Installation Switcher 3

Anleitung

Version 3.4
Peter K. Boxler, September 2022

Inhaltsverzeichnis

Installation Switcher 3

Dieses Dokument beschreibt die Installation des Switcher 3 auf einem Raspberry Pi

1. Setup SD Karte

1.1 Prinzip

Folgendes Vorgehen hat sich bei mir bewährt:

- Eine Minimal-Karte erstellen, nur Basis Packages installieren
- Backup (Image) dieser Karte erstellen
- Weitere benötigte Packages installieren
- Bei Bedarf ab dem Backup weitere Karten erstellen für andere Projekte
- Die Basiskarte alle ca. 1.5 Jahre erneuern mit neuesten OS

1.2 Download Pi OS

Zuerst einer der drei verfügbaren OS Varianten runterladen:

- Pi OS full: Mit Desktopn und aller recommended Software (?)
- Pi OS medium: Mit Desktop
- Pi OS lite: Minimal Version

Leider hat die Raspberry Foundation **nicht** spezifizert, was jeweils genau enthalten ist.

Ich lade meist nur die **Lite Version**, da ich den Desktop für meine Projekte nicht brauche. Zudem kann ich dann genau jene Packages installieren, die mein Projekt braucht.

1.3 SD Karte behandeln

Auf dem Mac verwende ich das geniale Progamm **SDClone** um das Image auf die Karte zu schreiben. Die App erlaubt auch Backup, Verify und Shrinken von SD Karten.

SD Clone

2. Weitere Vorbereitung der SD Karte

Damit der Pi schon beim **ersten Boot** Verbindung mit meinem WLAN aufnehmen kann und dass zudem SSH schon nach dem ersten BOOT **enabled** ist, muss man VOR DEM ERSTEN BOOT folgendes machen (solange die Karte noch im Mac mounted ist):

2.1 Für WLAN

Im Root Directory der Karte (Volume **Boot**) eine Datei anlegen mit dem Namen:

```
wpa_supplicant.conf
```

Diese Datei liegt im Betrieb hier:

```
/etc/wpa_supplicant/wpa_supplicant.conf
```

Der Inhalt der Datei muss so aussehen (mit Editor erstellen). Bei mir sind immer 2 Netzwerke definiert.

```
# Datei wpa_supplicant.conf in der Boot-Partition (Raspbian Jessie)
country=CH
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1
network={
ssid="Netzwerkname1"
psk="passwort"
priority=90
}
network={
ssid="Netzwerkname2"
psk="passwort"
priority=90
}
```

Note: darauf achten, dass in dieser Datei vor und nach den Gleichheitszeichen keine Leerzeichen vorhanden sind! Der Ländercode muss in Großbuchstaben angegeben werden. Fehlt er, wird der WLAN-Controller automatisch deaktiviert.

2.2 Für SSH enable

Im Root Directory der Karte (Volume Boot) eine **leere** Datei anlegen mit dem Namen SSH (ohne Dateiendung !). Dazu öffne ich Terminal Window auf dem Mac und mache

```
touch ssh
```

Siehe hier

Note:

Falls die bestehenden WLAN Einstellungen einer SD-Karte **geändert** werden sollen, ist das Vorgehen am Ende dieses Dokuments beschrieben.

3. Boot, Login und Packages installieren

Nachdem diese beiden Dateien angelegt sind, kann der Pi mit dieser SD-Karte gebootet werden. Leider wird bei ersten Boot meist das Filesystem expandiert, dazu später mehr.

Ein Terminal Fenster öffnen und bei Pi mit SSH einloggen (User: pi, Passwort raspberry):

```
ssh pi@raspberrypi
```

Danach rufe ich meist erstmal das Konfig-Programm des Pi auf und ändere die Konfig.

```
sudo raspi-config
```

- Locale: de_CH.UTF-8
- Tastatur-Layout : Deutsche Schweiz
- WiFI Country: CH
- Hostname nach Belieben
- Passwort für User Pi ändern
- Enable i2c (falls nötig)

Dann reboot und check, danach installieren der für Python3 notwendigen Packages:

```
sudo apt-get update
```

Bei Pi OS Lite diese Packages installieren:

```
sudo apt install python3-pip
pip3 install Rpi.GPIO
```

Damit ist die Basis Karte erstellt, alle anderen Packages werden später bei Bedarf installiert.

Note: Ich arbeite nur noch mit **Python3**, da Python2 nun schon vor mehreren Jahren durch Python3 abgelöst wurde. Bislang musste ich in meinen älteren Python2 Programmen nur alle **print**-Statements ändern: in Python3 ist print einen Funktion und die Syntax ist so: **print („hallo Welt“)**. Es sind also lediglich zwei Klammern ( ) einzufügen.

4. Karte Backup

Nachdem der Pi mit dieser Basiskarte ok läuft (noch ohne Applikation) mache ich einen Backup der SD Karte mit dem erwähnten Programm **SDClone**. Dann kann ich darauf zurückgreifen, falls ich später wieder eine vorkonfigurierte SD Karte brauche.

Aber: Falls das Filesystem schon vergrössert ist, wird es manchmal unmöglich, ab dem Backup später eine neue Karte zu erstellen (restore), da nicht alle SD Karten gleich gross sind. Zu oft ist mir passiert, dass ein Restore eines Image unmöglich war, weil das Image 15.92 GB gross ist, die neue SD Karte aber bloss 15.32 GB.
Ich verwende ausschliesslich 16 GB Karten.

Aus diesem Grunde wird die lauffähige Basis-Karte erst **shrinked** und dann erst wird ein Backup erstellt. Damit ist sichergestellt, dass ich in jedem Fall später einen Restore machen kann.

Nach dem Shrinken mache ich jedoch nochmals einen kurzen Test mit dem Pi. Bootet die Karte immer noch ok, dann kann Backup erstellt werden.

Diese Operationen sind alle mit SDClone möglich.

Das Problem zu kleine Karte ist hier beschrieben.

5. Verwendung der Basis-Karte

Die Basiskarte kann nun für Applikationen verwendet werden. Zuerst wrid die Karte aber updated mit folgenden Befehlen,
siehe hier

```
sudo apt-get update
sudo apt-get upgrade
```

Oder auch

```
sudo apt-get dist-upgrade
```

Dann ist Karte ready für ein neues Projekt.

6. Schritt 2

Partition erweitern mit sudo raspi-config (damit genug Platz auf Karte)
prüfen mit

```
df -h
```

Pi erneut booten und Pakage List erneuern mit

```
sudo apt-get update
```

Wenn ein Pi OS mit recommended Software vorliegt (full), fahre weiter mit Schritt 3a. Falls hingegen nur ein **Pi OS Lite** vorhanden ist, zuerst diese Packages installieren:

```
sudo apt install python3-pip
sudo apt install git
pip3 install Rpi.GPIO
```

7. Schritt 3a

Nun alle für Switcher3 notwendigen Packages installieren

```
pip3 install configparser
pip3 install psutil
pip3 install flask
pip3 install flask_socketio
```

```
pip3 install paho-mqtt
sudo apt install -y mosquitto mosquitto-clients
```

Note: falls **Schaltart 5** in einer Dose (**433 MHZ Funk mit Send Modul)** verwendet wird, ist auch noch dies zu installieren:

```
sudo apt-get install wiringpi
```

8. Schritt 4, Switcher3 Code von github clonen

Ins Home Dir des User Pi wechseln und dies eingeben

```
git clone https://github.com/dakota127/switcher3.git
```

damit entsteht ein Directory (Folder) switcher3, der allen notwendigen Code enthält.

9. Schritt 5, Setup Switcher

Ins Directory switcher3 wechseln und diese zwei Shell Scripts ausführen (erstes mit sudo und zweites ohne sudo !):

```
sudo chmod 755 sources/shell_scripts/setup_swi.sh
sources/shell_scripts/setup_swi.sh
```

Dieses Script macht folgendes:

- kopiert Inhalt des Folders **sources** in den switcher3 Hauptordner, löscht Folder sources
- kopiert User-Config-File und Passwort-File für mosquitto
- kopiert die Shellscripts für den Auto-Start des Switcher3 und des Flask Webservers
- installiert obige Shellscripts im systemd-Mechanismus des OS, damit werden Switcher3 und der Webserver nach dem booten gestartet. Zudem ist start/stop möglich.

Für **Uninstall** siehe Kapitel Notes (am Schluss des Dokuments))

Nach Ausführung der Shell Scripts einen Reboot des Pi auslösen:

```
sudo reboot
```

10. Schritt 6, Fixe IP-Adresse für den Raspberry Pi

Es kann sinnvoll sein, dem Switcher PI eine fixe IP-Adresse zuzuteilen - damit wird der MQTT Broker auch mit einer fixen IP-Adresse angesprochen. Da ja die Smart-Switches in der Tasmota Konfiguration die IP-Adresse des Brokers benötigen, ist es günstig, wenn die IP-Adresse fixiert wird.
Wie man das macht im Pi zeigt diese Anleitung im Netz:
Fixe IP-Adresse für Raspberry Pi

11. Definition der Dosen

Nach der Installation des Switchers sind 4 Dosen konfiguiert (in File **swdosen.ini**) und im Konfigfile **swconfig.ini** sind die maximal 5 Dosen mit Typ2 definiert (dose_n_schaltart = 2), also normale Funksteckdosen).

Die Anzahl der Dosen im erwähnten File **swdosen.ini** bestimmt, wieviele Dosen beim Start des Switchers konfiguriert werden. Dies kann im WebInterface **jederzeit** geändert werden, wenn der Switcher läuft.
Verwendet man andere Dosen (zB. Smart Switches), so ist deren Typ im File **swconfig.ini** anzupassen. Für Smart Switches (Sonoff) ist dies zu setzen: **dose_n_schaltart = 3,1,1**

Nach Installieren des Switchers (siehe Install Dokument) müssen im XML Steuerfile die Schaltzeiten und auch die Zimmerbezeichnungen mit einem Texteditor angepasst werden. Es empfiehlt sich, den Steuerfile anschliessend mit dem XML-Prüfprogamm zu testen.

12. Schritt 7

Enable autostart von mosquitto

```
sudo systemctl enable mosquitto.service
```

13. Schritt 8

Mosquitto prüfen mit

```
mosquitto -v  -c /etc/mosquitto/mosquitto.conf
```

14. Schritt 9, mosquitto starten/stoppen

```
sudo service mosquitto start
sudo service mosquitto restart
sudo service mosquitto stop
netstat -tln | grep 1883
```

Schauen, ob der Prozess läuft:

```
ps -ef | grep mosq
```

Statusabfrage mit:

```
sudo service mosquitto status
```

Log von mosquitto anschauen

```
sudo tail /var/log/mosquitto/mosquitto.log
```

15. Schritt 10, Mosquitto Config

After setup of switcher3 the broker is configured (default config file) to allow only clients with known user-id's in the LAN to connect. If you want to change any of this behaviour, you **need to have a user config file** and also acl file. See below.

Mosquittos default config file should **never be modified.**

How to add a user config file:

Create a user config file for mosquitto, description here
https://mosquitto.org/man/mosquitto-conf-5.html

Fortunately you already have one, check supplied folder **mosquitto_config -** there you will find four files

- my_mosquitto.conf private config file for mosquitto
- my_passw.txt password file for mosquitto
- my_aclfile.txt ACL permissions on a per user basis

See **end of document** for a schematic showing relationship between these files and Mosquitto is configured for these tests. This is important as of version 2.x of Mosquitto, read this if want to avoid headaches.

Note

Never change the **original** mosquitto config file which can be found here:

```
/etc/mosquitto/mosquitto.conf
```

This file looks like this (mosquitto v2.0.11)

```
# Place your local configuration in /etc/mosquitto/conf.d/
# A full description of the configuration file is at
# /usr/share/doc/mosquitto/examples/mosquitto.conf.example

pid_file /run/mosquitto/mosquitto.pid
 persistence true
persistence_location /var/lib/mosquitto/

log_dest file /var/log/mosquitto/mosquitto.log

include_dir /etc/mosquitto/conf.d
```

Mosquitto always looks into the folder /etc/mosquitto/conf.d/ for a additional user config file. Any file found there (no matter what the file name) is considered to be a user config file.
So we place our user config file into this folder.

Der default Passwort File **my_passw.txt** enthält 2 Paare User-ID/Passwort:

- UserID: dagobert, Passwort: itscool
- UserID: test127, Passwort: 123-123

Das erste Paar wird vom Switcher3 und den WiFi Schaltsteckdosen verwendet. Das andere Paar kann für andere MQTT Tests verwendet werden.

Aber Achtung:

> *MQTT provides username/password authentication as part of the protocol. Use the password_file option to define the valid usernames and passwords. Be sure to use network encryption if you are using this option otherwise the username and password will be vulnerable to interception. Use the per_listener_settings to control whether passwords are required globally or on a per-listener basis*

16. Schritt 11, Test mosquitto

IP-Adresse des Pi holen:

```
hostname -I
```

mosquitto testen mit Sub und Pub (Aktuelle IP-Adresse anpassen !)
In 2 Konsolen beim Pi anmelden mit SSH. Eine Konsole für Publish und eine für Subscribe. Falls der Broker **keine** User-ID/Passwort verlangt, in den Konsolen dies eingeben:

```
mosquitto_sub -h 192.168.1.130 -p 1883 -v -t test
mosquitto_pub -h 192.168.1.130 -p 1883 -t test -m „Hello world, Mosquitto“
```

Falls der Broker User-ID/Passwort verlangt, in den Konsolen dies eingeben:

```
mosquitto_sub -h 192.168.1.130 -p 1883 -v -t test -u test127 -P 123-123
mosquitto_pub -h 192.168.1.130 -p 1883 -t test -m „Hello world, Mosquitto“ test127 -P 123-123
```

Wenn Resultat ok ist, kann der Pi neu gebootet werden. Alles iO.

Es ist sinnvoll, auf einem Mobile Device eine **MQTT App** zu haben. Damit kann der Broker geprüft werden - auch können publish und subscribe abgesetzt werden.

- Ich verwende auf dem iPad eine **iOS App** genannt **MQTT Inspector**. Damit können Publish und Subscribe gemacht werden und es können beliebige Topics/Payloads gesant werden. Damit ist es einfach, den Smart Switch zu testen.
- Für **Android** (Google PlayStore) ist es die App **MyMQTT**, die dafür gut geeignet ist.

17. Schritt 12

Wenn der Pi ok bootet, kann das Webinterface im Browser aufgerufen werden mit (sofern Hostname swi1 gesetzt ist, sonst entsprechend)

```
swi1.local:4000
```

Läuft der Pi, so kann man ebenfalls vis SSH einlogen mit

```
ssh pi@swi1.local
```

Mit welchen WLAN sich der Pi verbunden hat kann man feststellen mittels:

```
iwgetid
```

oder

```
iwconfig
```

18. Start/Stop Switcher3

Für den Start von Switcher3 bei Boot des Pi wird der systemd Meccano verwendet.

siehe systemd

Durch die in Schritt 6 installierten Scripts sind folgende Kommandos möglich für Switcher3 und Webserver:

```
sudo systemctl start switcher3.service
sudo systemctl stop switcher3.service

sudo systemctl start swserver3.service
sudo systemctl stop swserver3.service
```

Abfragen

```
systemctl is-active switcher3.service
systemctl is-active swserver3.service
```

19. Schritt 13

Happy switching !

20. Uninstall Switcher 3

Ins Directory switcher3 wechseln und dieses Shell Script ausführen:

```
sudo shell_scripts/remove_swi.sh
```

Dieses Script stoppt die Switcher Programme und entfernt die systemd Einträge. Das Directory switcher3 bleibt bestehen. Soll dies auch entfernt werden, so wechsle man ins übergeordnete Directory (/home/pi) und führe diesen Command aus:

```
sudo rm -R -f switcher3
```

Danach ist switcher3 vollständig entfernt - Mosquitto bleibt jedoch bestehen.

Eventuell **Uninstall** von mosquitto:

```
sudo apt-get purge mosquitto
```

```
sudo apt-get --purge remove mosquitto
```

21. Konfiguration von Smart Switches (Tasmota)

Setup und Konfiguration von Smart Switches der Marke Sonoff ist in einem separaten Dokument beschrieben.

Setup_Sonoff_Devices.pdf

Dort wird das Flashen mit Tasmota Firmware und der Initial Setup für Verwendung mit Switcher3 beschrieben.

22. Anhang

22.1 Notizen zu Mosquitto:

Never change the **original** mosquitto config file which is here:

```
/etc/mosquitto/mosquitto.conf
```

Mosquitto always looks into the folder /etc/mosquitto/conf.d/ for a additional user config file. Any file found there (no matter what the file name) is considered to be a user config file. So we place our user config file into this folder.

Die **eigene mosquitto Configuration** ist im File **my_mosquitto.conf** in diesem Directory

```
/etc/mosquitto/conf.d
```

Der von Switcher3 gelieferte eigene Configfile kann verändert werden mit

```
sudo nano /etc/mosquitto/conf.d/my_mosquitto.conf
```

Der von Switcher3 gelieferte Passwort-File heisst my_passw.txt, dessen Name ist in my_mosquitto.conf definiert. Der Passwort-File von switcher3 enthält zwei User:

- dagobert mit Passwort itscool
- test127 mit Passwort 123-123

Der Passwort File für Switcher liegt hier:

```
sudo nano /etc/mosquitto/my_passw.txt
```

Info im Web für Mosquitto

https://iotbytes.wordpress.com/mosquitto-mqtt-broker-on-raspberry-pi/
http://www.steves-internet-guide.com/mosquitto-logging/

22.2 WLAN Einstellungen ändern auf einer bootbaren SD-Karte

Falls die WLAN Einstellungen einer SD-Karte geändert werden sollen (z.B. weil der Pi in neuer Umgebung laufen soll) ist folgendermassen vorzugehen:

Einen Pi mit LAN Stecker benutzen (Modell 2,3 oder 4), LAN Kabel einstecken und mit der SD-Karte booten. Pi verbindet sich mit dem LAN. IP-Adresse feststellen (Netzwerkmonitor), ein Terminal öffnen und einloggen mit ssh pi@ipadress

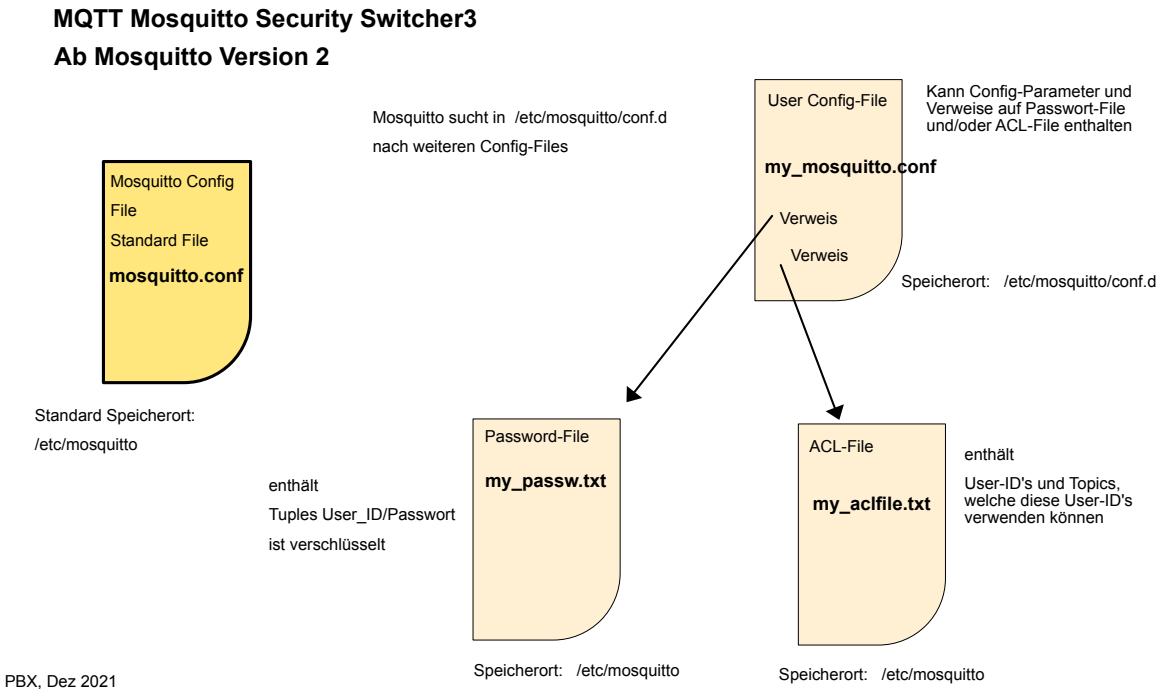
Dann die Datei wpa_supplicant.conf editieren mit folgendem Befehl (commandline):

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Einen Network Eintrag in der Datei ändern: ssid und psk neu setzen.

Exit aus dem login, LAN Kabel entfernen und Pi neu booten. Er wird sich mit dem (neuen) WLAN verbinden.

22.3 Config-Files for Mosquitto



As noted one should never change the standard moquitto configfile. Simply add a user configfile like this:

The User config file supplied with this package look like this:

An important line in this file is this:

```
listener 1883 0.0.0.0
```

This allows connection from other machines.

```
# Config file for mosquitto
#  personal config file
# See mosquitto.conf(5) for more information.

#  acl_file added Dez 2021 by Peter
#
persistence false
persistence_file mosquitto.db
allow_anonymous false
allow_zero_length_clientid true
log_timestamp true
per_listener_settings false
# set listener to 0.0.0.0 damit auch von extern auf den broker connected werden kann
# ohne 0.0.0.0 kann nur auf der lokalen machine connected werden !!!
listener 1883 0.0.0.0
listener 9001 127.0.0.1
password_file /etc/mosquitto/my_passw.txt
#
# acl file added
acl_file /etc/mosquitto/my_aclfile.txt
```

The password file looks like this (contains user-id/password combination). This file will be encrypted after install.

```
dagobert:itscool
test127:123-123
test:051054
```

The ACL file looks like this: It contains all the topics a specific user-id can subscribe to. Note: this is the acl-file used in my switcher3 project.

```
#  aclfile für switcher3 mqtt
#  December 2021 Peter Boxler
#---------------------------------------
# This affects access control for clients with no username.
topic read $SYS/#

# This only affects clients with username "test127"
user test127
topic test
topic prisca

# This only affects clients with username "dagobert"  (Switcher3)
# Switcher3 muss sich mit user_id dagobert beim broker anmelden (siehe configfile swconfig.ini))
user dagobert
topic test
topic swi/#
topic serv/#
topic aliste
topic wetter
topic home
topic response
#
# topics für smart switches
topic cmnd/#
topic stat/#

# This affects all clients.
pattern write $SYS/broker/connection/%c/state
```

22.4 Wichtige Commandline-Comands

Für den Betrieb und die Ueberwachung eines Pi sollte man folgende Comands kennen.

Installed Packages anzeigen:

```
dpkg -l
```

KeyEintrag für bestimmte IP-Adr löschen

```
ssh-keygen -R 192.168.1.99
```

Shutdown the Pi immediately

```
shutdown -h 0
```

CPU-Info anzeigen

```
cat /proc/cpuinfo
```

Laufende Python3 Prozesse anzeigen

```
ps -elf | grep python
```

WiFi-Info anzeigen

```
sudo iwlist wlan0 scan
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Raspberry Typ auslesen:

```
cat /sys/firmware/devicetree/base/model
```

Sammelinfo ausgeben mit

```
hostnamectl
```

Mehr in Info durch install eines Packages

```
sudo apt-get install lshw
```

danach

```
sudo lshw
```

oder:

```
sudo lshw | grep "product:" -m 1
```

Metwork Infos mit:

```
sudo lshw | tail -n 9 | grep -iE "size|capacity|capabilities|speed"
```

Weiter Info mit **vcgencmd**

```
vcgencmd commands
```

Aktuelle Config ausgeben:

```
vcgencmd get_config int
```

Firmware Version:

```
vcgencmd version
```

Boot Config mit:

```
cat /boot/config.txt
```

OS-Release ermitteln:

```
cat /etc/os-release
cat /etc/issue
```

Peter K. Boxler, im September 2022
First Version Juli 2018

end of document