V1.0.3

THIRTYSOMETHING

DIY-NAS

About my DIY-NAS

ThirtySomething

21.04.2022

Contents

Contents **1**

Change history **3**

1 Initial situation **4**

2 Search for alternatives **4**
2.1 Use a mini PC in front of the NAS 4
2.2 Hardware alternatives 4
2.3 The planned hardware 5
2.4 The bought hardware 5

3 The hardware build **6**

4 The software installation **6**
4.1 The operating system 6
4.2 The disks 7
4.3 The RAID storage 7
4.4 The file system 8
4.5 Shared folders 9
4.6 Users 10
4.7 CIFS shares 11
4.8 Tuning 12
4.8.1 SMB/CIFS 12

5 OMV plugins **12**
5.1 ClamAV 12
5.2 MiniDLNA 15
5.3 OMV extras 16
5.4 Autoshutdown 17

6 Docker **19**
6.1 Docker installation 20
6.2 Portainer 20
6.3 SCM-Manager 21

7 Ports **24**

A svnExport.sh **25**

List of Figures I

List of Tables II

Glossary III

Change history

Version	Date	Section	Description	Name
1.0.0	02.04.2022	–	• Start with description	ThirtySomething
1.0.1	16.04.2022	–	• Rename section WOL to Autoshutdown • Fill section Autoshutdown with content	ThirtySomething
1.0.2	16.04.2022	–	• Split file DIY-NAS-Content into separate files	ThirtySomething
1.0.3	21.04.2022	–	• Update to latest template version • Add section about used ports • Add svnExport.sh as appendix	ThirtySomething

Table 1: Change history

1 Initial situation

In 2010 I bought my first NAS. It was a Synology DS411slim. The device is running up to now. For a few years now, it is only supplied with security updates. That is quite remarkable for the fact that it has already 12 years on the hump.

Now I've run out of space - I have 4*2TB running there in RAID 10, which results in about 3.7TB. In addition, the performance is, well, not quite up to date.

2 Search for alternatives

2.1 Use a mini PC in front of the NAS

Realizing this situation I was searching for alternatives. The first idea was to use a mini PC in front of the NAS. This failed for some reasons.

- Using the NAS directly for Docker volumes
- The CPU of the mini PC does not support Intel VT
- The network as bottleneck for Docker containers

2.2 Hardware alternatives

Then I decided to by a new NAS system. But which kind of NAS will it be? I've spent a long time to search the internet for possible soultions. I didn't know about the possibility to pimp a TerraMaster NAS with a different OS. This is also possible for Western Digital NAS as described here and here. Also the variant with a mini PC and separated storage case was interesting. The idea is to get the most out of money, so the ranking here is done by price/performance ratio and the DIY NAS wins the comparison.

- DIY NAS system on PC base
- DIY NAS system with separated storage case and mini PC
- Commercial NAS system
- Commercial NAS with custom OS

2.3 The planned hardware

When comparing the different hardware solutions, I also made a comparison between an Intel and an AMD based NAS. The AMD variant won the price/power chapter, so I decided to buy:

- 32GB G.Skill RipJaws V black DDR4-3200 DIMM Dual Kit
- 250GB Samsung 970 Evo Plus M.2 2280
- 400 Watt be quiet! Pure Power 11 CM Modular 80+ Gold
- Black Fractal Design Node 304 cube without power supply
- 4x 4000GB WD Red Plus WD40EFZX 128MB 3.5”
- ASRock Fatal1ty B450 Gaming-ITX/AC AMD B450
- AMD Athlon 3000G with Radeon Vega Graphics 3.5GHz
- Noctua NH-L9a-AM4 topblow cooler

2.4 The bought hardware

„Life is what happens to you while you’re busy making other plans.“

Figure 1: John Lennon

This means that the availability of chips and other events affect the market and make it difficult to realize these plans. For example, motherboards in mini-ITX format with AM4 socket are really hard to get. That’s why I revised my decision on the components:

- 32GB G.Skill RipJaws V black DDR4-3200 DIMM Dual Kit
- 250GB Samsung 970 Evo Plus M.2 2280
- 400 Watt be quiet! Pure Power 11 CM Modular 80+ Gold
- Black Fractal Design Node 304 cube without power supply
- 4x 4000GB WD Red Plus WD40EFZX 128MB 3.5”

- ASRock H610M-ITX/AC mITX Intel H610 DDR4 S1700
- Intel Core I3-12100 tray
- Noctua NH-L9i-17xx topblow cooler

The difference is that the CPU is a latest generation Intel I3 processor. The corresponding motherboard is also brand new. Both were only launched in Q4/2022. This has some consequences: OMV 5 does not support the network chip of the motherboard. So I decided to install the OMV 6 beta on my DIY NAS. I also tried TrueNAS – it looks much more professional than OMV, but it is also a bit more complicated to understand and configure. It also doesn't have native support for Docker.

3 The hardware build

Although I build up my last PC more than 25 years ago assembling the hardware was a breeze. The only surprise was the mounting position of the power supply. As I understand how to do this everything was fine.

4 The software installation

4.1 The operating system

The basic installation of OMV works without any problems. This is known from debian. Additional Volker Theile, the founder of the OMV project, and his team have done a very good job. Thank you guys!

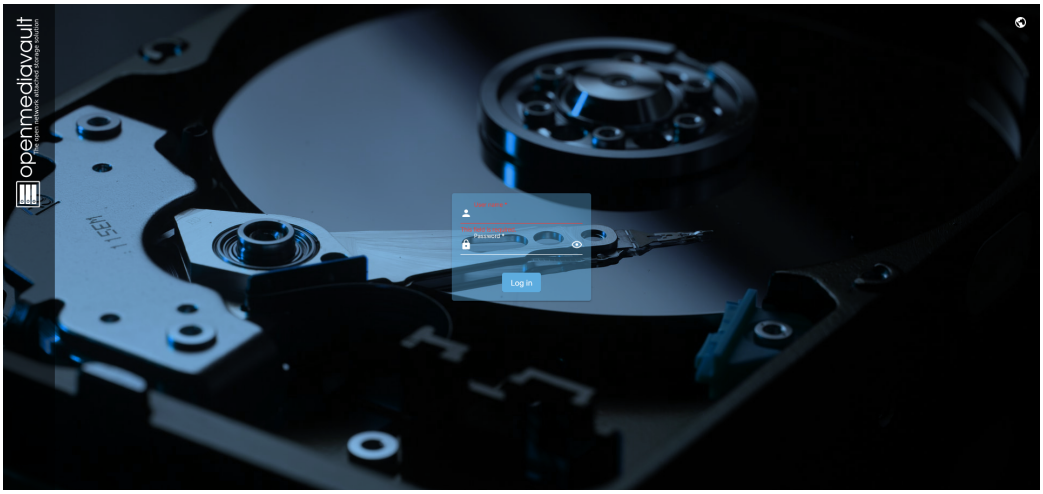

Figure 2: The OMV login page

4.2 The disks

All attached disks are working. The system disk is a NVME SSD with a 250 GB capacity. All others are 4 TB WD RED disks.

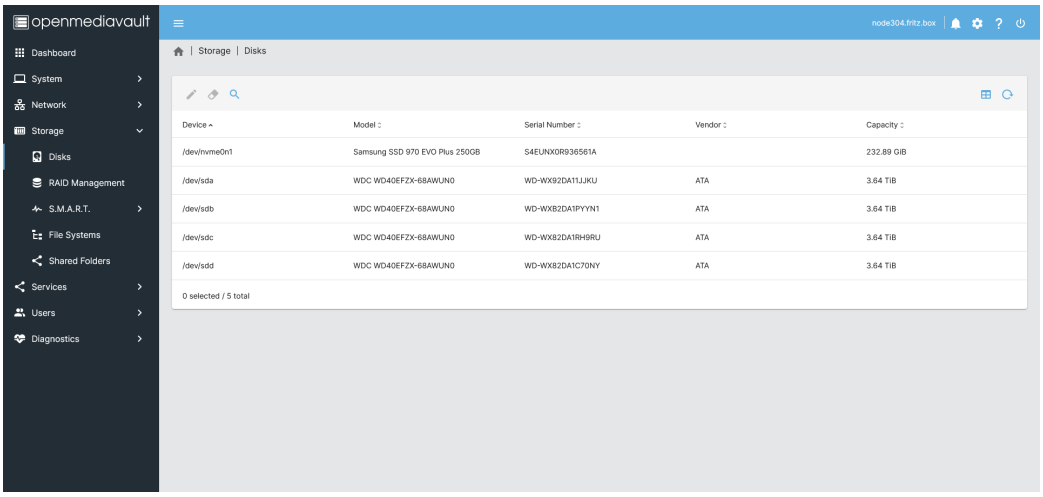

Figure 3: The OMV disks

4.3 The RAID storage

Configuring the RAID was also less problematic. But there is an important point: As long as the RAID system is created, do not use the RAID! During the RAID build I was playing around with OMV plugins to see the

diskstats – and damaged the RAID build. After a reboot the system hangs on the filesystem check. I was shocked - also about the large amount of blocks the fsck found and tried to repair. It took a while to understand what happens. Then I've startet from scratch and everything was okay. The creation of the file system at the end went without problems. A filesystem check found nothing to do.

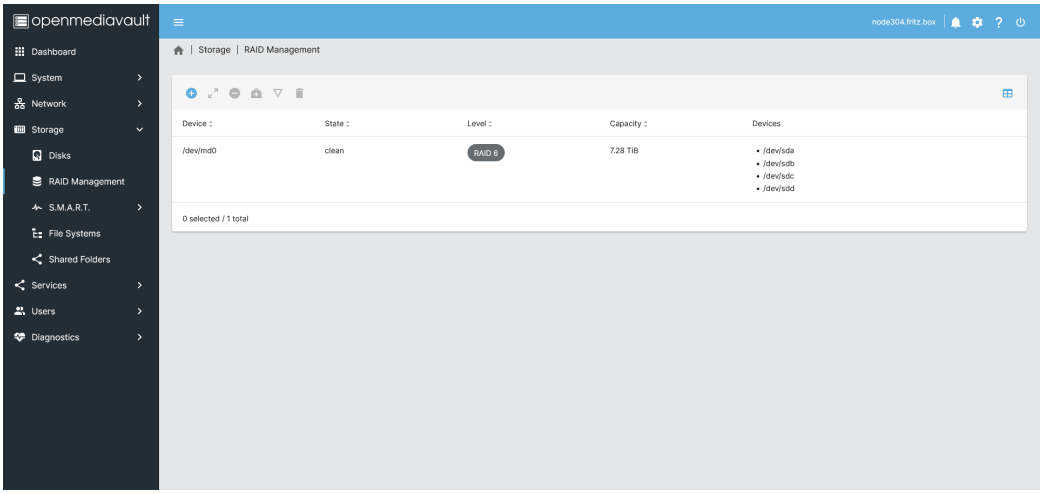

Figure 4: The OMV RAID

4.4 The file system

This was a simple step – just create a filesystem on the previously created RAID volume.

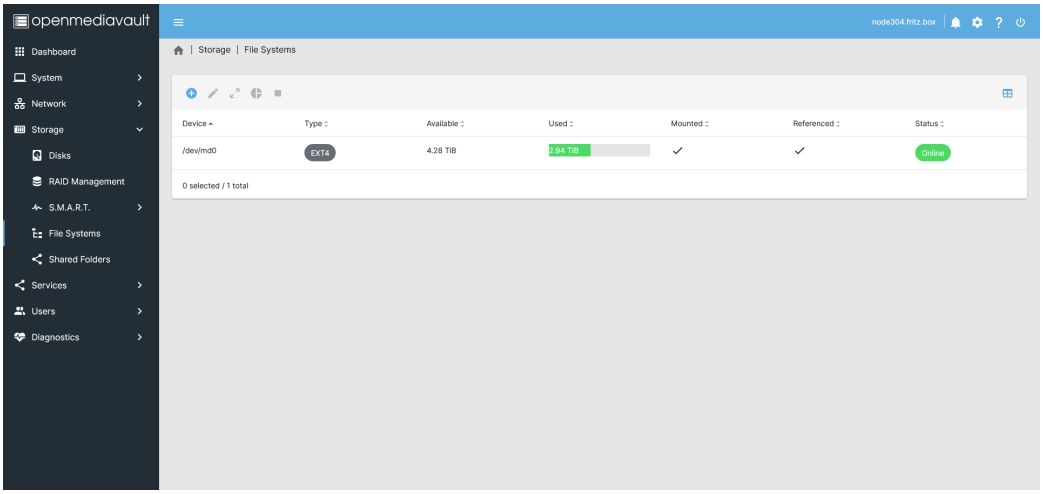

Figure 5: The OMV filesystem on the RAID

4.5 Shared folders

Some common folders should be defined before anything else is done:

- The folder `docker` for the use of Docker.
- The folder `homes` as base folder for the users home directories.
- The folder `music` for the miniDLNA plugin.
- The folder `quarantine` for the use of ClamAV.
- The folder `video` also for the miniDLNA plugin.

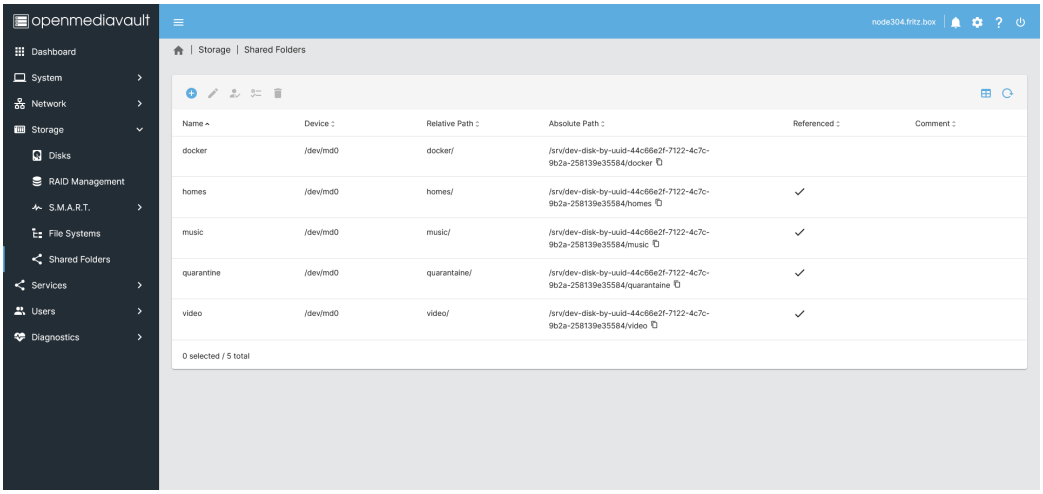

Figure 6: The OMV shared folders

This „shared folders“ are defined as container to be used inside of OMV. To make them accessible from the network you have to enable services for them.

4.6 Users

In the settings the option `User home directory` is enabled and points to the previously created (homes) folder. Then I created the users.

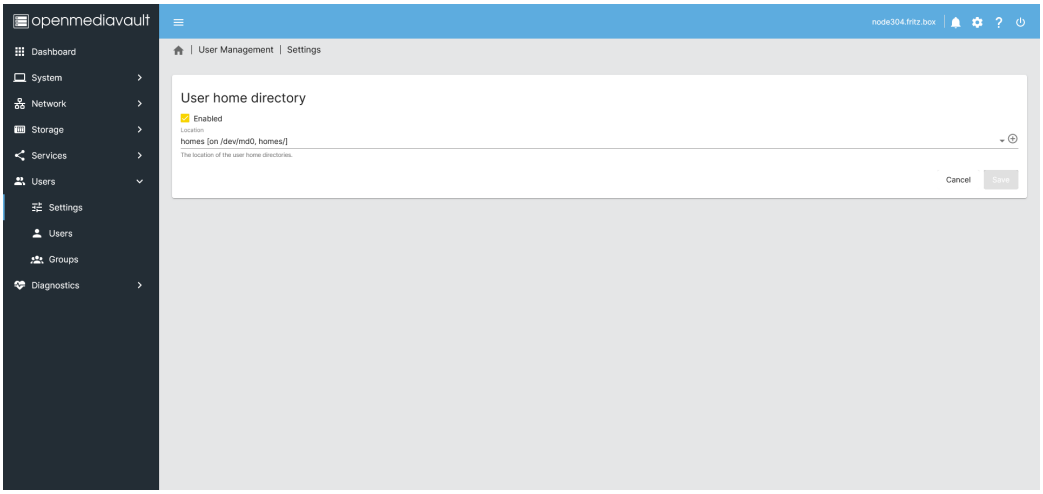

Figure 7: The OMV users home directory

4.7 CIFS shares

Simple - enable the `SMB/CIFS` service, enable the `home directories` and then create shares for the previously defined `shared folders`. Allow read/write access for the administrator of the shares (me), the others got read access to them.

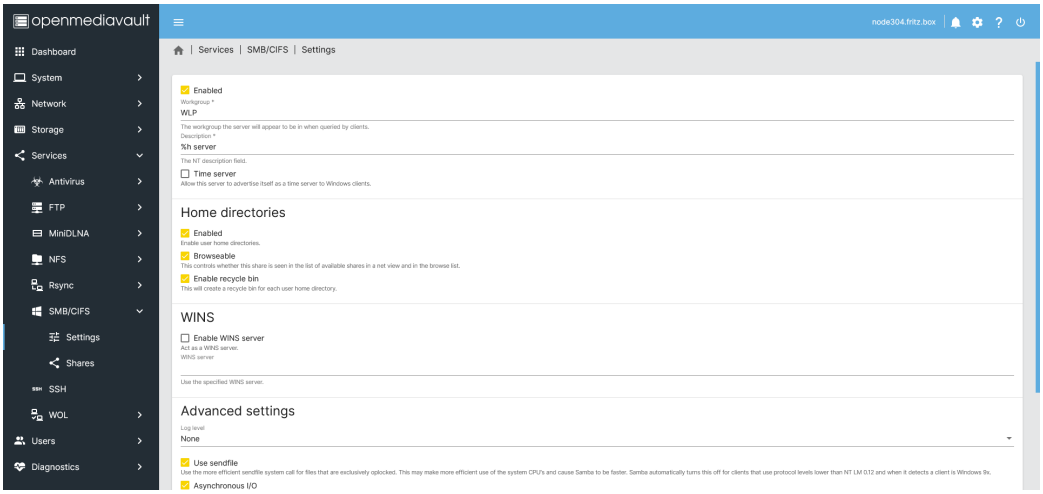

Figure 8: The OMV CIFS settings

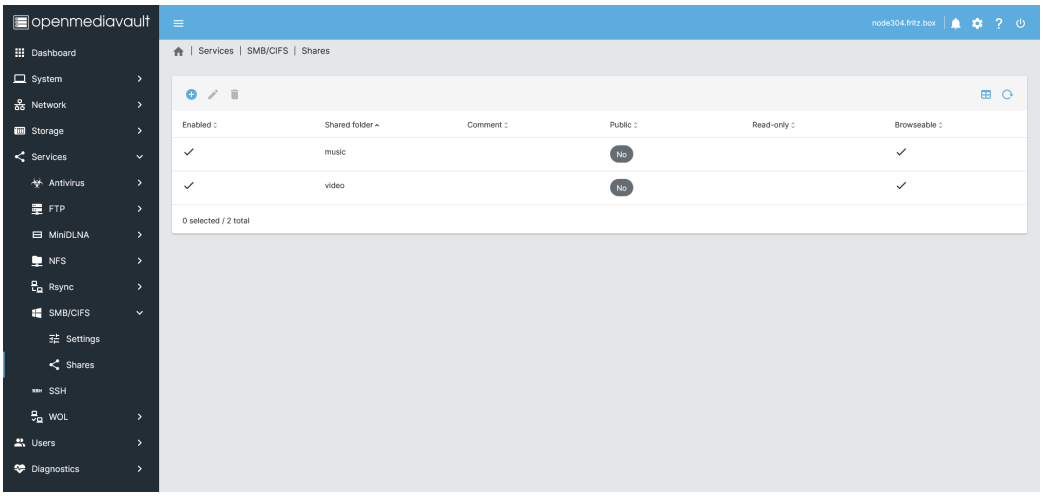

Figure 9: The OMV CIFS shares

As you can see there is actualy no public access to this shares. This means that only priviledged users can access them.

4.8 Tuning

4.8.1 SMB/CIFS

Transfering data from the old NAS to the new one takes a lot of time. To speed up the process I searched for some tuning and found this here:

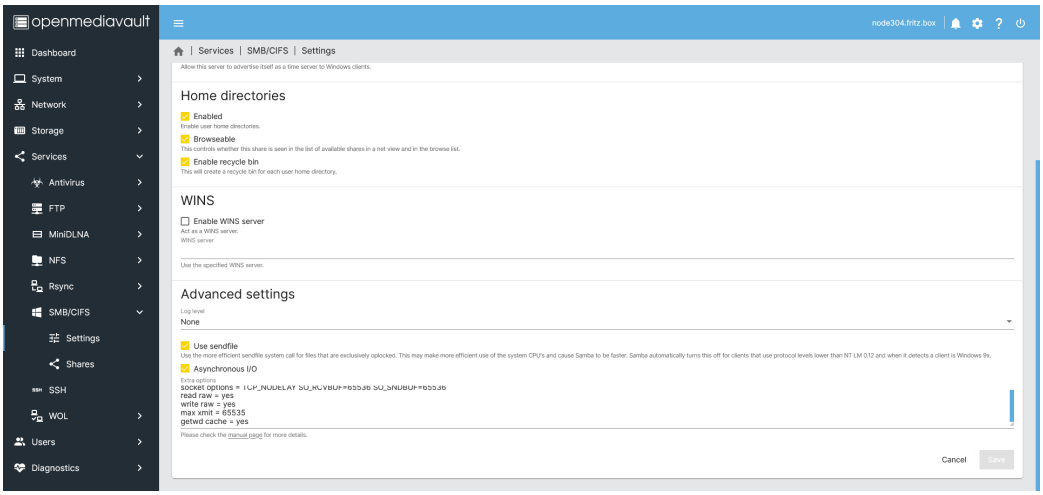

Figure 10: The OMV advanced CIFS settings

5 OMV plugins

5.1 ClamAV

To protect the data I want to use an antivirus program. As open source solution there is ClamAV available – and also as plugin for OMV.

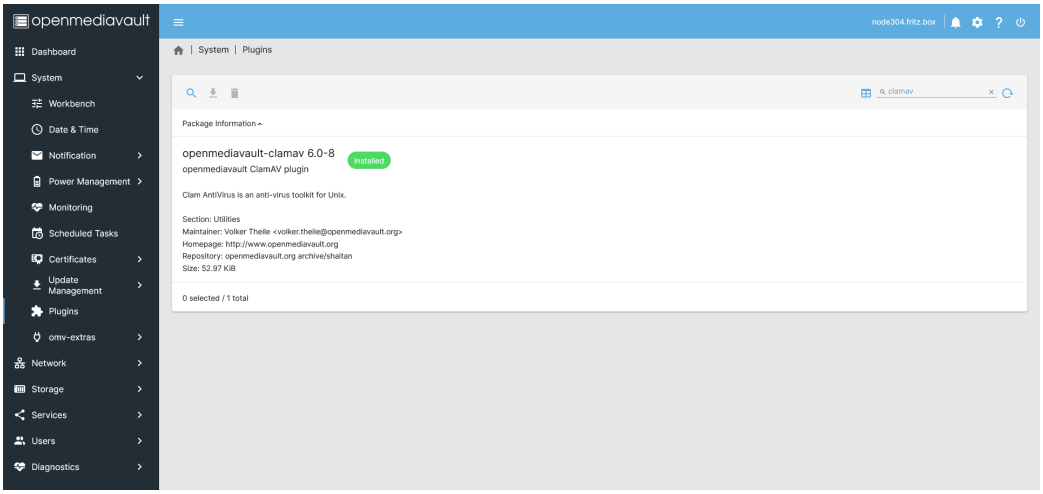

Figure 11: The OMV ClamAV plugin

In the setup we use the previously defined quarantine folder.

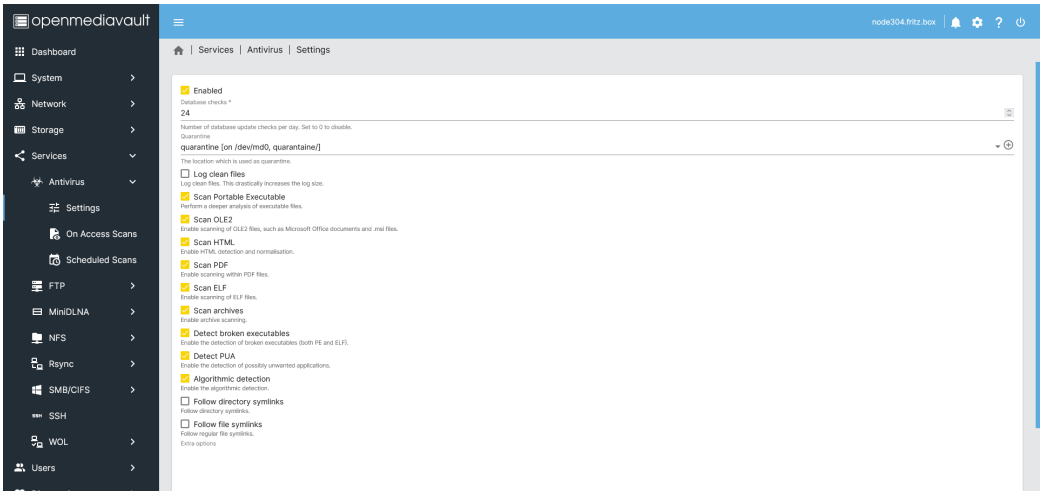

Figure 12: The antivirus settings

I enabled a scan on access for specific folders.

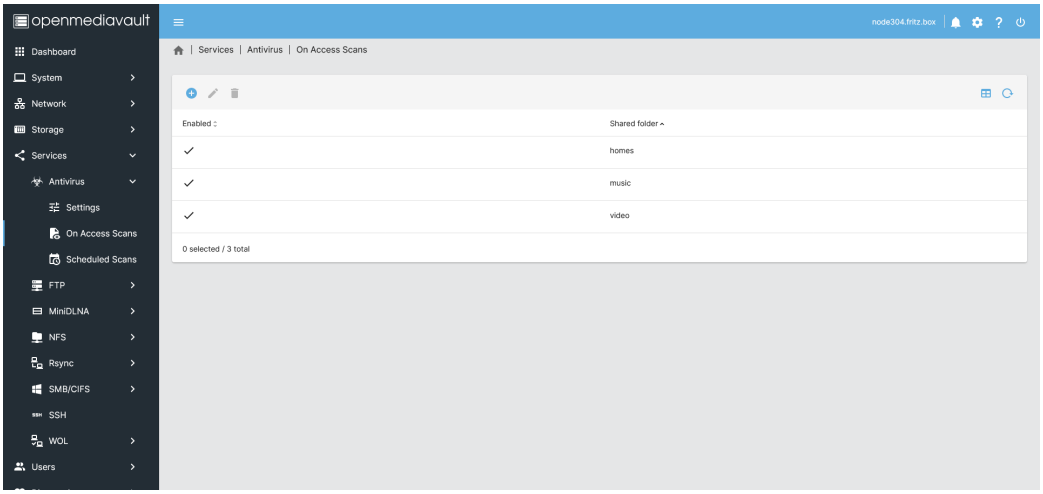

Figure 13: The antivirus on access scan

Also I've enabled a scheduled scan for these folders, too.

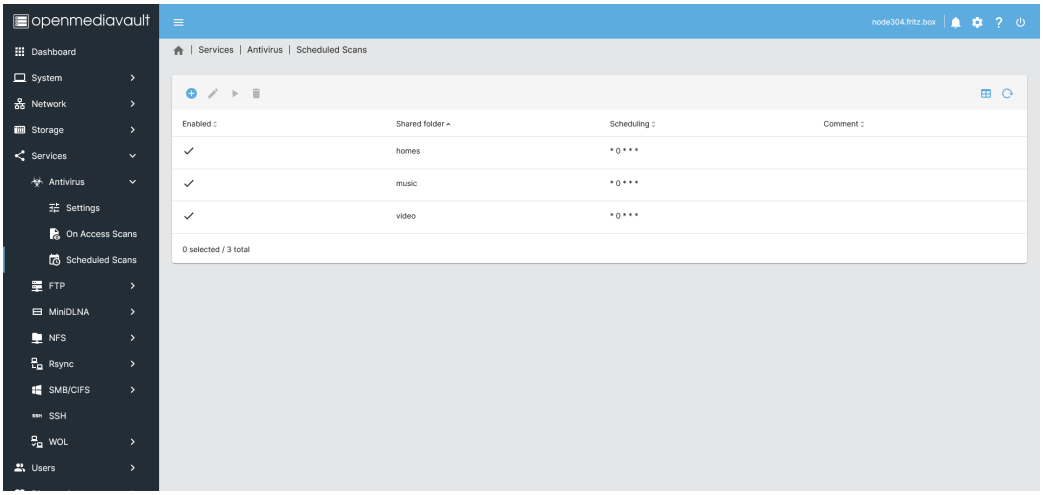

Figure 14: The antivirus scheduled scan

Maybe somebody will claim that all scheduled scans run on the same time. Yes – this was setup to check the power of the CPU. The system load increases to a load of about 3 – that's fantastic from my point of view! This means that there are more than enough reserves. When I'm spreading the schedule I can lower the load.

Figure 15: The antivirus system load on scan

5.2 MiniDLNA

To stream music and videos from the NAS to the network I'm using the ReadyDLNA media server software. In OMV this is used with the oldname `MiniDLNA`.

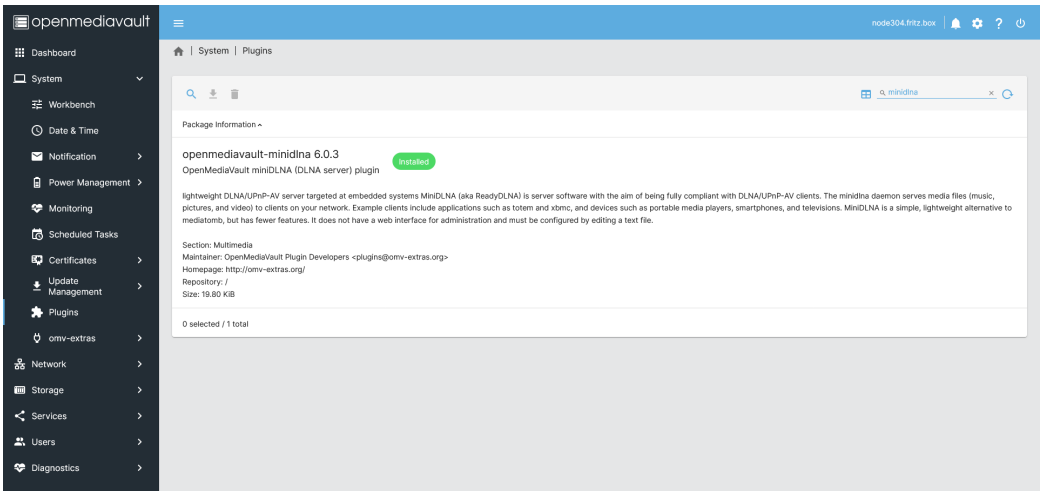

Figure 16: The MiniDLNA plugin

The basic setup is simple - I've checked the `Enable` box and that's it.

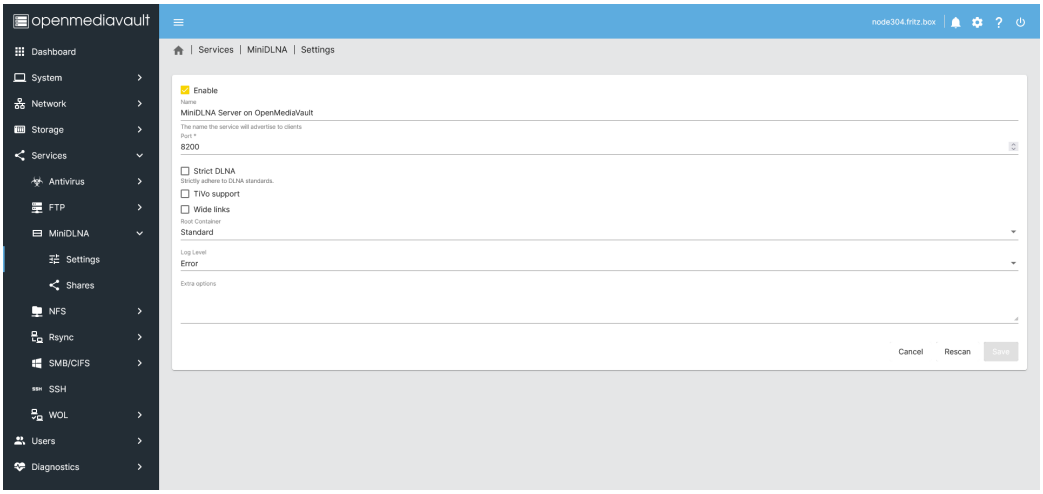

Figure 17: The MiniDLNA settings

Then I have to define the shares and the kind of content of the shares.

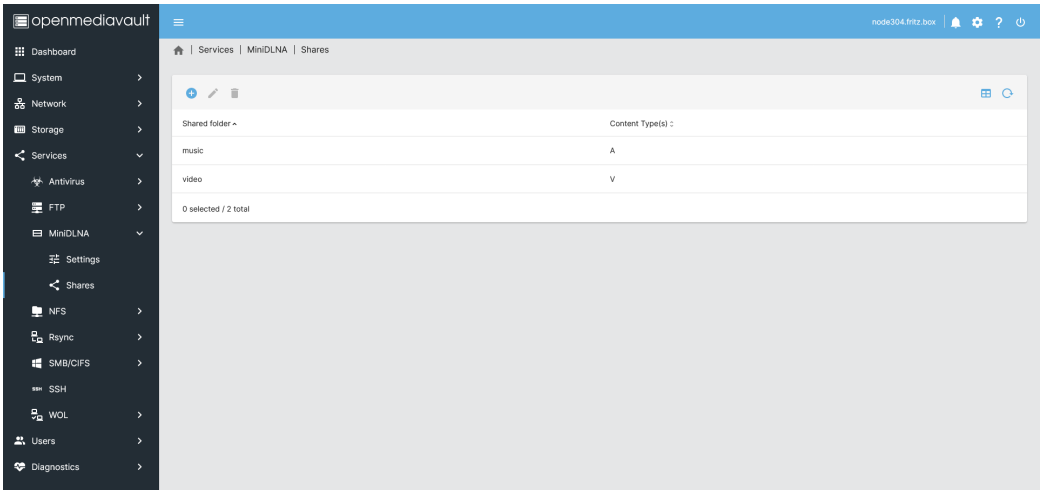

Figure 18: The MiniDLNA shares

5.3 OMV extras

The OMV extras are not available in the plugin list. The way to go to install them is described here. Login as user `root` using SSH and enter the following command:

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

Figure 19: The OMV extras installation

Using this plugin enables the Docker and Portainer installation to enhance the capabilities of the NAS.

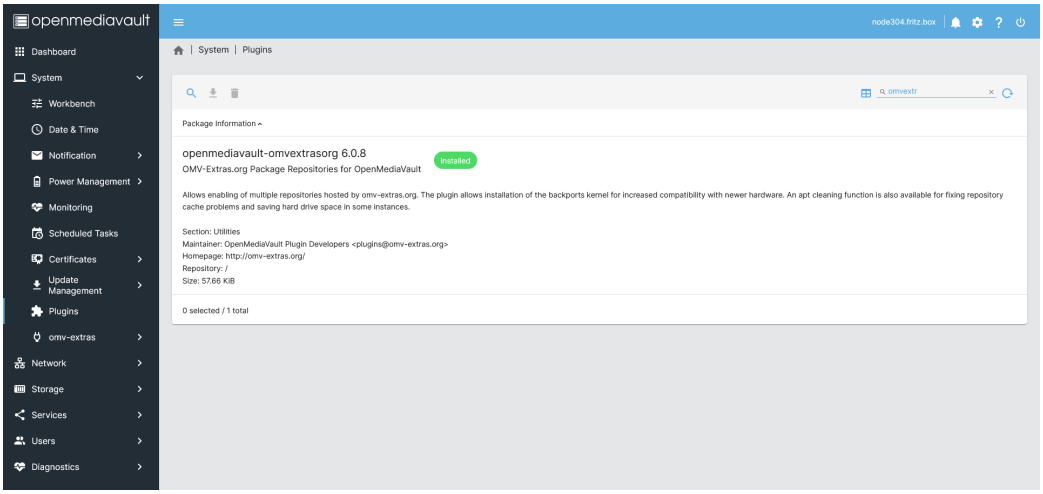

Figure 20: The OMV extras plugin

5.4 Autoshutdown

To save some energy we want the system to shutdown when not used and to wake up when used. So we want to use the autoshutdown plugin and the WOL feature. **NOTE:** Don't forget to enable WOL in the BIOS. For my system there have been two settings, one setting on the adapter itself and one setting in the boot options.

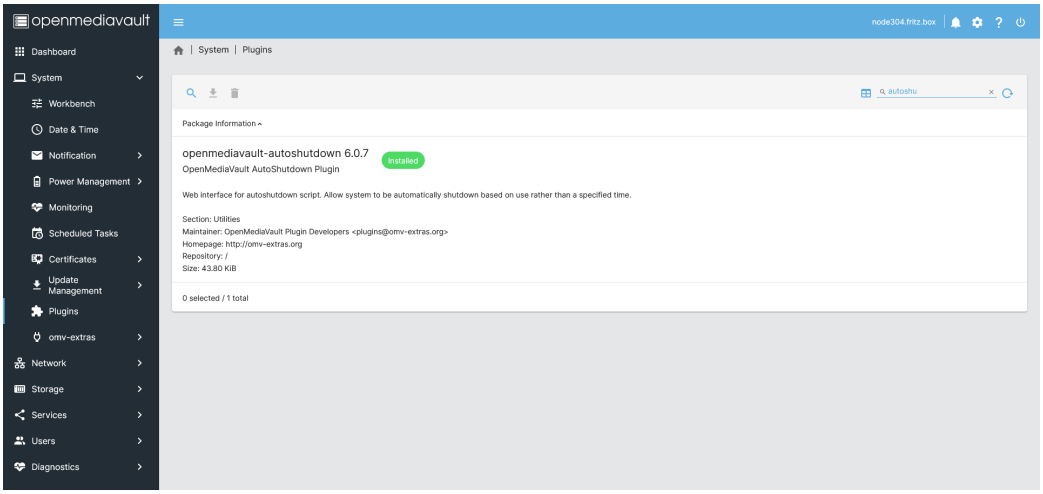

Figure 21: The OMV autoshutdown plugin

As first step we have to enable the WOL on the network card. You can find the checkbox as last option ot the `Advanced settings` section.

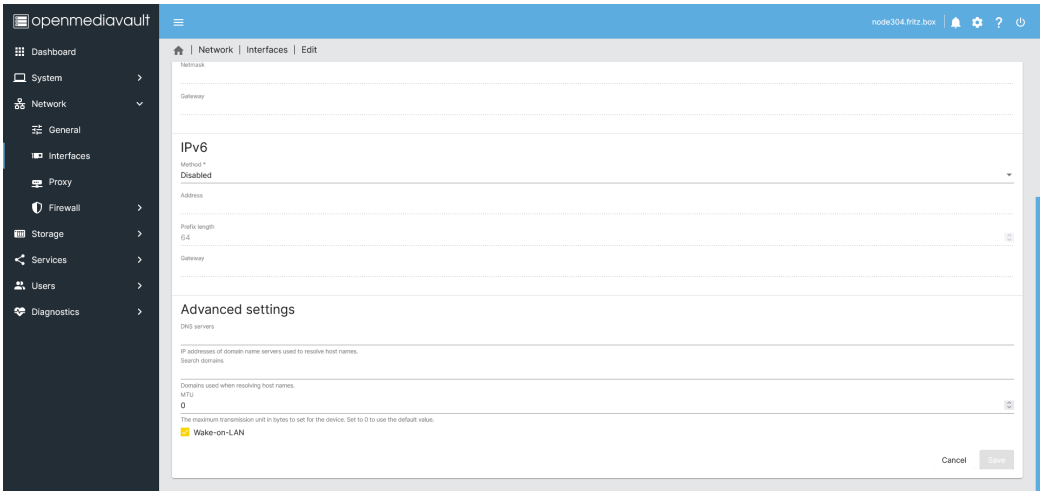

Figure 22: The OMV network card setup

Then we have to configure/setup the autoshutdown plugin. First step is to enable the autoshutdown. Second step is to force an uptime between 23:30 and 02:00 to ensure backup is done. The rest is left with default settings and might be updated at a later point.

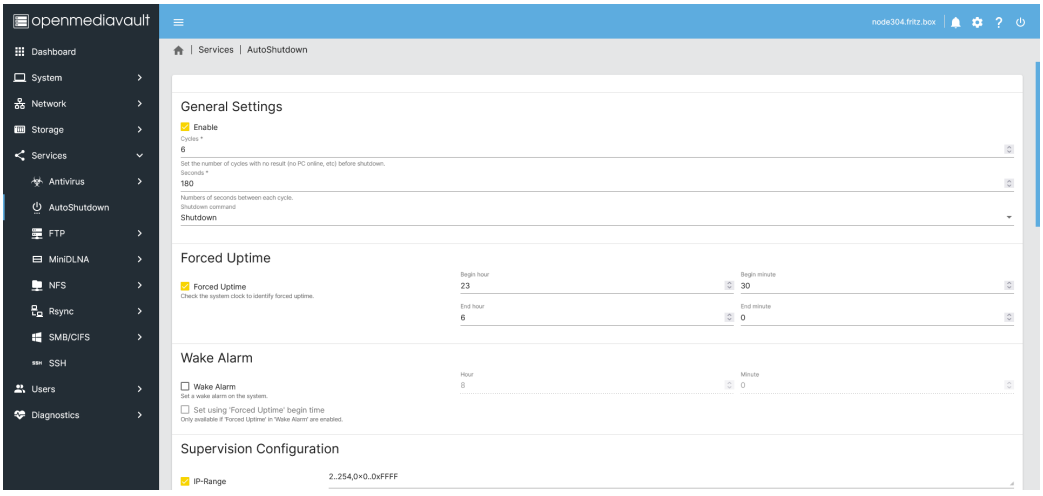

Figure 23: The autoshutdown setup 1

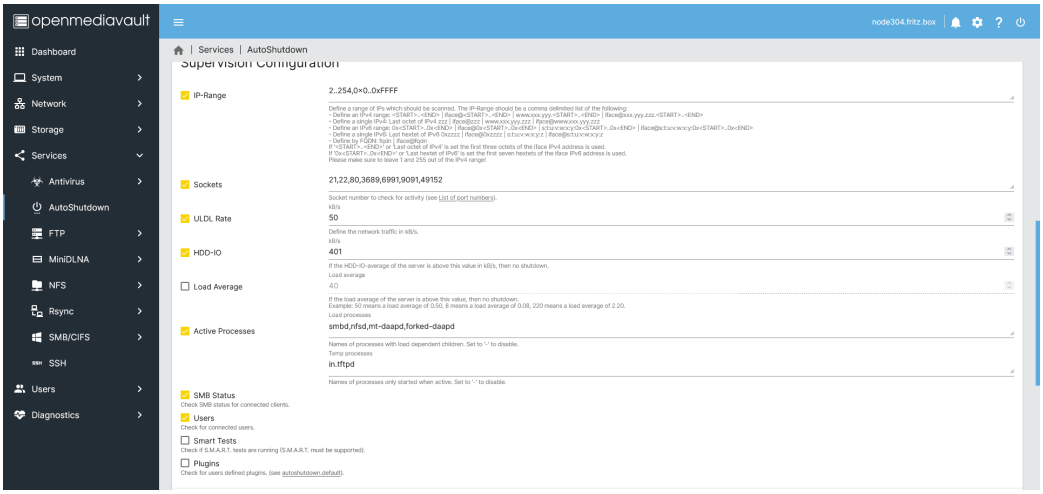

Figure 24: The autoshutdown setup 2

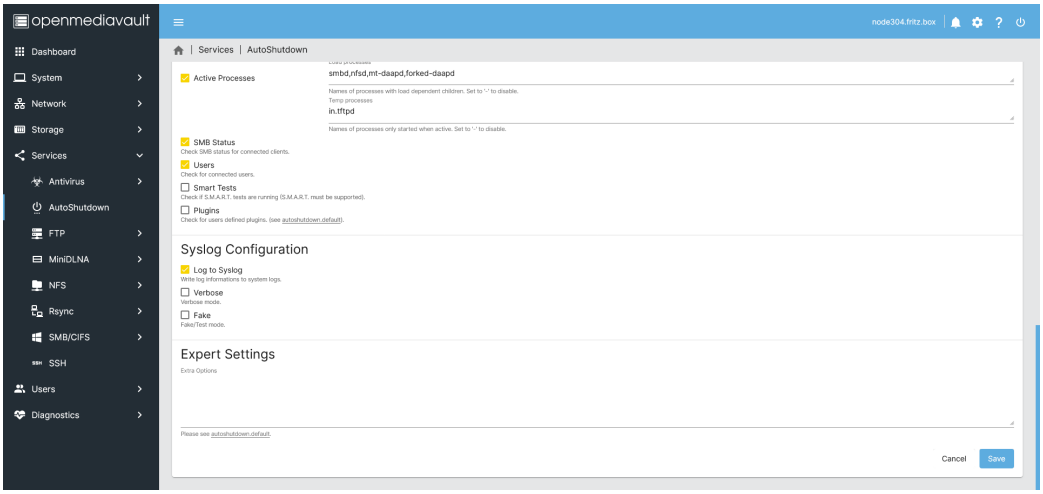

Figure 25: The autoshutdown setup 3

6 Docker

By using Docker I want to enhance the NAS with features which are not available out-of-the-box. Especially with services my previous NAS offers and which are not native available for OMV.

6.1 Docker installation

The default location of Docker is `/var/lib/docker` on the system disk. Allthough the system disk is a fast SSD, I want to install Docker on the slower RAID storage. So I have to change the path of Docker storage.

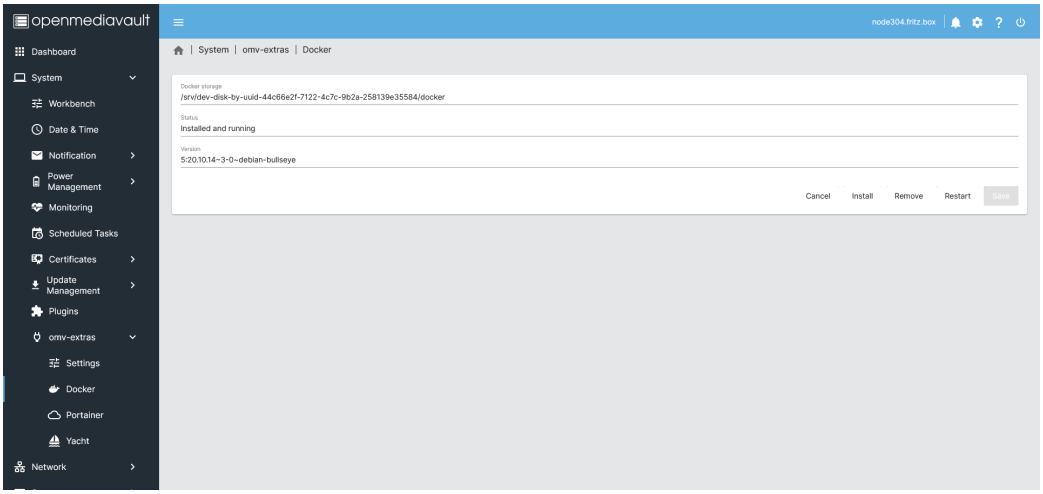

Figure 26: The Docker setup

In case Docker is already installed, see here how to move the Docker storage to another location.

6.2 Portainer

To have more comfort in dealing with Docker we install also Portainer from the OMV extras.

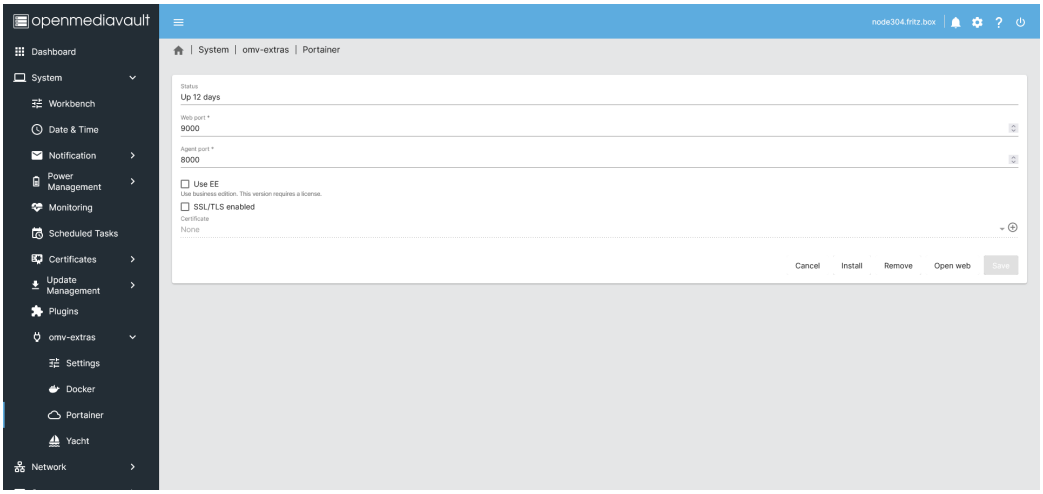

Figure 27: The Portainer setup

After installation Portainer is up and running.

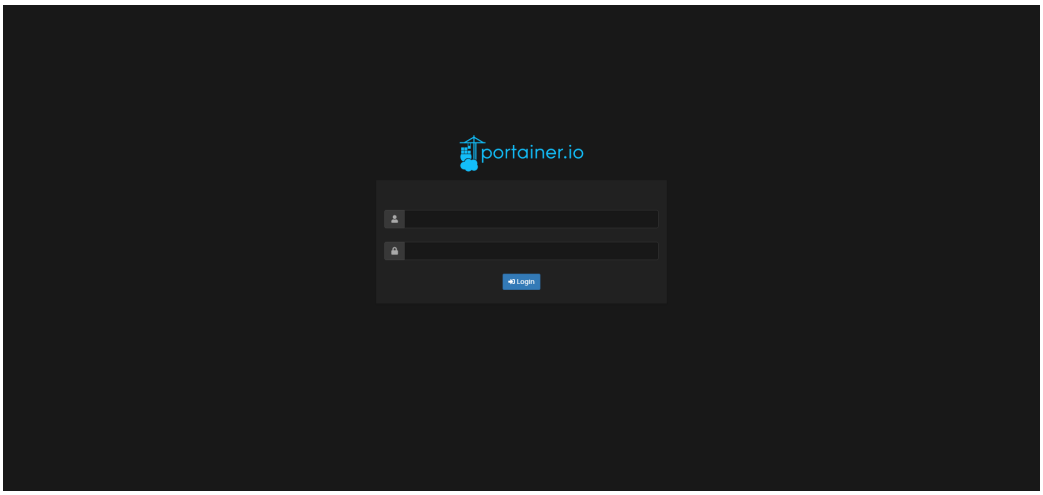

Figure 28: The Portainer login

6.3 SCM-Manager

SCM-Manager provides a comfortable user interface for git, Mercurial and Subversion. Up to now I use Subversion for version control. Some of my repositories are private and I will never publish them to a public hoster like GitHub allthough they offer private repositories. All my other repositories are hosted on my NAS – the plan is to move them to git if possible – just to have them on a more modern version control system.

First of all we have to create a Docker volume for SCM-Manager. This is important for running the backup script.

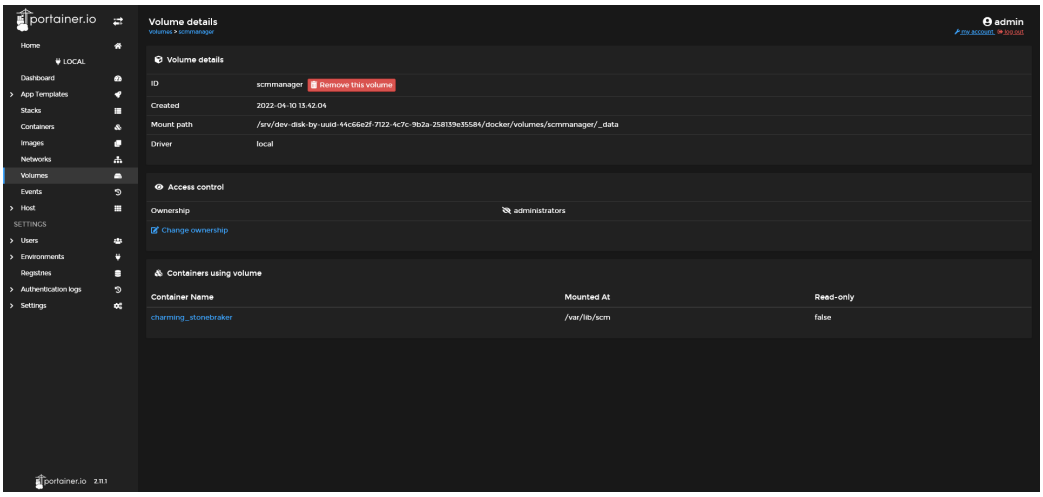

Figure 29: The docker volume scmmanager

Then we can setup the container:

- Mapping port `2222` to `2222` to enable the Subversion protocol
- Mapping port `8080` to `8080` to enable the http protocol
- Mapping of the SCM-Manager volume to `/var/lib/scm`

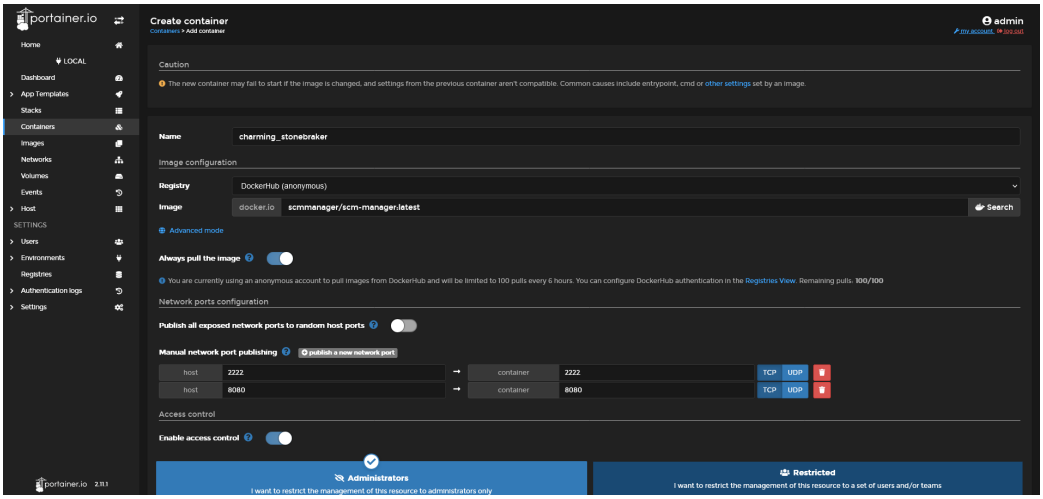

Figure 30: The docker container scmmanager setup 1

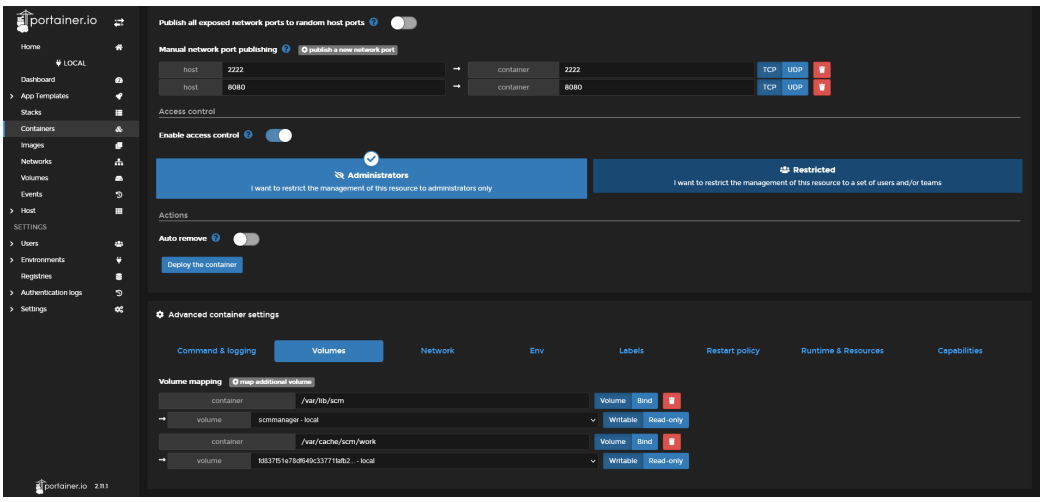

Figure 31: The docker container scmmanager setup 2

The startup of the container should work without any problems.

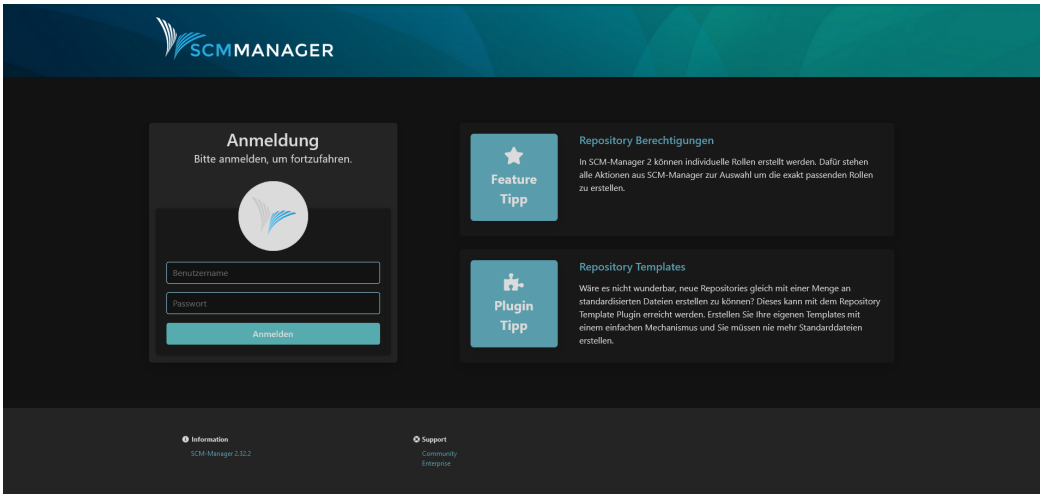

Figure 32: The SCM-Manager login

After importing my repositories, I've setup a backup job. For this I've updated my script svnExport.sh that it will work for SCM-Manager. Setting up a cronjob inside a Docker container is a pita. The previously created volume is required to dynamically export all existing Subversion repositories. The command will be something like this one:

```
/bin/bash /srv/<omv-raid>/<path-to-script>/svnExport.sh >>
/srv/<omv-raid>/<path-to-script>/svnExport.log 2>&1 &
```

Figure 33: Cron command for svnExport

For the script please see appendix A.

The job is then setup in OMV scheduled tasks accessing the script. **NOTE:** The path to the SCM-Manager repositories needs to be set inside in the script.

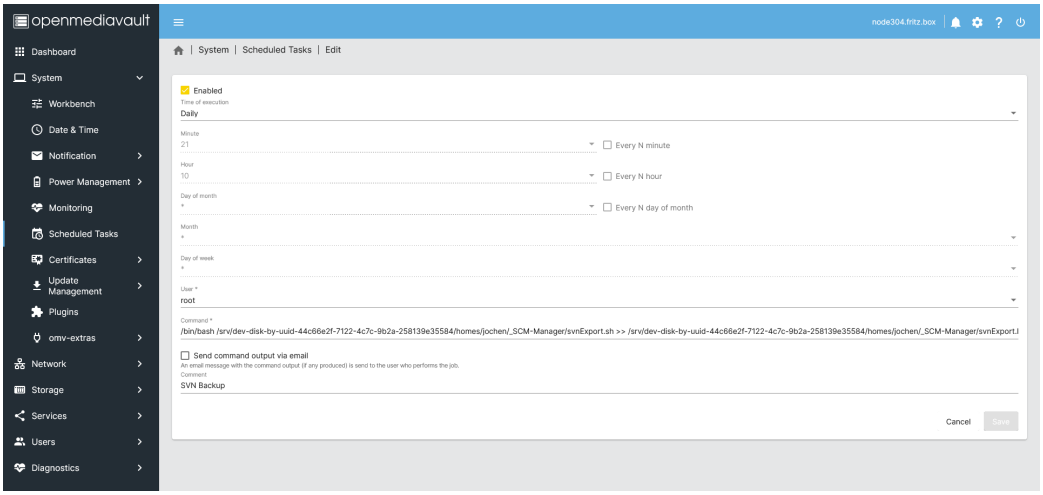

Figure 34: The SCM-Manager backup task

7 Ports

It's always a good idea to remember the used ports. This section will give you an overview about the used ports and the name of the program using this this port.

Program	Port
OMV GUI	80
Portainer	9000
SCM-Manager UI	8080
SCM-Manager ssh	2222

Table 2: Used ports

A svnExport.sh

```
#!/bin/bash
#+------------------------------------------------------------------------------+
#| Script to export configured svn repositories                                 |
#+------------------------------------------------------------------------------+
# Comment line out for debugging purposes
# set -x
#+------------------------------------------------------------------------------+
#| Variable definitions                                                         |
#+------------------------------------------------------------------------------+
# location of all repositories
VAR_PATH_SVN="/srv/dev-disk-by-uuid-44c66e2f-7122-4c7c-9b2a-258139e35584/docker/volumes/scmmanager/_data/repositories/*"
# suffix for SCM organized repositories
SUFFIX_SVN=data
# filename containing real repository name
META_SVN=metadata.xml
# maximum days to keep a backup
INT_AGE=5
# file extension
STR_EXT=gz
# current date
STR_DATE=$(date +%Y-%m-%d)
# get current name of backup folder
DIR_EXPORT=$(dirname "${0}")
DIR_EXPORT=$(realpath "${DIR_EXPORT}")

#+------------------------------------------------------------------------------+
#| Check for valid SVN repository                                               |
#+------------------------------------------------------------------------------+
function is_svn_repository {
    svnlook info "${1}" >/dev/null 2>&1
    echo $?
}

#+------------------------------------------------------------------------------+
#| Get base name for SVN repository                                             |
#+------------------------------------------------------------------------------+
function get_svn_base_name {
    REPOBASE="$(dirname "${1}")/${META_SVN}"
    REPONAME=$(xmlstarlet sel -T -t -m "/repositories/name" -v "/repositories/name" < "${REPOBASE}")
    echo "${REPONAME}"
}

#+------------------------------------------------------------------------------+
#| Create name for SVN respoistory for export                                   |
#+------------------------------------------------------------------------------+
function get_svn_destination_name {
    REPOBASE=$(get_svn_base_name "${1}")
    echo "${DIR_EXPORT}/${REPOBASE}-${STR_DATE}.${STR_EXT}"
}

#+------------------------------------------------------------------------------+
#| Delete backups older than specified age                                      |
#+------------------------------------------------------------------------------+
function drop_old_exports {
    PATTERN=$(get_svn_base_name "${1}")
    FTK=$((1 + "${INT_AGE}"))
    echo "$(date +'%Y%m%d-%H:%M:%S'): Keep the last ${INT_AGE} backups"
    COUNTER=0
    for CURRENT_DUMP in $(find "${DIR_EXPORT}" -name "${PATTERN}-*.${STR_EXT}" | sort -nr); do
        if [[ "${COUNTER}" -lt "${FTK}" ]]; then
            echo "Keep dump [${CURRENT_DUMP}]"
        else
            echo "Delete dump [${CURRENT_DUMP}]"
            rm "${CURRENT_DUMP}"
        fi
        COUNTER=$((COUNTER + 1))
    done
    echo ""
}

#+------------------------------------------------------------------------------+
#| Export SVN repository                                                        |
#+------------------------------------------------------------------------------+
function export_svn_repository {
    VAR_DEST_NAME=$(get_svn_destination_name "${1}")
    echo "$(date +'%Y%m%d-%H:%M:%S'): Dumping repo [${1}] to [${VAR_DEST_NAME}]"
    svnadmin dump "${1}" | gzip > "${VAR_DEST_NAME}"
}

#+------------------------------------------------------------------------------+
```

```
#| Loop over all repositories                                                  |
#+-----------------------------------------------------------------------------+
echo "-------------------------------------------------------------------------------"
echo "***** Start export of SVN repositories for date [${STR_DATE}]"
for VAR_CURRENT_DIR in ${VAR_PATH_SVN}; do
    SVN_REPO="${VAR_CURRENT_DIR}/${SUFFIX_SVN}"
    if [[ $(is_svn_repository "${SVN_REPO}") -eq 1 ]]; then
        echo "$(date +'%Y%m%d-%H:%M:%S'): Skip non SVN repository [${SVN_REPO}]."
        echo ""
    else
        export_svn_repository "${SVN_REPO}"
        drop_old_exports "${SVN_REPO}"
    fi
done
echo ""
```

List of Figures

1 John Lennon . . . 5
2 The OMV login page . . . 7
3 The OMV disks . . . 7
4 The OMV RAID . . . 8
5 The OMV filesystem on the RAID . . . 9
6 The OMV shared folders . . . 10
7 The OMV users home directory . . . 10
8 The OMV CIFS settings . . . 11
9 The OMV CIFS shares . . . 11
10 The OMV advanced CIFS settings . . . 12
11 The OMV ClamAV plugin . . . 13
12 The antivirus settings . . . 13
13 The antivirus on access scan . . . 14
14 The antivirus scheduled scan . . . 14
15 The antivirus system load on scan . . . 15
16 The MiniDLNA plugin . . . 15
17 The MiniDLNA settings . . . 16
18 The MiniDLNA shares . . . 16
19 The OMV extras installation . . . 16
20 The OMV extras plugin . . . 17
21 The OMV autoshutdown plugin . . . 17
22 The OMV network card setup . . . 18
23 The autoshutdown setup 1 . . . 18
24 The autoshutdown setup 2 . . . 19
25 The autoshutdown setup 3 . . . 19
26 The Docker setup . . . 20
27 The Portainer setup . . . 21
28 The Portainer login . . . 21
29 The docker volume scmmanager . . . 22
30 The docker container scmmanager setup 1 . . . 22
31 The docker container scmmanager setup 2 . . . 23
32 The SCM-Manager login . . . 23
33 Cron command for svnExport . . . 23
34 The SCM-Manager backup task . . . 24

List of Tables

1 Change history 3
2 Used ports 24

Glossary

Docker Container based virtualization 3, 5, 8, 16, 18, 19, 21, 22

git git, a version control system 20

NAS Network Attached Storage 3–5, 11, 14, 16, 18, 20

OMV Open Media Vault, a NAS operating system 5, 9, 11, 14, 18, 23

OMV extras Open Media Vault Extras, value added plugins to OMV 15, 19

Portainer Portainer, a container management tool 16, 19, 20, 23

SCM-Manager SCM-Manager, a version control system server 20–23

Subversion Subversion, a version control system 20–22

Synology Manufacturer of NAS systems 3

WOL WOL, starts computer on specific network signal 16, 17