

Lab 15 – MQTT Sikkerhet II



1 OpenSSL software

Make sure that you have OpenSSL available.

Windows: <https://slproweb.com/products/Win32OpenSSL.html>

On Linux: You probably have it already. Download via channels if you need to.

2 Mosquitto and TLS

2.1 Edit the mosquitto.conf file (assuming “clean” config file)

These settings must be updated. Some must wait until you have created certificates.

- Change the port option to listen to **8883**.
- Set cafile (or capath) -- here we will have the CA certificate
- Set certfile to the the broker certificate
- Set keyfile to the broker (private) keyfile
- Only permit TLSv1.2
- bind_address <192.168.0.103>¹ # This optional, but useful

Store the file in the **config** directory. Give it a sensible name.

It is useful to start the broker with “**mosquitto -c config/mosquitto.conf -v**” to get some logging outputs and to be explicit about the config file.

¹ The actual address or hostname. All examples and certificates here are with this as the broker address.

3 Creating certificates

In the **mosquitto** directory: create a directory **certificates** . Then cd there.

Note 1:

There seems to be some trouble with the Windows installs of OpenSSL. I did the certificate job on **Linux Mint 19** and **OpenSSL 1.1.0**. It worked fine. Should also work on the RPI.

The certificates could not be created on Windows with OpenSSL 1.1.1.

3.1 The CA certificate

User input in **yellow**. The CA files are called “dat235ca” (crt & key).

```
$ openssl genrsa -out dat235ca.key 4096
Generating RSA private key, 4096 bit long modulus
.....++
.....++
....++
e is 65537 (0x010001)

$ openssl req -x509 -new -nodes -key dat235ca.key -sha256 -days 3650 -out dat235ca.crt
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:NO
State or Province Name (full name) [Some-State]:AGDER
Locality Name (eg, city) []:GRIMSTAD
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MOSQUITTO CA
Organizational Unit Name (eg, section) []:.
Common Name (e.g. server FQDN or YOUR name) []:MOSQUITTO CA
Email Address []:mosquittoca@none.com

$ openssl x509 -in dat235ca.crt -noout -text
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            b1:04:c0:9f:7d:46:81:b0
        Signature Algorithm: sha256WithRSAEncryption
        Issuer: C = NO, ST = AGDER, L = GRIMSTAD, O = MOSQUITTO CA, CN = MOSQUITTO CA,
        emailAddress = mosquittoca@none.com
        Validity
            Not Before: Oct 18 13:31:27 2018 GMT
            Not After : Oct 15 13:31:27 2028 GMT
        Subject: C = NO, ST = AGDER, L = GRIMSTAD, O = MOSQUITTO CA, CN = MOSQUITTO CA,
        emailAddress = mosquittoca@none.com
        Subject Public Key Info:
```

```
Public Key Algorithm: rsaEncryption
Public-Key: (4096 bit)
Modulus:
    00:e4:85:69:94:d1:5f:be:e1:0b:7b:59:b5:55:69:
        :      :
    Skipped many lines here
        :      :
    cd:2d:79:46:d0:be:7b:5e:b0:d2:e1:d7:ac:a6:71:
    1c:00:b7
Exponent: 65537 (0x10001)
X509v3 extensions:
X509v3 Subject Key Identifier:
    19:E1:DA:4C:09:0F:FC:05:B4:40:80:2C:4E:98:0D:E6:0C:EB:83:CC
X509v3 Authority Key Identifier:
    3eyed:19:E1:DA:4C:09:0F:FC:05:B4:40:80:2C:4E:98:0D:E6:0C:EB:83:CC

X509v3 Basic Constraints: critical
CA:TRUE
Signature Algorithm: sha256WithRSAEncryption
    5f:00:ce:4b:0f:67:55:35:59:be:06:d9:76:4f:d8:e6:99:81:
        :      :
    Skipped many lines here
        :      :
    f6:6b:f5:4f:33:17:34:86
```

Now, you have the CA certificate and keyfile:

- dat235ca.key
- dat235ca.crt

The certificate is in PEM format.

3.2 The broker certificate

Before we start, we need to know either the domain name or the IP address where the broker is running.

Check this out: <https://www.lifewire.com/192-168-1-101-818390>

Don't use "localhost" -- it will likely not work. Similarly, use of "127.0.0.1" is not a good idea.

- 1) Generate a server keyfile
- 2) Generate a server certificate request (csr) file
- 3) Grant the request and produce the signed server certificate (signed with CA key)

```
$ openssl genrsa -out server.key 4096
Generating RSA private key, 4096 bit long modulus
.....
.....++
.....++
e is 65537 (0x010001)

$ openssl req -new -key server.key -out server.csr
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.

Country Name (2 letter code) [AU]:NO
State or Province Name (full name) [Some-State]:AGDER
Locality Name (eg, city) []:GRIMSTAD
Organization Name (eg, company) [Internet Widgits Pty Ltd]:MQTT SERVER
Organizational Unit Name (eg, section) []:MQTT
Common Name (e.g. server FQDN or YOUR name) []:<domain name|IP address>
Email Address []:mosquitto@none.com

Please enter the following 'extra' attributes
to be sent with your certificate request
A challenge password []:.
An optional company name []:Mosquitto

$ openssl x509 -req -in server.csr -CA dat235ca.crt -CAkey dat235ca.key -Ccreateserial -out
server.crt -days 730 -sha256
Signature ok
subject=C = NO, ST = AGDER, L = GRIMSTAD, O = MQTT SERVER, OU = MQTT, CN = 192.168.0.103,
emailAddress = mosquitto@none.com
Getting CA Private Key

$ openssl x509 -in server.crt -noout -text
Certificate:
    Data:
        Version: 1 (0x0)
        Serial Number:
            d8:ee:52:91:20:ab:cf:f3
        Signature Algorithm: sha256WithRSAEncryption
```

```
Issuer: C = NO, ST = AGDER, L = GRIMSTAD, O = MOSQUITTO CA, CN = MOSQUITTO CA,
emailAddress = mosquittoca@none.com
Validity
  Not Before: Oct 18 13:43:17 2018 GMT
  Not After : Oct 17 13:43:17 2020 GMT
Subject: C = NO, ST = AGDER, L = GRIMSTAD, O = MQTT SERVER, OU = MQTT, CN =
192.168.0.103, emailAddress = mosquitto@none.com
Subject Public Key Info:
  Public Key Algorithm: rsaEncryption
  Public-Key: (4096 bit)
  Modulus:
    00:ac:3f:fa:03:55:2b:9c:37:a1:76:64:95:d4:f1:
    :
    :
    Skipped many lines here
    :
    :
    f3:f3:f4:87:11:a9:ed:cf:05:9f:1b:95:6a:95:9a:
    13:bf:ad
  Exponent: 65537 (0x10001)
Signature Algorithm: sha256WithRSAEncryption
  4d:3e:ae:b0:73:e2:dd:64:5b:02:33:f9:33:a1:98:b4:63:a1:
  :
  :
  Skipped many lines here
  :
  :
  b7:26:44:d8:62:3b:39:1e
```

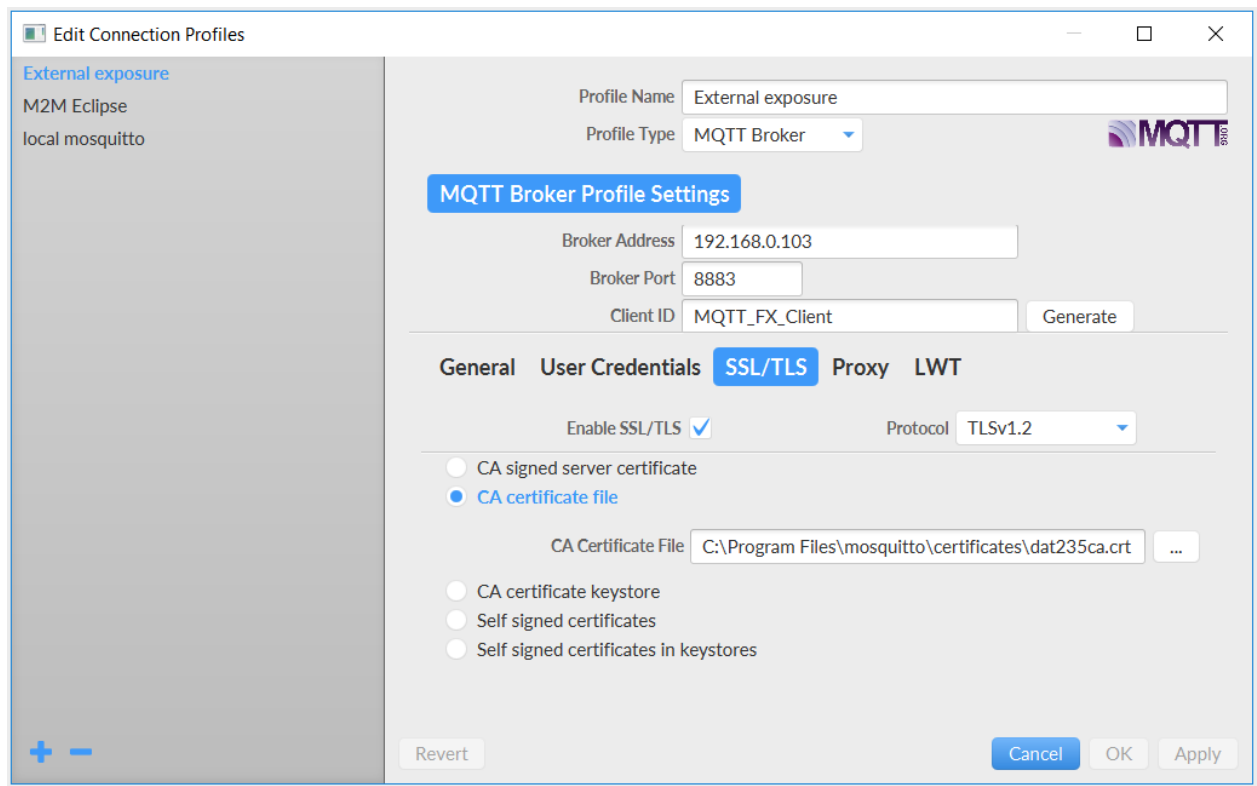
Now, you have the CA certificate and keyfile:

- dat235ca.key & dat235ca.crt
- server.crt & server.key (and server.csr, but you don't need it anymore)

4 Trying to connect to the secure MQTT server with MQTT.fx

Let's try to set up the MQTT.fx utility to use the secure broker.

- Set the port
- Enter **SSL/TLS**:
 - Enable SSL/TLS & choose TLSv1.2
 - Choose CA certificate file
- The User Credentials settings should be cleared now



Connect and watch the show.

Output from the broker:

```
C:\..\mosquitto> mosquitto -c mosquitto.conf -v
1539873366: mosquitto version 1.5.3 starting
1539873366: Config loaded from mosquitto.conf.
1539873366: Opening ipv6 listen socket on port 8883.
1539873366: Opening ipv4 listen socket on port 8883.
1539873481: New connection from 192.168.0.103 on port 8883.
1539873481: New client connected from 192.168.0.103 as MQTT_FX_Client (c1, k60).
1539873481: No will message specified.
1539873481: Sending CONNACK to MQTT_FX_Client (0, 0)
```

5 Trying to connect to the secure MQTT server – paho

Note 2:

There seems to be some incompatibilities between Python 3.7 SSL support and the Python Paho library. Or, it may be attributed to the platform (Windows) support.

Whatever the case, the code presented in the Pings.py and Pongs.py programs will not work properly on Python 3.7 on Windows with paho-mqtt 1.4.0.

On Raspbian, using Python 3.5.3 and paho-mqtt 1.4.0, things work as expected.


The follow code is run on Raspbian, paho-mqtt 1.4.0 an Python 3.5.3

Note 3:

Filesystem naming and file access is specific to each os. Beware also that the “\” character used by windows as a separator is an escape character in Python. This means the you will need “\\” in Python to get this right.

You need to edit the python programs to accept your host & CA cert, etc.

The setup:

- Ensure the Mosquitto is running with mosquitto.conf set to secured connection
 - Start MQTT.fx
 - Pings_v1.py – To be started after Pongs_v1.py
 - Pongs_v1.py
- 

Run on the RPI

6 Secure MQTT server & password/acl

6.1 The Broker

Make a config file, and include all TLS settings and all pw/acl settings from Lab-14.

Store the file in **config**.

Start the broker with the new configuration.

6.2 Set up MQTT.fx

Configure MQTT.fx with:

- TLS as previously described
- “User Credentials” as in Lab-14 (User Name: pang, Password: Random_03)

Connect with new settings

6.3 Set up Pings_v2 & Pongs_v2

These are mostly ready to be run on the RPI.

You need to edit the python programs to accept your host & CA cert, etc.

Check out the programs thoroughly.

Run the programs (Pongs_v2 first).