

# INSTALLATION D'UN SERVEUR DOCKER SOUS RASPBERRY PI

Raspberry Pi - Debian Bullseye  
**Configuration de base**

Tutoriel **DOCKER** - RASPBERRY PI

David GOÏTRÉ

## Table des matières

|  |   |
|--|---|
| Introduction .....                             | 1 |
| 1. Pré requis .....                            | 1 |
| 2. Connexion au serveur .....                  | 1 |
| 3a. Paramétrage Ethernet du serveur .....      | 2 |
| 3b. Paramétrage Wifi du serveur .....          | 3 |
| 4. Optimisation du système.....                | 3 |
| 5. Installation de Docker.....                 | 3 |
| 6. Vérification du Docker .....                | 4 |
| 7. Désinstallation de Docker .....             | 5 |
| 8. Installation de Portainer pour Docker ..... | 5 |
| 9. Configuration de Portainer .....            | 6 |
| 10. Mise à jour de Portainer .....             | 7 |
| 11. Installation Portainer 1.24 .....          | 7 |
| 12. Erreurs Docker et Portainer .....          | 8 |
| 13. Commandes RaspberryPi.....                 | 8 |
| 14. Liens annexes .....                        | 8 |
| 15. Conclusion .....                           | 9 |

## Introduction

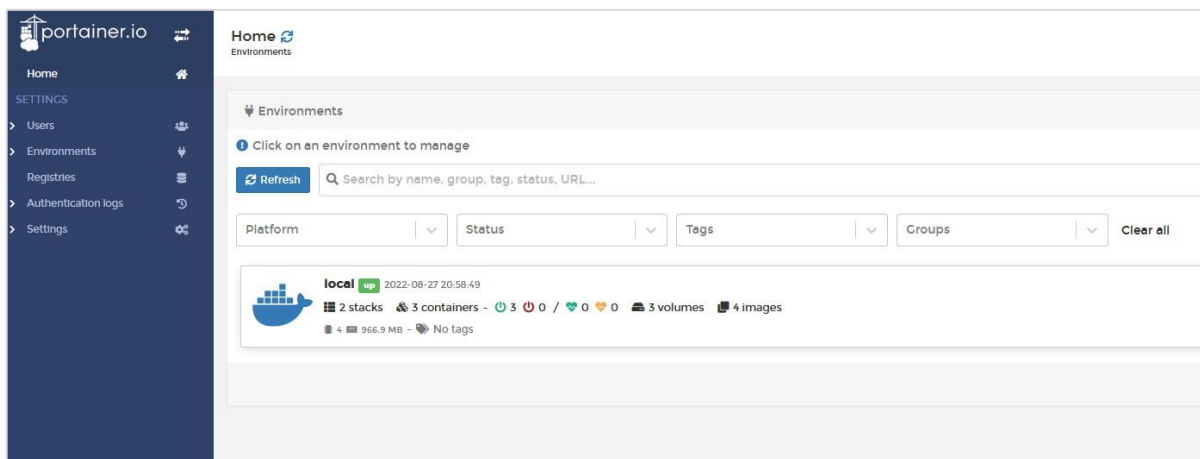
**DOCKER 2.0** est un logiciel libre Open Source qui permet d'automatiser le déploiement d'applications. Il a été développé par Solomon Hykes de la société dotCloud et a été distribué à partir de mars 2013. C'est une plateforme de virtualisation par conteneur qui va permettre de concevoir, tester et déployer des applications rapidement. Grâce à Docker, il est facile de déployer et dimensionner les applications dans n'importe quel environnement en s'assurant que le code s'exécutera automatiquement.

## 1. Pré requis

On a besoin des différents matériels et logiciels pour la création d'un Serveur DOCKER 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 VPN
- 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 **DOCKER** 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/dhcpd.conf # ouvre le fichier de configuration réseau
```

c) Copier le texte ci-dessous à la fin du fichier **dhcpd.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
```

```
Raspberry Pi Software Configuration Tool (raspi-config)
1 System Options          Configure system settings
2 Display Options        Configure display settings
3 Interface Options      Configure connections to peripherals
4 Performance Options    Configure performance settings
5 Localisation Options   Configure language and regional settings
6 Advanced Options       Configure advanced settings
8 Update                 Update this tool to the latest version
9 About raspi-config     Information about this configuration tool
```

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

```
Raspberry Pi Software Configuration Tool (raspi-config)
S1 Wireless LAN          Enter SSID and passphrase
S2 Audio                 Select audio out through HDMI or 3.5mm jack
S3 Password              Change password for the 'pi' user
S4 Hostname              Set name for this computer on a network
S5 Boot / Auto Login     Select boot into desktop or to command line
S6 Network at Boot      Select wait for network connection on boot
S7 Splash Screen        Choose graphical splash screen or text boot
S8 Power LED             Set behaviour of power LED
```

### 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. Optimisation du système

Si on utilise le Raspberry Pi pour DOCKER sans écran connecté, il est recommandé d'affecter le minimum de RAM à la partie vidéo. Il suffit de se connecter en SSH et d'éditer le fichier config

```
# sudo nano /boot/config.txt
```

c) Ajouter ou modifier les lignes du fichier config, comme ci-dessous :

```
gpu_mem=16
disable_l2cache=0
gpu_freq=250
```

d) Rebooter le Raspberry

### 5. Installation de Docker

Par défaut, le paquet **DOCKER** est disponible dans le référentiel de Debian Bullseye, mais celui-ci n'est pas souvent mise à jour. On va utiliser le dépôt officiel de Docker pour être sûr de travailler sur la dernière version.

a) Installer les packages nécessaires avec les commandes suivantes

```
# sudo apt-get remove docker docker-engine docker.io containerd runc
# sudo apt update --fix-missing
# sudo curl -fsSL https://get.docker.com -o get-docker.sh
# sudo sh get-docker.sh
ou
# sudo curl -sSL https://get.docker.com | sh
ou
# sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

## 6. Vérification du Docker

### a) Vérifier l'installation finale

```
# sudo docker version
```

```
pi@raspberrypi:~$ sudo docker version
Client: Docker Engine - Community
 Version: 20.10.17
 API version: 1.41
 Go version: go1.17.11
 Git commit: 100c701
 Built: Mon Jun 6 23:04:01 2022
 OS/Arch: linux/arm
 Context: default
 Experimental: true

Server: Docker Engine - Community
 Engine:
  Version: 20.10.17
  API version: 1.41 (minimum version 1.12)
  Go version: go1.17.11
  Git commit: a89b842
  Built: Mon Jun 6 23:04:58 2022
```

### b) Vérifier le statut du Docker

```
# sudo systemctl status docker
```

```
linuxtechi@debian11:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
   Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabled)
   Active: active (running) since Sun 2021-10-17 01:40:11 EDT; 2min 36s ago
   TriggeredBy: ● docker.socket
   Docs: https://docs.docker.com
   Main PID: 3458 (dockerd)
   Tasks: 7
   Memory: 32.2M
   CPU: 589ms
   CGroup: /system.slice/docker.service
           └─3458 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

Oct 17 01:40:10 debian11 dockerd[3458]: time="2021-10-17T01:40:10.375509429-04:00" level=info msg="scheme \"unix\" not registered"
Oct 17 01:40:10 debian11 dockerd[3458]: time="2021-10-17T01:40:10.375642801-04:00" level=info msg="ccResolverWrapper: sending update"
Oct 17 01:40:10 debian11 dockerd[3458]: time="2021-10-17T01:40:10.375760109-04:00" level=info msg="ClientConn switching balance"
Oct 17 01:40:10 debian11 dockerd[3458]: time="2021-10-17T01:40:10.487452940-04:00" level=info msg="Loading containers: start."
Oct 17 01:40:10 debian11 dockerd[3458]: time="2021-10-17T01:40:10.808453307-04:00" level=info msg="Default bridge (docker0) is"
Oct 17 01:40:11 debian11 dockerd[3458]: time="2021-10-17T01:40:11.140507568-04:00" level=info msg="Loading containers: done."
Oct 17 01:40:11 debian11 dockerd[3458]: time="2021-10-17T01:40:11.275556065-04:00" level=info msg="Docker daemon" commit=79ea9c5"
Oct 17 01:40:11 debian11 dockerd[3458]: time="2021-10-17T01:40:11.276097600-04:00" level=info msg="Daemon has completed initial"
Oct 17 01:40:11 debian11 systemd[1]: Started Docker Application Container Engine.
Oct 17 01:40:11 debian11 dockerd[3458]: time="2021-10-17T01:40:11.368746882-04:00" level=info msg="API listen on /run/docker.sock"
lines 1-22/22 (END)
```

### c) Exécuter Docker

```
# sudo systemctl start docker
```

### d) Vérifier le fonctionnement du Docker

```
# sudo docker run hello-world
```

```
linuxtechi@debian11:~$ sudo docker run hello-world
Unable to find image 'hello-world:latest' locally
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:37a0b92b08d4919615c3ee023f7ddb068d12b8387475d64c622ac30f45c29c51
Status: Downloaded newer image for hello-world:latest

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

## 7. Désinstallation de Docker

a) Désinstaller docker

```
# sudo apt-get autoremove --purge docker*
```

b) Supprimer les conteneurs et les images stockés

```
# sudo rm -rf /var/lib/docker
```

c) Supprimer les certificats et fichiers de configurations liés à Docker

```
# sudo rm -rf /etc/docker  
# sudo rm /etc/systemd/system/docker.service  
# sudo rm /etc/init/d/docker*
```

d) La finition

```
# sudo apt-get autoremove && apt-get autoclean
```

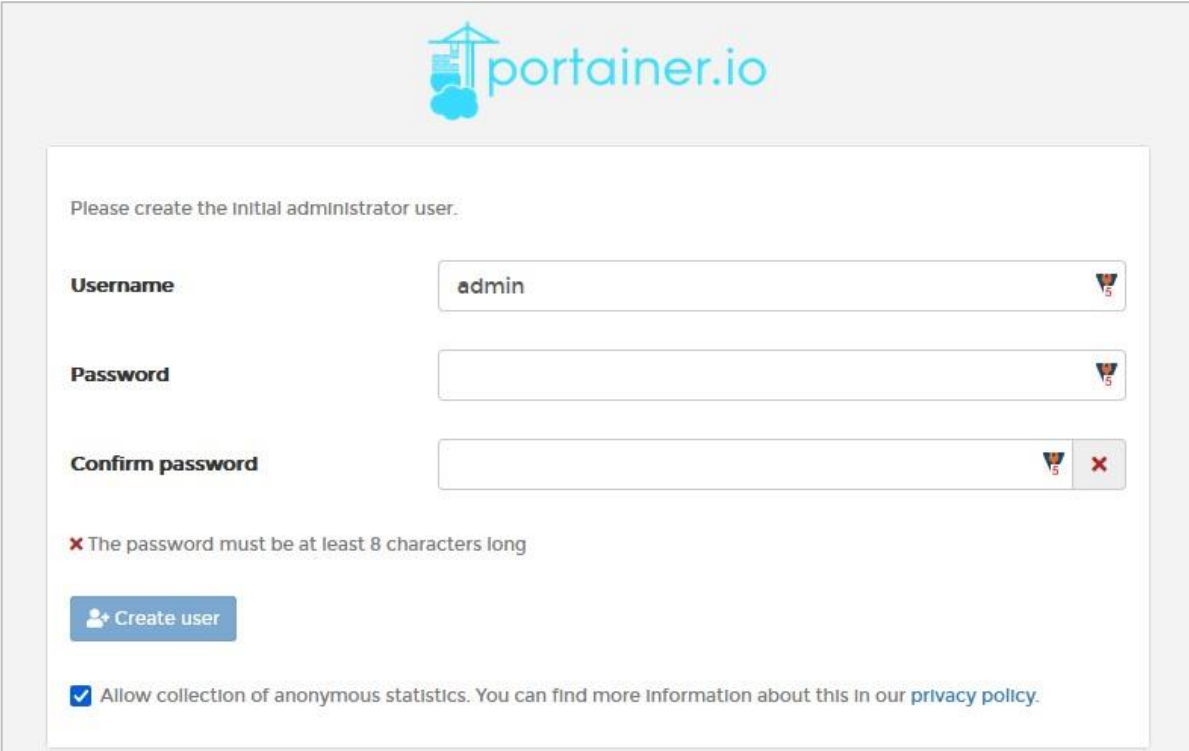
## 8. Installation de Portainer pour Docker

a) Installer l'interface graphique pour l'administration

```
# sudo docker pull portainer/portainer-ce  
# sudo docker volume create portainer_data  
# sudo docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -v  
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-  
ce
```

b) Saisir dans le navigateur : @IPserveur:9000

c) Saisir un mot de passe et cliquer sur le bouton **Create user**

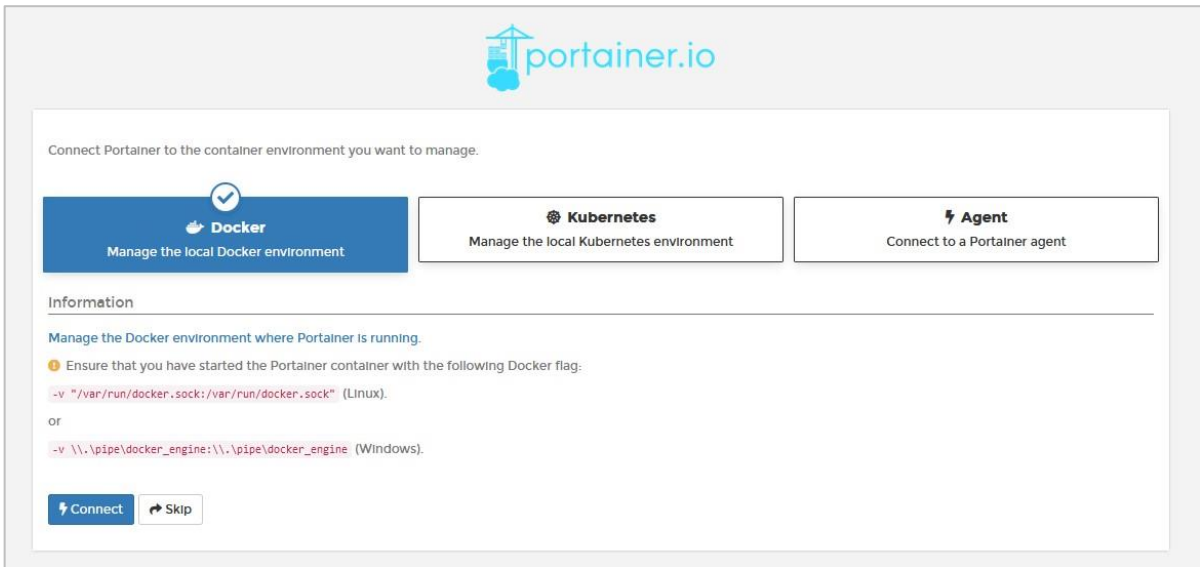


The screenshot shows the Portainer.io user creation interface. At the top, there is the Portainer.io logo. Below it, a message reads "Please create the initial administrator user.". There are three input fields: "Username" with the value "admin", "Password", and "Confirm password". Each field has a strength indicator icon on the right. Below the fields, there is a red error message: "✗ The password must be at least 8 characters long". At the bottom left, there is a blue button labeled "Create user" with a user icon. At the bottom right, there is a checked checkbox with the text "Allow collection of anonymous statistics. You can find more information about this in our [privacy policy](#)."

d) Cliquer sur le bouton **Docker**, puis sur le bouton **Connect**



e) Cliquer sur le bouton **Connect** pour confirmer

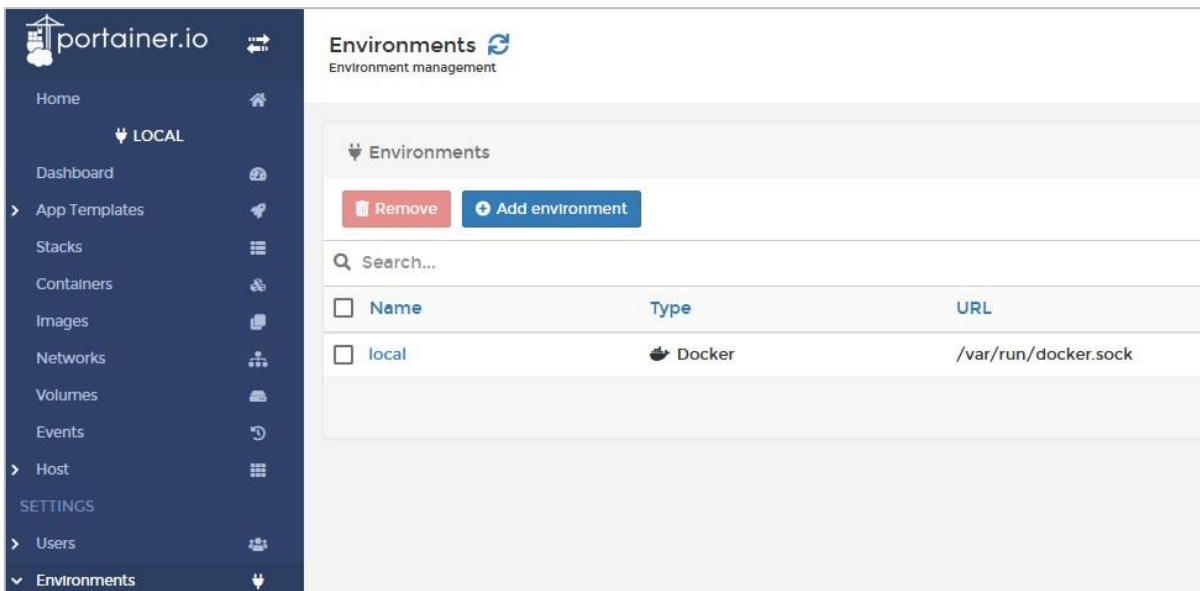


## 9. Configuration de Portainer

Il arrive parfois que Portainer ne se configure pas correctement. Pour éviter les erreurs de fonctionnement, il faut vérifier la configuration de Portainer.

a) Cliquer sur le bouton **Environnement (Endpoints)**

b) Cliquer sur le lien **local** pour éditer l'environnement





c) Saisir dans le champ **Public IP**, l'@IP du serveur Docker. Si celui-ci est vide

The screenshot shows the Portainer.io interface. On the left is a dark blue sidebar with navigation options: Home, SETTINGS, Users, Environments (expanded), Groups, Tags, Registries, Authentication logs, and Settings. The main content area is titled 'Environment details' with a refresh icon and a breadcrumb 'Environments > local'. Below this, there are two sections: 'Configuration' and 'Metadata'. The 'Configuration' section contains three input fields: 'Name' with the value 'local', 'Environment URL' with the value '/var/run/docker.sock', and 'Public IP' with the placeholder 'e.g. 10.0.0.10 or mydocker.mydomain.com'. The 'Metadata' section contains two input fields: 'Group' with the value 'Unassigned' and 'Tags' with the value 'Select...'. At the bottom of the form are two buttons: 'Update environment' (in blue) and 'Cancel' (in white).

d) Saisir des **Metadata Group** et **Tags**, en cas de besoin

e) Cliquer sur le bouton **Update environment**

## 10. Mise à jour de Portainer

a) Pour mettre à jour, nous devons d'abord désinstaller Portainer puis installer la mise à jour

```
# sudo docker ps # Liste les dockers
# sudo docker stop portainer # arrête le docker
# sudo docker rm portainer # supprime le container
# sudo docker pull portainer/portainer-ce
# sudo docker run -d -p 8000:8000 -p 9000:9000 --name=portainer --restart=always -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-
ce
```

## 11. Installation Portainer 1.24

a) Si on veut tester l'ancienne version de Portainer, utiliser les commandes ci-dessous

```
# sudo docker pull portainer/portainer:linux-arm
# sudo docker run --restart always -d -p 9000:9000 -v
/var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data
portainer/portainer/linux:arm
```

## 12. Erreurs Docker et Portainer

Il arrive parfois que des erreurs de connexion, téléchargements, etc... arrivent. On peut rencontrer des messages d'erreurs comme ci-dessous :

```
Error response from daemon: Get "https://registry-1.docker.io/v2/": net/http: request canceled while waiting for connection (Client.Timeout exceeded while awaiting headers)
```

```
Error response from daemon: Get "https://registry-1.docker.io/v2/": context deadline exceeded
```

```
Error response from daemon: Get "https://registry-1.docker.io/v2/": context deadline exceeded (Client.Timeout exceeded while awaiting headers)
```

a) Voici les commandes à effectuer pour régler ces problèmes

```
# sudo docker info | grep Proxy
# sudo systemctl daemon-reload
# sudo systemctl restart docker
# sudo docker logout registry-1.docker.io
# sudo reboot
```

b) Vérifier que les paramètres d'@IP de(s) fichier(s) ci-dessous, soient bien configurés en fonction de votre réseau :

```
# sudo nano /etc/network/interfaces
# sudo nano /etc/resolv.conf
# sudo nano /etc/dhcpd.conf
```

## 13. Commandes RaspberryPi

Liste des commandes basiques à la gestion du serveur RaspberryPi

```
# sudo -i # passe en mode root
# shutdown -h now # éteint le serveur en toute sécurité
# shutdown -r now # redémarre le serveur en toute sécurité
# apt install openssh-server # installe le SSH
# systemctl enable sshd.service # active le service SSH au démarrage
##### Désactive la mise en veille #####
# systemctl mask sleep.target suspend.target hibernate.target hybrid-sleep.target
```

## 14. Liens annexes

Liste de contenu à télécharger pour Docker et Portainer

- [Support et ressources](#)
- [Forum Docker](#)
- [Docker et les containers](#)
- [Installation Docker en ligne de commande](#)
- [Support Portainer](#)

## 15. Conclusion

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

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

Pour **Debian** : <https://www.duhaz.fr/blog/tag/docker>

Pour **Debian** : <https://www.forum-nas.fr/threads/tuto-installation>

Pour **Ubuntu** : <https://www.duhaz.fr/blog/installation-de-docker-sous-linux>