Setting up TLS certificates for Mosquitto

Prerequisites

This tutorial assumes you are using IOT Stack for Raspberry Pi and the following containers are available:

- Portainer
- Mosquitto
- Node-RED

It has been tested on Debian + Raspbian, warranty void if using another OS. For the Let's Encrypt example `certbot` has to be installed.

Self signed certificates

In this step we are going to create our own fake-CA and sign our own certificates. Then we will set those up for use with Mosquitto.

Create the certificates

Using openssl (`sudo apt install openssl`).

```
openssl genrsa -out ca.key 4096
openssl req -new -x509 -days 1826 -key ca.key -out ca.crt
openssl genrsa -out server.key 2048
openssl req -new -out server.csr -key server.key
openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out
server.crt -days 360
```

Changes for Mosquitto

1. Create a `cert` folder in `services/mosquitto`

2. Copy `ca.crt`, `server.crt` and `server.key`

3. Update the docker compose file:

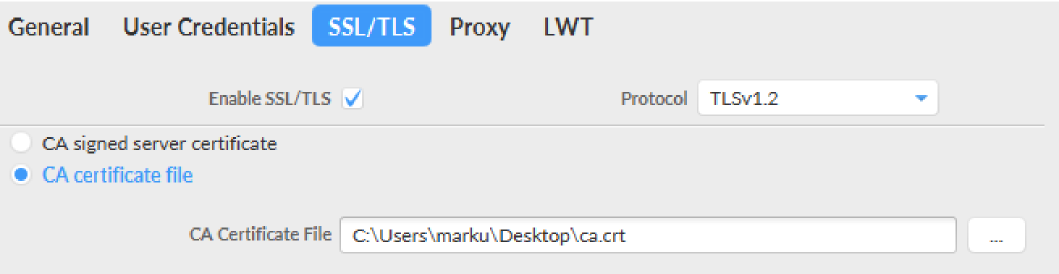
    1. Add volume mapping for certificates: `- ./services/mosquitto/certs:/mosquitto/certs`
    2. Add port mapping for secure port: `- 8883:8883`

4. Change `mosquitto.conf`:

    1. add `listener 8883`
    2. add `cafile /mosquitto/certs/ca.crt`
    3. add `keyfile /mosquitto/certs/server.key`
    4. add `certfile /mosquitto/certs/server.crt`
    5. add `tls_version tlsv1.2`

Changes for MQTT.FX

- Setup SSL/TLS for the broker

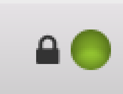
  - Port is now `8883`
  - Use option `CA certification file` and supply the `ca.crt` file previously created
  - 

    Enable SSL/TLS ☑ Protocol TLSv1.2

    ○ CA signed server certificate

    ● CA certificate file

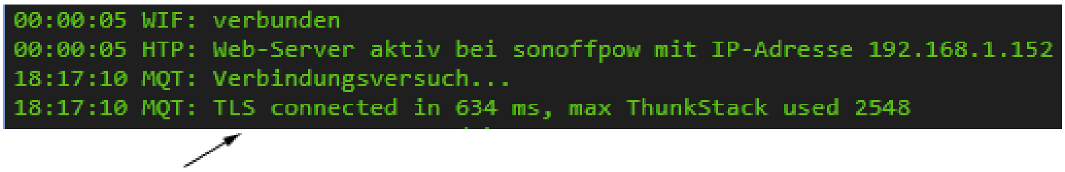
    CA Certificate File C:\Users\marku\Desktop\ca.crt

- Check if connection is encrypted:
  - 

Changes for Node-RED

- Update `mqtt-broker` configuration
  - Select `enable secure connection`
  - Update/Add a TLS configuration
    - Remove `verify server certificate` checkmark
  - Change port to `8883`

Changes for Tasmota

- Compile from source required
- Clone project from GitHub
- Create `user_config_override.h` (based on sample in git repo)
  - add `#define USE_MQTT_TLS`
  - add `#undef MQTT_PORT`
  - add `#define MQTT_PORT 8883`
- Create `platformio_override.ini` (based on sample in git repo)
  - enable `-DUSE_CONFIG_OVERRIDE` build_flag
  - also add `tasmota-DE` for `default_envs` (build target)
- Flash firmware
  - You may want to use Tasmota PyFlasher
- Perform device setup (WLAN, broker,...) as usual
- If everything is setup correctly Tasmota console output should look like this:
  - 

    ```
    00:00:05 WIF: verbunden
    00:00:05 HTP: Web-Server aktiv bei sonoffpow mit IP-Adresse 192.168.1.152
    18:17:10 MQT: Verbindungsversuch...
    18:17:10 MQT: TLS connected in 634 ms, max ThunkStack used 2548
    ```

Let's Encrypt

Overall very similar to the process for using self signed certificates. Assuming you previously setup self signed certificates apply the following changes to switch to Let's Encrypt certificates.

First of all you'll need a domain (e.g. DynDNS + subdomain, with port forwarding).

Let's Encrypt performs its challenge for verifying domain ownership via port 80, so you'll need that in addition to port 8883 (at least temporarily).

Use the `sudo certbot certonly --standalone` command to retrieve certificates for the domain (as specified via `certbot` inquiries). This will yield you four files:

- cert.pem
- chain.pem
- privkey.pem
- fullchain.pem

Copy three of them (`cert.pem`, `chain.pem` & `privkey.pem`) to the certs folder created previously. We won't need `fullchain.pem`. Make the following changes to the `mosquitto.conf` file:

- certfile: path to `cert.pem`
- cafile: path to `chain.pem`
- keyfile: path to `privkey.pem`

You might need to reboot to apply all changes.

Changes for MQTT.FX

- Change broker address to the one defined in the certificate (has to be available of course)
- If you selected a port different than `8883` (port forwarding) input the correct one
- In the SSL/TLS tab switch to `CA signed server certificate`

Changes for Node-RED

- Change the mqtt-broker
  - Change server to the domain specified for the certificate
  - Optionally change port
- Change TLS config
  - Remove the (self signed) CA certificate
  - Tick `verify server certificate`

Changes for Tasmota

Again we need to compile our own version of the firmware.

- Apply all the changes required for the self signed certificate version (if not done so already)
- Add to `user_config_override.h`:
  - `#define USE_MQTT_TLS_CA_CERT`
- On the device:
  - Set broker hostname to domain (as defined in certificate)
  - Optionally change port

Let's Encrypt signed certificates should now be set up and usable by both the Tasmota flashed device and applications like MQTT.FX and Node-RED.

Please note that no automatic certificate renewal has been set up and Let's Encrypt certificates expire within three months.