Mise en place

Avant de commencer

Install docker - CLi / .sh

Set up the repository

1. Update the `apt` package index and install packages to allow apt to use a repository over HTTPS:

```
sudo apt-get update
sudo apt-get install \
    ca-certificates \
    curl \
    gnupg \
    lsb-release
```

2. Add Docker’s official GPG key:

```
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/debian/gpg | sudo gpg --dearmor -o
/etc/apt/keyrings/docker.gpg
```

3. Use the following command to set up the repository:

```
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg]
https://download.docker.com/linux/debian \
  $(lsb_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
```

Install Docker Engine

1. Update the `apt` package index:

```
sudo apt-get update
```

2. Install Docker Engine, containerd, and Docker Compose:

```
sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin
```

3. You can verify that Docker Engine is installed correctly by running the `hello-world` image.

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

Install Portainer

1. Create Docker Volume to store the data:

```
docker volume create portainer_data
```

2. Install Portainer Server:

```
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:latest
```

Access Portainer Dashboard

1. In a browser, visit the following address:

```
http://<yourmachineipadress>:9000
```

2. The first time you access Portainer, the system asks to create a password for the admin user. Type the password twice and select the **Create user** button.
3. Select the **Get Started** button to go to the dashboard and start using Portainer in the local environment only.

Install docker with script from `https://get.docker.com`:

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

https://github.com/docker/docker-install

Portainer Apps Templates

Original : https://raw.githubusercontent.com/portainer/templates/master/templates-2.0.json

Technorabilia :

- For Portainer v1: https://raw.githubusercontent.com/technorabilia/portainer-templates/main/lsio/templates/templates-1.20.0.json
- For Portainer v2: https://raw.githubusercontent.com/technorabilia/portainer-templates/main/lsio/templates/templates-2.0.json

SelfhostedPro :

- For Portainer v1: https://raw.githubusercontent.com/SelfhostedPro/selfhosted_templates/master/Template/omv-v1.json
- For Portainer v2: https://raw.githubusercontent.com/SelfhostedPro/selfhosted_templates/master/Template/omv-v2.json

DBTech : https://raw.githubusercontent.com/dnburgess/self-hosted-template/master/template.json

Mikestraney : https://raw.githubusercontent.com/mikestraney/portainer-templates/master/templates.json

ARM64

ARM64

Se procurer l'OS

Raspberry Pi OS 64bits

https://downloads.raspberrypi.org/raspios_lite_arm64/images/

ARM64

Premières bidouilles

New Pi password

```
passwd
```

New root password

```
sudo passwd
```

Config

```
sudo raspi-config #change gpu, hostname, password, TZ
sudo apt update
sudo apt upgrade -y
sudo reboot now
#(sudo apt update && sudo apt full-upgrade -y)
```

Check RAM & SWAP

```
free -m
```

Remove Pi user

```
sudo adduser steph
sudo adduser steph sudo
sudo cp /etc/sudoers.d/010_pi-nopasswd /etc/sudoers.d/010_steph-nopasswd
sudo chmod u+w /etc/sudoers.d/010_steph-nopasswd
sudo nano /etc/sudoers.d/010_steph-nopasswd (replace pi &gt; steph)
sudo chmod u-w /etc/sudoers.d/010_steph-nopasswd
sudo reboot
```

Then login as your new user

```
sudo deluser -remove-home pi

sudo rm -vf /etc/sudoers.d/010_pi-nopasswd)
```

ARM64

Increase Swap

1. Before we can increase our Raspberry Pi’s swap file, we must first temporarily stop it.

The swap file cannot be in use while we increase it.

To stop the operating system from using the current swap file, run the following command.

```
xxxxxxxxxxx
```

```
sudo dphys-swapfile swapoff
```

2. Next, we need to modify the swap file configuration file.

We can open this file using nano by using the command below.

```
xxxxxxxxxx
```

```
sudo nano /etc/dphys-swapfile
```

3. Within this config file, find the following line of text.

You can use `CTRL` + `W` to search for text within the file.

```
xxxxxxxxxx
```

```
CONF_SWAPSIZE=100
```

To increase or decrease the swap file, all you need to do is modify the numerical value you find here.

This number is the size of the swap in **megabytes**.

For example, if we wanted to increase our swap size to **1GB**, we would change that line to the following.

```
xxxxxxxxxx
```

```
CONF_SWAPSIZE=1024
```

Whatever size you set, you must have that space available on your SD card.

4. Once you have made the change, save the file by pressing `CTRL + X`, followed by `Y`, then `ENTER`.

5. We can now re-initialize the Raspberry Pi’s swap file by running the command below.

Running this command will delete the original swap file and recreate it to fit the newly defined size.

```
xxxxxxxxxx
```

```
sudo dphys-swapfile setup
```

6. With the swap now recreated to the newly defined size, we can now turn the swap back on.

To start the operating systems swap system, run the following command.

```
xxxxxxxxxx
```

```
sudo dphys-swapfile swapon
```

While the new swapfile is now switched on, programs will not know that this new memory exists until they restart.

7. If you want all programs to be reloaded with access to the new memory pool, then the easiest way is to restart your device.

To restart your Raspberry Pi, all you need to do is run the command below.

```
xxxxxxxxxx
```

```
sudo reboot
```

ARM64

Install Open Media Vault

```
wget -O - https://github.com/OpenMediaVault-Plugin-Developers/installScript/raw/master/install
| sudo bash
sudo reboot now
```

Go to local IP

Default login : admin

Default password : openmediavault

- General settings > Change port to 82
- reconnect
- General settings > Auto logout > 60min
- General settings > Web admin > change password
- Check Time Zone
- Notifications
- fail2ban plug in
- Disks > Select > Wipe
- File systems > create > select hd > name Files > format > mount
- Shared folders > add > Files > select hd > Everyone read/write
- Shared folders > add > Config > select hd > Everyone read/write
- Shared folders > add > Databases > select hd > Everyone read/write
- Shared folders > add > Nextcloud > select hd > Everyone read/write
- SMB/CIFS : Enable / Shares > add > choose Files + config etc... > Public : only guests

Check dans windows : \\192.168.x.x et drag & drop

ARM64

Install Docker et Portainer

OMV-Extras : install docker + portainer

AMD64

Se procurer l'OS

OpenMediaVault

- Debian + OMV: https://www.openmediavault.org/?page_id=77

- Debian: https://cdimage.debian.org/debian-cd/current/amd64/iso-cd/debian-11.2.0-amd64-netinst.iso

AMD64

Increase swap

Link

Step 1 – Checking the System for Swap Information

Before we begin, we can check if the system already has some swap space available. It is possible to have multiple swap files or swap partitions, but generally one should be enough.

We can see if the system has any configured swap by typing:

```
sudo swapon --show
```

If you don’t get back any output, this means your system does not have swap space available currently.

You can verify that there is no active swap using the `free` utility:

```
free -h
```

Step 2 – Checking Available Space on the Hard Drive Partition

Before we create our swap file, we’ll check our current disk usage to make sure we have enough space. Do this by entering:

```
df -h
```

Output

```
Filesystem      Size  Used Avail Use% Mounted on
udev            488M     0  488M   0% /dev
```

```
tmpfs           100M  4.5M   96M   5% /run
/dev/vda1        25G  989M   23G   5% /
tmpfs           499M     0  499M   0% /dev/shm
tmpfs           5.0M     0  5.0M   0% /run/lock
tmpfs           499M     0  499M   0% /sys/fs/cgroup
tmpfs           100M     0  100M   0% /run/user/1001
```

The device with `/` in the `Mounted on` column is our disk in this case. We have plenty of space available in this example (only 1.4G used). Your usage will probably be different.

Although there are many opinions about the appropriate size of a swap space, it really depends on your personal preferences and your application requirements. Generally, an amount equal to or double the amount of RAM on your system is a good starting point. Another good rule of thumb is that anything over 4G of swap is probably unnecessary if you are just using it as a RAM fallback.

Step 3 – Creating a Swap File

Now that we know our available hard drive space, we can create a swap file on our filesystem. We will allocate a file of the swap size that we want called `swapfile` in our root (/) directory.

The best way of creating a swap file is with the `fallocate` program. This command instantly creates a file of the specified size.

Since the server in our example has 1G of RAM, we will create a 1G file in this guide. Adjust this to meet the needs of your own server:

```
sudo fallocate -l 1G /swapfile
```

We can verify that the correct amount of space was reserved by typing:

```
ls -lh /swapfile
```

Output

```
-rw-r--r-- 1 root root 1.0G May 29 17:34 /swapfile
```

Our file has been created with the correct amount of space set aside.

Step 4 – Enabling the Swap File

Now that we have a file of the correct size available, we need to actually turn this into swap space.

First, we need to lock down the permissions of the file so that only the users with **root** privileges can read the contents. This prevents normal users from being able to access the file, which would have significant security implications.

Make the file only accessible to **root** by typing:

```
sudo chmod 600 /swapfile
```

Verify the permissions change by typing:

```
ls -lh /swapfile
```

Output

```
-rw------- 1 root root 1.0G May 29 17:34 /swapfile
```

As you can see, only the **root** user has the read and write flags enabled.

We can now mark the file as swap space by typing:

```
sudo mkswap /swapfile
```

Output

```
Setting up swapspace version 1, size = 1024 MiB (1073737728 bytes)
no label, UUID=b591444e-c12b-45a6-90fc-e8b24c67c006f
```

After marking the file, we can enable the swap file, allowing our system to start using it:

```
sudo swapon /swapfile
```

Verify that the swap is available by typing:

```
sudo swapon --show
```

Output

```
NAME      TYPE  SIZE USED PRIO
/swapfile file 1024M   0B   -2
```

We can check the output of the `free` utility again to corroborate our findings:

```
free -h
```

Output

```
              total        used        free      shared  buff/cache   available
Mem:          990Mi        37Mi       860Mi       4.0Mi        92Mi       834Mi
Swap:         1.0Gi          0B       1.0Gi
```

Our swap has been set up successfully and our operating system will begin to use it as necessary.

Step 5 – Making the Swap File Permanent

Our recent changes have enabled the swap file for the current session. However, if we reboot, the server will not retain the swap settings automatically. We can change this by adding the swap file to our `/etc/fstab` file.

Back up the `/etc/fstab` file in case anything goes wrong:

```
sudo cp /etc/fstab /etc/fstab.bak
```

Add the swap file information to the end of your `/etc/fstab` file by typing:

```
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
```

Next we’ll review some settings we can update to tune our swap space.

AMD64

Premières bidouilles

New Pi password

```
passwd
```

New root password

```
sudo passwd
```

Config

```
sudo raspi-config #change gpu, hostname, password, TZ
sudo apt update
sudo apt upgrade -y
sudo reboot now
#(sudo apt update && sudo apt full-upgrade -y)
```

Check RAM & SWAP

```
free -m
```

Remove Pi user

```
sudo adduser steph
sudo adduser steph sudo
sudo cp /etc/sudoers.d/010_pi-nopasswd /etc/sudoers.d/010_steph-nopasswd
sudo chmod u+w /etc/sudoers.d/010_steph-nopasswd
sudo nano /etc/sudoers.d/010_steph-nopasswd (replace pi &gt; steph)
sudo chmod u-w /etc/sudoers.d/010_steph-nopasswd
sudo reboot
```

Then login as your new user

```
sudo deluser -remove-home pi

sudo rm -vf /etc/sudoers.d/010_pi-nopasswd)
```

AMD64

Installer OMV Extras

OMV5

```
sudo wget -O - https://github.com/OpenMediaVault-Plugin-
Developers/installScript/raw/master/install | sudo bash
sudo dpkg --configure -a
```

OMV6

```
 wget -O - https://github.com/OpenMediaVault-Plugin-Developers/packages/raw/master/install |
bash
```

Go to local IP

Default login : admin

Default password : openmediavault

- General settings > Change port to 82

- reconnect

- General settings > Auto logout > 60min

- General settings > Web admin > change password

- Check Time Zone

- Notifications

- fail2ban plug in

- Disks > Select > Wipe

- File systems > create > select hd > name Files > format > mount

- Shared folders > add > Files > select hd > Everyone read/write

- Shared folders > add > Config > select hd > Everyone read/write

- Shared folders > add > Databases > select hd > Everyone read/write

- Shared folders > add > Nextcloud > select hd > Everyone read/write

- SMB/CIFS : Enable / Shares > add > choose Files + config etc... > Public : only guests

Check dans windows : \\192.168.x.x et drag & drop

Install Docker and Portainer on Debian for Self-Hosting

When I started tinkering with self-hosting, Docker was by far my biggest hurdle. But to learn more about Docker, we need to figure out how to install it first. Then, later we will install and use Portainer to manage and monitor our Docker containers.

Table of Contents

Docker does a good job at explaining how to install Docker on specific distros. However, I always found it intimidating with how the instructions are laid out. So this guide is from a different perspective. A perspective from someone who once was lost and was learning like you are.

I'll lay it out in a much more simple format for beginners to move forward as quickly as possible. Aftrall, we live in a "I want it now" society and quite frankly, that's how I would want to see it done!

Update and upgrade the Host Machine

It's important to keep your host machine up to date. This can include important patches that keep your host safe from vulnerabilities.

Step 1. Update the host.

```
apt update
```

Step 2. Upgrade the host.

```
apt upgrade -y
```

Install Docker on your Host Machine

Docker has made this very simple by creating an official install script that does all the work for you. You can see what this script includes and how it works on the Docker Github repo.

> The purpose of the install script is for a convenience for quickly installing the latest Docker-CE releases on the supported linux distros. It is not recommended to depend on this script for deployment to production systems. For more

> thorough instructions for installing on the supported distros, see the install instructions.

I'll be using Ubuntu 23.04 standard on Proxmox to run these commands as root. You may need to add sudo at the beginning of these commands if you are not logged in as root.

Step 1. Install curl.

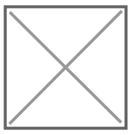
```
apt install curl
```

Step 2. Install Docker.

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

Step 3. Check if Docker is functioning.

```
systemctl status docker
```



Step 4. Force Docker to start at boot.

```
systemctl enable docker
```

This step is not necessary but can make the process of starting docker quicker by doing it automatically. This way you don't have to manually start Docker when you reboot your host machine. I haven't seen any official documentation but Docker always seems to start automatically regardless after testing on my hardware.

As you get more comfortable with Docker and setting up self-hosted applications, you'll also find it useful to install Docker Compose and Git along the way.

Install Docker Compose on the Host Machine

Run the following command to install Docker Compose

```
apt install docker-compose
```

Install Git on the Host Machine

You may come across a project that requires you to clone it to your host machine from Github. To do this, you will need to install Git.

```
apt install git -y
```

Now you can use the `git clone` command to clone repositories to your host machine.

Setting up Portainer on the Host Machine

I think it's a great idea to learn basic Docker commands because they will come in handy at a later time. Especially when you want to update Portainer. Portainer makes managing Docker containers easier and in my opinion, faster. You want it now right? □□ Well Portainer will give you a nice graphic interface for what would otherwise be a much larger learning curve using only the CLI (command line interface).

Step 1. Install Portainer.

Now that Docker is installed, you can install Portainer using Docker run. Open your terminal and run the following command to install Portainer.

```
docker run -d \ --name="portainer" \ --restart on-failure \ -p 9000:9000 \ -p 8000:8000 \ -v /var/run/docker.sock:/var/run/docker.sock \ -v portainer_data:/data \ portainer/portainer-ce:latest
```

When complete, navigate to your host IP on port 9000. Example: 192.168.1.5:9000. You will see the Portainer registration page.

If you don't know the host machine IP, you can use the following command to output the information.

```
ip addr
```

Mine is usually always listed under number 2 in the output.

Step 2. Create a Portainer admin account.

I uncheck the collection of anonymous statistics. I don't need anyone knowing what I do or when I do it, but I am happy to give feedback of course.

Then click "Create user" and log in!

Final Notes and Thoughts

For now, this is a great place to stop and get familiar with Portainer and the menu within the dashboard. Browse around and check out the different settings but try not to get overwhelmed by all the technical jargon you may not understand.

In the next article, we will go over Portainer basics and launching your first self-hosted application using Portainer and the Docker Compose stacks feature.