319MB
root@buster:~# *
```

```
root@buster:~# docker run -d -p 8001:80 --name wiki01 mprasil/dokuwiki
7556305e00c0fcd1af1f931164b199edbe26cfd41b998964c640f3fe6b958e31
```

‘-d’ : Indique un lancement en arrière plan

‘-p 8001:80’ : redirige le port 8001 de la machine physique vers le port 80 du conteneur

‘–name wiki01’ : Affecte un nom au conteneur. En l’occurrence Wiki01

‘mprasil/dokuwiki’ : Nom de l’image sur le hub docker

V. Commandes Utiles

Mode Console :

docker exec –it wiki01 bash

Après avoir exécuté cette commande, vous serez placé dans le mode console du conteneur, où vous pourrez exécuter des commandes comme si vous étiez à l'intérieur d'une machine virtuelle ou d'un système d'exploitation physique. Pour quitter le mode console du conteneur, vous pouvez simplement taper la commande exit.

‘docker exec’: C'est la commande Docker pour exécuter une commande à l'intérieur d'un conteneur existant.
‘-it’: Ces options rendent l'interaction avec le conteneur interactive et fournissent un terminal pseudo-TTY.
‘wiki01’: C'est le nom ou l'ID du conteneur dans lequel vous souhaitez exécuter la commande.
‘bash’: C'est la commande à exécuter à l'intérieur du conteneur. Dans ce cas, il s'agit du shell Bash.

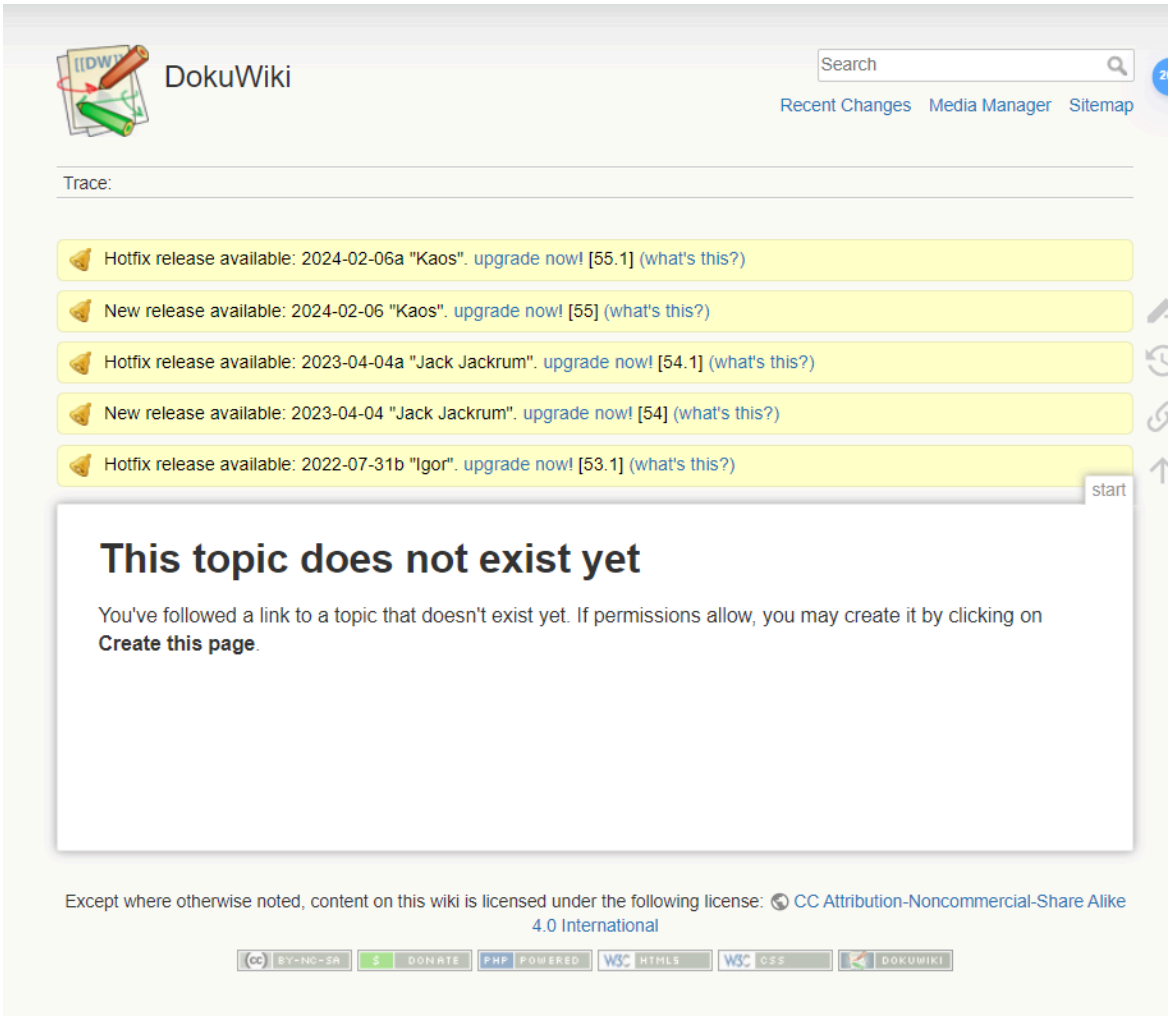
Pour vérifier sa création et/ou la présence des autres conteneurs, on utilise la commande **docker ps -a** permettant d'afficher tous les conteneurs en cours d’exécution. En revanche la commande **docker container ls -a** permet d’afficher tous les conteneurs existant.

```
root@buster:~# docker ps -a
CONTAINER ID   IMAGE              COMMAND             CREATED          STATUS          PORTS
                     NAMES
7556305e00c0   mprasil/dokuwiki   "/startup.sh run"   15 minutes ago   Up 15 minutes   0.0.0.0:8001-
>80/tcp, :::8001->80/tcp   wiki01
```

On peut démarrer un conteneur avec **docker start NOM_CONTENEUR**

VI. Vérification :

Dans un navigateur de la machine hôte ou cliente 192.168.56.30:8001



Test d'installation d'Apache

On crée un conteneur avec l'image ubuntu sur le port 8002. Il installa automatiquement l'image ubuntu comme convenu.

```
root@buster:~# docker run -d -p 8002:80 --name apache ubuntu
Unable to find image 'ubuntu:latest' locally
latest: Pulling from library/ubuntu
3c645031de29: Pull complete
Digest: sha256:1b8d8ff4777f36f19bfe73ee4df61e3a0b789caeff29caa019539ec7c9a57f95
Status: Downloaded newer image for ubuntu:latest
2396e740158982728e82ce2ed249f08eb7f06655304f04427884ce7484373b04
```

Nous allons lancer l'image avec

docker run ubuntu

Vérifier les conteneurs actifs avec

docker ps -a

```
root@buster:~# docker ps -a
CONTAINER ID   IMAGE              COMMAND                  CREATED          STATUS                      P
ORTS                                    NAMES
2396e7401589   ubuntu             "/bin/bash"              38 seconds ago   Exited (0) 36 seconds ago
                                        apache
7556305e00c0   mprasil/dokuwiki   "/startup.sh run"        48 minutes ago   Up 48 minutes               0
.0.0.0:8001->80/tcp, :::8001->80/tcp   wiki01
```

docker run ubuntu cat /etc/lsb-release

Vous pouvez utiliser la commande **docker run** pour lancer un conteneur en lui faisant exécuter une commande spécifique. Dans cet exemple, nous lançons un conteneur à partir de l'image Ubuntu et nous lui demandons d'exécuter la commande **cat /etc/lsb-release**.

```
root@buster:~# docker run ubuntu cat /etc/lsb-release
DISTRIB_ID=Ubuntu
DISTRIB_RELEASE=22.04
DISTRIB_CODENAME=jammy
DISTRIB_DESCRIPTION="Ubuntu 22.04.4 LTS"
root@buster:~# docker run -ti ubuntu bash
```

Pour entrer dans le conteneur :

```
root@buster:~# docker run -ti ubuntu bash
root@95e9e2d9f6d1:/#
```

Un conteneur **Docker** est une unité d'exécution logicielle légère et autonome qui contient tout le nécessaire pour exécuter une application, y compris le code, les bibliothèques système, les outils système et les dépendances. Les conteneurs **Docker** utilisent la technologie de virtualisation au niveau du système d'exploitation pour isoler et encapsuler les applications, ce qui permet de les **exécuter** de manière cohérente et isolée sur n'importe quel environnement compatible avec **Docker**.

Dans le conteneur, faire:

apt update
apt install apache2

VII. Créer ou récupérer l'image d'une archive

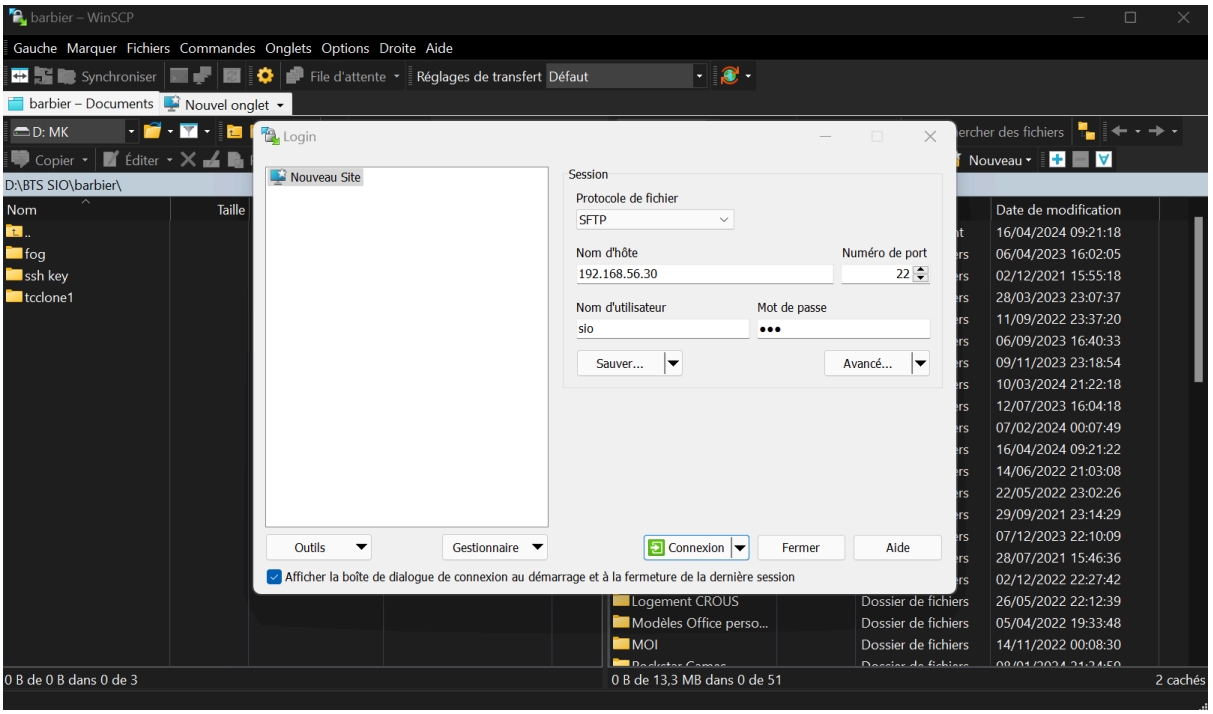
Pour commencer, il faut compresser l'image au format **tar** avec la commande suivante

docker save -o dokiwiki.tar mprasil/dokuwiki

Bien le faire dans le dossier /home/sio pour le récupérer après.

```
root@buster:/home/sio# docker save -o dokuwiki.tar mprasil/dokuwiki
root@buster:/home/sio# ls
dokuwiki.tar
```

Pour pouvoir récupérer le fichier, nous allons utiliser **Winscp.**



En cas de problème de droits avec l'archive sur le profil **sio**, utilisez cette commande :

chown sio:sio dokuwiki.tar

Vous pourrez ensuite vérifier la modification des droits avec :

ls -l

VIII. Nginx

A. Qu'est-ce que Nginx ?

Nginx est un serveur web open-source très performant, souvent utilisé comme **serveur proxy inverse**, équilibreur de charge **HTTP**, **cache web**, et plus encore. Conçu pour gérer un grand nombre de connexions simultanées, il est **réputé** pour sa faible utilisation de ressources système et sa capacité à gérer efficacement des charges de trafic élevées.

Nginx est un serveur web puissant et polyvalent largement utilisé pour sa haute performance, sa fiabilité et sa capacité à gérer efficacement des charges de trafic importantes sur le web moderne.

B. Procédure d'installation

On installe l'image **Nginx** avec la commande suivante :

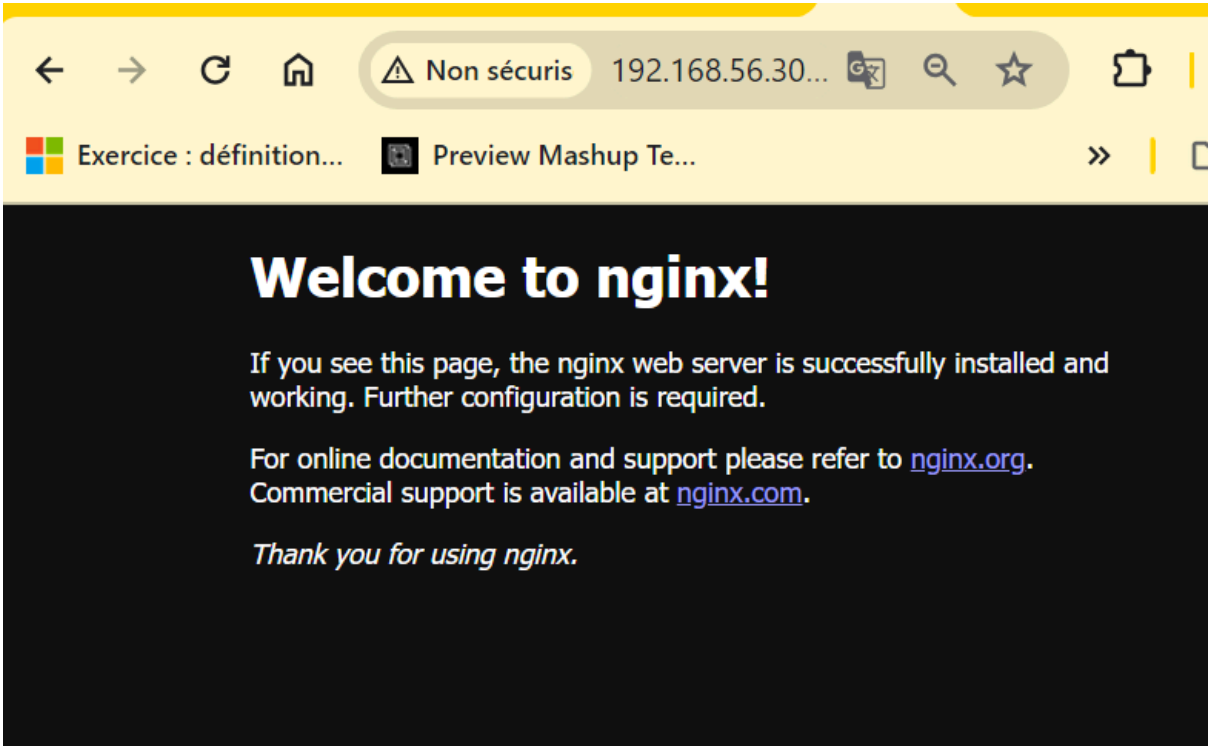
docker pull nginx

```
root@buster:/# docker pull nginx
Using default tag: latest
latest: Pulling from library/nginx
13808c22b207: Downloading  6.859MB/29.13MB
ad34559c894f: Downloading  6.839MB/41.81MB
e871086e0ba8: Download complete
1cdf2e0600bb: Download complete
c791c3089e36: Waiting
5941304296e7: Waiting
c7cbceefe40c: Waiting
```

On va ensuite créer un conteneur comme précédemment :

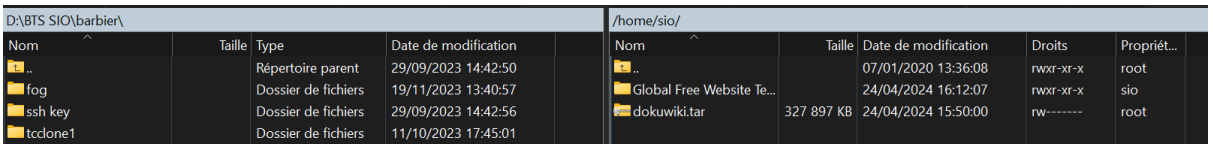
```
root@buster:/# docker run -d -p 9901:80 --name nginx01 nginx
ed7f80a1ab6ddbdc9e74c8bd8e0f44fb8ab1228a456c8a1efb249c0553f424dd
```

On peut vérifier que Nginx fonctionne bien avec le navigateur .



C. Personnalisation de la page

Nous allons utiliser un template pour personnaliser la page html.



On déplace ensuite le dossier concerné avec la commande :

mv DOSSIER /root

On va procéder au déplacement du répertoire **Template** de la machine debian au conteneur Nginx avec la commande suivante :

```
root@buster:~# docker cp /root/template/ ed7f80a1ab6d:/root
Successfully copied 1.72MB to ed7f80a1ab6d:/root
```

On se connecte ensuite au conteneur :

docker exec -it nginx /bin/bash

On va commencer par mettre à jour les paquets dans le conteneur avec apt update

Ensuite, Pour que le **template** soit pris en charge, il faut supprimer le fichier **index.html** par défaut :

find / -name index.html

La commande précédente permet de trouver le fichier qu'on cherche.

```
root@ed7f80a1ab6d:/usr/share/nginx/html# find / -name index.html
find: '/proc/29/map_files': Permission denied
find: '/proc/30/map_files': Permission denied
/root/template/global-master/index.html
root@ed7f80a1ab6d:/usr/share/nginx/html#
```

On va ensuite supprimer le fichier dans : **/usr/share/nginx/html**

rm index.html

Il faut par la suite, remplacer le fichier supprimer par le template choisi.

Utilisez un simple mv du contenu template pour le déplacer dans **/usr/share/nginx/html**

Nous pouvons vérifier le bon fonctionnement avec l'adresse et le port attribué.