

# Enabling and Using MQTT on Anybus CompactCom 40 IIoT Secure

## APPLICATION NOTE

SCM-1202-171 1.2 en-US ENGLISH



# 1      **About this Document**

Application note presenting how to setup and configure an Anybus CompactCom 40 IIoT Secure Ethernet module to enable MQTT, and how to subscribe to published data using the third party tools Mosquitto and MQTT.fx.

MQTT can be used both with and without TLS encryption enabled. This application note will go through the steps needed to generate and install certificates to be able to run TLS encryption. If TLS encryption is not needed, the sections about certificates can be ignored.

## 2 Prerequisites

- Download and install MQTT.fx (<http://mqttfx.jensd.de>).
- Download and install Mosquitto (<https://mosquitto.org>). Guides on how to install Mosquitto on different platforms are available on the internet.
- To generate certificates, download the Anybus Certificate Generator from <http://www.anybus.com> or OpenSSL from <http://www.openssl.org>.
- An Anybus CompactCom 40 series IIoT Secure module supporting MQTT.
- Possibility to modify and update the host application.

### 2.1 Updating the Host Application

To enable the MQTT client on the Anybus CompactCom 40 module, the MQTT host object (E2h) must be implemented in the host application. Attribute #1, MQTT Mode, must be set to the value 1. Other attributes and services are optional to implement to customize the MQTT client's behavior.

The definition of the MQTT host object is available in the IIoT Network Guide of each product (see the support pages for Anybus CompactCom 40 EtherNet/IP™ IIoT Secure and Anybus CompactCom 40 PROFINET IIoT Secure available at [www.anybus.com/support](http://www.anybus.com/support)).

If using the Anybus Host Application Example Code available in the Anybus CompactCom Starterkit, the MQTT host object and the attribute MQTT Mode are enabled and configured in `abcc_adapt\abcc_obj_cfg.h`. Support for the MQTT host object is included from version 3.05 of the Anybus Host Application Example Code.



Implementing attribute #1, MQTT Mode of the MQTT host object is required to enable MQTT on Anybus CompactCom 40 devices.

The host application must support the Application object command `Get_Data_Notification` to be able to publish data on MQTT. The Anybus CompactCom 40 will yield publication resources to the Host Application by sending this request. The Host Application then responds to the request once it has data to publish.

In the Anybus Host Application Example Code, the `Get_Data_Notification` command is enabled by defining `APP_CMD_GET_DATA_NOTIFICATION_ENABLE` to `TRUE`.



Implementing Application Object command `Get_Data_Notification` is required in order to publish MQTT data on Anybus CompactCom 40 devices.

MQTT topic strings will be automatically generated by default. For published ADI datasets, the default topic string is "Parameter/<ADI name>". A topic example for an ADI named "SPEED" (as in the default ADI mapping example in the Host Application Example Code) would be "Parameter/SPEED". See the IIoT Network guide for more information regarding topic strings.

## 2.2 Certificates

If TLS encryption is required, there are two scenarios depending on the needed security level:

**Example 1** – the Client authenticates the server before the connection is established.

- The Client (CompactCom) will need a CA certificate (used to generate the server certificate) installed.
- The Broker (Mosquitto in this example) will need the CA certificate, a server certificate, and a server private key file.

**Example 2** – both the Client and the server authenticates each other before a connection is established.

- The Client (CompactCom) will need a CA certificate (used to generate the server certificate) installed. It also needs a client certificate (generated with the same CA certificate) and a client private key.
- The Broker (Mosquitto in this example) will need the CA certificate, a server certificate, and a server private key file.



*A client certificate identifies the client and a server certificate identifies the server.*

---



*For mutual authentication, the client and server must use the same CA for the client and server certificates.*

---

### 2.2.1 Example 1 (client authenticates the server)

1. Generate a CA-certificate either with OpenSSL or with the Anybus Certificate Generator.
2. Generate a server certificate and a server private key based on the CA-certificate.



*The Common Name (CN) of the server certificate must match the IP address/URL of the broker. It is the same value that is configured in Instance #50 in the Network Configuration object (04h).*

---

3. Install the CA-certificate in the CompactCom.
4. Copy the CA certificate and the server certificate including the private key to a folder, e.g. /mosquitto/certs/.
5. Update the mosquitto configuration file (mosquitto.conf) with the certificate information.

```
cafile C:\Program Files\mosquitto\certs\CA Certificate in PEM format.crt
certfile C:\Program Files\mosquitto\certs\Certificate in PEM format.crt
keyfile C:\Program Files\mosquitto\certs\Private key for certificate in
PEM format.key
require_certificate false
```

### 2.2.2 Example 2 (client and server mutual authentication)

1. Create a CA-certificate either with OpenSSL or with the Anybus Certificate Generator.
2. Create a server certificate and a server private key based on the CA-certificate.



*The Common Name (CN) of the server certificate must match the IP address/URL of the broker. It is the same value that is configured in Instance #50 in the Network Configuration object (04h).*

---

3. Create a client certificate and a client private key based on the CA-certificate.
4. Install the CA-certificate and the client certificate including the private key in the CompactCom.
5. Copy the CA-certificate and the server certificate including the private key to a folder, e.g. /mosquitto/certs/.
6. Update the mosquitto configuration file (mosquitto.conf) with the certificate information.

```
cafile C:\Program Files\mosquitto\certs\CA Certificate in PEM format.crt
certfile C:\Program Files\mosquitto\certs\Certificate in PEM format.crt
keyfile C:\Program Files\mosquitto\certs\Private key for certificate in
PEM format.key
require_certificate true
```

### 3 Configuration

Configure MQTT either via the web page or via the Network Configuration Object (04h), Instances #50-#57. After changes are made, a restart is required for the new values to be valid.

To run with TLS encryption, set the TLS instance (#57) to 'Enable'.

An easy check to verify that MQTT is enabled in the host application when the module has started is to enter the configuration or the status web page of the module. Both the configuration and status web page shall have an MQTT section present. From the configuration web page it is possible to configure connection and publish options of MQTT. Once a broker URL has been configured and the CompactCom module has been restarted, it is possible to find the connection status in the MQTT section of the status web page. If the CompactCom module fails to resolve the configured URL or fails to authenticate the connection, this will be communicated through that page.

The screenshot shows the 'Anybus CompactCom' web interface. The top header includes the Anybus logo and the text 'Anybus CompactCom'. Below the header, there is a navigation menu on the left with options: MODULE, Overview, Parameters, NETWORK, Status, Configuration, SERVICES, SMTP, SHICP, SECURITY, Certificates, and Accounts. The main content area is divided into several sections:

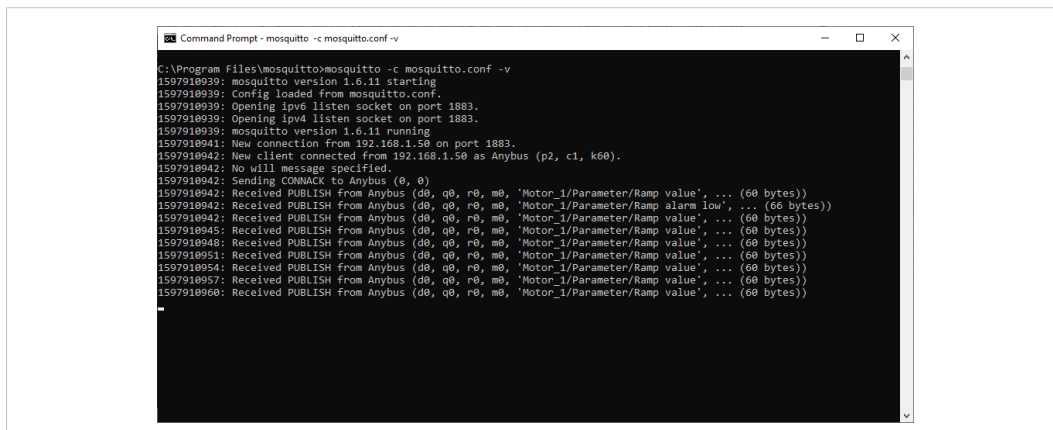
- IP Configuration:** Includes fields for DHCP (set to Disabled), IP Address (192.168.1.50), Subnet Mask (255.255.255.0), Gateway Address (0.0.0.0), Host Name, Domain name, DNS Server #1 (0.0.0.0), and DNS Server #2 (0.0.0.0). A 'Save settings' button is present.
- Ethernet Configuration:** Includes fields for Port 1 and Port 2, both set to Auto. A 'Save settings' button is present.
- MQTT Configuration:** Includes fields for Broker URL (192.168.1.40:1883), TLS (set to Enable), Client Identifier (Anybus), Keep alive time (s) (60), Base topic (Motor\_1), and Quality of service (QoS 0). A 'Save settings' button is present.
- MQTT Broker Account Configuration:** Includes fields for Username and Password. A 'Save settings' button is present.

Fig. 1 Anybus CompactCom 40 MQTT web page

### 3.1 Mosquitto

Run mosquitto from a command line and specify the config file to use. A good idea is also to enable all debug printout with the `-v` switch.

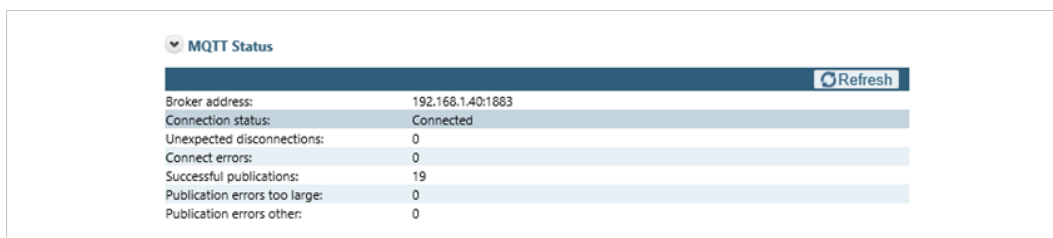
```
C:\Program Files\mosquitto\mosquitto -c mosquitto.conf -v
```



```
Command Prompt - mosquitto -c mosquitto.conf -v
C:\Program Files\mosquitto>mosquitto -c mosquitto.conf -v
1597910939: mosquitto version 1.6.11 starting
1597910939: Config loaded from mosquitto.conf.
1597910939: Opening ipv6 listen socket on port 1883.
1597910939: Opening ipv4 listen socket on port 1883.
1597910939: mosquitto version 1.6.11 running
1597910941: New connection from 192.168.1.50 on port 1883.
1597910942: New client connected from 192.168.1.50 as Anybus (p2, c1, k60).
1597910942: No will message specified.
1597910942: Sending CONNACK to Anybus (0, 0)
1597910942: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910942: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp alarm low', ... (60 bytes))
1597910942: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910945: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910948: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910951: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910954: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910957: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
1597910960: Received PUBLISH from Anybus (d0, q0, r0, m0, 'Motor_1/Parameter/Ramp value', ... (60 bytes))
```

Fig. 2 Mosquitto command line.

The CompactCom device will automatically try to connect to the specified broker. Status information for the connection is shown in the Status tab.



MQTT Status	
Broker address:	192.168.1.40:1883
Connection status:	Connected
Unexpected disconnections:	0
Connect errors:	0
Successful publications:	19
Publication errors too large:	0
Publication errors other:	0

Fig. 3 Connection status.



## 4 Publishing Data Notifications

To receive data notifications from the CompactCom 40 module the PC client MQTT.fx will be used. To connect to the same Mosquitto instance as the CompactCom 40 module, enter the Connection profiles view from the Extras menu (Extras - Edit Connection Profiles). Configure the connection details and other options needed for your Mosquitto server.

Once done, apply the configuration and exit this view.

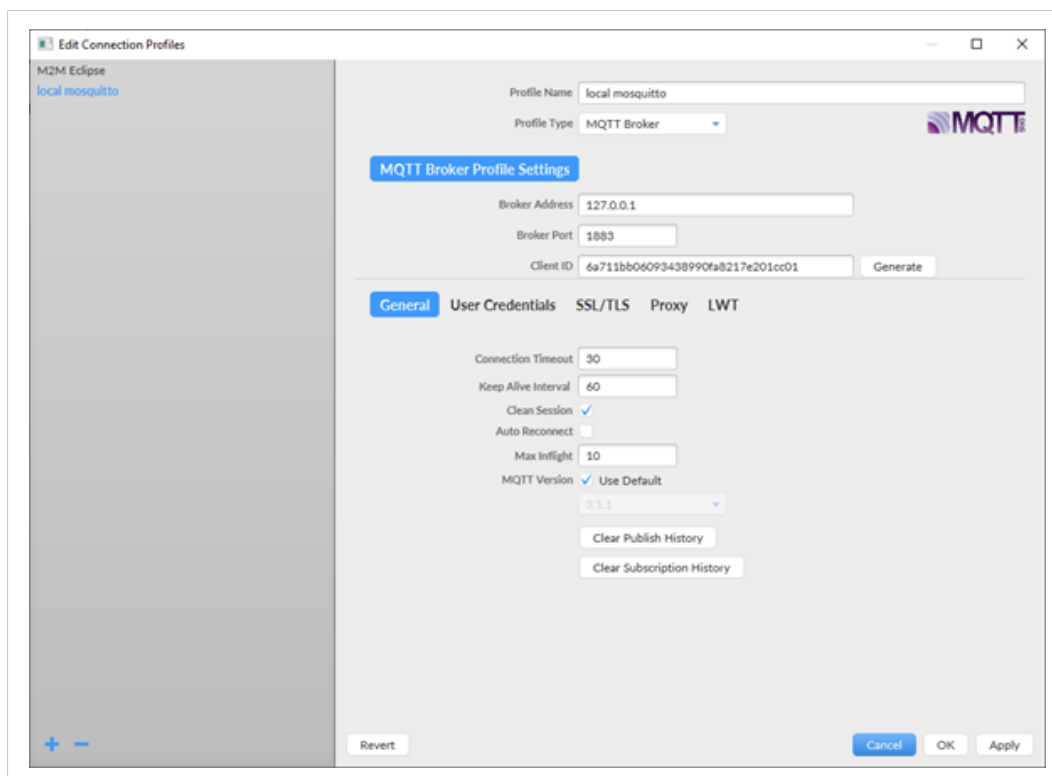
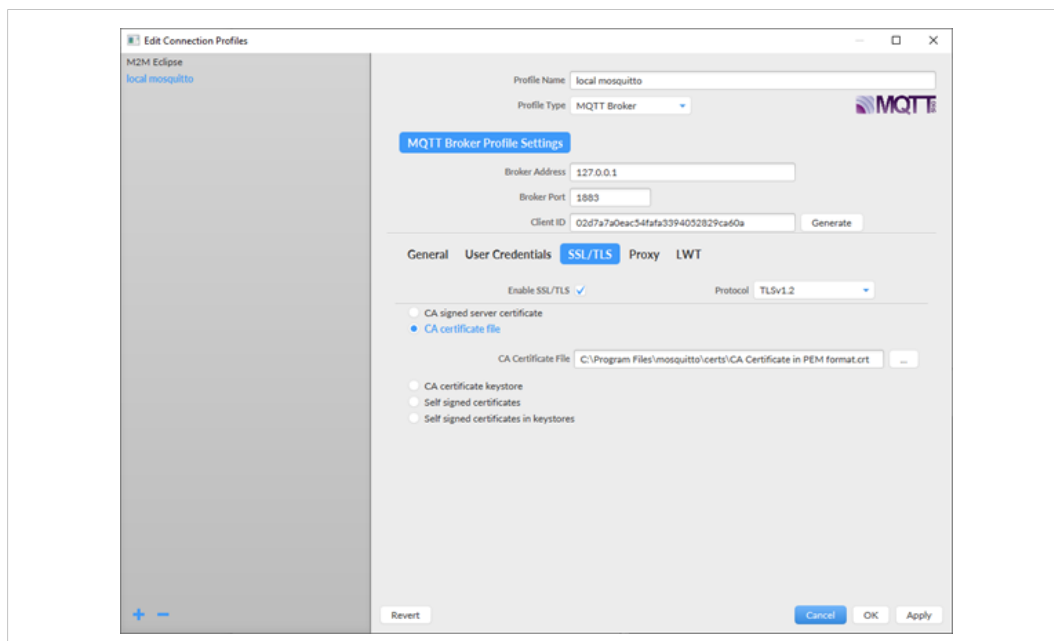


Fig. 4 MQTT.fx Connection Profiles view

**Settings for example 1:**

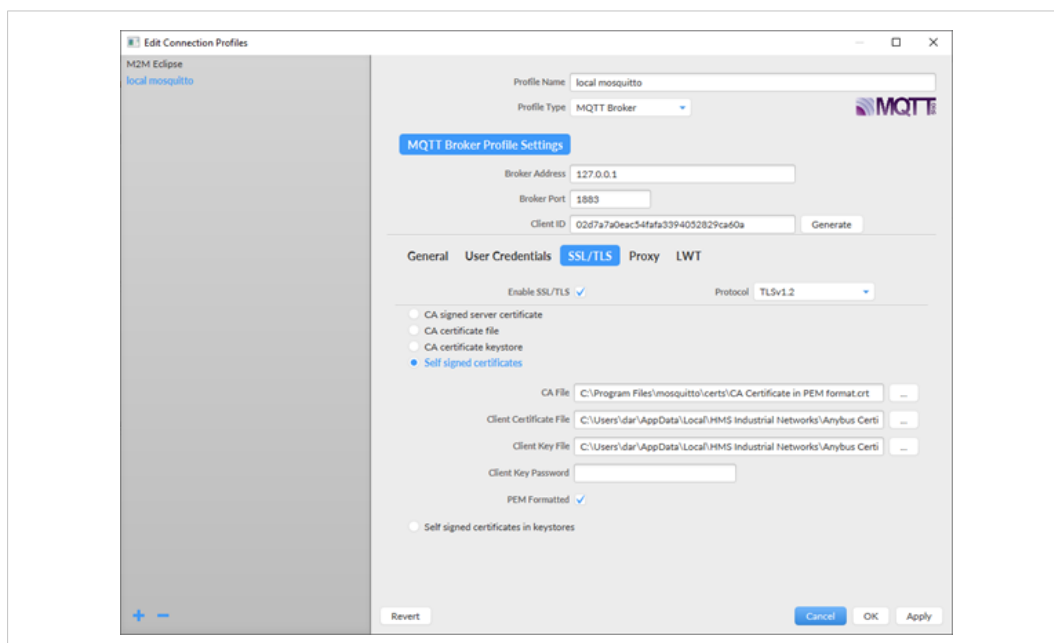
The path to the CA certificate file used when creating the server certificate must be added on the SSL/TLS tab of the broker settings in the 'CA certificate file' section.



**Fig. 5** Example 1: Only the CA-certificate file is needed to authenticate the server.

**Settings for example 2:**

When using this example, a client certificate is also needed for MQTT.fx. The path to the files is added in the 'Self signed certificates' section.



**Fig. 6** Example 2: Both the CA-certificate file and a client certificate for MQTT.fx, including the private key, is needed.

Now, select the newly created connection profile at the top of the main window and press the Connect button. Once connected, it is possible to enter the Broker status tab and subscribe for status of the Mosquitto server. As the CompactCom 40 module should already be connected, the status should tell two clients are connected. At this point, if a data notification event is triggered by the host application of the CompactCom 40 module, one should see the message and traffic counters increment.

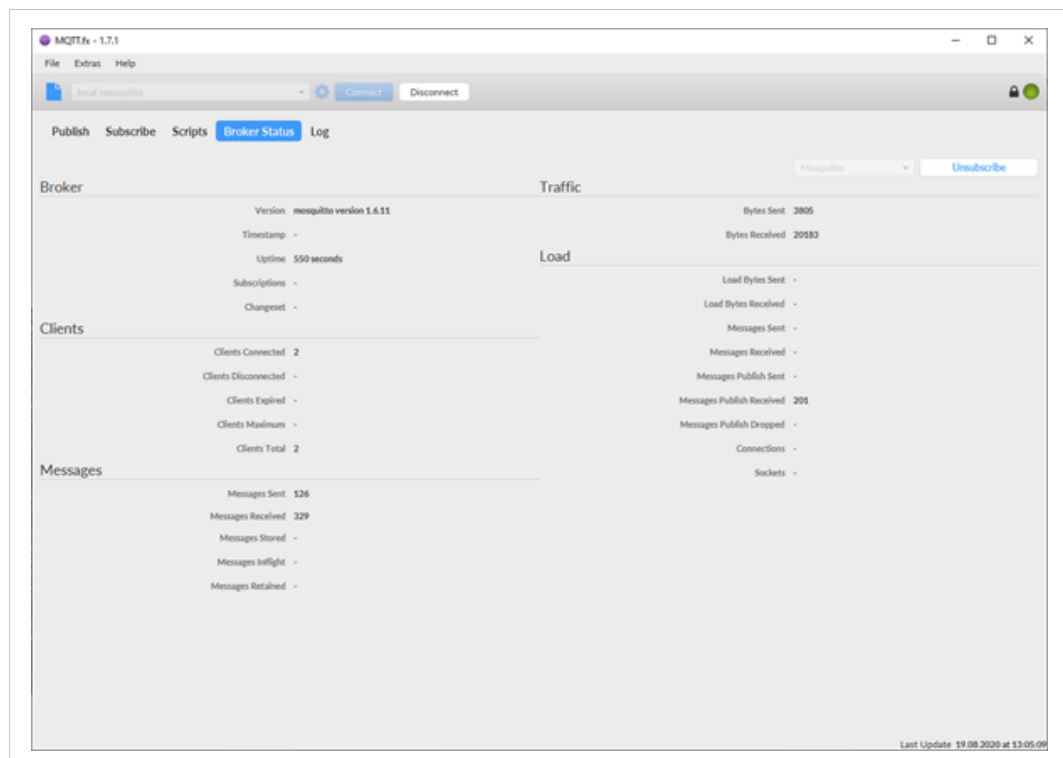


Fig. 7 MQTT.fx Broker Status tab

To receive the data notifications published by the CompactCom 40 module, a subscription must be registered to the Mosquitto server. To do this, open the Subscribe tab and enter the topic string that the CompactCom 40 module publishes data on in the text box. Then press the Subscribe button. When the application has data to publish, e.g. if data is changed or if someone pushes a button, it sends the data to the CompactCom 40 and the publish message will be forwarded by the Mosquitto server to the MQTT.fx client and show up in the console of the Subscribe tab.

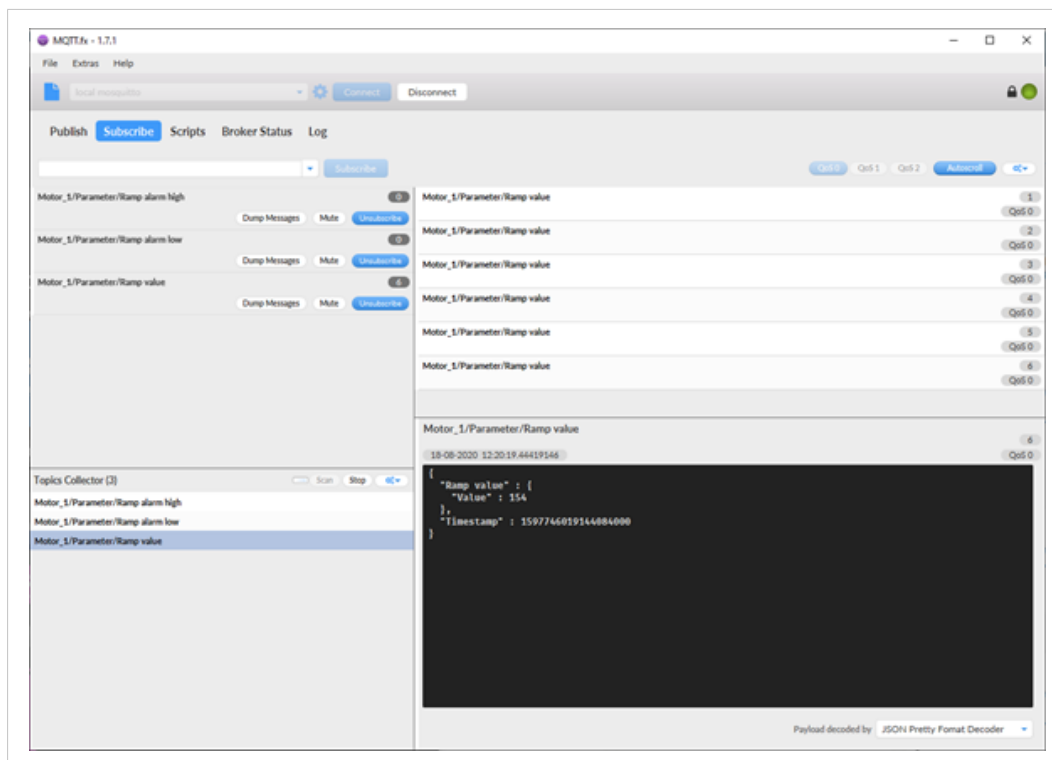


Fig. 8 Subscribing to topics with MQTT.fx

Contact HMS support for an example of how to use the `Get_Data_Notification` command with MQTT.

**This page intentionally left blank**

