Raspberry pi mit Raspbian als SSH/SFTP-Server

Nach der Installation von Open SSH mit "apt-get install ssh" vom Terminal aus ist sinnvollerweise eine **Konfiguration des LAN** auf eine statische IP-Adresse vorzunehmen. Dazu wird in "/etc/network/interfaces" editiert:

nach: iface lo inet loopback
Zeile: iface etho inet dhcp ändern in: iface etho inet static
darunter zusätzlich z.B.: address 172.16.1.2
netmask 255.255.255.240
gateway 172.16.1.1
dns-nameserver 172.16.1.1

Ab Raspbian Strech besser "/etc/network/interfaces" unverändert lassen und die statische IP-Adresse in der „/etc/ dhcpcd.conf festlegen (nicht in beiden!). Dazu am Ende einfügen und „sudo service dhcpcd restart“:

```
interface eth0
static ip_address=172.16.1.2/24
static routers= 172.16.1.1
static domain_name_servers= 172.16.1.1
```

Kontrolle im Terminal mit: "ifconfig -a".

Um Ressourcen zu sparen: Im Menü „Preferences > Raspberry Pi Configuration > System > Boot > To CLI auswählen“ (oder auch mit „sudo raspi-config“), um einen Start von X-Window abzuschalten.

Vorbereitung von Gruppen und Nutzern vom Terminal aus**:**

```
addgroup dat
addgroup gastdat
adduser --ingroup dat user01
adduser --ingroup dat user02
adduser --ingroup dat user03
adduser --ingroup gastdat gast01
```

(Kontrolle mit Editor in "/etc/group" bzw. "/etc/passwd" möglich)
Falls nicht bereits geschehen, Rechte auf: root root 755 für „/home“
zusätzlich: root root 755 für „/home/<username>“
Verzeichnisse erstellen: /home/<username>/Datei
je mit den Rechten root root 777 für „/home/<username>/Datei“

Das geht sehr bequem im Filemanager (diesen als Root neu starten); ansonsten im Terminal mit den Befehlen chown, chmod sowie mkdir).
Kontrolle mit „ls -lasi /home“ usw. (andere Rechte liefern nicht wirklich mehr, siehe auch unten USB-Festplatte).

Änderungen in "/etc/ssh/sshd_config" mit Editor (es sind nur die hier notwendigen Zeilen angegeben):

```
AllowUsers pi user01 user02 user03 gast01
PermitRootLogin no
ChallengeResponseAuthentication no
PasswordAuthentication yes              #(weil PAM nicht mit Match zusammenarbeitet)

Subsystem sftp /usr/lib/openssh/sftp-server
# Subsystem     sftp    internal-sftp
```

```
Match group dat
    ChrootDirectory /home
    ForceCommand internal-sftp
    AllowTCPForwarding no
    X11Forwarding no
Match user gast01
    ChrootDirectory /home/gast01
    ForceCommand internal-sftp
    AllowTCPForwarding no
    X11Forwarding no

# UsePAM yes                    (weil es mit Match nicht zusammenarbeitet)
```

Danach Restart des SSH-Servers im Terminal:
/etc/init.d/ssh restart (Ein Log befindet sich in "/var/log/auth.log".)

Ergebnis:

Nutzer pi darf alles sehen und sich über SSH-Clients (auch mit einem Terminalclient) anmelden. So kann darüber der Raspberry pi bedient werden (z.B. mit PuTTy und VcXsrv; In PuTTy kann „X11 forwarding“ aktiviert und VcXsrv mit seinem Pfad eingetragen werden, dann in PuTTy „ nach Anmeldung „lxpanel“ starten und nach Rechtsklick auf dieses Fenster über „Panel Settings“ so konfigurieren, dass alles bequem gestartet werden kann; jedes Tool/Programm wird in einem eigenen Fenster gestartet; der Prozess „lxpanel“ ist in PuTTy zum Schluss mit „Str C“ zu beenden).

Für die Nutzer user01 user02 user03 gibt es ausschließlich SFTP über SSH und sie sehen nichts oberhalb "/home". Sie können sich gegenseitig sehen und lesen (auch in gast01) aber nur in allen ihren Verzeichnissen „Datei“ schreiben.
Für Nutzer gast01 gibt es ausschließlich SFTP über SSH und er sieht nichts oberhalb "/home/gast01", er kann nur in seinem "/home/gast01/Datei" schreiben.
Mit ChrootDirectory "/home/%u" würde sich niemand gegenseitig sehen, alle könnten nur in ihrem Verzeichnis "<username>/Datei" schreiben. (<username> je durch konkreten Usernamen ersetzen.)

Dieser SSH-Server nutzt die bekannten Sicherheiten (SSH-Verschlüsselung, nur autorisierte Nutzer, kein root-Login, chroot-Umgebung), lässt aber die Bedienung für Nutzer pi über SSH zu.

Verlegen der Verzeichnisse „/home /<username>/Dateien“ auf eine USB – Festplatte

Diese ist mit NTFS formatiert, um sie auch mit Windows nutzen zu können. Dadurch haben alle Verzeichnisse und Dateien das Recht „root root 777“, weshalb ein Verlegen der Verzeichnisse „<username>“ selbst nicht sinnvoll ist. Als Lösung auf der Festplatte Verzeichnisse „Datei01“ bis „Datei03“ und „DateiG1“ anlegen und jeweils als „/home/<username>/Datei“ mounten. Unter „/“ kann z.B. das Verzeichnis „/Dokumente“ als Mount Point für die Festplatte angelegt werden.
Das erfolgt am besten in
„/etc/fstab“ mit einem Editor und wird dann bei jedem Booten ausgeführt.

```
/dev/sda1              /Dokumente              ntfs-3g  defaults,noatime  0  1
/Dokumente/Datei01     /home/ user01/Datei     none     bind              0  0
/Dokumente/Datei02     /home/ user02/Datei     none     bind              0  0
/Dokumente/Datei03     /home/ user03/Datei     none     bind              0  0
/Dokumente/DateiG1     /home/gast01/Datei      none     bind              0  0
```

Der „ntfs-3g“ Treiber (für Lese- und Schreibzugriff notwendig) kann mit „apt-get install ntfs-3g“ installiert werden.
Die Zugriffsmöglichkeiten der User bleiben mit dieser Lösung wie oben beschrieben.

Als Virenscanner wurde ClamAV installiert (im repository verfügbar im Raspberry Pi aber leider ohne Hintergrundscanner – sicher wegen zu geringer Ressourcen). Mit "apt-get install clamav " installiert, können z.B. jede Nacht einmal alle Dateien in „/home“ gescannt werden. Dazu wird in die „/etc/crontab mit einem Editor z.B. hinzugefügt:
47 0 * * * root /usr/bin/clamscan /home –r #um 0:47 h Scannen (hier hohe CPU-Auslastung)
Die Virenupdatefunktion „freshclam „ wird beim Booten gestartet. In der „/etc/clamav/freshclam.conf“ kann aber von 24-mal am Tag mit einem Editor z.B. auf 1 bis 2-mal reduziert werden:
Check for new database 24 times a day
Checks 24
auf 2 mal geaendert
Checks 2
In Raspberry Pi OS Buster kann clamd in /var/run (ein Link zu /run) kein Verzeichnis clamav anlegen (permission denied) als „workaround“ in /etc/rc.local vor exit 0 einfügen:
#Ausführung als „root“
export USER='root'
export LOGNAME='root'
erzeugt /run/clamav mit clamav:clamav 774
mkdir /run/clamav && chown clamav:clamav -R /run/clamav && chmod 774 /run/clamav
Nur das Erstellen dieses Verzeichnisses reicht nicht, weil nach jedem Booten /run und so auch /var/run neu erstellt wird.

Soll auf den Server von außerhalb zugegriffen werden (nachdem er in unserem internen Netzwerk bereits läuft und leider nur eine dynamische IP vom Provider zugewiesen wurde), muss ein Dienst für dynamisches DNS genutzt werden, um eine feste URL zu erhalten.

Der Router ist dazu so zu konfigurieren, dass Anfragen von außen an den richtigen Host weitergeleitet werden und seine Firewall den entsprechenden Port (z.B. Standard Port 22; besser einen unbenutzten hohen Port, da sonst unzählige Angriffe) freigibt.

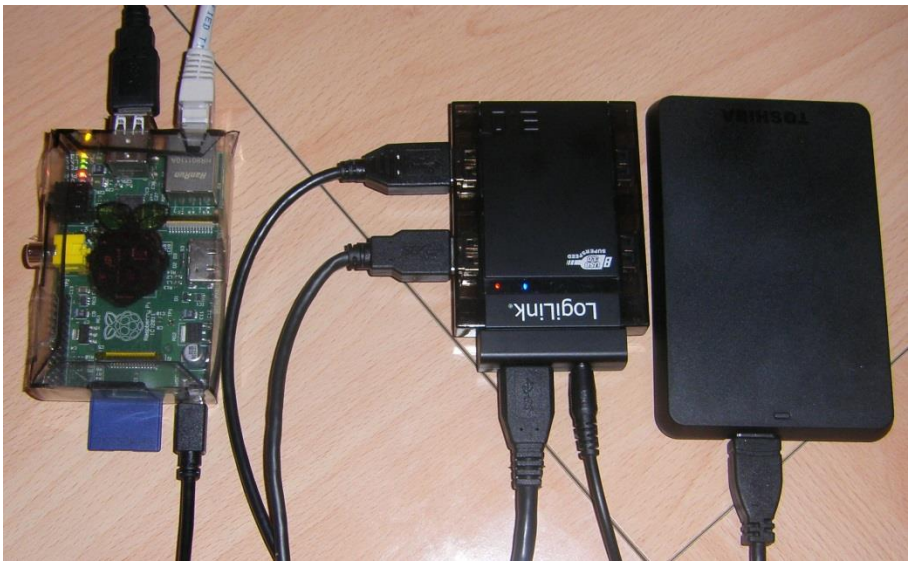
Eine Nutzung ist z.B. mit „Filezilla“ oder WinSCP als SFTP-Client zum Dateitransfer für Windows möglich.

Der Hardwareaufbau besteht aus einem Raspberry Pi, einer USB 3.0 Laptopfestplatte und einem USB 3.0 HUB (1.1 und 2.0 kompatibel; Ports bis 1 A belastbar) sowie dem Netzteil des HUB (5 V, 2 A). Der Raspberry Pi wird vom HUB über ein USB 2.0 Kabel (Stecker A, micro Stecker B) versorgt. Die Festplatte wird nur durch den HUB versorgt. Am Raspberry Pi sind der HUB und Ethernet angeschlossen.

Bei mir ist offensichtlich der DSL-Zugang der Flaschenhals für die Übertragungsgeschwindigkeit. Der gesamte Leistungsbedarf (gemessen auf der 230 V Seite des Netzteils) liegt bei Inaktivität bei 3,9 W und während der Datenübertragung steigt er teilweise bis auf 5,3 W (bei heruntergefahrenem Pi werden noch 2 W gezogen). Der fertige Aufbau (Netzteil außerhalb des Bildes):

Mit Raspberry Pi 4b kann die USB-Festplatte direkt am Raspberry Pi 4b über USB 3 angeschlossen werden und dann das dafür vorgesehene Netzteil genutzt werden. Der gesamte Leistungsbedarf (gemessen auf der 230 V Seite des Netzteils) liegt bei Inaktivität auch hier bei 3,9 W und während der Datenübertragung steigt er teilweise bis auf 6,1 W.

Der Raspberry Pi 4b hat einen Kühlkörper „Joy-IT Armor Gehäuse Block für Raspberry Pi 4“ bekommen, so liegt die CPU Temperatur auch bei Dauerbetrieb um 40°C.



Raspberry Pi 1b

Raspberry Pi 4b mit Joy-IT Armor Gehäuse Block