Allgemeines

- Alle Cronjobs von allen Usern auflisten
- IPTables Basics
- Nützliche Bash-Einzeiler
- Prozess unter einem anderen Benutzer starten
- Random Strings generieren
- Erstellen von SSH Keys
- Umgang mit Dateien
- Wiederherstellung des root Passworts
- SA-MP
    - SA-MP Server installieren
    - Gamemodes auf einem Linux System kompilieren
- Wireguard Server installieren & Clients einrichten
- cloud-init Deaktivieren
- Fehlermeldung beim Updaten der Paketlisten "... öffentlicher Schlüssel nicht verfügbar ..." beheben
- Arbeiten mit systemd
    - Dienste verwalten
    - Bearbeiten von Unit-Dateien

Alle Cronjobs von allen Usern auflisten

```
#!/bin/bash

# System-wide crontab file and cron job directory. Change these for your system.
CRONTAB='/etc/crontab'
CRONDIR='/etc/cron.d'

# Single tab character. Annoyingly necessary.
tab=$(echo -en "\t")

# Given a stream of crontab lines, exclude non-cron job lines, replace
# whitespace characters with a single space, and remove any spaces from the
# beginning of each line.
function clean_cron_lines() {
    while read line ; do
        echo "${line}" |
            egrep --invert-match '^($|\s*#|\s*[[:alnum:]_]+=)' |
            sed --regexp-extended "s/\s+/ /g" |
            sed --regexp-extended "s/^ //"
    done;
}

# Given a stream of cleaned crontab lines, echo any that don't include the
# run-parts command, and for those that do, show each job file in the run-parts
# directory as if it were scheduled explicitly.
function lookup_run_parts() {
    while read line ; do
        match=$(echo "${line}" | egrep -o 'run-parts (-{1,2}\S+ )*\S+')

        if [[ -z "${match}" ]] ; then
            echo "${line}"
        else
            cron_fields=$(echo "${line}" | cut -f1-6 -d' ')
```

```
            cron_job_dir=$(echo  "${match}" | awk '{print $NF}')

            if [[ -d "${cron_job_dir}" ]] ; then
                for cron_job_file in "${cron_job_dir}"/* ; do  # */ <not a comment>
                    [[ -f "${cron_job_file}" ]] && echo "${cron_fields} ${cron_job_file}"
                done
            fi
        fi
    done;
}

# Temporary file for crontab lines.
temp=$(mktemp) || exit 1

# Add all of the jobs from the system-wide crontab file.
cat "${CRONTAB}" | clean_cron_lines | lookup_run_parts >"${temp}"

# Add all of the jobs from the system-wide cron directory.
cat "${CRONDIR}"/* | clean_cron_lines >>"${temp}"  # */ <not a comment>

# Add each user's crontab (if it exists). Insert the user's name between the
# five time fields and the command.
while read user ; do
    crontab -l -u "${user}" 2>/dev/null |
        clean_cron_lines |
        sed --regexp-extended "s/^((\S+ +){5})(.+)$/\1${user} \3/" >>"${temp}"
done < <(cut --fields=1 --delimiter=: /etc/passwd)

# Output the collected crontab lines. Replace the single spaces between the
# fields with tab characters, sort the lines by hour and minute, insert the
# header line, and format the results as a table.
cat "${temp}" |
    sed --regexp-extended "s/^(\S+) +(\S+) +(\S+) +(\S+) +(\S+) +(\S+) +(.*)$/\1\t\2\t\3\t\4\t\5\t\6\t\7/" |
    sort --numeric-sort --field-separator="${tab}" --key=2,1 |
    sed "1i\mi\th\td\tm\tw\tuser\tcommand" |
    column -s"${tab}" -t

rm --force "${temp}"
```

IPTables Basics

Eine IP sperren

```
iptables -A INPUT -s xxx.xxx.xxx.xxx -j DROP
```

Beispiel:

```
iptables -A INPUT -s 65.55.44.100 -j DROP
```

Einzelnen Port für eine IP sperren

```
iptables -A INPUT -s 65.55.44.100 -p tcp --destination-port 25 -j DROP
```

IP wieder freigeben (Regel löschen)

```
iptables -D INPUT -s xxx.xxx.xxx.xxx -j DROP
```

Alternativ kann man auch so vorgehen:

1. Alle Regeln / Einträge auflisten lassen mit `iptables -L --line-numbers`
2. Die gewünschte IP aus der Liste raussuchen und die Zeilennummer und den Regelnamen merken (Bspw. f2b-sshd)
3. `iptables -D f2b-sshd 1` um den ersten Eintrag aus der Regel f2b-sshd zu löschen

Nützliche Bash-Einzeiler

Logausgabe verfolgen und gleichzeitig nach einem bestimmten String suchen

```
tail -f file | grep --line-buffered my_pattern
```

SSH Keys von Github importieren

```
ssh-import-id-gh <username>


ODER falls auf dem System dieses Tool nicht vorinstalliert ist


mkdir -m 700 ~/.ssh; curl https://github.com/<username>.keys >> ~/.ssh/authorized_keys
```

Inhalte in Dateien ersetzen

```
sed -i 's/old-text/new-text/g' input.txt
```

Passwort generieren

```
date +%s | sha256sum | base64 | head -c 32 ; echo
```

String hashen

```
echo -n "yourstring" | shasum -a 512
```

Timestamp zu Datum konvertieren

```
date -d @<TIMESTAMP>


ex.
date -d @1616934873
```

```
Sun 28 Mar 2021 02:34:33 PM CEST
```

Datum zu Timestamp konvertieren

Linux

```
date -d '06/12/2012 07:21:22' +"%s"
```

Beachte: Amerikanisches Datumsformat, sprich *MM/DD/YYYY*

Mac OS

```
date -j -u -f "%a %b %d %T %Z %Y" "Tue Sep 28 19:35:15 EDT 2010" "+%s"
```

Dateien von Gerät A zu Gerät C über Gerät B kopieren (Jumphost-like)

```
scp -oProxyCommand="ssh -W %h:%p user@host_jumphost" quelldatei user@host_ziel:zielverzeichnis
```

Backup des gesamten Systems anfertigen

```
rsync -aAXHv --exclude={"/dev/*","/proc/*","/sys/*","/tmp/*","/run/*","/mnt/*","/media/*","/lost+found"} /
/path/to/backup
```

Alle Dateien aus Unterordnern in das aktuelle Verzeichnis verschieben

```
find . -mindepth 2 -type f -print -exec mv {} . \;
```

Alle selber erstellten User auflisten

```
awk -F'[/:]' '{if ($3 >= 1000 && $3 != 65534) print $1}' /etc/passwd
```

> Es werden keine Systemuser aufgeführt (welche i.d.R. eine UserID < 1000 haben)

IP Adressen in Webserver-Zugriffslogs auszählen

```
awk '{print $1}' /var/log/nginx/access.log | sort | uniq -c | sort -nr
```

Prozess unter einem anderen Benutzer starten

```
sudo -H -u otheruser bash -c 'echo "I am $USER, with uid $UID"'
```

Wobei mit `-c` erst der eigentlich auszuführende Befehl festgelegt wird

Auszug aus den sudo-Manpages:

> -H The -H (HOME) option requests that the security policy set the HOME environment variable to the home directory of the target user (root by default) as specified by the password database. Depending on the policy, this may be the default behavior.

> -u user The -u (user) option causes sudo to run the specified command as a user other than root. To specify a uid instead of a user name, use #uid. When running commands as a uid, many shells require that the '#' be escaped with a backslash ('\'). Security policies may restrict uids to those listed in the password database. The sudoers policy allows uids that are not in the password database as long as the targetpw option is not set. Other security policies may not support this.

Random Strings generieren

```
#!/bin/bash
# bash generate random alphanumeric string
#

# bash generate random 32 character alphanumeric string (upper and lowercase) and
NEW_UUID=$(cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 32 | head -n 1)

# bash generate random 32 character alphanumeric string (lowercase only)
cat /dev/urandom | tr -dc 'a-zA-Z0-9' | fold -w 32 | head -n 1

# Random numbers in a range, more randomly distributed than $RANDOM which is not
# very random in terms of distribution of numbers.

# bash generate random number between 0 and 9
cat /dev/urandom | tr -dc '0-9' | fold -w 256 | head -n 1 | head --bytes 1

# bash generate random number between 0 and 99
NUMBER=$(cat /dev/urandom | tr -dc '0-9' | fold -w 256 | head -n 1 | sed -e 's/^0*//' | head --bytes 2)
if [ "$NUMBER" == "" ]; then
  NUMBER=0
fi

# bash generate random number between 0 and 999
NUMBER=$(cat /dev/urandom | tr -dc '0-9' | fold -w 256 | head -n 1 | sed -e 's/^0*//' | head --bytes 3)
if [ "$NUMBER" == "" ]; then
  NUMBER=0
fi
```

Erstellen von SSH Keys

Key generieren (die einfachste Variante)

```
ssh-keygen -C
```

Key generieren im Ed25519 Format (das aktuellste & zZt. sicherste Format)

```
ssh-keygen -o -a 100 -t ed25519 -f ~/.ssh/id_ed25519 -C "name@email.de"
```

Was bedeuten die eben genutzten Parameter?

`-o` : Speichert den Key im neuen OpenSSH Format anstelle des alten PEM Formats. Muss angegeben werden wenn Ed25519 Keys generiert werden sollen.

`-a`: Bestimmt die Anzahl der Schlüsselableitungen (engl. KDF - *Key Derivation Function*). Eine höhere Zahl hier sorgt für eine langsamere Überprüfung der Key-Passphrase (Passwort mit dem der Schlüssel zusätzlich geschützt ist) um Brute-Force Angriffe zu erschweren sollte der private Key kompromittiert worden sein.

`-t`: Legt den Typ des Keys fest der generiert werden soll, in diesem Fall Ed25519. Mögliche Werte hier wären DSA, RSD, ECDSA oder Ed25519.

`-f`: Legt den Dateinamen des erstellten Keys fest. Soll der Key automatisch vom verwendeten SSH Agent erkannt werden sollte der Default-Pfad ~/.ssh/ beibehalten werden.

`-C`: Hier kann ein Kommentar angegeben werden. Es gibt keine Vorgabe was hier einzutragen ist, üblicherweise wird der Besitzer des Schlüssels im Format <username>@<hostname> hier eingetragen (Bspw. *nielsperetzke@breadfish.de-prod*).

Umgang mit Dateien

Dateien vor Veränderung/Löschen schützen

Aktuellen Status von Dateien anzeigen

```
lsattr
```

Beispiel:

```
root@server:~/.ssh# lsattr
----------e--- ./authorized_keys
```

Mögliche Attribute

Attribut	Bedeutung
A	Bei Dateien mit diesem Attribut wird das Datum des letzten Zugriffes nicht gespeichert.
a	Dateien mit diesem Attribut können nur im append-Modus zum Schreiben geöffnet werden. Es kann also nur Inhalt an die Datei dran gehangen werden aber nicht gelöscht oder überschrieben werden. Dieses Attribut kann nur mit Root-Rechten gesetzt und entfernt werden.
c	Dateien mit diesem Attribut werden automatisch vom Kernel gepackt auf der Platte gespeichert. Wird sie ausgelesen, wird sie automatisch wieder entpackt. Dieses Attribut hat momentan noch keine Auswirkungen auf ext2- und ext3-Dateisystemen.
D	Wenn ein Ordner dieses Attribut besitzt und verändert wird, werden diese Veränderungen synchron auf die Festplatte geschrieben.
d	Dateien mit diesem Attribut werden von dem Programm „dump“ ignoriert.

Attribut	Bedeutung
E	Dieses Attribut kann nicht gesetzt werden. Es gibt an, ob eine Datei, die vom Kernel gepackt wurde, einen Kompressions-Fehler besitzt.
I	Dieses Attribut kann nicht gesetzt werden. Es gibt an, ob ein Ordner über gehashte trees (Bäume) indexiert wird.
i	Dateien mit diesem Attribut können nicht verändert werden. Sie können nicht gelöscht oder modifiziert werden und man kann keinen harten Link (Hardlink) auf die Datei erstellen. Symbolische Links (Softlinks) sind weiterhin möglich. Dieses Attribut kann nur mit Root-Rechten gesetzt und entfernt werden.
j	Besitzt eine Datei dieses Attribut, wird ihr ganzer Inhalt erst in das Journal geschrieben bevor es auf die Festplatte geschrieben wird. Es hat nur Auswirkungen auf ext3-Dateisystemen und nur, wenn es im ordered- oder writeback-Modus läuft. Dieses Attribut kann nur mit Root-Rechten gesetzt und entfernt werden.
s	Wenn eine Datei mit diesem Attribut gelöscht wird, werden seine Daten auf der Festplatte mit Nullen überschrieben. Dieses Attribut hat momentan noch keine Auswirkungen auf ext2- und ext3-Dateisystemen.
S	Wenn eine Datei dieses Attribut besitzt und verändert wird, werden diese Veränderungen synchron auf die Festplatte geschrieben.
T	Ein Ordner mit diesem Attribut wird vom Orlov block allocator □□□□ behandelt, als wäre es der erste Ordner in der Hierarchie. Zugriffe auf diesen Ordner werden dadurch beschleunigt.
t	Eine Datei auf der Festplatte wird sich den letzten Block, auf dem sie liegt, nicht mit einer anderen Datei teilen (tail-merging). Dieses Attribut hat momentan noch keine Auswirkungen auf ext2- und ext3-Dateisystemen, da diese generell kein tail-merging unterstützen außer in experimentellen Patches.
u	Wenn eine Datei mit diesem Attribut gelöscht wird, wird ihr Inhalt gespeichert, so dass ein User sie später wieder herstellen kann. Dieses Attribut hat momentan noch keine Auswirkungen auf ext2- und ext3-Dateisystemen.
X	Dieses Attribut kann nicht gesetzt werden. Es gibt an, ob eine vom Kernel gepackte Datei auch unentpackt gelesen werden kann.
Z	Dieses Attribut kann nicht gesetzt werden. Es gibt an, ob eine Datei, die vom Kernel gepackt, wurde einen Fehler besitzt.

Datei vor allen Veränderungen schützen

```
chattr +i [filename]
```

Datei nur für Erweiterungen freigeben, Löschen der Datei und seiner Inhalte verbieten

```
chattr +a [filename]
```

Attribute wieder entfernen

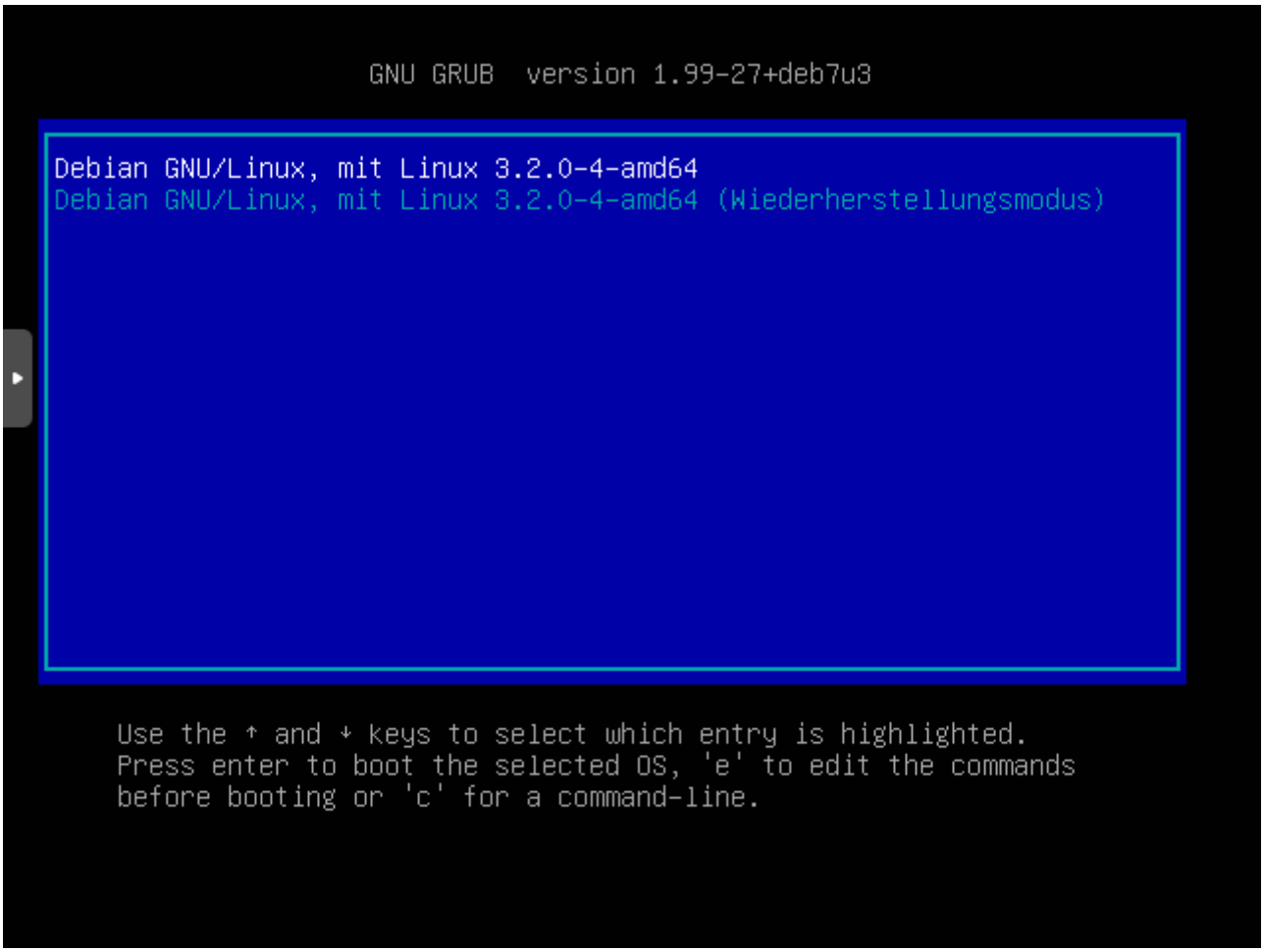
```
chattr -[Attribut] [filename]
```

Wiederherstellung des root Passworts

alte Methode (ältere Installationen von Debian und Ubuntu)

1. Server neustarten (Strg-Alt-Entf löst einen Reboot aus auch ohne dass man eingeloggt ist)

2. Wenn der folgende Bildschirm erscheint, Taste **e** drücken (GRUB Bootloader)



3. Im folgenden Bildschirm muss die mit

```
linux /boot/vmlinuz-...
```

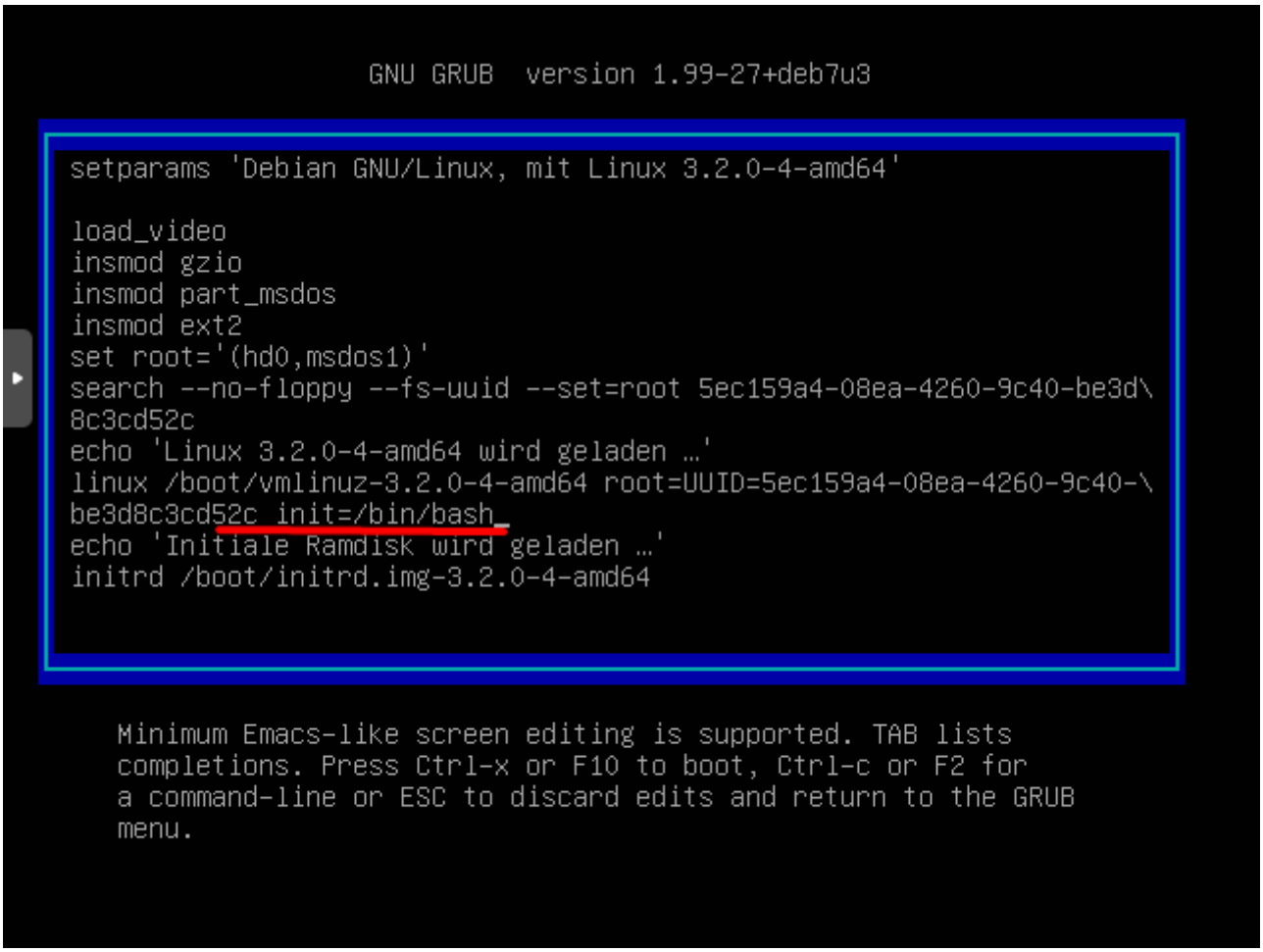
beginnende Zeile ausgewählt und an dessen Ende gesprungen werden. Dort werden die letzten beiden Teile

```
ro quiet
```

gelöscht und durch

```
init=/bin/bash
```

ersetzt.

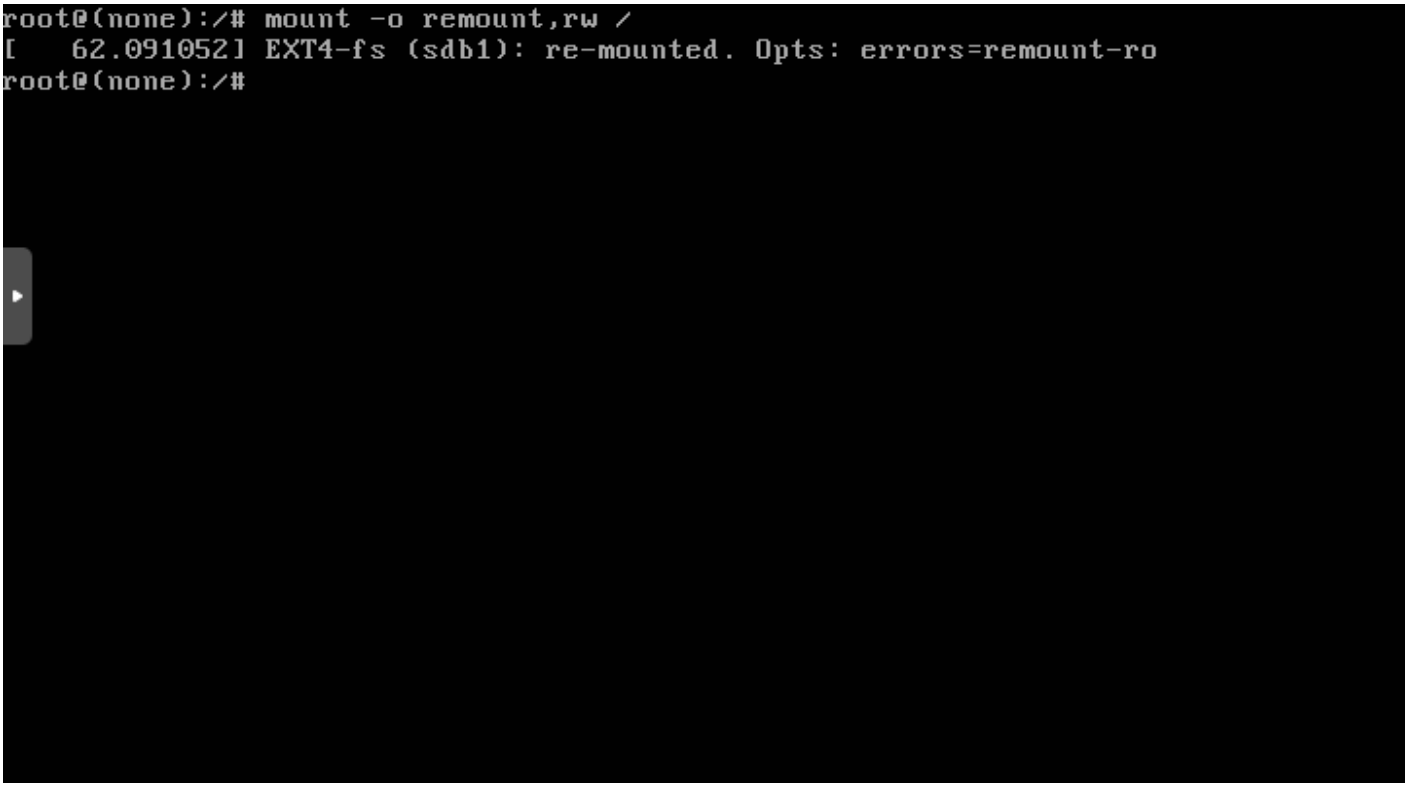


> **ACHTUNG:** Standardmäßig wird immer ein englisches Tastaturlayout verwendet, desshalb müssen für die verwendeten Sonderzeichen folgende Tasten verwendet werden

US Layout	Deutsches Layout
=	´ (Akzenttaste)
/	- (Bindestrich)
-	ß (Eszet)

4. Die geänderten Eingaben werden mit **F10** bestätigt und der Server bootet direkt in die Shell. Nun muss noch das root-Dateisystem schreibbar gemountet (geladen) werden. Dazu wird in die Konsole folgender Befehl eingegeben

```
mount -o remount,rw /
```



5. Nun kann das Passwort geändert werden, dazu einfach

```
passwd
```

eingeben und ein neues Kennwort vergeben

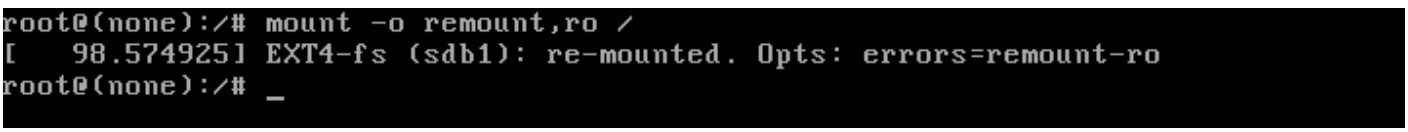
> *Hinweis: es werden keine Eingaben oder Sterne beim Tippen angezeigt; zudem bietet es sich an für den Moment an ein einfaches Kennwort einzugeben da man beim direkten Login über die Konsole noch nicht mit der Zwischenablage arbeiten kann*

> **UNMITTELBAR nach dem ersten Login via SSH ein starkes Kennwort vergeben!!!**

6. Wenn das Ändern des Passworts mit einer Erfolgsmeldung quittiert wurde muss das Dateisystem wieder nur-lesend gemounted (geladen) werden.

Dies geschiet mit dem Befehl

```
mount -o remount,ro /
```



Wenn auch dies mit einer Erfolgsmeldung quittiert wurde kann der Server neugestartet werden durch Betätigen der Reset-Taste oder halten des Startbuttons

SA-MP

Installation von SA-MP Servern und weitere nützliche Dinge

SA-MP Server installieren

Host aktualisieren und benötigte Pakete installieren

```
sudo apt update && sudo apt upgrade -y
sudo apt install lib32stdc++6 lib32z1 screen wget
```

Benutzerkonto anlegen

```
sudo adduser sampserver --gecos ""
```

> Passwort nach Aufforderung eingeben, mit Enter quittieren und nochmal eingeben (zur Bestätigung)

Serverdateien herunterladen

Hierzu wechseln wir in den eben erstellten Benutzer

```
sudo su - sampserver
```

und laden dann die Serverdateien herunter

```
wget -q -O - https://files.sa-mp.de/server/0.3.X/0.3.7/linux/samp037svr_R3.tar.gz | tar xvzf -
```

> Die Serverdateien wurden automatisch entpackt und liegen nun im Verzeichnis **samp03**

Abschließende Schritte

Um den Server starten zu können muss das RCON Passwort geändert werden

```
cd samp03/
sed -i 's/changeme/DEIN_NEUES_RCON_PASSWORT/g' server.cfg
```

Nun kann der Server gestartet werden

```
./samp03svr
```

Server im Hintergrund betreiben

Würde man nun das Terminalfenster (bspw. PuTTY) schließen würde auch unser SA-MP Server beendet werden. Um dies zu verhindern kann man sich dieses kleinen Skriptes bedienen

```
#!/usr/bin/env bash

start() {
  echo "Starte den SA-MP Server"
  screen -S samp_server -m -d sh -c './samp03svr'
  sleep 2
  echo "SA-MP Server wurde mit der PID $(ps fax | grep samp03svr | grep -v grep | head -1 | awk '{print $1}')
gestartet"
  echo "Status: ["$(tput setaf 2)OK$(tput sgr0)"]"
}

stop() {
  echo "Stoppe den SA-MP Server"
  screen -X -S samp_server quit
  sleep 2
  echo "Status: ["$(tput setaf 2)OK$(tput sgr0)"]"
}

case "$1" in
  start) start ;;
  stop)  stop;;
  *) echo "Auswahl: $0 start|stop" >&2
     exit 1
     ;;
esac
```

Den gesamten Code kopieren und in eine neue Datei einfügen

```
nano control.sh
--- Code einfügen, Editor mit CTRL-X -> Y -> ENTER beenden ---
chmod +x control.sh
```

SA-MP

Gamemodes auf einem Linux System kompilieren

Compiler herunterladen

Wir finden den Compiler auf GitHub, einfach den aktuellsten Release herunterladen

```
mkdir -p /tmp/pawn && cc /tmp/pawn
wget -q -O - https://github.com/pawn-lang/compiler/releases/download/v3.10.10/pawnc-3.10.10-linux.tar.gz | tar xvzf -
chmod a+x /tmp/pawn/pawnc-3.10.10-linux/bin/pawncc
```

Wir finden die Compiler-Executable dann im Verzeichnis `/tmp/pawn/pawnc-3.10.10-linux/bin/pawncc`

Library installieren

Um den Compiler ausführen zu können muss eine mitgelieferte Library-Datei in das passende Verzeichnis auf dem System kopiert werden

```
cp /tmp/pawn/pawnc-3.10.10-linux/lib/libpawnc.so /usr/lib
```

Compiler installieren

Installieren ist vielleicht das falsche Wort, wir kopieren einfach die Binary in ein Verzeichnis unserer Wahl, z.b.

```
mkdir ~/pawn-compiler
cp /tmp/pawn/pawnc-3.10.10-linux/bin/pawncc ~/pawn-compiler/
```

Compiler ausführen und .amx Datei erzeugen

```
cd ~/dein-gamemode/
~/pawn-compiler/pawncc "-iinclude" "gamemodes/gamemode.pwn"  "-;+" "-v2" "-d3" "-Z+" "-(+"
```

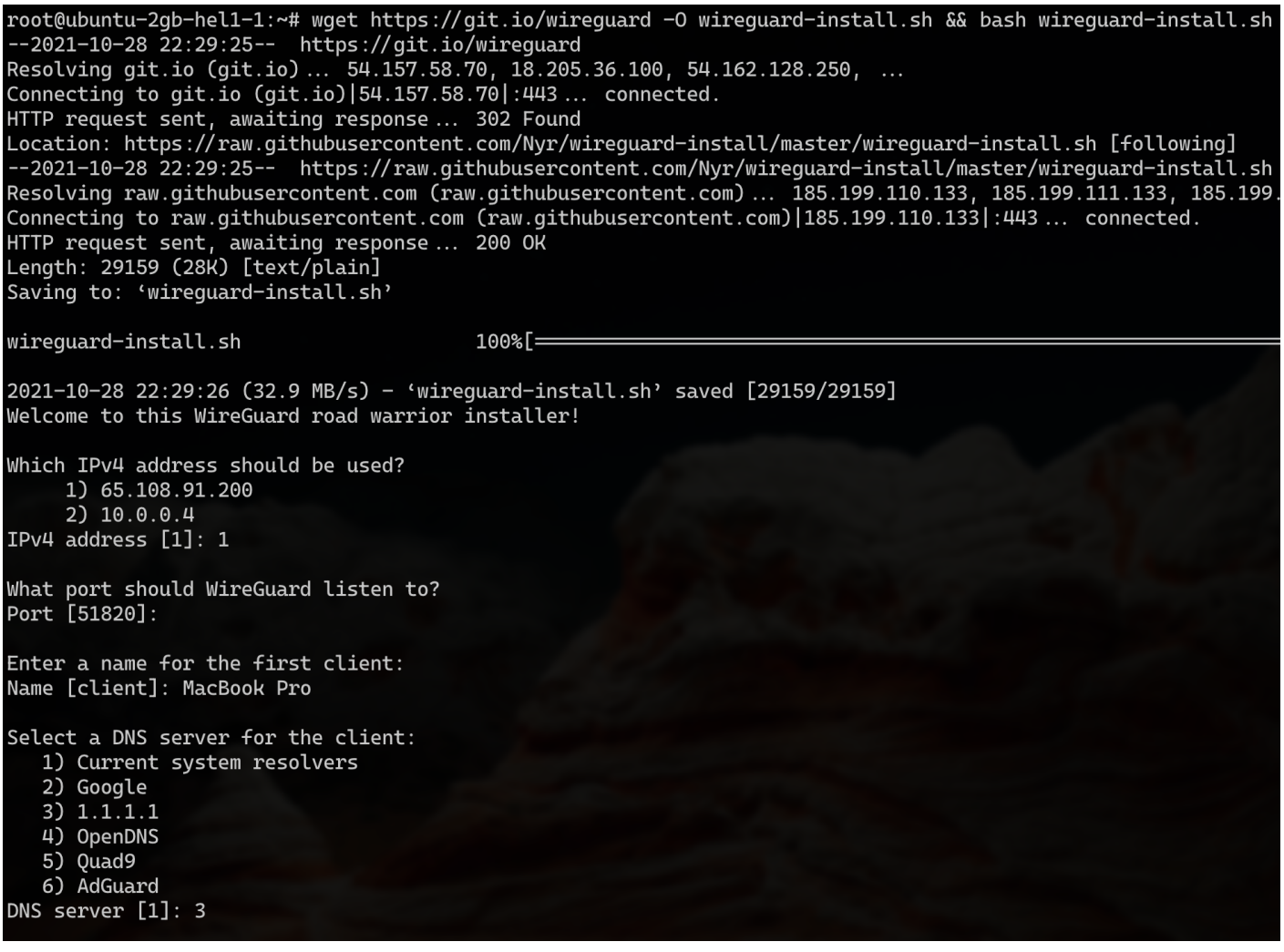
Wichtig ist hierbei dass dein Gamemode (die .pwn Datei) sich im Verzeichnis*gamemodes* sowie alle benötigten Includes sich im Verzeichnis *include* befinden und nicht wie vom Windows PC gewohnt im Ordner *pawno/includes*

Wireguard Server installieren & Clients einrichten

Server installieren

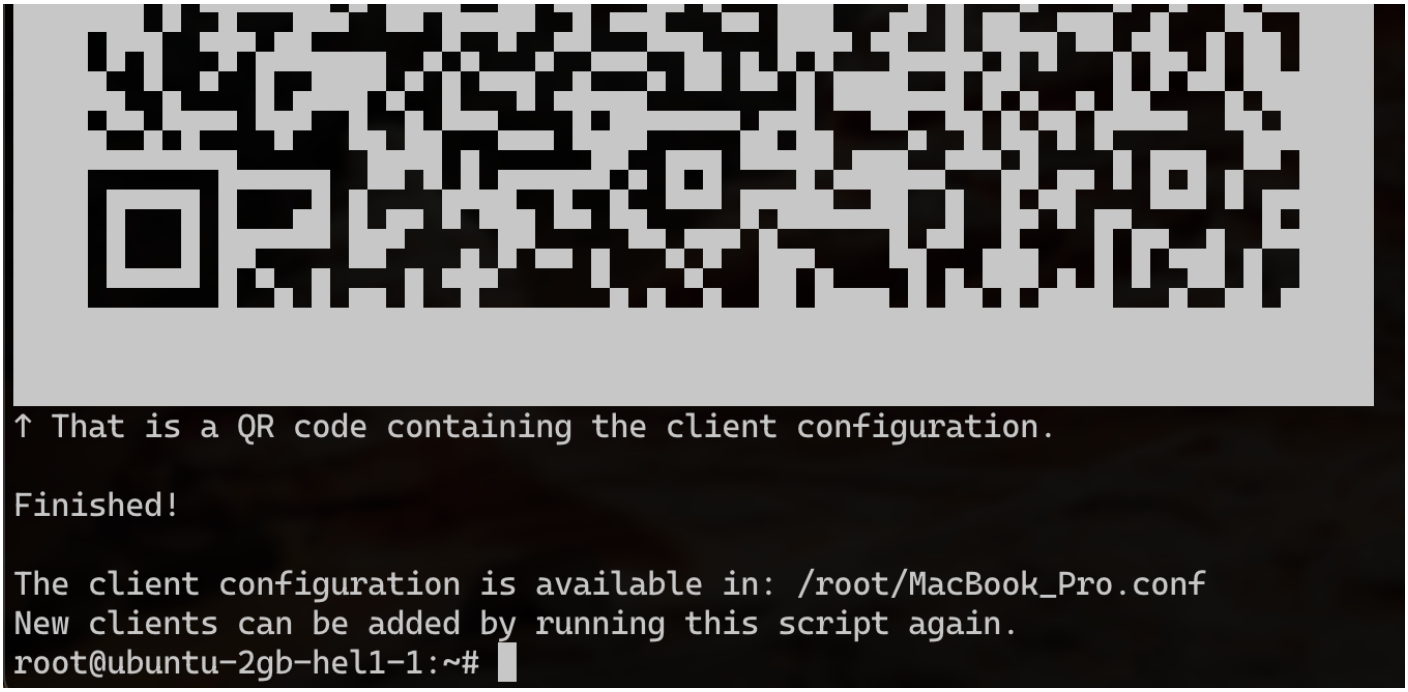
Wir bedienen uns dazu eines praktischen Tools welches die Installation und Verwaltung der Clients sehr einfach gestaltet

```
wget https://git.io/wireguard -O wireguard-install.sh && bash wireguard-install.sh
```



> Nach dem Absenden des oben genannten Befehls werden ein paar Fragen gestellt, diese werden im Zweifelsfall mit den Standardwerten bestätigt (ENTER Taste)

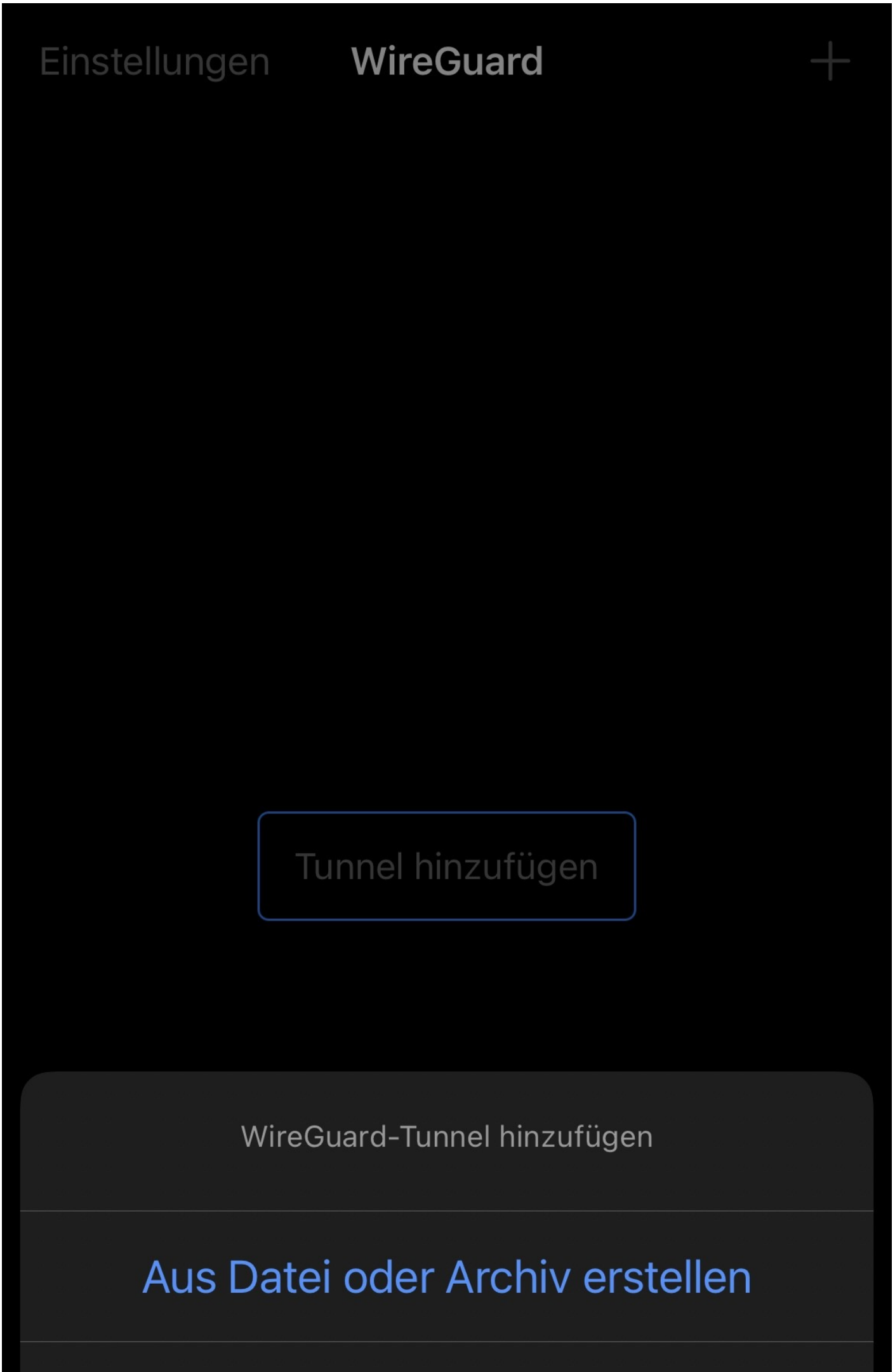
Am Ende der Installation wird im aktuellen Verzeichnis eine .conf Datei erstellt welche man in alle gängigen Wireguard-Clienten importieren kann, zudem erscheint ein QR Code mit welchem man die Mobilen Version von Wireguard für iOS und Android kinderleicht konfigurieren kann



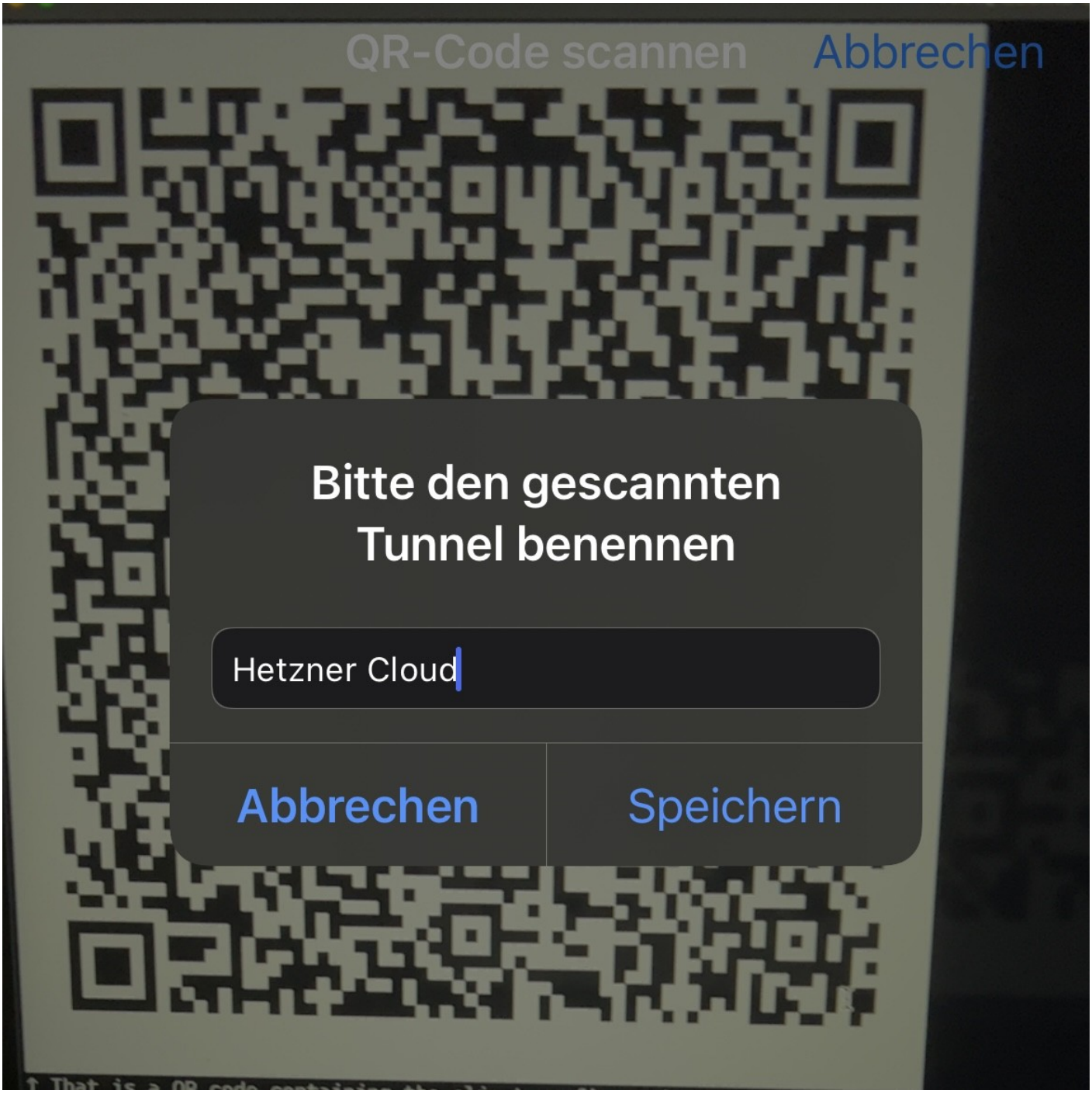
Konfiguration des iOS Clienten

Die Einrichtung ist selbsterklärend, hier die einzelnen Schritte in Bildern

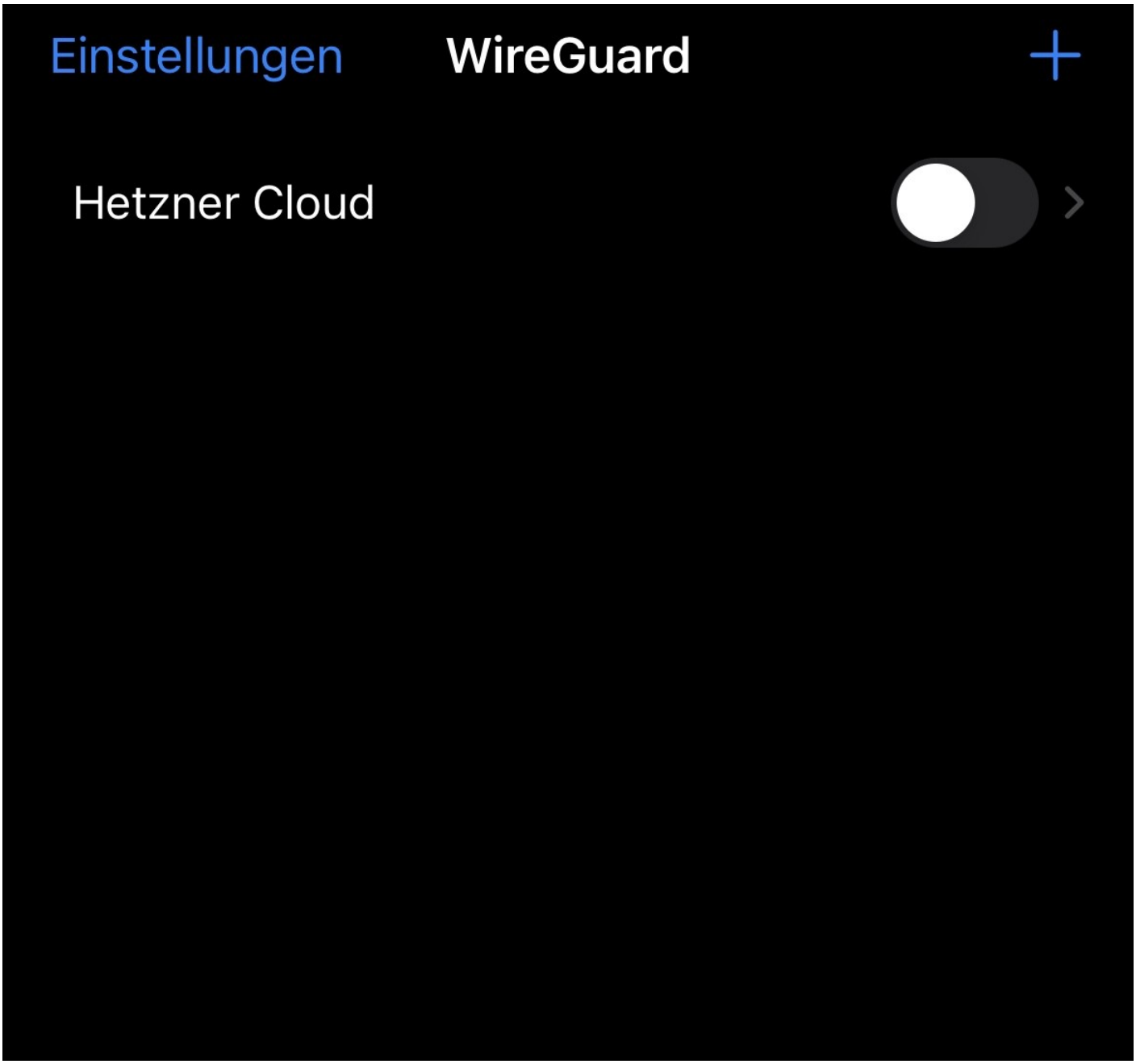
Schritt 1: Wireguard App installieren und starten, anschließend auf *Tunnel hinzufügen* gehen und *Aus QR-Code erstellen* anwählen


Einstellungen
WireGuard
Tunnel hinzufügen
WireGuard-Tunnel hinzufügen
Aus Datei oder Archiv erstellen

Schritt 2: QR-Code scannen und die Konfiguration mit einem selbst gewählten Namen versehen, die im Anschluss folgende Abfrage ob das Profil wirklich installiert werden soll bestätigen



Schritt 3: Verbindung aktivieren



Schritt 4: Fertig! Wir sind nun über unseren eigenen Wireguard Server mit dem Internet verbunden

MYIP.IS

Google Search

Your IP address is:

65.108.91.200

Your Hostname is:

static.200.91.108.65.clients.your-server.de

The IP used to connect to this webserver is 162.158.203.36 which is your proxy server's address. You connect to the proxy server using 65.108.91.200

- Click your IP address to copy it to the clipboard
- Set myip.is as your home page
- Add myip.is to your favorites
- add MyIP.Is to your homepage!

info@myip.is

cloud-init Deaktivieren

Start verhindern

Einfach eine leere Datei erstellen um den Start der cloud-init Dienste zu verhindern

```
sudo touch /etc/cloud/cloud-init.disabled
```

Deinstallieren

Alle cloud-init Dienste deaktivieren (Alle Optionen abwählen außer "None"):

```
sudo dpkg-reconfigure cloud-init
```

Pakete entfernen und die Konfigurationsordner von cloud-init löschen

```
sudo dpkg-reconfigure cloud-init
sudo apt-get purge cloud-init
sudo rm -rf /etc/cloud/ && sudo rm -rf /var/lib/cloud/
```

System neustarten

```
sudo reboot
```

Fehlermeldung beim Updaten der Paketlisten "... öffentlicher Schlüssel nicht verfügbar ..." beheben

Beim Ausführen von `apt update` erscheint die folgende oder eine ähnliche Fehlermeldung:

> Die folgenden Signaturen konnten nicht überprüft werden, weil ihr öffentlicher Schlüssel nicht verfügbar ist: NO_PUBKEY 0E98404D386FA1D9 NO_PUBKEY 6ED0E7B82643E131

Ich hatte dieses Problem bspw. auf einem Raspberry Pi welcher länger nicht aktualisiert wurde. Vermutlich fehlt dann nach einiger Zeit die Referenz von bspw. einem abgelaufenen Key auf einen Nachfolgeschlüssel.

Die Lösung dieses Problems ist denkbar einfach - man fragt die nicht vorhandenen Public-Keys einfach neu an

```
root@controller:~# apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 0E98404D386FA1D9
Executing: /tmp/apt-key-gpghome.Wn3QlHvSLW/gpg.1.sh --keyserver keyserver.ubuntu.com --recv-keys 0E98404D386FA1D9
gpg: Schlüssel 73A4F27B8DD47936: Öffentlicher Schlüssel "Debian Archive Automatic Signing Key (11/bullseye) <ftpmaster@debian.org>" importiert
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                importiert: 1

root@controller:~# apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 6ED0E7B82643E131
Executing: /tmp/apt-key-gpghome.49UdKYdmCn/gpg.1.sh --keyserver keyserver.ubuntu.com --recv-keys 6ED0E7B82643E131
```

```
gpg: Schlüssel B7C5D7D6350947F8: Öffentlicher Schlüssel "Debian Archive Automatic Signing Key
(12/bookworm) <ftpmaster@debian.org>" importiert
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                importiert: 1
```

Danach sollte ein `apt update` wieder möglich sein

Arbeiten mit systemd

Dienste verwalten

Starten eines Dienstes

```
sudo systemctl start <dienstname>
```

Startet den angegebenen Dienst.

Stoppen eines Dienstes

```
sudo systemctl stop <dienstname>
```

Stoppt den angegebenen Dienst.

Neustarten eines Dienstes

```
sudo systemctl restart <dienstname>
```

Startet den angegebenen Dienst neu. Dies ist nützlich, um Konfigurationsänderungen zu übernehmen.

Neuladen der Dienstkonfiguration

```
sudo systemctl reload <dienstname>
```

Lädt die Konfiguration des angegebenen Dienstes neu, ohne ihn vollständig neu zu starten.

Überprüfen des Status eines Dienstes

```
sudo systemctl status <dienstname>
```

Zeigt den aktuellen Status des angegebenen Dienstes an, einschließlich Informationen darüber, ob er aktiv ist und ob Fehler aufgetreten sind.

Aktivieren eines Dienstes beim Systemstart

```
sudo systemctl enable <dienstname>
```

Konfiguriert den angegebenen Dienst so, dass er beim Systemstart automatisch gestartet wird.

Deaktivieren eines Dienstes beim Systemstart

```
sudo systemctl disable <dienstname>
```

Verhindert, dass der angegebene Dienst beim Systemstart automatisch gestartet wird.

Überprüfen, ob ein Dienst aktiviert ist

```
sudo systemctl is-enabled <dienstname>
```

Überprüft, ob der angegebene Dienst so konfiguriert ist, dass er beim Systemstart automatisch gestartet wird.

Anzeigen aller aktiven Dienste

```
sudo systemctl list-units --type=service --state=active
```

Listet alle derzeit aktiven Dienste auf.

Anzeigen aller fehlgeschlagenen Dienste

```
sudo systemctl list-units --type=service --state=failed
```

Listet alle Dienste auf, die fehlgeschlagen sind.

Maskieren eines Dienstes

```
sudo systemctl mask <dienstname>
```

Verhindert, dass der angegebene Dienst manuell oder automatisch gestartet wird, indem er auf `/dev/null` verlinkt wird.

Demaskieren eines Dienstes

```
sudo systemctl unmask <dienstname>
```

Hebt die Maskierung des angegebenen Dienstes auf, sodass er wieder gestartet werden kann.

Neustarten des Systems

```
sudo systemctl reboot
```

Startet das System neu.

Herunterfahren des Systems

```
sudo systemctl poweroff
```

Fährt das System herunter.

Bearbeiten von Unit-Dateien

Anzeigen einer Unit-Datei

```
sudo systemctl cat <service>.service
```

Zeigt den Inhalt der Unit-Datei für den angegebenen Service an.

Erstellen einer neuen Unit-Datei

```
sudo systemctl edit --force --full <service>.service
```

Erstellt eine neue Unit-Datei für den angegebenen Service oder bearbeitet eine vorhandene vollständig.

Bearbeiten einer vorhandenen Unit-Datei

```
sudo systemctl edit --full <service>.service
```

Bearbeitet die vorhandene Unit-Datei für den angegebenen Service vollständig.

Überschreiben einer Unit-Datei

```
sudo systemctl edit <service>.service
```

Erstellt eine Override-Datei für den angegebenen Service, um bestimmte Einstellungen zu ändern oder hinzuzufügen.

Anzeigen der Überschreibungen einer Unit-Datei

```
sudo systemctl status <service>.service
```

Zeigt den Status des angegebenen Services an, einschließlich aller Überschreibungen.

Rückgängigmachen von Überschreibungen

```
sudo rm /etc/systemd/system/<service>.service.d/override.conf
```

Entfernt die Override-Datei für den angegebenen Service.

Neu Laden der Systemd-Konfiguration

```
sudo systemctl daemon-reload
```

Lädt die Systemd-Konfiguration neu, um Änderungen an Unit-Dateien zu übernehmen.

Neustarten eines Services

```
sudo systemctl restart <service>.service
```

Startet den angegebenen Service neu, um Änderungen wirksam werden zu lassen.