Proxmox VE Homelab

# Was ist Proxmox VE

## Merkmale von Proxmox VE

Forum: https://forum.proxmox.com/)

## Kernel-based Virtual Machine versus Linux Container

Quick Emulator; https://www.qemu.org/

## Unterschiede zu Softwarealternativen (Beispiele)

# Proxmox VE installieren

## Systemanforderungen

## Proxmox Virtual Environment installieren

Proxmox VE ISO-Abbild https://www.proxmox.com/de/downloads/proxmox-virtual-environment/iso

### Proxmox als bare-metal-server

balenaEtcher https://www.balena.io/etcher/

df

sudo umount /dev/sda3

sudo mkfs.vfat /dev/sda -I

sudo dd if=/dev/null of=/dev/sda bs=1MB

sudo dd if=/dev/zero of=/dev/sda bs=1MB

##### Schritt 2: USB-Boot ermöglichen

automatische Installation:  
 https://pve.proxmox.com/wiki/Automated\_Installation#Overview.

### Proxmox VE auf einem Raspberry Pi

Raspberry Pi Imager (<https://www.raspberrypi.com/software/>

ssh pi@<ihre-pi-ip-adresse> (z.B.: ssh pi@192.168.178.38)

sudo raspi-config

ssh-keygen -R <server>

sudo apt-get update

sudo apt-get upgrade

sudo apt-get install curl

sudo nano /etc/hosts

127.0.0.1 raspberrypi5

hostname –ip-address

sudo passwd root

Proxmox Ports Repository’s (https://github.com/jiangcuo/Proxmox-Port)

curl -L https://mirrors.apqa.cn/proxmox/debian/pveport.gpg | sudo tee /usr/share/keyrings/pveport.gpg >/dev/null

echo "deb [deb=arm64 signed-by=/usr/share/keyrings/pveport.gpg] https://mirrors.apqa.cn/proxmox/debian/pve bookworm port" | sudo tee /etc/apt/sources.list.d/pveport.list

sudo apt update

sudo apt-get install ifupdown2

sudo apt install proxmox-ve postfix open-iscsi pve-edk2-firmware-aarch64

hostname -I

<https://pve.proxmox.com/wiki/Proxmox_VE_inside_VirtualBox>

### Proxmox VE in einem Browser starten

https://www.wundertech.net/how-to-rename-a-node-in-proxmox/).

### Proxmox VE Installations-Nachbereitungen (post install)

<https://johnscs.com/remove-proxmox51-subscription-notice/>

sed -Ezi.bak "s/(Ext.Msg.show\(\{\s+title: gettext\('No valid sub)/void\(\{ \/\/\1/g" /usr/share/javascript/proxmox-widget-toolkit/proxmoxlib.js && systemctl restart pveproxy.service

# Proxmox VE administrieren

## Das Proxmox VE Graphical User Interface

### Die GUI-Kopfzeile

### Der Ressourcenbaum

### Das Content Panel (Hauptbereich)

### Das Log Panel (Protokollbereich)

## Das Rechenzentrum administrieren

<https://www.spice-space.org/download.html>

### Storage

### Backup

dir /mnt/pve/extSSD1/dump

scp -r benutzername@zielserver.de:/pfad/zum/remote/verzeichnis /pfad/zum/lokalen/verzeichnis

scp -r root@192.168.178.29:/var/lib/vz/dump C:\Users\mike\Downloads

scp -r /pfad/zum/lokalen/verzeichnis benutzername@zielserver.de:/pfad/zum/remote/verzeichnis

<https://filezilla-project.org/>

https://pve.proxmox.com/wiki/Backup\_and\_Restore#vzdump\_jobs

### Replizierung

### Rechte

https://pve.proxmox.com/wiki/User\_Management

### HA

### SDN

### ACME

### Firewall

https://pve.proxmox.com/wiki/Firewall

### Metrikserver

### Resource-Mappings

### Benachrichtigungen

apt install postfix libsasl2-modules -y

nano /etc/postfix/main.cf

…

mydestination = $myhostname, localhost.$mydomain, localhost

#relayhost =

mynetworks = 127.0.0.0/8

…

relayhost = <smtp.mailserver.de>:<port>

smtp\_sasl\_auth\_enable = yes

smtp\_sasl\_password\_maps = hash:/etc/postfix/sasl\_passwd

smtp\_sasl\_security\_options = noanonymous

smtp\_tls\_CAfile = /etc/postfix/cacert.pem

smtp\_use\_tls = yes

sender\_canonical\_classes = envelope\_sender, header\_sender

sender\_canonical\_maps = regexp:/etc/postfix/sender\_canonical\_maps

smtp\_header\_checks = regexp:/etc/postfix/header\_check

nano /etc/postfix/sender\_canonical\_maps

/.+/ <absender@adresse.de>

nano /etc/postfix/header\_check

/From:.\*/ REPLACE From: <absender@adresse.de>

nano /etc/postfix/sasl\_passwd

<smtp.mailserver.de>:<port> <user>:<passwort>

postmap /etc/postfix/sasl\_passwd

chown root:root /etc/postfix/sasl\_passwd /etc/postfix/sasl\_passwd.db

chmod 0600 /etc/postfix/sasl\_passwd /etc/postfix/sasl\_passwd.db

cat /etc/ssl/certs/ssl-cert-snakeoil.pem | tee -a /etc/postfix/cacert.pem

postfix reload

echo "Proxmox SMTP Test" | mail -s "SMTP Test" <empfänger@adresse.de> -r <absender@adresse.de>

### Support

## Einen PVE-Node administrieren

### Suche

### Übersicht

### Shell

### System

https://pve.proxmox.com/wiki/Network\_Configuration.

### DNS

### Hosts

### Optionen

### Zeit

### System Log

### Updates

apt-get update

apt-get dist-upgrade

### Firewall

### Disks / Ceph

### Replizierung

### Task History

grep -R "keyword" /var/log/pve/tasks

grep -R "VM" /var/log/pve/tasks

### Subskription

## Brevier Linux-Befehle

dmesg -c

lsusb

lsblk

udevadm info -a -n /dev/sdb

gdisc

blkid -o list

cat /proc/partitions

cat /etc/mtab

mount /dev/sdb1 /mnt/pve/extHD1

fdisk -l

mkfs.ext4 /dev/sdb1

dmesg

fdisk -l

fdisk /dev/sdb

fdisk /dev/sdb

mkfs.ext4 /dev/sdb1

e2label /dev/sdb1 disk-part1

mkfs -t vfat /dev/sdb1

mkfs.vfat /dev/sdb1 -n EXTHD

mkfs.ntfs /dev/sdb1

mkfs.ntfs /dev/sdb1 -f -L EXTHD

mount

# Festplatten-Datenspeicher

## SSD Spezifikationen

## Das Proxmox VE-Speichermodell

### Speichertypen

<https://docs.ceph.com/en/quincy/>

## Festplattenspeicher einrichten

https://pve.proxmox.com/wiki/ZFS\_on\_Linux

## Aktionen Festplattenspeicher

### Vergrößern

### Verkleinern

lvdisplay | grep "LV Path\|LV Size"

e2fsck -fy /dev/pve/vm-100-disk-0

resize2fs /dev/pve/vm-100-disk-0 6G

lvreduce -L 6G /dev/pve/vm-100-disk-0

nano /etc/pve/lxc/100.conf

rootfs: local-lvm:vm-100-disk-0,size=6G

pct start 100

pct enter 100

df -h

### SSD-Trim

hdparm -I /dev/sda| grep -i TRIM

pct fstrim <ID>

bash -c "$(wget -qLO - https://github.com/tteck/Proxmox/raw/main/misc/fstrim.sh)"

systemctl status fstrim.timer

fstrim -av

dd if=/dev/zero of=/mytempfile

rm -f /mytempfile

qm shutdown <VM-ID>.

qemu-img convert -O qcow2 image.qcow2 image-new.qcow2

pvesm path extSSD1:150/vm-150-disk-0.qcow2

cd /mnt/pve/extSSD1/images/150

qemu-img info vm-150-disk-0.qcow2

df -h

sudo dd if=/dev/urandom of=/tempFile bs=1M count=3800

df -h

qemu-img info vm-150-disk-0.qcow2

sudo rm /tempFile

df -h

(sudo) dd if=/dev/zero of=/mytempfile

(sudo) rm -f /mytempfile

qm shutdown 150

cd /mnt/pve/extSSD1/images/150

qemu-img convert -O qcow2 \  
 vm-150-disk-0.qcow2 \  
 vm-150-disk-0.qcow2-new

qemu-img info vm-150-disk-0.qcow2-new

### Festplatten-Speicher verschieben

## Gästen weiteren Festplatten-Speicher zuordnen

## Troubleshooting

# LXC-Container

## LXC-Container erstellen

### das manuelle Erstellen von LXC-Containern

http://download.proxmox.com/images/system/

<https://www.turnkeylinux.org/>

apt update

apt upgrade

apt autoremove -y

dpkg-reconfigure tzdata

adduser mike

apt install sudo

usermod -aG sudo mike

### das Skript gesteuerte Erstellen von LXC-Containern

<https://github.com/community-scripts/ProxmoxVE>

bash -c "$(wget -qLO - https://github.com/tteck/Proxmox/raw/main/ct/mqtt.sh)"

## LXC-Container administrieren

(https://linuxcontainers.org/

### Betrieb des LXC-Containers / Grundfunktionen

lxc-attach --name 100

ssh mike@192.168.178.150

https://ping.flenny.net/2016/ssh-into-a-proxmox-lxc-container/.

### Funktionen der Administrationsleiste im rechten Seitenbereich

[https://pve.proxmox.com/pve-docs/pve-admin-guide.html#markdown\_basics](https://pve.proxmox.com/pve-docs/pve-admin-guide.html" \l "markdown_basics)

<https://www.markdownguide.org/>

## Sonderfälle LXC-Container administrieren

### Einen unprivilegierten in einen privilegierten überführen

### Ein USB Gerät durchreichen

mkdir /media/usbstick

ls -la /media

mount -t ext4 -o rw /dev/sdb1 /media/usbstick

apt install ntfs-3g -y

mount -t ntfs-3g -o rw /dev/sdb1 /media/usbstick

nano /etc/pve/lxc/<Gast-Nr>.conf

nano /etc/pve/lxc/105.conf

mp0:/media/usbstick,mp=/media/usbstick

nano /etc/udev/rules.d/99-usb-sereial.rules

SUBSYSTEM=="<Subsystem>", ATTRS{idVendor}=="<Vendor-ID>", ATTRS{idProduct}=="<Produkt-ID>" SYMLINK+="<Symlink-Name>"

KERNEL=="ttyUSB[0-9]\*",MODE="0666"

udevadm trigger

ls -la /dev/ttyM\*

nano /etc/pve/lxc/100.conf

lxc.mount.entry: /dev/ttyMC0 dev/ttyUSB1 none bind,optional,create=file

ls -la /dev/tty\*

### Debuggen

lxc-start -n 104 -d –logfile 104-lxc-log.log -l debug

tail -f 104-lxc-log.log

nano 104-lxc-log.log

### LXC-Container und Virtuelle Maschinen löschen

dir /etc/pve/lxc

qm destroy <ID> --purge

qm unlock <ID>

### LXC-Container umbenennen

pct set <VMID> --hostname <newname>

pct set 101 --hostname mqtt

### Mit SSH auf einen LXC-Container zugreifen

systemctl status ssh

apt install openssh-server

nano /etc/ssh/sshd\_config

..

# Athentification

LoginGraceTime 120

PermitRootLogin yes

StrictModes yes

PasswordAuthentication yes

..

systemctl restart ssh

ssh root@192.168.178.101

scp root@192.168.178.101:/test\_file C:\Users\mike\Downloads

# Kernel-based Virtual Maschines

## KVM’s erstellen

### Das manuelle Erstellen über das GUI

https://cdimage.debian.org/cdimage/archive/12.4.0/amd64/iso-cd/debian-12.4.0-amd64-netinst.iso f

[https://ubuntu.com/download/desktop/](https://ubuntu.com/download/desktop/thank-you?version=24.04&architecture=amd64&lts=true)

https://ubuntu.com/download/server/

### Das manuelle Erstellen mit Hilfe des CLI

qm create 600 --name debianqcow --memory 4096 --sockets 1 --cores 2 --net0 virtio,bridge=vmbr0

qm disk import 600 /mnt/pve/extSSD1/images/500/vm-500-disk-0.qcow2 extSSD1 --format qcow2

qm set 600 --scsi0 extSSD1:600/vm-600-disk-0.qcow2,discard=on --boot order='scsi0;net0'

qm start 600

### das Erstellen über Cloud-Init

Ubuntu: <https://cloud-images.ubuntu.com/>

Debian: <https://cloud.debian.org/images/cloud/>

<https://cloud-images.ubuntu.com/noble/current/noble-server-cloudimg-amd64.img>)

qm create 5000 --memory 2048 --core 2 --name ubuntu-24-04-LTS --net0 virtio,bridge=vmbr0

cd /var/lib/vz/template/iso/

qm importdisk 5000 noble-server-cloudimg-amd64.img <YOUR-STORAGE-HERE>

qm importdisk 5000 noble-server-cloudimg-amd64.img local-lvm

qm set 5000 --scsihw virtio-scsi-pci --scsi0 <YOUR STORAGE HERE>:vm-5000-disk-0

qm set 5000 --scsihw virtio-scsi-pci --scsi0 local-lvm:vm-5000-disk-0

qm set 5000 --boot c --bootdisk scsi0

qm set 5000 --serial0 socket --vga serial0

qm set 5000 --ide2 <YOUR STORGE HERE>:cloudinit

qm set 5000 --ide2 local-lvm:cloudinit

<https://pve.proxmox.com/wiki/Cloud-Init_Support>.

### Das Skript gesteuerte Erstellen von KVM’s

<https://tteck.github.io/Proxmox/>

bash -c "$(wget -qLO - https://github.com/tteck/Proxmox/raw/main/vm/debian-vm.sh)"

## Fertige Maschinen übertragen

### VMware ESXi  ablösen

[https://pve.proxmox.com/wiki/Migrate\_to\_Proxmox\_VE#Migration](https://pve.proxmox.com/wiki/Migrate_to_Proxmox_VE" \l "Migration).

<https://www.youtube.com/watch?v=H1t6hxCoiZw>. I

### VMs aus VirtualBox als KVM in Proxmox importieren

scp -r /pfad/zum/lokalen/verzeichnis benutzername@zielserver.de:/pfad/zum/remote/verzeichnis

scp ./Documents/"Money Box"/"Money Box-disk001.vmdk" root@192.168.178.29:/tmp

scp ./"VirtualBox VMs"/"Money Box"/"Debian Custum CTF-disk001.vdi" root@192.168.178.29:/tmp

cd /temp

qemu-img convert -f vmdk 'Money Box-disk001.vmdk' -O qcow2 'Money Box-disk001.qcow2'

qemu-img convert -f vdi 'Debian Custum CTF-disk001.vdi' -O qcow2 'Debian Custum CTF-disk001.qcow2'

cd /temp

qm disk import 800 'Money Box-disk001.vmdk' extSSD1 --format qcow2

### Eine Maschine mit Clonezilla portieren

### Eine fertige ova-Datei importieren

## Funktionen der Administrationsleiste im Hauptbereich

## KVM’s administrieren Sonderfälle

### Der Shutdown einer KVM scheitert

ps aux | grep <VMID>

kill -9 <PID>

### USB-Port an eine KVM durchreichen

### CLI Anweisungen

<https://pve.proxmox.com/pve-docs/qm.1.html>.

### Umwandlung zwischen Abbildformaten

pvesm path extSSD1:500/vm-500-disk-0.raw

cd /mnt/pve/extSSD1/images/500

qemu-img convert -p -f raw -O qcow2 <source> <target>

qemu-img convert -p -f raw -O qcow2 \  
 vm-500-disk-0.raw \  
 vm-500-disk-0.qcow2

# Der Proxmox-Backup-Server

<https://www.proxmox.com/de/proxmox-backup-server/uebersicht>

## Systemanforderungen

## Proxmox Backup Server installieren

https://www.proxmox.com/de/downloads.

https://enterprise.proxmox.com/iso/proxmox-backup-server\_3.2-1.iso

## Über das Web Interface zugreifen und administrieren

sgdisk --zap-all /dev/sdX

## Integration in Proxmox VE

<https://liersch.it/2023/09/proxmox-backup-server-auf-raspberry-pi-installieren/>).

https://www.youtube.com/watch?v=4Jf2T45tfEU

# Clonzilla

(<https://clonezilla.org/>

## Clonzilla auf Proxmox-VE installieren und starten

<https://clonezilla.org/downloads.php>

## Ein qcow2-Image verkleinern

pvesm path extSSD1:150/vm-150-disk-0.qcow2

cd /mnt/pve/extSSD1/images/150

qemu-img info vm-150-disk-0.qcow2

pvesm path extSSD1:501/vm-501-disk-0.qcow2

qemu-img convert -O qcow2 \  
 /mnt/pve/extSSD1/images/150/vm-150-disk-0.qcow2 \  
 /mnt/pve/extSSD1/images/501/vm-501-disk-0.qcow2

cd /mnt/pve/extSSD1/images/501

ls -as

qemu-img info vm-501-disk-1.qcow2

## Eine Maschine mit Clonezilla nach Proxmox VE portieren

# Home Automation und Home Automation Stack

https://www.home-assistant.io/

## Das Konzept von Home Assistant

## Installation von Home Assistant

bash -c "$(wget -qLO - https://github.com/tteck/Proxmox/raw/main/vm/haos-vm.sh)"

## erste Schritte

[http://homeassistant.local:8123](http://homeassistant.local:8123/),

### Automatisch erkannte Geräte

### Integrationen hinzufügen

### Automatisierungen

### Add-ons

### Home Assistant und YAML-Programmierung

alias: "Shelly\_RGB-Lampe\_Timer "  
description: ""  
trigger:  
 - platform: time  
 at: "20:00:00"  
condition: []  
action:  
 - type: turn\_on  
 device\_id: 0193d2437b79e92d6e9eb9bf40fedf03  
 entity\_id: a3f10c55f7329cc3c9dd93bbea34d5b5  
 domain: light  
 - delay:  
 hours: 0  
 minutes: 30  
 seconds: 0  
 milliseconds: 0  
 - service: light.turn\_off  
 metadata: {}  
 data: {}  
 target:  
 entity\_id: light.shellycolorbulb\_3494546b2c8d  
mode: single

# Mosquitto

## Mosquitto installieren

apt-get update

apt-get upgrade

apt-get install mosquitto

## Administration / Nachbereitung

systemctl status mosquitto

systemctl restart mosquitto

systemctl start mosquitto

systemctl stop mosquitto

mosquitto\_sub -h localhost -v -t test

nano /etc/mosquitto/mosquitto.conf

...

listener 1883

allow\_anonymous true

socket\_domain ipv4

apt-get install mosquitto-clients

mosquitto\_sub -h localhost -v -t test oder

mosquitto\_sub -h 192.168.178.151 -v -t test oder

mosquitto\_sub -h mqtt -v -t test

mosquitto\_pub -h localhost -t test -m 'Hallo' oder

mosquitto\_pub -h 192.168.178.151 -t test -m 'Hallo' oder

mosquitto\_pub -h mqtt -t test -m 'Hallo'

http://www.steves-internet-guide.com/mqtt-username-password-example/.

nano /etc/mosquitto/passwords.txt

mosquitto\_passwd -U /etc/mosquitto/passwords.txt

...:

listener 1883

allow\_anonymous false

password\_file /etc/mosquitto/passwords.txt

socket\_domain ipv4

mosquitto\_sub -h mqtt -v -t test -u tasmota -P raspberry

mosquitto\_pub -h mqtt -t test -m "Hallo" -u tasmota -P raspberry

## Mosquitto an Home Assistant anbinden

## Beispiel Tasmota

<https://sonoff.tech/>

<https://www.tasmota.info/>

mosquitto\_pub -h mqtt -t cmnd/sonoff\_01/POWER -m toggle -u tasmota -P raspberry

mosquitto\_sub -h mqtt -v -t stat/sonoff\_01/POWER -u tasmota -P raspberry

https://tasmota.github.io/docs/Commands#setoptions/.

mqtt:  
 switch:  
 - unique\_id: tasmota\_switch\_02  
 name: "Tasmota Switch 02"  
 state\_topic: "stat/tasmota\_02/RESULT"   
 value\_template: "{{ value\_json.POWER }}"  
 command\_topic: "cmnd/sonoff\_02/POWER"  
 payload\_on: "ON"  
 payload\_off: "OFF"  
 availability\_topic: "tele/sonoff\_02/LWT"  
 payload\_available: "Online"  
 payload\_not\_available: "Offline"

# Docker

<https://www.docker.com/>

## Docker Installation

apt update && apt upgrade && apt autoremove -y

adduser mike

usermod -aG sudo mike

sudo nano /etc/sudoers

mike

sudo apt install docker.io curl net-tools -y

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

sudo sh get-docker.sh

sudo systemctl enable docker

sudo systemctl start docker

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

sudo chmod +x /usr/local/bin/docker-compose

sudo ln -s /usr/local/bin/docker-compose /usr/bin/docker-compose

sudo usermod -a -G docker mike

sudo docker run -d -p 8000:8000 -p 9000:9000 --name portainer --restart unless-stopped -v /var/run/docker.sock:/var/run/docker.sock -v /opt/portainer:/data portainer/portainer-ce:latest

sudo docker ps

docker ps -a

docker stop <container\_name> / <container\_id>

docker start <container\_name> / <container\_id>

docker restart <container\_name> / <container\_id>

docker rm <container\_name> / <container\_id>

docker images

docker rmi <image\_name>

docker rmi <image\_name> -force

docker system prune

docker system prune -a

https://devtodevops.com/fix-no-space-left-on-device-docker/.

## Gotify

<https://gotify.net/>

curl "http://<IP-Adresse-Gotify>:<Port>/message?token=<App-Token>" -F "title=<Titel>" -F "message=<Nachricht>" -F "priority=5"

curl "http://192.168.178.154:6767/message?token=As6U5lK.22p6\_sp" -F "title=Test" -F "message=Dies ist eine Testnachricht" -F "priority=5"

## PDF-Dateien bearbeiten

<https://www.stirlingpdf.com/>

<https://github.com/Stirling-Tools/Stirling-PDF>

mkdir -p /opt/stirling-pdf/trainingData

mkdir /opt/stirling-pdf/extraConfigs

docker run -d \

-p 8080:8080 \

-v /opt/stirling-pdf/trainingData:/usr/share/tesseract-ocr/5/tessdata \

-v /opt/stirling-pdf/extraConfigs:/configs \

-v /opt/stirling-pdf/logs:/logs \

-e DOCKER\_ENABLE\_SECURITY=true \

--name stirling-pdf \

--restart always \

frooodle/s-pdf:latest

cd /opt/stirling-pdf/trainingData

wget https://github.com/tesseract-ocr/tessdata/raw/main/deu.traineddata # full

wget https://github.com/tesseract-ocr/tessdata\_fast/raw/main/deu.traineddata # fast

## Geräte-Verfügbarkeit im Netzwerk überwachen

https://uptime.kuma.pet/

mkdir -p /opt/uptime-kuma

docker run -d \  
 -p 3001:3001 \  
 -v /opt/uptime-kuma:/app/data \  
 --name uptime-kuma \  
 --restart=always \  
 louislam/uptime-kuma:1

# Zigbee2MQTT

<https://www.zigbee2mqtt.io/>

## Installation

<https://helper-scripts.com/scripts?id=Zigbee2MQTT>

<https://github.com/tteck/Proxmox/discussions/2917>

<sudo> apt-get update

<sudo> apt-get install curl

curl -fsSL https://deb.nodesource.com/setup\_20.x | sudo -E bash -

<sudo> apt-get install -y nodejs git make g++ gcc libsystemd-dev make

node --version

npm --version

<sudo> mkdir /opt/zigbee2mqtt

<sudo> chown -R ${USER}: /opt/zigbee2mqtt

git clone --depth 1 https://github.com/Koenkk/zigbee2mqtt.git /opt/zigbee2mqtt

cd /opt/zigbee2mqtt

npm ci

npm run build

## ZigBee-Stick USB-Paththrough einrichten

nano /etc/udev/rules.d/99-usb-sereial.rules

SUBSYSTEM=="tty", ATTRS{idVendor}=="10c4", ATTRS{idProduct}=="ea60" SYMLINK+="ttyZIGBEE0"

KERNEL=="ttyUSB[0-9]\*" , MODE="0666"

udevadm trigger

ls -la /dev/ttyZ\*

nano /etc/pve/lxc/103.conf

lxc.mount.entry: /dev/ttyZIGBEE0 dev/ttyUSB0 none bind,optional,create=file

lxc.mount.entry: /dev/serial/by-id dev/serial/by-id none bind,optional,create=dir

test -w /dev/ttyUSB0 && echo success || echo failure

ls -l /dev/serial/by-id

## zigbee2mqtt konfigurieren und starten

nano /opt/zigbee2mqtt/data/configuration.yaml

# keine Anbindung an Home Assistant

homeassistant: false

# Zugang über einen Browser aktiviert

frontend: true

# mqtt-Broker-Einstellungen

mqtt:

# alle Topics beginnen mit ...

base\_topic: zigbee2mqtt

# server Buchkapitel mit dem Namen mqtt; mqtt://localhost

server: mqtt:mqtt

# Benutzername und Passwort; entfallen bei ungesch. Broker

user: tasmota

password: raspberry

serial:

# port: für diesen Zigbee-Stick /dev/serial/by-id

port: >-

/dev/serial/by-id/usb-Nabu\_Casa\_SkyConnect\_v1.0\_38be2388b09aed11b7657c08a8669f5d-if00-port0

# adapter: für diesen Zigbee-Stick ezsp

adapter: ezsp

cd /opt/zigbee2mqtt

npm start

## Mit zigbee2mqtt arbeiten

mosquitto\_pub -h mqtt -t zigbee2mqtt/bridge/request/permit\_join -m "true" -u tasmota -P raspberry

devices:

'0x90ab96fffe17d02f':

friendly\_name: '0x90ab96fffe17d02f'

mosquitto\_pub -t 'zigbee2mqtt/0x90ab96fffe17d02f/set' -m '{ "state": "ON" }'

mosquitto\_pub -t 'zigbee2mqtt/0x90ab96fffe17d02f/set' -m '{ "state": "OFF" }'

mosquitto\_pub -t 'zigbee2mqtt/SiedeboardLampe/set' -m '{ "state": "ON" }'

mosquitto\_pub -t 'zigbee2mqtt/SiedeboardLampe/set' -m '{ "state": "OFF" }'

<sudo> nano /etc/systemd/system/zigbee2mqtt.service

...

[Unit]

Description=zigbee2mqtt

After=network.target

[Service]

Environment=NODE\_ENV=production

Type=notify

ExecStart=/usr/bin/node index.js

WorkingDirectory=/opt/zigbee2mqtt

StandardOutput=inherit

# Or use StandardOutput=null if you don't want Zigbee2MQTT messages filling syslog, for more options see systemd.exec(5)

StandardError=inherit

WatchdogSec=10s

Restart=always

RestartSec=10s

User=root

[Install]

WantedBy=multi-user.target

...

systemctl daemon-reload

systemctl start zigbee2mqtt

systemctl status zigbee2mqtt

systemctl restart zigbee2mqtt

systemctl stop zigbee2mqtt

<sudo> journalctl -xeu zigbee2mqtt.service

<sudo> journalctl -u zigbee2mqtt.service -f

sudo systemctl enable zigbee2mqtt.service

# InfluxDB / Grafana

## InfluxDB

https://www.influxdata.com/

### Das Datenbankkonzept

### Installation InfluxDB 2.x und erste Schritte

https://helper-scripts.com/scripts?id=InfluxDB

apt install dpkg curl

curl -LO https://download.influxdata.com/influxdb/releases/influxdb2\_2.7.6-1\_amd64.deb

(sudo) dpkg -i influxdb2\_2.7.6-1\_amd64.deb

systemctl status influxdb

systemctl restart influxdb

systemctl start influxdb

systemctl stop influxdb

wget https://download.influxdata.com/influxdb/releases/influxdb2-client-2.7.5-linux-amd64.tar.gz

tar xvzf ./influxdb2-client-2.7.5-linux-amd64.tar.gz

(sudo) cp ./influx /usr/local/bin/Üblicherweise ist jedes influx-Kommando um den InfluxDB-Host, den API-Token und den Organisationsnamen zu ergänzen.

influx bucket list -o myHome -t FqsezC0-xJQuc-4MITmACg518PtK9vIkN9W8ybw2kcAOmg7TRAvSOwwXdNgfARYqLP-ynD76\_pqkumjVzQnrPA==

influx config create --config-name local-config \  
 --host-url http://localhost:8086 \  
 --org myHome \  
 --token FqsezC0-xJQuc-4MITmACg518PtK9vIkN9W8ybw2kcAOmg7TRAvSOwwXdNgfARYqLP-ynD76\_pqkumjVzQnrPA==\  
 --active

influx bucket list

### InfluxDB als Proxmox VE Metrikserver einrichten

### InfluxDB und Home Assistant

influxdb:  
 api\_version: 2  
 ssl: false  
 host: 192.168.178.155  
 port: 8086  
 token: FqsezC0-xJQuc-4MITmACg518PtK9vIkN9W8ybw2kcAOmg7TRAvSOwwXdNgfARYqLP-ynD76\_pqkumjVzQnrPA==  
 organization: myHome  
 bucket: home\_assistant  
 tags:  
 source: HA  
 tags\_attributes:  
 - friendly\_name  
 default\_measurement: units

...

tags\_attributes:  
 - friendly\_name  
 default\_measurement: units  
 exclude:  
 entities:  
 - zone.home  
 domains:  
 - persistent\_notification  
 - person  
 include:  
 domains:  
 - sensor  
 - binary\_sensor  
 - sun  
 entities:  
 - weather.home

influx query '  
from(bucket: "proxmox")  
 |> range(start: -1m)  
 |> filter(fn: (r) => r["\_measurement"] == "system")

|> filter(fn: (r) => r["\_field"] == "used")

|> filter(fn: (r) => r["host"] == "local")

|> filter(fn: (r) => r["object"] == "storages")  
 |> last(column: "\_value")  
 |> keep(columns: ["\_value"])  
'

sensor:  
 - platform: influxdb  
 api\_version: 2  
 ssl: false  
 host: 192.168.178.155  
 port: 8086  
 token: FqsezC0-xJQuc-4MITmACg518PtK9vIkN9W8ybw2kcAOmg7TRAvSOwwXdNgfARYqLP-ynD76\_pqkumjVzQnrPA==  
 organization: myHome  
 bucket: proxmox  
 queries\_flux:  
 - name: "Proxmox used memory"  
 range\_start: "-1m"  
 unit\_of\_measurement: GB  
 value\_template: "{{ (value|float / 1000000000.0)|round(2) }}"  
 bucket: proxmox  
 queries\_flux:  
 - name: "Proxmox used local storage"  
 range\_start: "-1y"  
 unit\_of\_measurement: GB  
 value\_template: "{{ (value|float / 1000000000.0)|round(2) }}"  
 query: >  
 filter(fn: (r) => r["\_measurement"] == "system")  
 |> filter(fn: (r) => r["object"] == "storages")  
 |> filter(fn: (r) => r["\_field"] == "used")  
 |> filter(fn: (r) => r["host"] == "local")  
 |> last(column: "\_value")  
 |> keep(columns: ["\_value"])

## Grafana

<https://grafana.com/>

### Installation und Erstanmeldung

(sudo) apt-get install -y apt-transport-https software-properties-common wget

(sudo) mkdir -p /etc/apt/keyrings/

wget -q -O - https://apt.grafana.com/gpg.key | gpg --dearmor | sudo tee /etc/apt/keyrings/grafana.gpg > /dev/null

echo "deb [signed-by=/etc/apt/keyrings/grafana.gpg] https://apt.grafana.com stable main" | sudo tee -a /etc/apt/sources.list.d/grafana.list

systemctl status grafana-server

systemctl restart grafana-server

systemctl start grafana-server

systemctl stop grafana-server

### Anzeigen von Daten aus der InfluxDB2

from(bucket: "proxmox")  
 |> range(start: v.timeRangeStart, stop: v.timeRangeStop)  
 |> filter(fn: (r) => r["\_measurement"] == "system")  
 |> filter(fn: (r) => r["\_field"] == "used")  
 |> filter(fn: (r) => r["host"] == "local")  
 |> filter(fn: (r) => r["object"] == "storages")  
 |> aggregateWindow(every: v.windowPeriod, fn: mean, createEmpty: false)  
 |> yield(name: "mean")

### Grafana in Home Assistant

[auth.anonymous]

# enable anonymous access

enabled = true

# set to true if you want to allow browsers to render Grafana in a <frame>, <iframe>, <embed> or <object>. default is f>

allow\_embedding = true

grafana-cli plugins install grafana-image-renderer

type: iframe

url: >-  
 http://192.168.178.155:3000/d-solo/edo9fdjmarn5sd/proxmox-system?orgId=1&refresh=10s&from=now-5m&to=now&panelId=1&fullscreen&kiosk=1

aspect\_ratio: 60%

# Das Homelab im Netz

## Das Domain Name System

<https://www.ssllabs.com/ssltest>

## Der externe Zugriff auf das lokale Netz

https://github.com/traefik/whoami

docker run -d \  
 -p 9080:80 \  
 --name whoami \  
 --restart=always \  
 traefik/whoami

## Die eigene Domain

## Das Dynamic Domain Name System

curl -B4 checkip.dyndns.com 2>&1 | grep -o '[0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}\.[0-9]\{1,3\}'

<https://www.duckdns.org/>

services:

duckdns:

image: lscr.io/linuxserver/duckdns:latest

container\_name: duckdns

network\_mode: host #optional

environment:

- PUID=1000 #optional

- PGID=1000 #optional

- TZ=Etc/UTC #optional

- SUBDOMAINS=subdomain1,subdomain2

- TOKEN=token

- UPDATE\_IP=ipv4 #optional

- LOG\_FILE=false #optional

volumes:

- /path/to/duckdns/config:/config #optional

restart: unless-stopped

https://hub.docker.com/r/linuxserver/duckdns.

## Der VPN-Tunnel

https://ngrok.com/

docker run --net=host -it \  
 --name ngrok \  
 -e NGROK\_AUTHTOKEN=<Ihr\_Ath\_Token> \  
 ngrok/ngrok:latest \  
 http 9080

mkdir /opt/ngrok

cd /opt/ngrok

nano docker-compose.yml

version: "3"

services:

ngrok:

image: ngrok/ngrok:latest

restart: unless-stopped

command:

- "start"

- "--all"

- "--config"

- "/etc/ngrok.yml"

volumes:

- /opt/ngrok/ngrok.yml:/etc/ngrok.yml

ports:

- 4040:4040

nano ngrok.yml

# Version of the ngrok Agent Configuration file. Required.

version: 3

# Agent Configuration

agent:

authtoken: <Ihr\_Auth\_Token>

# Endpoints Configuration

endpoints:

- name: example1

url: https://example1.ngrok-free.app

# Der eigene DNS-Server

<https://technitium.com/dns/>

## Technitium installieren

https://github.com/TechnitiumSoftware/DnsServer

apt-get update

apt-get upgrade

apt-get install curl

curl -sSL https://download.technitium.com/dns/install.sh | sudo bash

systemctl status dns

<http://technitium-dns:5380/>

## Mit Technitium-DNS arbeiten

# Proxies und selbst gehosteter Passwortmanager

## NGINX / NGINX Proxy Manager

<https://nginx.org/>

<https://nginxproxymanager.com/>

<https://nginx.org/en/linux_packages.html>

<https://tteck.github.io/Proxmox/>)

systemctl --type=service –state=runningstatus  
systemctl status pveproxy.service

https://pve.proxmox.com/wiki/Web\_Interface\_Via\_Nginx\_Proxy

mkdir -p /opt/npm

mkdir -p /opt/letsencrypt (optional)

mkdir -p /opt/mysql

cd /opt/npm

nano docker-compose.yml

..

version: "3"

services:

app:

image: 'jc21/nginx-proxy-manager:latest'

container\_name: 'npm-app'

restart: always

ports:

# Public HTTP Port:

- '80:80'

# Public HTTPS Port:

- '443:443'

# Admin Web Port:

- '81:81'

environment:

# These are the settings to access your db

DB\_MYSQL\_HOST: "db"

DB\_MYSQL\_PORT: 3306

DB\_MYSQL\_USER: "npm"

DB\_MYSQL\_PASSWORD: "npm"

DB\_MYSQL\_NAME: "npm"

# If you would rather use Sqlite uncomment this

# and remove all DB\_MYSQL\_\* lines above

# DB\_SQLITE\_FILE: "/data/database.sqlite"

# Uncomment this if IPv6 is not enabled on host

DISABLE\_IPV6: 'true'

volumes:

- /opt/npm:/data

- /opt/letsencrypt:/etc/letsencrypt

depends\_on:

- db

db:

image: 'jc21/mariadb-aria:latest'

container\_name: 'npm-db'

restart: always

environment:

MYSQL\_ROOT\_PASSWORD: 'npm'

MYSQL\_DATABASE: 'npm'

MYSQL\_USER: 'npm'

MYSQL\_PASSWORD: 'npm'

volumes:

- /opt/mysql:/var/lib/mysql

docker-compose up -d

docker compose pull

docker compose up -d

## Caddy

<https://caddyserver.com/docs/>

### Installation von Caddy

apt install -y debian-keyring debian-archive-keyring apt-transport-https curl

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-stable-archive-keyring.gpg

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/stable/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-stable.list

apt update

apt install caddy

systemctl status caddy

systemctl restart caddy

caddy run

caddy stop

caddy reload

curl localhost:2019/config/

ls -alh /etc/caddy/Caddyfile

nano /etc/caddy/Caddyfile

:2015 {

respond "Hello, world!"

}

nano /usr/share/caddy/caddy.html

<!doctype html>

<html>

<head>

<meta charset="utf-8">

<title>Hello, Caddy!</title>

</head>

<body>

<h1>Hello, Caddy!</h1>

<p> Hello Current year is: {{now | date "2006"}</p>

<p>We have just configured our Caddy web server on Ubuntu Server!</p>

</body>

</html>

:80 {

# Pfad zum Root-Verzeicnis für die Seiten

root \* /usr/share/caddy

# Enable Templates

templates

# Enable Sstatic File Server.

file\_server browse

}

caddy fmt /etc/caddy/Caddyfile --overwrite

:2080 {

reverse\_proxy 192.168.178.154:9080

}

https://github.com/caddyserver/xcaddy

https://go.dev/doc/install

wget https://go.dev/dl/go1.23.2.linux-amd64.tar.gz

rm -rf /usr/local/go && tar -C /usr/local -xzf go1.23.2.linux-amd64.tar.gz

ls -alh /usr/local

nano /etc/profile

export PATH=$PATH:/usr/local/go/bin

source /etc/profile

echo $PATH

go version

nano helloworld.go

package main

// Import OS and fmt packages

import (

"fmt"

"os"

)

// Let us start

func main() {

fmt.Println("Hello, world!") // simple text on screen

fmt.Println(os.Getenv("USER"), ", Let's be friends!") // Read Linux $USER environment variable

}

go run helloworld.go

<https://github.com/caddyserver/xcaddy> u

apt install -y debian-keyring debian-archive-keyring apt-transport-https curl

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/xcaddy/gpg.key' | sudo gpg --dearmor -o /usr/share/keyrings/caddy-xcaddy-archive-keyring.gpg

curl -1sLf 'https://dl.cloudsmith.io/public/caddy/xcaddy/debian.deb.txt' | sudo tee /etc/apt/sources.list.d/caddy-xcaddy.list

apt update

apt install xcaddy -y

https://caddyserver.com/docs/modules/

xcaddy build --with github.com/caddy-dns/duckdns

././caddy version z

systemctl stop caddy

cp ./caddy /usr/bin/

nano /etc/caddy/Caddyfile

# General Options

{

# debug

email <Ihre\_E-Mail\_Adresse>

acme\_dns duckdns <Ihr\_Auth\_Token>

}

iiotnerds.duckdns.org {

log {

output file /var/log/caddy/access.log

}

# tls {#

# dns duckdns {

# api\_token <Ihr\_Auth\_Token>

# }

# }

reverse\_proxy 192.168.178.154:9080

ls -als $HOME/.local/share/caddy

cat /var/log/caddy/access.log

cat /dev/null > /var/log/caddy/access.log

caddy run -w -c /etc/caddy/Caddyfile

## Der Passwortmanager Vaultwarden

<https://github.com/dani-garcia/vaultwarden>

### Installation

mkdir -p /opt/vaultwarden

docker run -d \  
 -e ADMIN\_TOKEN=raspberry \  
 -p 3080:80 \  
 -v /opt/vaultwarden:/data \  
 --name vaultwarden \  
 --restart=always \  
 vaultwarden/server:latest

docker stop vaultwarden

docker rm vaultwarden

### Mit Vaultwarden arbeiten

# Medienmanagement im Homelab

## Der Zugriff auf externe (Medien-)Daten

<https://www.samba.org/samba/>

lxc-ls -f

cat /var/lib/lxc/500/config

apt install cifs-utils -y

mkdir /mnt/samba

nano /etc/fstab

//192.168.178.105/SambaPublic /mnt/samba cifs uid=1000,gid=1000,rw,user,username=pi,password=raspberry,domain=workgroup 0 0

systemctl daemon-reload

mount -a

ls -l /mnt/samba

[https://pve.proxmox.com/wiki/Unprivileged\_LXC\_containers#Using\_local\_directory\_bind\_mount\_points](https://pve.proxmox.com/wiki/Unprivileged_LXC_containers" \l "Using_local_directory_bind_mount_points))

apt install cifs-utils -y

groupadd -g 10000 lxc\_shares

cat /etc/group

usermod -aG lxc\_shares root

mkdir -p /mnt/samba

sudo nano /etc/fstab

//192.168.178.105/SambaPublic /mnt/samba cifs   
\_netdev,x-systemd.automount,noatime,uid=100000,gid=110000,dir\_mode=0770,file\_mode=0770,user=pi,pass=raspberry 0 0

systemctl daemon-reload

mount /mnt/samba

mount -a

nano /etc/pve/lxc/501.conf

mp0: /mnt/samba/,mp=/mnt/lxc\_mnt\_samba

ls -l /mnt/lxc\_mnt\_samba

## Ein NAS mit OpenMediaVault

[https://optimizeddocs.com/blogs/backups/backup%20hardware/truenas%20vs%20openmediavault](https://optimizeddocs.com/blogs/backups/backup hardware/truenas vs openmediavault)

<https://flexense.com/truenas_vs_openmediavault_nas_performance_review.html>)

### OpenMediaVault

### Installation und erster Start

https://docs.openmediavault.org/en/stable/installation/index.html

<https://www.openmediavault.org/download.html>

### Das OpenMediaVault Web-Interface

http://openmediavault

### Die Erstkonfiguration

### Weitere Einstellungen

## Der Media-Server Jellyfin

### Jellyfin

### Installation und erster Start

<https://community-scripts.github.io/ProxmoxVE/scripts?id=jellyfin>

apt install curl -y

curl https://repo.jellyfin.org/install-debuntu.sh | sudo bash

systemctl status jellyfin.service

### Das Jellyfin Web-Interface

mkdir -p /data/jellyfin/pictures

scp -r C:\Users\udobr.LAPTOP-HP\Pictures\IoT-Grizzly.png root@192.168.178.158:/data/jellyfin/pictures