Index

1 Introduction....2
2 Installing MQTT broker....2
3 Generate server and client certificate and key....2
4 Test with Mosquitto MQTT client....3
5 Test with MQTT.fx client....3
6 Test with Mosquitto broker and Cinterion EHS6/ELS61– TLS enabled....4
7 History....7

1 Introduction

This technical memo provides details about:

- Installing Mosquitto MQTT broker on an UBUNTU server
- Enable MQTT over SSL
- Generating the required server certificates and keys
- Generate certificates and keys for a MQTT client (Mosquitto MQTT client and MQTT.fx)

The purpose is to deploy Mosquitto MQTT broker on a Ubuntu server and test secure MQTT communication over TLS by using MQTT.fx client.

2 Installing MQTT broker

Prerequisite:

- Ubuntu 16.04 (Xenial) installed.
- Server certificate, key and CA file are available.

Install Mosquitto from terminal:

```
$sudo apt-get install mosquitto
```

Edit Mosquitto configuration file: /etc/mosquitto/mosquitto.conf
And check following options are set:

• port 8883	Port for MQTT with TLS
• protocol MQTT	MQTT or Websocket
• cafile /etc/mosquitto/certs/ca.crt	Path and name of the CA file
• certfile /etc/mosquitto/certs/server.crt	Path and name of the server certificate
• keyfile /etc/mosquitto/certs/server.key	Path and name of the server key file
• require_certificate true	Request certificate for authentication
• tls_version tlsv1.2	Use TLS 1.2
• use_identity_as_user_name true	To avoid using a user name & password.

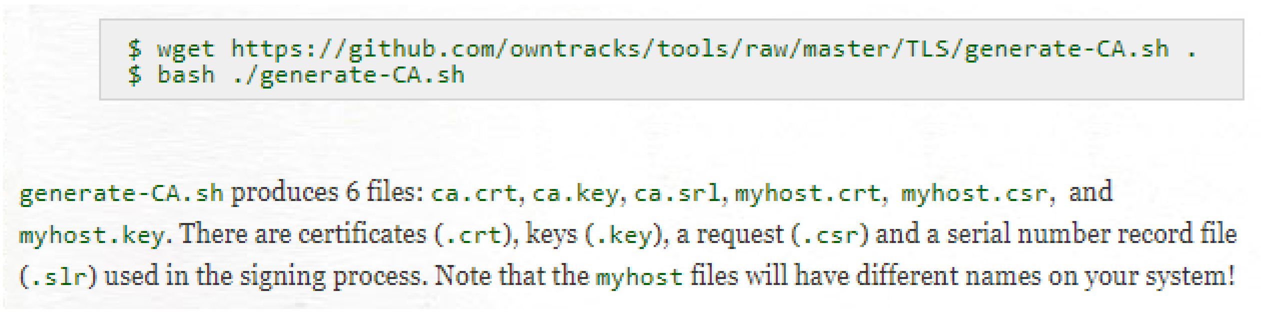
Start Mosquitto broker:

```
mosquitto -c mosquitto.conf -v
```

Port can be specified in the mosquitto.conf or by using the following parameter: '-p 1883'/'-p 8883'

3 Generate server and client certificate and key

Download and run the `generate-CA.sh` script from the OwnTracks project. The script creates the certificate authority (CA) files, generates server certificates, and uses the CA to sign the certificates.

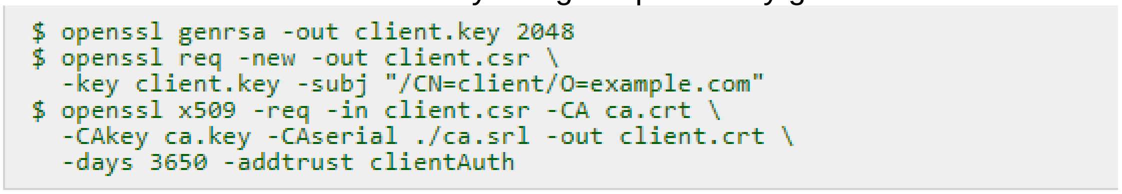


```
$ wget https://github.com/owntracks/tools/raw/master/TLS/generate-CA.sh .
$ bash ./generate-CA.sh
```

`generate-CA.sh` produces 6 files: `ca.crt`, `ca.key`, `ca.srl`, `myhost.crt`, `myhost.csr`, and `myhost.key`. There are certificates (`.crt`), keys (`.key`), a request (`.csr`) and a serial number record file (`.slr`) used in the signing process. Note that the `myhost` files will have different names on your system!

Copy ca.crt, <myhost>.crt and <myhost>.key to the location matching with the one stated in mosquitto.conf file.

Generate client certificate and key using the previously generated CA file:



```
$ openssl genrsa -out client.key 2048
$ openssl req -new -out client.csr \
  -key client.key -subj "/CN=client/O=example.com"
$ openssl x509 -req -in client.csr -CA ca.crt \
  -CAkey ca.key -CAserial ./ca.srl -out client.crt \
  -days 3650 -addtrust clientAuth
```

4 Test with Mosquitto MQTT client

Install Mosquitto MQTT client:

sudo apt-get install mosquitto-clients

Subscribe to “mqtt” topic on the local host:

- mosquitto_sub –h localhost --cafile ca.crt --cert client.crt --key client.key –p 8883 --tls-version tlsv1.2 -q 1 –t “mqtt” -v

Publish message “I am alive” on “mqtt” topic on the local host:

- mosquitto_pub –h localhost --cafile ca.crt --cert client.crt --key client.key –p 8883 --tls-version tlsv1.2 -q 1 –t “mqtt” –m “I am alive” –v

Publish zipped firmware on “mqtt” topic on the local host:

- mosquitto_pub –h localhost --cafile ca.crt --cert client.crt --key client.key –p 8883 --tls-version tlsv1.2 -q 1 –t “mqtt” –f firmware.zip -v

Notes:

- port 8883 indicates that MQTT over TLS is used.
- ‘-q’ is used to specify the quality of service.

5 Test with MQTT.fx client

This client requires to use PEM formatted certificates.

It is then required to convert CRT certificates to PEM certificates. Run openssl as root or use ‘sudo’:

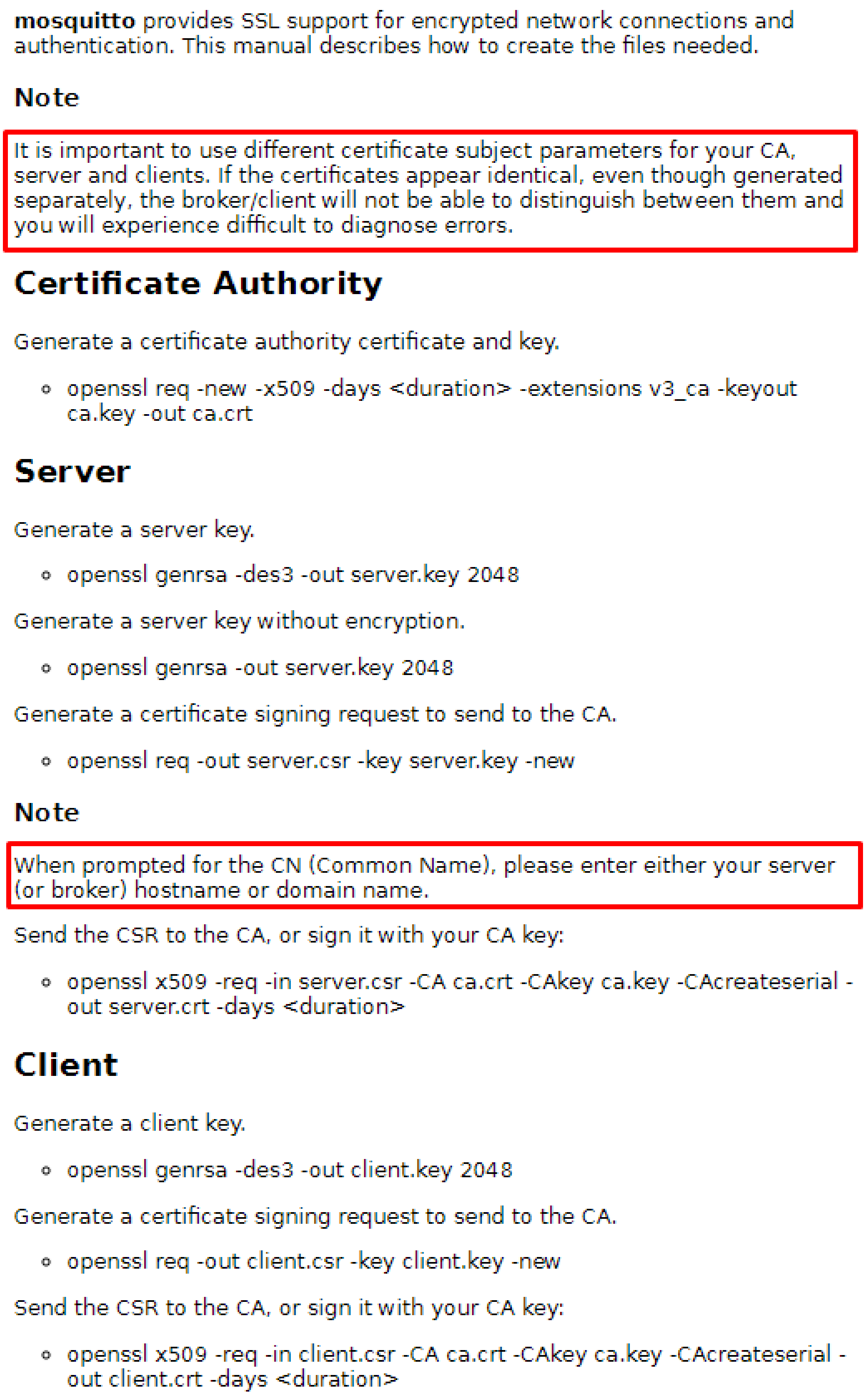
```
sudo openssl x509 –in ca.crt –out ca.der –outform DER
sudo openssl x509 –in ca.der –inform DER –out ca.pem –outform PEM
```

Proceed likewise with “client.crt” to generate client.pem certificate.

Client.key doesn’t require any change.

6 Test with Mosquitto broker and Cinterion EHS6/ELS61– TLS enabled

CA, server and client certificates must be generated strictly as described on Mosquitto.org (https://mosquitto.org/man/mosquitto-tls-7.html):



mosquitto provides SSL support for encrypted network connections and authentication. This manual describes how to create the files needed.

Note

It is important to use different certificate subject parameters for your CA, server and clients. If the certificates appear identical, even though generated separately, the broker/client will not be able to distinguish between them and you will experience difficult to diagnose errors.

Certificate Authority

Generate a certificate authority certificate and key.

- openssl req -new -x509 -days <duration> -extensions v3_ca -keyout ca.key -out ca.crt

Server

Generate a server key.

- openssl genrsa -des3 -out server.key 2048

Generate a server key without encryption.

- openssl genrsa -out server.key 2048

Generate a certificate signing request to send to the CA.

- openssl req -out server.csr -key server.key -new

Note

When prompted for the CN (Common Name), please enter either your server (or broker) hostname or domain name.

Send the CSR to the CA, or sign it with your CA key:

- openssl x509 -req -in server.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out server.crt -days <duration>

Client

Generate a client key.

- openssl genrsa -des3 -out client.key 2048

Generate a certificate signing request to send to the CA.

- openssl req -out client.csr -key client.key -new

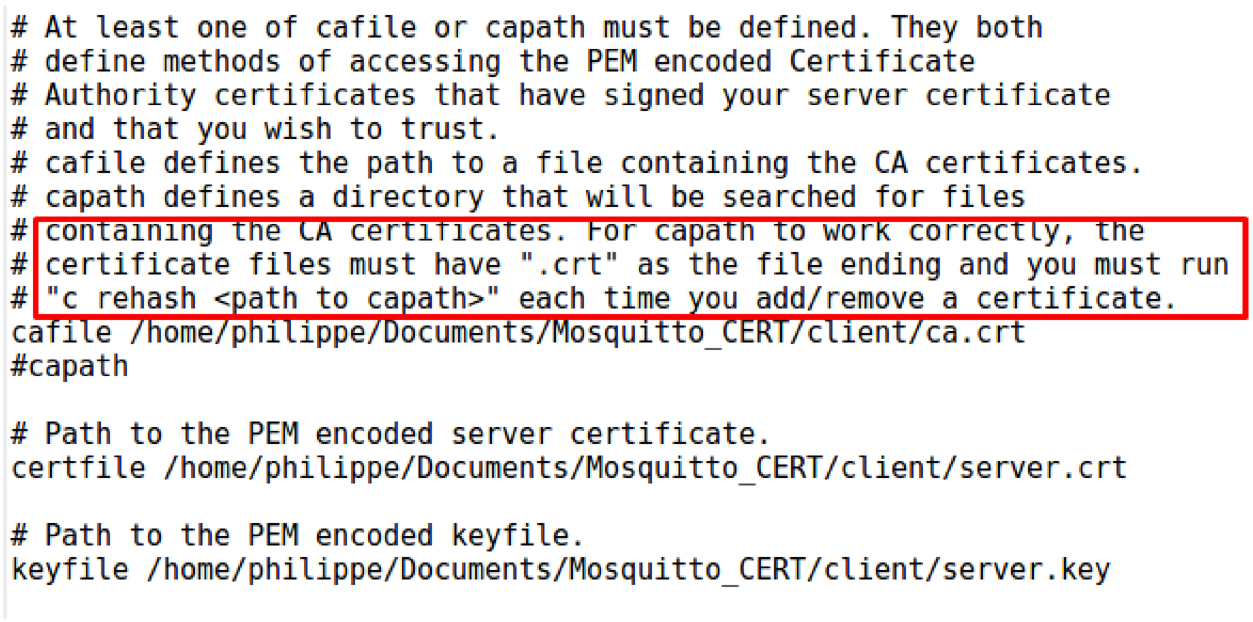
Send the CSR to the CA, or sign it with your CA key:

- openssl x509 -req -in client.csr -CA ca.crt -CAkey ca.key -CAcreateserial -out client.crt -days <duration>

It is essential to comply with the recommendations highlighted in red.

Keys encryption is not mandatory (‘des3’ option) and the process is smoother if not applied.

Then the Mosquitto configuration file should be updated to correctly point to the CA and server certificates, and the server’s key:



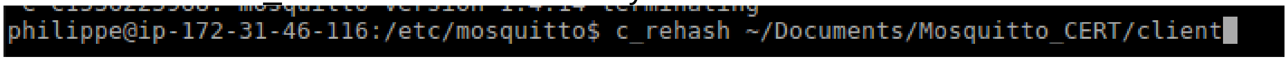
```
# At least one of cafile or capath must be defined. They both
# define methods of accessing the PEM encoded Certificate
# Authority certificates that have signed your server certificate
# and that you wish to trust.
# cafile defines the path to a file containing the CA certificates.
# capath defines a directory that will be searched for files
# containing the CA certificates. For capath to work correctly, the
# certificate files must have ".crt" as the file ending and you must run
# "c rehash <path to capath>" each time you add/remove a certificate.
cafile /home/philippe/Documents/Mosquitto_CERT/client/ca.crt
#capath

# Path to the PEM encoded server certificate.
certfile /home/philippe/Documents/Mosquitto_CERT/client/server.crt

# Path to the PEM encoded keyfile.
keyfile /home/philippe/Documents/Mosquitto_CERT/client/server.key
```

Before (re)starting the Mosquitto broker, it is crucial to regenerate the CA and server certificates hash.

For this execute ‘c_rehash’ from the directory where the broker has been started:

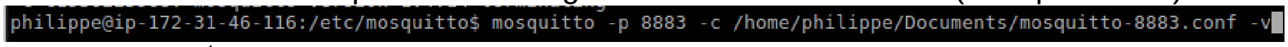


```
philippe@ip-172-31-46-116:/etc/mosquitto$ c_rehash ~/Documents/Mosquitto_CERT/client
```

And then restart Mosquitto broker.

Note: Multiple Mosquitto broker instances can be started, for instance if the user wants to have one instance running with SSL and another without.

To achieve this different ports and configuration files must be used (mosquitto.conf):



```
philippe@ip-172-31-46-116:/etc/mosquitto$ mosquitto -p 8883 -c /home/philippe/Documents/mosquitto-8883.conf -v
```

Will start a 2nd instance of the broker running on TLS and using parameters from a new configuration file (mosquitto-8883.conf).

7 History

Author	Date	Comments
Philippe Hervieu	16/01/2018	Creation.
Philippe Hervieu	05/02/2018	Update Mosquitto_pub/sub commands.
Philippe Hervieu	30/07/2018	Mosquitto_pub with a file
Philippe Hervieu	06/09/2018	Mosquitto broker with ELS61