ragtime

Release 1

jan jansen

Dec 27, 2024

CONTENTS:

1 Intro **1**

1.1 Homelab: (did not want to create separate repo) 1

2 readthedocs **3**

2.1 sphinx 3

2.2 integration github 3

3 install something from docker **5**

3.1 using portainer: 5

3.2 using github to build containers: 5

3.3 Project Dockerfiles: 5

3.4 leecher: 5

3.5 embedder: 5

3.6 postgres: 5

3.7 Containers 5

3.8 compose 6

3.9 how to copy pdf files to the container? 6

3.10 using github actions to build docker containers 6

4 TRICK : multiple containers with github **7**

4.1 elasticsearch: 7

5 Elasticsearch **9**

6 storing vector + metadata **11**

7 Github Actions **13**

7.1 build docker images in github 13

7.2 docker publish 13

7.3 adapt docker-publish.yml 13

7.4 where to find the published containers? 13

8 python technology **15**

9 application: **17**

10 technology **19**

11 Welcome to proxmox-dox's documentation! **21**

11.1 intro 21

11.2 Upgrading homelab HP DL 380p 21

11.3 Where can I find more Templates? . . . 22
11.4 reading data from USB stick . . . 22
11.5 ramdisk . . . 23
11.6 sharing a directory for LXC . . . 23
11.7 moving a LXC container . . . 23
11.8 nfs network between linux containers . . . 24
11.9 NFS Host and Client Setup in Proxmox . . . 24
11.10 nfs client . . . 26
11.11 Portainer . . . 26
11.12 server for blender . . . 27
11.13 ollama install and use . . . 27
11.14 configure elasticview . . . 28
11.15 mounting . . . 28
11.16 edit /etc/fstab . . . 28
11.17 Backup pve . . . 28
11.18 RAID . . . 28
11.19 melborp server . . . 29

12 Indices and tables **31**

CHAPTER ONE

INTRO

These are some building blocks in what should become a local LLM for financial info. I do like investing, but I rather not read all the publications.

This software should be able to :

- leech reports
- embed them into a vectordb
- use a local LLM to retrieve and bundle information
- generate a report

1.1 Homelab: (did not want to create separate repo)

I have an old HP dl380p gen8, which I modified: - removed a SAS controller card, and have disks on internal controller - add a 1TB NVME M2 disk, on PCI-e 3 adapter - installed Proxmox

todo: - boot on sata disk on cdrom port - remove RAID1 and use disks as such: space over security - insert NVIDIA P4 single slot

CHAPTER

TWO

READTHEDOCS

- use restructured text

2.1 sphinx

pip install sphinx sphinx-quickstart

2.2 integration github

https://readthedocs.org/dashboard/

include .readthedocs.yml in github repository

```
version: 2

build:
   os: ubuntu-22.04
   tools:
      python: "3.12"

sphinx:
   configuration: conf.py
```

CHAPTER

THREE

INSTALL SOMETHING FROM DOCKER

the docker build that comes with the apt-install from ubuntu does not always cut the cake

> https://docs.docker.com/engine/install/ubuntu/ https://download.docker.com/linux/ubuntu/dists/jammy/pool/stable/amd64/

3.1 using portainer:

> sudo docker volume create portainer_data sudo docker run -d -p 8000:8000 -p 9443:9443 –name portainer –restart=always -v /var/run/docker.sock:/var/run/docker.sock -v portainer_data:/data portainer/portainer-ce:2.21.3

3.2 using github to build containers:

see github actions (and actions.rst in the doc)

3.3 Project Dockerfiles:

the project dockerfiles are within their own directory: - leecher - embedder - postgres

3.4 leecher:

this container waits for an incoming file in the dockervolume : and then converts pdf to txt

3.5 embedder:

this container chops the txt-file and inserts into a postgres database

3.6 postgres:

this a combination of management and a vectordatabase (contains the script to create the ‘document_chunks’ table

3.7 Containers

- the Dockerfiles help to build docker images
- the docker compose file help to build containers

3.8 compose

there are docker-compose.yml files in : - postgres - rag

The one in postgres : - creates 2 containers (database + management) - inits the postgres database as a vectordatabase and creates a table

The one in rag : - creates a volume where you can copy pdf files - creates a volume where converted text files are stored - defines environment variables to access the database (to be changed on your environment!) *docker compose up -d*

3.9 how to copy pdf files to the container?

- the dockervolumes are created using the docker-compose files
- dockervolumes link a directory in docker to a directory on the filesystem

cp 270123.pdf /var/snap/docker/common/var-lib-docker/volumes/rag_leech_data/_data

3.10 using github actions to build docker containers

each time there is is a push toward the github repository, automatically a build of the docker images gets triggered.

I use multiple Dockerfiles, thus multiple Docker images, and I couldn't not figure out the easy-way how to build them with a single script.

So … multiple scripts, which each build a single image.

By default the image gets a name like this repo:main. This can be modified!

CHAPTER

FOUR

TRICK : MULTIPLE CONTAINERS WITH GITHUB

- copy docker-publish.yml to docker-publish2.yml
- change IMAGE_NAME: 'najnesnaj/embed'

(./embedder is de directory in the repo that contains the Dockerfile)

and change :

> **# Build and push Docker image for embedder**
>
> - name: Build and push Docker image (embedder) id: build-and-push-embedder uses: docker/build-push-action@0565240e2d4ab88bba5387d719585280857ece09 # v5.0.0 with:
>
>   context: ./embedder

first login to github

echo "ghp_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxzK" | docker login ghcr.io -u najnesnaj –password-stdin

Login Succeeded

now I can download the image naj@naj-Latitude-5520:/usr/src/ragtime$ sudo docker pull ghcr.io/najnesnaj/embed:main

4.1 elasticsearch:

- sudo docker run -d –name elasticsearch -p 9200:9200 -e "discovery.type=single-node" docker.elastic.co/elasticsearch/elasticsearch:8.15.3
- sudo docker run –name kibana -p 5601:5601 –link elasticsearch:elasticsearch kibana:8.15.3

Kibana has not been configured.

Go to http://0.0.0.0:5601/?code=003763 to get started.

in the elasticsearch container generate token (bin/elasticsearch-create-enrollment-token -s kibana) elasticsearch-users useradd test elasticsearch-users passwd test elasticsearch-users roles -a kibana_admin test

CHAPTER

FIVE

ELASTICSEARCH

for installation see: docker

https://www.elastic.co/guide/en/elasticsearch/reference/current/semantic-search-elser.html

https://www.elastic.co/search-labs/tutorials/examples

CHAPTER

SIX

STORING VECTOR + METADATA

PostgreSQL + pgvector Example If you prefer using PostgreSQL, you can use the pgvector extension, which allows you to store vector embeddings in a PostgreSQL table alongside metadata.

Steps:

- Install pgvector: First, install the pgvector extension.
- Create a table: Create a table that stores both vectors and metadata.
- Insert vectors and metadata: Insert each chunk’s vector along with its metadata.

sql

CHAPTER

SEVEN

GITHUB ACTIONS

7.1 build docker images in github

//github.com/najnesnaj/ragtime/actions/new

Continuous integration

- Publish Docker Container
- Build, test and push Docker image to GitHub Packages.

7.2 docker publish

in https://github.com/najnesnaj/ragtime/tree/main/.github/workflows, docker-publish.yml is added

7.3 adapt docker-publish.yml

Dockerfile definitions in this repository are to be found in subdirectories the docker-publish script has to be made aware

7.4 where to find the published containers?

https://github.com/najnesnaj/ragtime/pkgs/container/ragtime

```
version: ‘3’

services:

    nginx:

        image: nginx:latest

        container_name: nginx_proxy

        ports:

          • “9443:9443”

        volumes:

          • ./nginx.conf:/etc/nginx/nginx.conf

          • /etc/letsencrypt:/etc/letsencrypt # Mount certs

        networks:

          • proxy

    certbot:
        image: certbot/certbot
        container_name: certbot
        command: certonly –webroot –webroot-path=/var/www/certbot -d www.melborp.solutions
        volumes:
          • /etc/letsencrypt:/etc/letsencrypt # Persist certs
          • /var/www/certbot:/var/www/certbot
        networks:
          • proxy
    app:
        image: your_app_image
        container_name: your_app
        expose:
          • “8080” # Internal port
        networks:
          • proxy
networks:
    proxy:
        external: true
```

CHAPTER

EIGHT

PYTHON TECHNOLOGY

A simple application was created with the help of chatgpt.

CHAPTER

NINE

APPLICATION:

- upload pdf
- convert pdf
- correct text
- split text

CHAPTER

TEN

TECHNOLOGY

- Flask : created html templates, which got javascript to handle text, for me: too complex
- FastAPI : similar to Flask
- streamlit : not bothered by html, small python application
-

CHAPTER

ELEVEN

WELCOME TO PROXMOX-DOX'S DOCUMENTATION!

11.1 intro

This doc does not belong here, but did not want to create multiple github repo's.

This is a personal account of venturing into the Proxmox virutualised world:

- running containers
- running local LLM
- no GPU
- cheap

I try to document as much as I can, to avoid solving the same problem twice.

11.1.1 homelab

I bought hp dl380p g8 (generation8) with 16 cores and 32 threads and 256 GIGA bytes of RAM!! for the price of a raspberry pi.

I plan it on using during winter, so its heat is not lost. During rendering or running a LLM it generates 400 Watt. (or consumes for 400 watt expensive electricity)

It could use a GPU, but I hate to spend more money, and it would need a special riser to give way to PCIe x16 double slot.

results sofar :

- LLM runs at 5 tokens / second (llavafile)
- blender takes half an hour for rendering (CYCLES) a single picture
- using it as a ramdisk give 1,5G/s readspead!

11.1.2 software

I choose proxmox as a virtualisation platform, it runs linux containers (LXC), which are kind of cool since they are created quickly, launched quickly. Some problems arise since there are still shared resources with the host… hence the use of the included templates

11.2 Upgrading homelab HP DL 380p

11.2.1 latest BIOS

- download latest firmware (BIOS) for linux in RPM format

- firmware-system-p70-2019.05.24-1.1
- on ubuntu (unpack with Ark)
- look for CPQP7013.6B8 (4MB in size)
- use ilo to upload firmware (update)

11.2.2 M2 disk drive

- M.2 NGFF SSD naar PCI-E 3.0 X16 High-Speed SSD (ashata = 5euro)
- need MVME (one notch) m2 card
- it shows up in BIOS / PCI devices
- on linux :#lsblk it should show up

11.2.3 Sata

- m2 to sata adapter case (aliexpress)
- cable female sata to female Slimline 13pin 7 + 6 (aliexpress)
- m2 sata disk 128G

11.2.4 Bootconfig

there is a - SAS controller - SATA controller (cdrom)

the controllor has a bootorder as well !!!! In order to boot from sata, the sata controller has to boot first!!!

**this is the way to boot an expensive hp server from a simple 2,5 inch sata disk (laptop 5V) ,using the onboard slimline cd-rom connector

11.3 Where can I find more Templates?

- Use:

pveam update

- to update the container template database, then:

pveam available

11.4 reading data from USB stick

the USB stick is readable from the proxmox host:

(do a dmesg to get the device: in this case /dev/sdb1) mount /dev/sdb1 /usbdrive

11.4.1 create a mp on CT (container02)

/dev/mapper/pve-vm–102–disk–1 51290592 28 48652740 1% /container02mp

11.4.2 transfer to CT (name = container02)

on the host mkdir /drive_container02 mount /dev/mapper/pve-vm–102–disk–1 /drive_container02/

11.5 ramdisk

My dl380p gen8 has 256Gb of RAM

11.5.1 Can I use RAM as a disk?

mkdir /tmp/ramdisk chmod 777 /tmp/ramdisk mount -t tmpfs -o size=1024m myramdisk /tmp/ramdisk

or to get something extra…

sudo mount -t tmpfs -o size=10G myramdisk /tmp/ramdisk

11.5.2 speedtest

For Write: dd if=/dev/zero of=/dev/shm/ram bs=1048576 count=4096 oflag=nocache conv=fsync 4096+0 records in 4096+0 records out 4294967296 bytes (4.3 GB, 4.0 GiB) copied, 2.79948 s, 1.5 GB/s

or: dd if=/dev/zero of=/tmp/ramdisk/blok bs=1048576 count=1024 oflag=nocache conv=fsync 1024+0 records in 1024+0 records out 1073741824 bytes (1.1 GB, 1.0 GiB) copied, 0.560324 s, 1.9 GB/s

For Read:

dd if=/tmp/ramdisk/blok of=/dev/null bs=1048576 iflag=nocache,sync conv=nocreat

dd if=/tmp/ramdisk/blok of=/dev/null bs=1048576 iflag=nocache,sync conv=nocreat 1024+0 records in 1024+0 records out 1073741824 bytes (1.1 GB, 1.0 GiB) copied, 0.240446 s, 4.5 GB/s

---

modprobe zram echo 80G | tee /sys/block/zram0/disksize (80G ramdisk) mkfs.ext4 /dev/zram0 (make a filesystem) mkdir /RAM (create a mountingpoint) mount /dev/zram0 /RAM

now you can use the /RAM directory (which will be gone after poweroff)

11.6 sharing a directory for LXC

on the host (pve node) create a shared directory (jan): (created on M2 storage) chown -R nobody:nogroup jan chmod 777 jan

root@pve:/etc/pve/lxc#

11.6.1 modify the lxc

create a directory /mnt/shared (which will be the mounting point)

11.6.2 on PVE modify the lxc config

add this: (mp0: /mnt/pve/M2P2/jan,mp=/mnt/shared)

arch: amd64 cores: 4 features: nesting=1 hostname: decode memory: 5120 net0: name=eth0,bridge=vmbr0,firewall=1,hwaddr=BC:24:11:82:79:94,ip=dhcp,type=veth ostype: ubuntu rootfs: local-lvm:vm-101-disk-0,size=20G swap: 512 unprivileged: 1 mp0: /mnt/pve/M2P2/jan,mp=/mnt/shared

11.7 moving a LXC container

I had a container on a logical volume SCSI and I wanted to move it to logical volume M2

Cloning?

The container shared a directory, and cloning and displacing would mess this up.

Solution: backup & restore from backup on other volume

11.8 nfs network between linux containers

11.8.1 set up a bridge

A Linux bridge interface (commonly called vmbrX) is needed to connect guests to the underlying physical network. It can be thought of as a virtual switch which the guests and physical interfaces are connected to.

11.8.2 define an extra network interface in range 10.0.0.X

11.9 NFS Host and Client Setup in Proxmox

This guide will explain how to set up an NFS (Network File Sharing) server and add it as a remote storage in Proxmox.

1. **Install NFS Server**

   First, log in to the LXC container or the machine where the NFS server will be hosted. Then update the package list and install NFS:

   ```
   apt-get update
   apt-get install sudo -y
   sudo apt install nfs-kernel-server
   ```

   Once installed, create a shared folder:

   ```
   sudo mkdir /home/sharedfolder
   sudo chmod 777 /home/sharedfolder
   ```

   Next, edit the `/etc/exports` file to configure the shared directory for export:

   ```
   sudo nano /etc/exports
   ```

   Add the following line (adjust the IP address and folder accordingly):

   ```
   /home/sharedfolder 192.168.1.0/24(rw,sync,no_subtree_check)
   ```

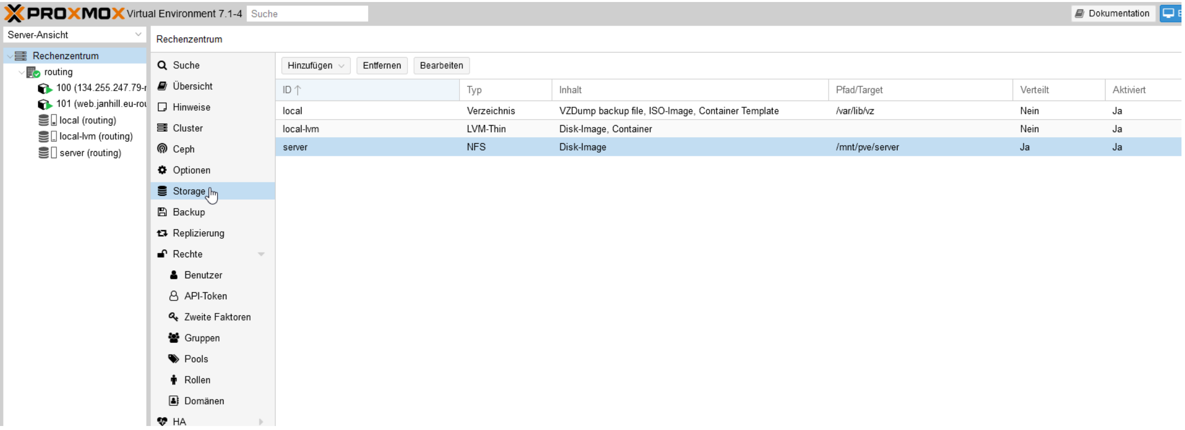
   Save the file and restart the NFS server:

   ```
   sudo exportfs -ra
   sudo systemctl restart nfs-kernel-server
   ```

2. **Configure NFS Storage in Proxmox**

   Now log into your Proxmox host (the machine that will receive the NFS storage) and navigate to:

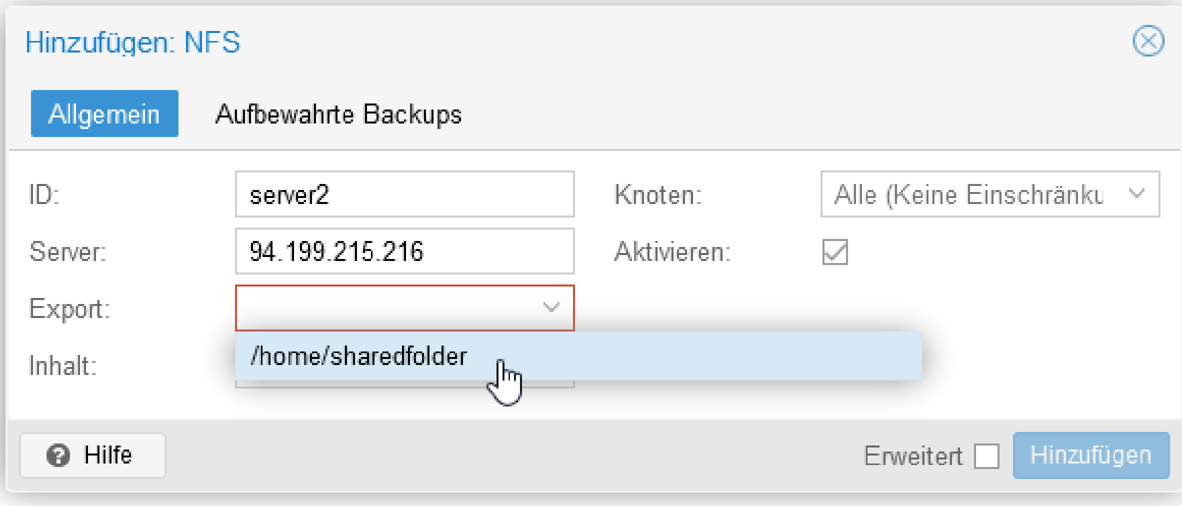
   Datacenter > Storage > Add > NFS



Rechenzentrum

ID ↑	Typ	Inhalt	Pfad/Target	Verteilt	Aktiviert
local	Verzeichnis	VZDump backup file, ISO-Image, Container Template	/var/lib/vz	Nein	Ja
local-lvm	LVM-Thin	Disk-Image, Container		Nein	Ja
server	NFS	Disk-Image	/mnt/pve/server	Ja	Ja

Here, enter the NFS server's IP address and select the shared directory.

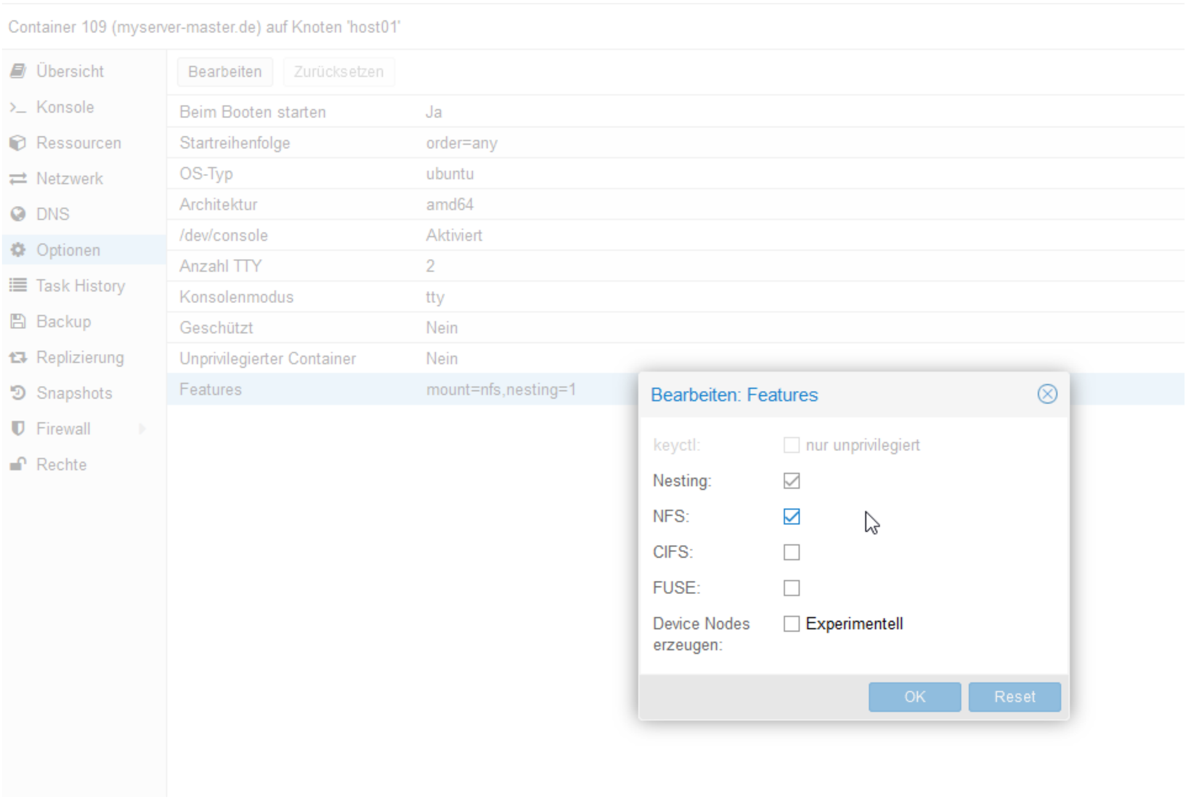


ID:	server2	Knoten:	Alle (Keine Einschränku
Server:	94.199.215.216	Aktivieren:	☑
Export:	/home/sharedfolder		
Inhalt:			

Choose the desired contents for the storage (ISO images, containers, backups, etc.) and click *Add*.

3. **Set Permissions on LXC Container (If Applicable)**

   If the NFS share will be used in an LXC container, ensure that permissions for NFS usage are set correctly:



Container 109 (myserver-master.de) auf Knoten 'host01'

Beim Booten starten	Ja
Startreihenfolge	order=any
OS-Typ	ubuntu
Architektur	amd64
/dev/console	Aktiviert
Anzahl TTY	2
Konsolenmodus	tty
Geschützt	Nein
Unprivilegierter Container	Nein
Features	mount=nfs,nesting=1

Bearbeiten: Features

keyctl:	□ nur unprivilegiert
Nesting:	☑
NFS:	☑
CIFS:	□
FUSE:	□
Device Nodes erzeugen:	□ Experimentell

Check the *NFS* box under *Options* for the LXC container.

That’s it! You have successfully set up an NFS server and added it to Proxmox as remote storage.

11.10 nfs client

apt install nfs-common

mount -t nfs 10.0.0.104:/share /mnt/nfson104

11.10.1 under node pve

watch out for features

rch: amd64 cores: 32 *features: mount=nfs,nesting=1* hostname: llama memory: 64000 net0: name=eth0,bridge=vmbr0,firewall=1,hwaddr=BC:24:11:58:0C:3A,ip=dhcp,type=veth net1: name=eth5,bridge=vmbr1,firewall=1,hwaddr=BC:24:11:3A:F0:4A,ip=10.0.0.100/24,type=veth ostype: ubuntu rootfs: local-lvm:vm-100-disk-0,size=40G swap: 512

11.11 Portainer

11.11.1 using a script

use a template for the linux container

a script from : https://raw.githubusercontent.com/tteck/Proxmox/refs/heads/main/install/docker-install.sh

systemctl start docker systemctl status docker

https://192.168.0.182:9443 (your IP)

11.11.2 by hand

- create CT ubuntu22 with template
- apt update
- sudo apt install docker.io -y
- sudo systemctl status docker
- sudo usermod -aG docker $USER (add current logged on user to docker group)
- docker pull portainer/portainer-ce:latest
- docker run -d -p 9000:9000 –restart always -v /var/run/docker.sock:/var/run/docker.sock portainer/portainer-ce:latest

11.11.3 exporting & importing

this seems to work between systems:

(origin) sudo docker save ollama/ollama:latest > my-ollama.tar (target) sudo docker load < my-ollama.tar

11.12 server for blender

Although the hp dl380 was 5 times faster than my i7 laptop, it is still slow, takes half an hour to render a picture.

One can adjust setting within blender to speed up things a bit, but still …

11.12.1 howto render?

- download blender 4.2

cd blender-4.2.0-linux-x64/

./blender -b /home/naj/misvormde-donut1.blend -E CYCLES -f 1

11.12.2 faster / less good

./blender -b /home/naj/misvormde-donut1.blend -E BLENDER_EEVEE_NEXT -f 1

11.12.3 at the movies

./blender -b /home/naj/misvormde-donut1.blend -E BLENDER_EEVEE_NEXT -s 10 -e 500 -t 2 -a ./blender -b /home/naj/misvormde-donut1.blend -E BLENDER_EEVEE_NEXT -s 1 -e 100 -t 2 -a

11.13 ollama install and use

curl -fsSL https://ollama.com/install.sh | sh

11.13.1 using a container

ollama-model-gemma2 was mounted using a volume and the image exported

sudo docker import ollama.tar ollama:latest

11.14 configure elasticview

11.14.1 elasticsearch password

connect with term to container: bin/elasticsearch-reset-password -u elastic bin/elasticsearch-reset-password -u elastic -i (this allows for setting it yourself)

11.14.2 testing elasticview connection

curl -X GET -k -u elastic:ebktuBhBHtE7N+JeBbIV "https://192.168.0.121:9200/_cluster/health/?pretty"

{

> "cluster_name" : "docker-cluster", "status" : "green", "timed_out" : false, "number_of_nodes" : 1, "number_of_data_nodes" : 1, "active_primary_shards" : 1, "active_shards" : 1, "relocating_shards" : 0, "initializing_shards" : 0, "unassigned_shards" : 0, "delayed_unassigned_shards" : 0, "number_of_pending_tasks" : 0, "number_of_in_flight_fetch" : 0, "task_max_waiting_in_queue_millis" : 0, "active_shards_percent_as_number" : 100.0

}

11.14.3 getting website certificate

connect to http://192.168.0.121

firefox/settings/privicy&security/certificates to add exception

11.15 mounting

mkdir /SCSIdata for the data on the scsidisk /dev/sdb mkdir /M2data for the data on the M2 disk /dev/nvme0n1p2

# mount /dev/sdb /SCSIdata

# mount /dev/nvme0n1p2 /M2data

11.16 edit /etc/fstab

/dev/nvme0n1p2 /M2data ext4 defaults 0 2 /dev/sdb /SCSIdata ext4 defaults 0 2

11.17 Backup pve

Proxmox node now start from a sata m2 in the CDrom slot

A tar copy is made to M2data from /etc directory

In case of crash : reinstall proxmox on sata disk en copy backup.tar to /etc

11.18 RAID

previously 2 600GB SAS disks were in RAID1 in the same volumegroup.

I have no spare disk, nor do I want to spend money on old tech.

Reconfigure : each disk is within own volume group, no more RAID, more (free) space. Tool would not let me otherwise.

11.18.1 No more raid :

machine on one disk are backupped to other disk, and vice versa one disk remains VG (volume group) within proxmox : used for LXC en VM images, backup other disk contains ext4 filesystem and is a directory

11.19 melborp server

11.19.1 solution for problem running pgadmin container and nginx

Configure Nginx: Create a new configuration file for your domain in /etc/nginx/sites-available/melborp.solutions:

nginx

```
server {
    listen 80; server_name www.melborp.solutions;

    location / {
        proxy_pass http://localhost:8888; # Forward traffic to the pgAdmin container proxy_set_header
        Host $host; proxy_set_header X-Real-IP $remote_addr; proxy_set_header X-Forwarded-For
        $proxy_add_x_forwarded_for; proxy_set_header X-Forwarded-Proto $scheme;

    }

}
```

CHAPTER

TWELVE

INDICES AND TABLES

- genindex
- modindex
- search