Type1SC MQTT Service Application Guide

Version: 1.0
Release Date: 5/18/2023

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr _ 1129
Wednesday, August 16, 2023
Change may apply without notification.

Please be aware that an important notice concerning availability, standard warranty and use in critical applications of Murata products and disclaimers thereto appears at the end of this application guide.

Preliminary & Confidential
< Specification may be changed by Murata without notice >
Murata Manufacturing Co., Ltd.

The revision history of the application guide

Version	Release Date	Comments
1.0	5/18/2023	Initial release

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

TABLE OF CONTENTS

Confidential, Released only from my Murata
rooney.jang @ codezoo.co.kr
Wednesday, August 16, 2017 12:29
Change may apply without notification

LIST OF FIGURES

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

1. Introduction

1.1 Scope

This document describes possible host interface scenarios to use AT commands to connect the Murata Type1SC module to LTE network and use embedded MQTT application to implement networking applications.

1.2 Audience

This document is intended for software/firmware engineers to evaluate and develop applications with Murata's Type1SC LTE Cat-M1/NB-IoT module.

1.3 Contact Information and Support

Contact Murata at ciotsupport@murata.com for technical support services, technical questions, and documentation error reporting.

1.4 Text Conventions

Danger – This information MUST be followed, or catastrophic equipment failure or bodily injury may occur.

Caution/Warning
Alerts the user to important points about using the product; if these points are not followed, the product and end user equipment may fail or malfunction.

Tip/Information – Provides advice and suggestions that may be useful when using the product.

1.5 Acronyms

Acronym	Meaning
3GPP	3rd Generation Partnership Project
API	Application Programming Interface
AT	Attention
CLI	Command Line Interface
ECM	Embedded Connection Manager
eMTC	enhanced Machine-Type Communication
EPS	Evolved Packet System
FW	Firmware
GPIO	General Purpose Input/Output
GUI	Graphical User Interface
IoT	Internet of Things
IP	Internet Protocol
LPWAN	Low Power Wide Area Network
LTE	Long Term Evolution
TLS	Transport Layer Security
M2M	Machine to Machine
MSC	Message Sequence Chart
MT	Mobile Termination
NB-IoT	Narrow Band IoT
NVM	Non-Volatile Memory

Confidential Reference only, from my Murata rooney.jang@... 2023 without notification.

PC	Personal Computer
PDN	Packet Data Network
RAT	Radio Access Technology
RF	Radio Frequency
RRC	Radio Resource Control
UART	Universal Asynchronous Receiver/Transmitter
URC	Unsolicited Result Code

1.6 Related Documents

[1] Murata, "Type1SC AT Commands Reference"
[2] Murata, "Type1SC Software Application Master Guide"
[3] 3GPP TS 27.007 specification and rules: http://www.3gpp.org/ftp/Specs/archive/27_series/27.007/
[4] MQTT Version 3.1.1: http://docs.oasis-open.org/mqtt/mqtt/v3.1.1/os/mqtt-v3.1.1-os.html

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

2. MQTT Service Overview

MQTT is a client server publish/subscribe messaging transport protocol. It is light weight, open, simple, and designed to be easy to implement. These characteristics make it ideal for communication in M2M and IoT contexts where a small code footprint is required and/or network bandwidth is at a premium.

The protocol runs over TLS, TCP, and other network protocols that provide ordered, lossless, bi-directional connections. Its features include:

- Use of the publish/subscribe message pattern which provides one-to-many message distribution and decoupling of applications.
- A messaging transport that is agnostic to the content of the payload
- A small transport overhead and protocol exchanges minimized to reduce network traffic.
- A mechanism to notify interested parties when an abnormal disconnection occurs.

The supported MQTT version is MQTT Version 3.1.1. For the details of the protocol please refer the official documentation as specified at reference [3]

Embedded MQTT service simplifies the host application -- there is no need for external IP stacks, and they allow data transfer without preventing the host from issuing AT commands and receiving URCs. For the details of AT commands used in this note, please see *Type1SC AT Commands Reference* [1].

See *Type1SC Software Application Master Guide* [2] for suggested steps to setup the module and establish the LTE connection necessary to support this application protocol.

This note describes the specified behavior of this feature. The actual performance may differ due to firmware limitation. Please refer to the corresponding firmware release note for any deviation.

2.1 AT commands for MQTT Service

The following commands are used for the MQTT operation over TLS or TCP connections:

- AT%MQTTCFG – used to configure MQTT connection parameters.
- AT%MQTTCMD – used to communicate with MQTT server (broker).
- AT%MQTTEV – used to notify about MQTT events

2.2 MQTT URC Events

The following events are used to notify about MQTT URC events.

- CONCONF - Connect procedure confirmation status
- DISCONF - Graceful disconnect procedure confirmation status
- SUBCONF - Subscribe procedure confirmation status
- UNSCONF - Unsubscribe procedure confirmation status
- PUBCONF - Outgoing publication procedure confirmation status
- PUBRCV - Incoming publication message received
- CONNFAIL - Connection failure

2.3 TLS certificate commands

The following commands are used to manage the certificates for the TLS socket operation:

- AT%CERTCMD – used to read/write/delete/list user certificates to/from NV
- AT%CERTCFG – used to add/delete certificate profiles into TLS certificate profiles config file

Confidential Reference only from Murata - August 2023 - Changes may apply without notification.

3. MQTT Usage Examples

This section provides a few examples of MQTT service usage.

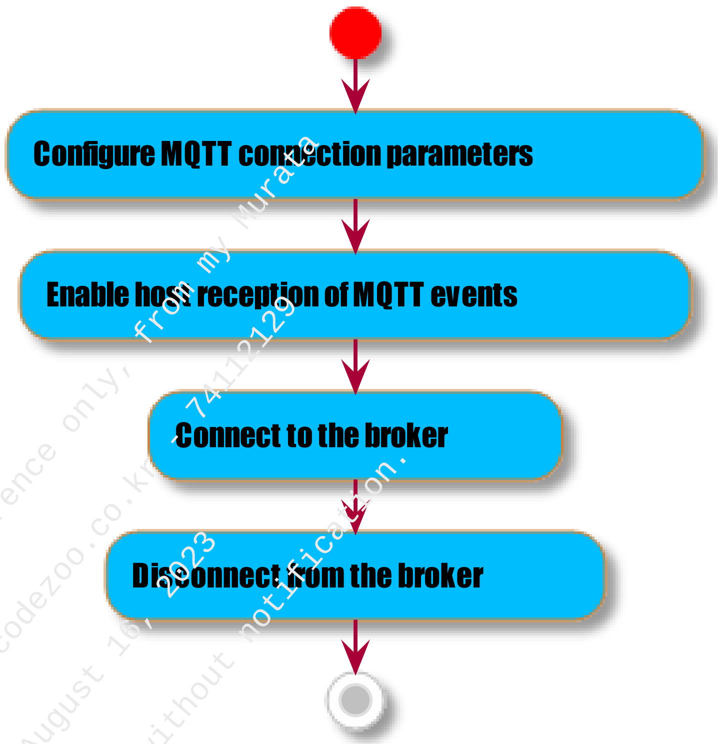
3.1 Non-Secure connection examples

This section provides a few examples of MQTT service over a non-secure connection.

3.1.1 Configure and connect to a broker

This example shows a basic MQTT scenario running over a non-secure connection. The procedure for this example is to configure and connect to a public MQTT broker. In this example, HiveMQ open-source public broker is used, and their website is https://www.hivemq.com/public-mqtt-broker/.

The following is a simple illustration of the procedure for this example:


Configure MQTT connection parameters
Enable host reception of MQTT events
Connect to the broker
Disconnect from the broker

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 7411212129
Wednesday, August 16, 2023
change may apply without notification.

Figure 1 Configure and connect to a broker

The following are the detailed steps for this example:

- Configure MQTT connection parameters

```
AT%MQTTCFG="clear",1
OK

<1> - connection id


AT%MQTTCFG="nodes",1,"ClientName","broker.hivemq.com"
OK
```

```
<ClientName> - unique client ID used to connect to the broker
<broker.hivemq.com> - broker URL or IP address

AT%MQTTCFG="IP",1,,0,1883
OK

<0> - preferred IP type for connection is IPv4v6
<1883> - MQTT broker's listening port number

AT%MQTTCFG="PROTOCOL",1,0,1200,1
OK

<0> - MQTT protocol type for connection is MQTT
<1200> - keep-alive time in seconds
<1> - clean session
```

- Enable all MQTT events

```
AT%MQTTEV="all",1
OK
```

- Connect to the broker:

```
AT%MQTTCMD="connect",1
OK
```

  Receive a connect procedure confirmation status URC

```
%MQTTEVU:"CONCONF",1,0

<0> - success
```

- Teardown connection:

```
AT%MQTTCMD="disconnect",1
OK
```

  Receive a Disconnect Procedure Confirmation Status URC as below,

```
%MQTTEVU:"DISCONF",1,0

<0> - success
```

For the detailed process description, refer to the message sequence chart diagram below:

Confidential Reference only, from my Murata
Mooney.Jung @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

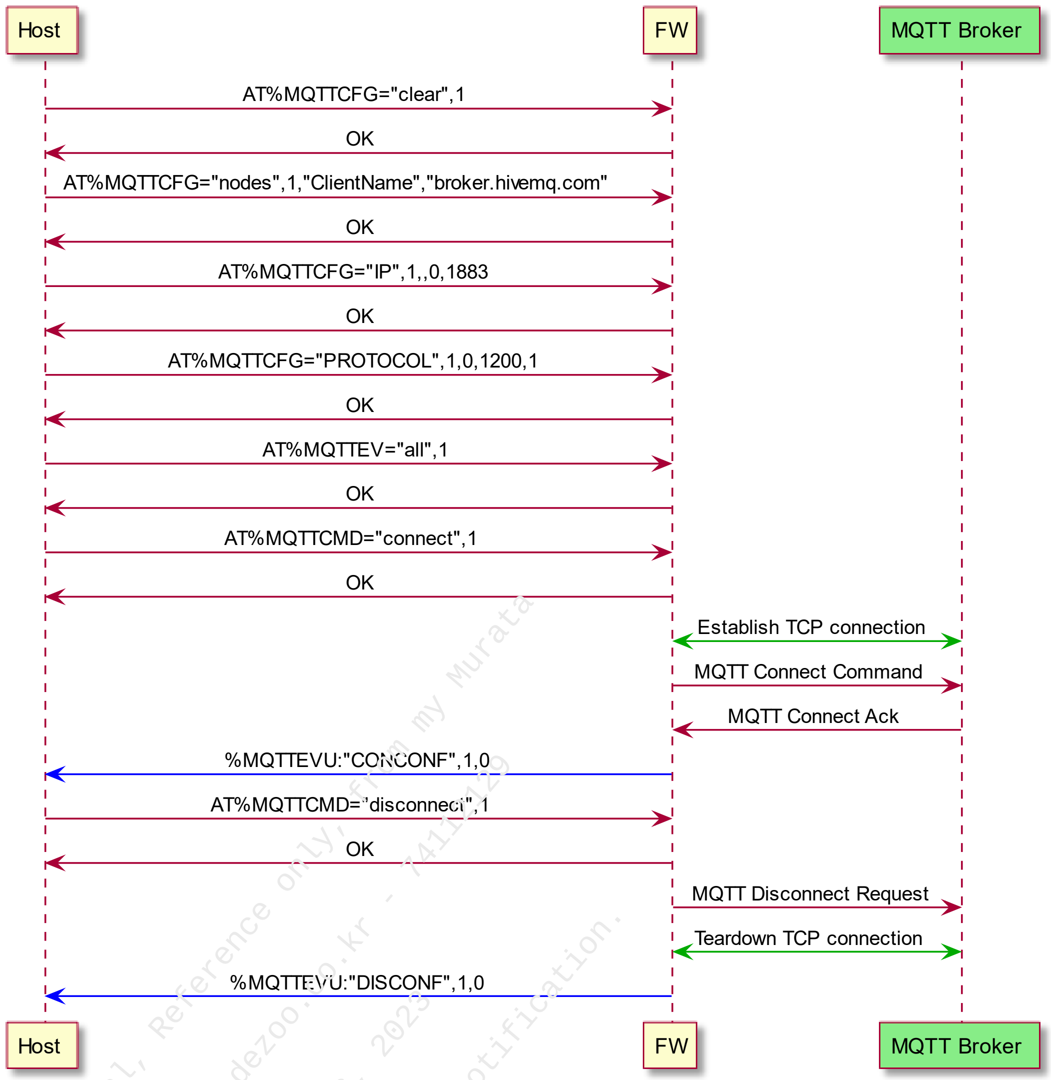


Figure 2 MSC - Configure and connect to a broker

3.1.2 Subscribe to a topic with QOS=0

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from 1SC host, then publish a message to same topic from a PC host and verify that the message is received by 1SC host.

In this example, both QoS of the subscribing and publishing are 0. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool **mosquitto_pub** is used to send a message to the broker; see https://mosquitto.org/man/mosquitto_pub-1.html for the detailed instruction.

The following is a simple illustration of the procedure for this example:

Confidential, Reference only, From my Murata
Hongyang @ codezoo.co.kr - 74137129
Wednesday, August 16, 2023
Changes may apply without notification.

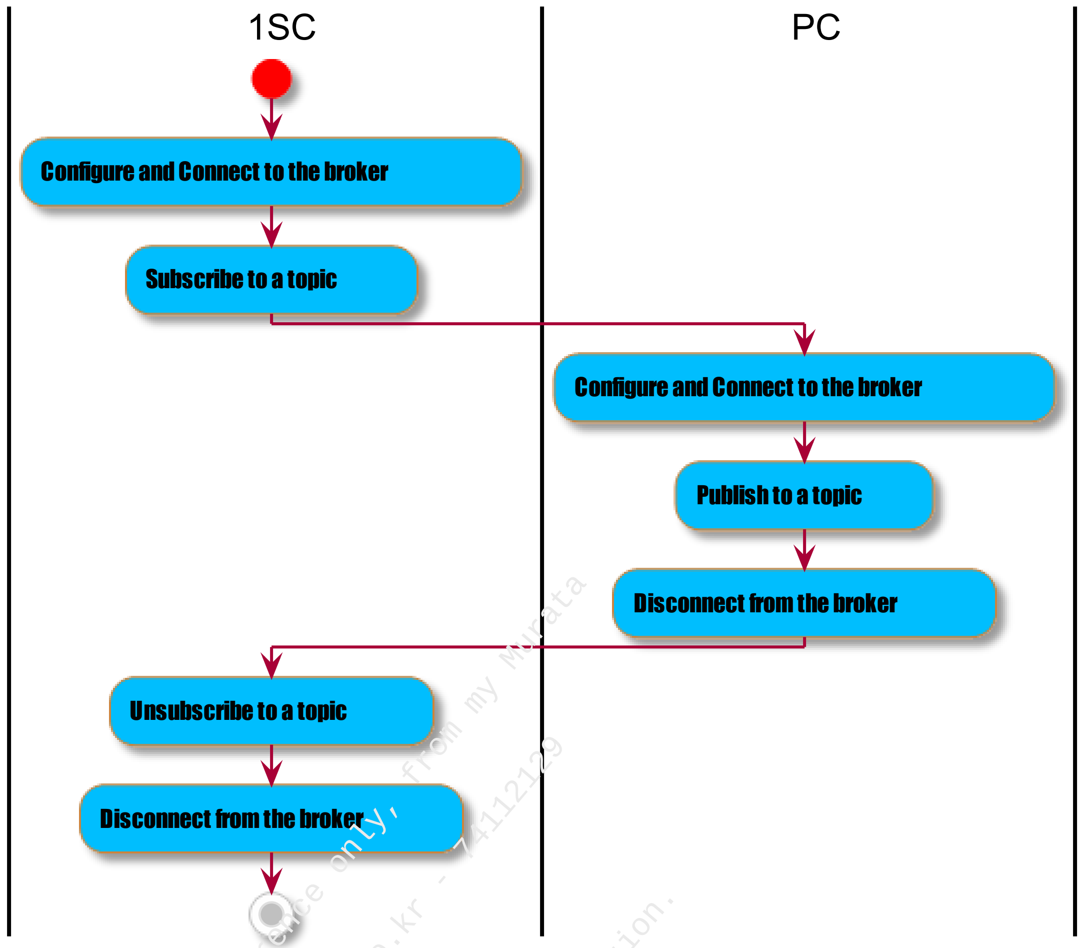

1SC
PC
Configure and Connect to the broker
Subscribe to a topic
Configure and Connect to the broker
Publish to a topic
Disconnect from the broker
Unsubscribe to a topic
Disconnect from the broker

Figure 3 Subscribe to a topic

Confidential, Reference only, from my Murata
coontal.jang @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Changes may apply without notification.

The following are the detailed steps for this example:

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

```
AT%MQTTCMD="subscribe",1,0,"TopicName"
%MQTTCMD: 1
OK
```

```
<0> - Subscription QoS level 0
```

Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0
```

```
<1> - message ID
<0> - success
```

- Publish a message to the broker from a PC

```
mosquitto_pub -d -h broker.hivemq.com -p 1883 -t TopicName -m 11111
```

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,0,"TopicName",5
11111

<0> - message ID. It may be zero (undefined) for QoS=0
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 2
OK

%MQTTEVU:"UNSCONF",1,2,0

<2> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

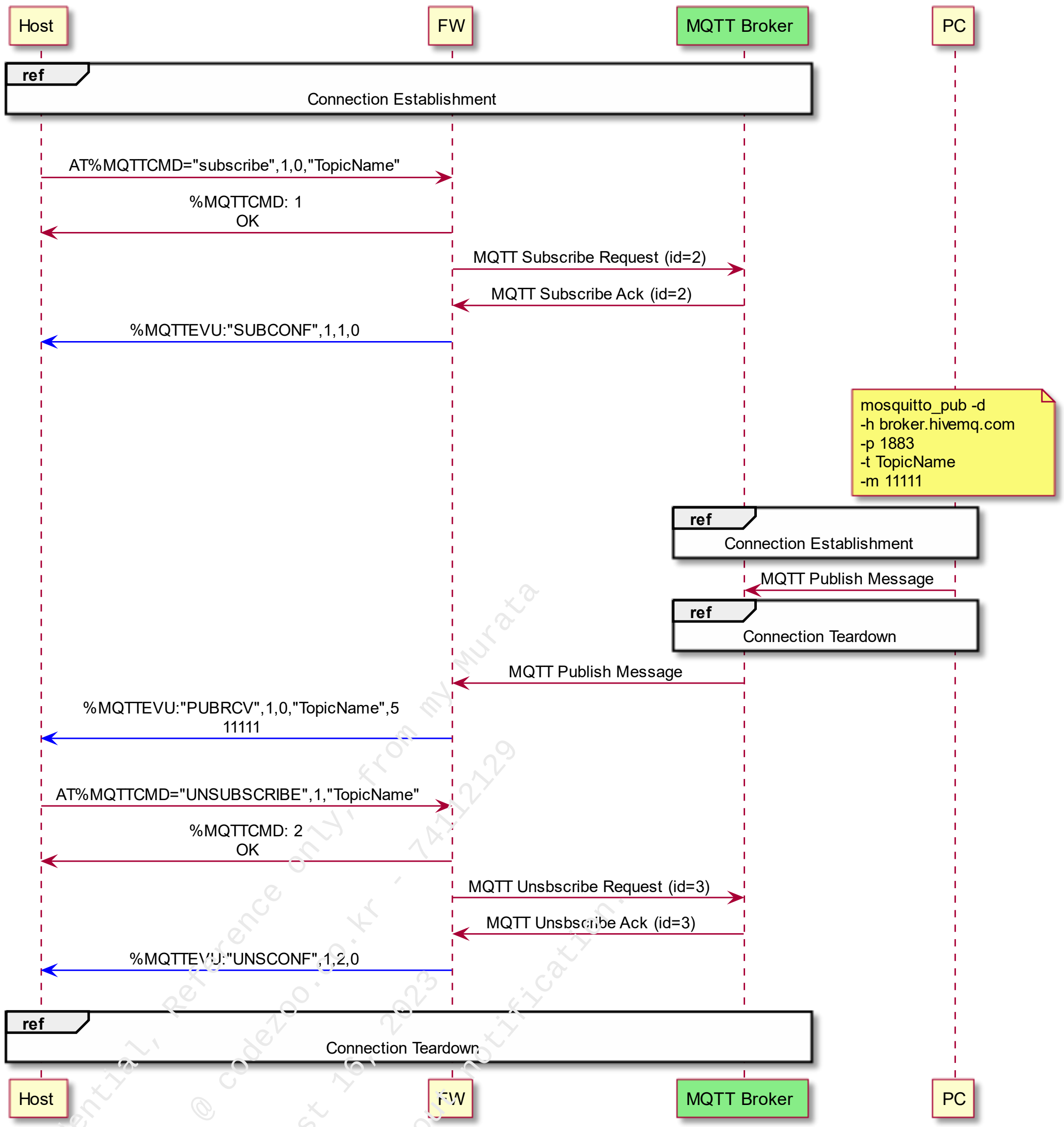


Figure 4 MSC - Subscription at QOS=0

3.1.3 Subscribe to a topic with QOS=1

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from 1SC host, then publish a message to same topic from a PC host and verify that the message is received by 1SC host.

In this example, both QoS of the subscribing and publishing are 1. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool **mosquitto_pub** is used to send a message to the broker; see https://mosquitto.org/man/mosquitto_pub-1.html for the detailed instruction.

The illustration of the procedure for this example is same as in Figure 3.

The following are the detailed steps for this example:

Confidential, Reference only, from my Murata
money.jang @ codezoo.co.kr - 7412129
August 16, 2023
Cheangman apply without notification

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

  ```
  AT%MQTTCMD="subscribe",1,1,"TopicName"
  %MQTTCMD: 1
  OK

  <1> - Subscription QoS level 1
  ```

  Receive a subscribe procedure confirmation status URC as below,

  ```
  %MQTTEVU:"SUBCONF",1,1,0

  <1> - message ID
  <0> - success
  ```

- Publish a message to the broker from a PC

  ```
  mosquitto_pub -d -h broker.hivemq.com -p 1883 -q 1 -t TopicName -m 11111
  ```

- Receive an Incoming Publication Message Received URC as below,

  ```
  %MQTTEVU:"PUBRCV",1,51,"TopicName",5
  11111

  <51> - message ID. It may be zero (undefined) for QoS=0
  ```

- Unsubscribe from the broker

  ```
  AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
  %MQTTCMD: 2
  OK

  %MQTTEVU:"UNSCONF",1,2,0

  <2> - message ID
  <0> - success
  ```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential, reference only, from Murata
Hoonjeong.jang @ codezoo.co.kr - 41129
Wednesday, August 16, 2023
Change may apply without notification.

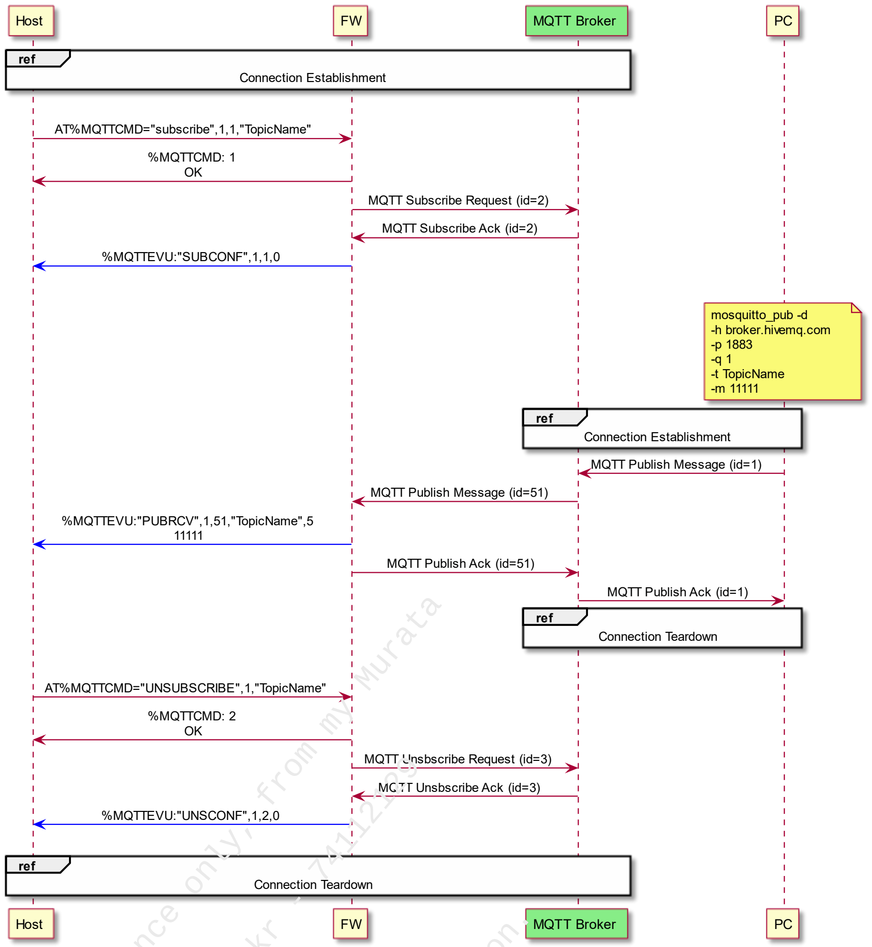


Figure 5 MSC - Subscription at QOS=1

3.1.4 Subscribe to a topic with QOS=2

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from 1SC host, then publish a message to same topic from a PC host and verify that the message is received by 1SC host.

In this example, both QoS of the subscribing and publishing are 2. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool **mosquitto_pub** is used to send a message to the broker; see https://mosquitto.org/man/mosquitto_pub-1.html for the detailed instruction.

The illustration of the procedure for this example is same as in Figure 3.

The following are the detailed steps for this example:

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

  ```
  AT%MQTTCMD="subscribe",1,2,"TopicName"
  ```

```
%MQTTCMD: 1
OK
```

```
<2> - Subscription QoS level 2
```

Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0
```

```
<1> - message ID
<0> - success
```

- Publish a message to the broker from a PC

```
mosquitto_pub -d -h broker.hivemq.com -p 1883 -q 2 -t TopicName -m 11111
```

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,51,"TopicName",5
11111
```

```
<51> - message ID. It may be zero (undefined) for QoS=0
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 2
OK
```

```
%MQTTEVU:"UNSCONF",1,2,0
```

```
<2> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential Reference only, from Murata
rooney.jang @ codezoo.co.kr - 72112129
Wednesday, August 16, 2023
Change may apply without notification.

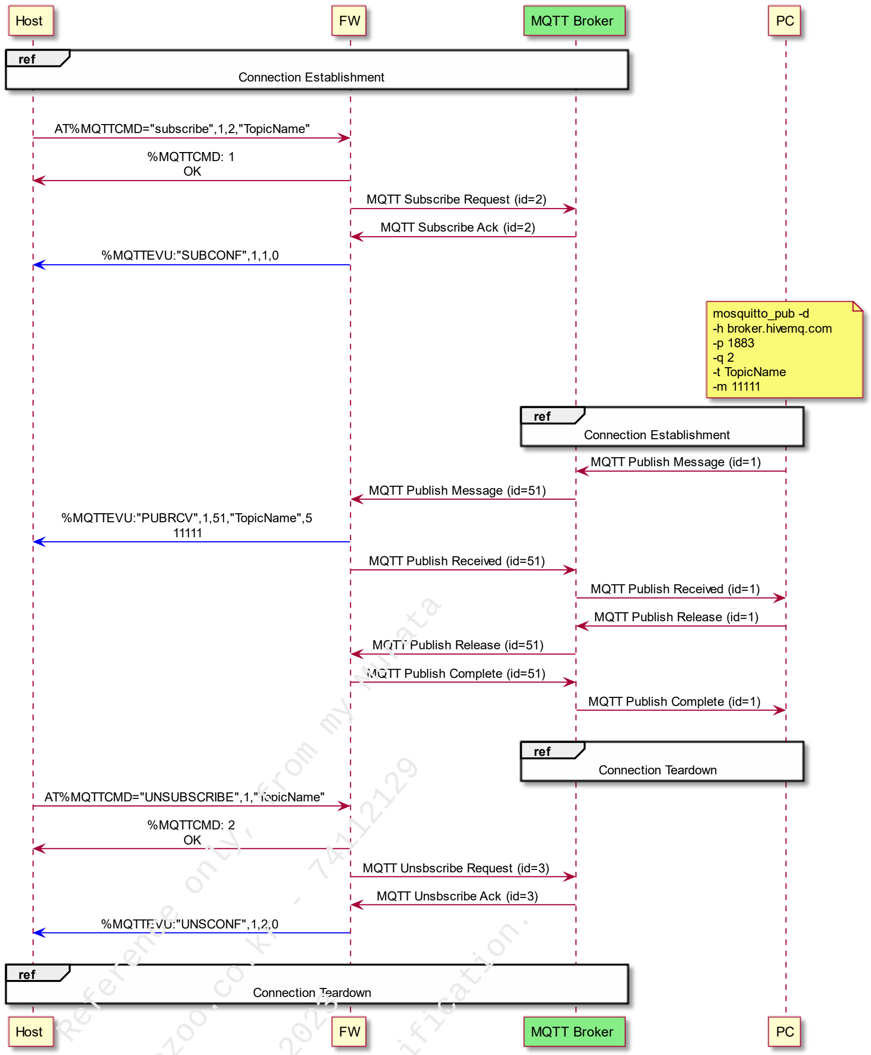

Host
FW
MQTT Broker
PC
ref
Connection Establishment
AT%MQTTCMD="subscribe",1,2,"TopicName"
%MQTTCMD: 1
OK
MQTT Subscribe Request (id=2)
MQTT Subscribe Ack (id=2)
%MQTTEVU:"SUBCONF",1,1,0
mosquitto_pub -d
-h broker.hivemq.com
-p 1883
-q 2
-t TopicName
-m 11111
ref
Connection Establishment
MQTT Publish Message (id=1)
MQTT Publish Message (id=51)
%MQTTEVU:"PUBRCV",1,51,"TopicName",5
11111
MQTT Publish Received (id=51)
MQTT Publish Received (id=1)
MQTT Publish Release (id=1)
MQTT Publish Release (id=51)
MQTT Publish Complete (id=51)
MQTT Publish Complete (id=1)
ref
Connection Teardown
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 2
OK
MQTT Unsbscribe Request (id=3)
MQTT Unsbscribe Ack (id=3)
%MQTTEVU:"UNSCONF",1,2,0
ref
Connection Teardown
Host
FW
MQTT Broker
PC

Figure 6 MSC - Subscription at QOS=2

3.1.5 Publish to a topic with QOS=0

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from a PC host, then publish a message to same topic from a 1SC host and verify that the message is received by PC host.

In this example, both QoS of the subscribing and publishing are 0. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool **mosquitto_sub** is used to subscribe to a topic; see https://mosquitto.org/man/mosquitto_sub-1.html for the detailed instruction.

The following is a simple illustration of the procedure for this example:

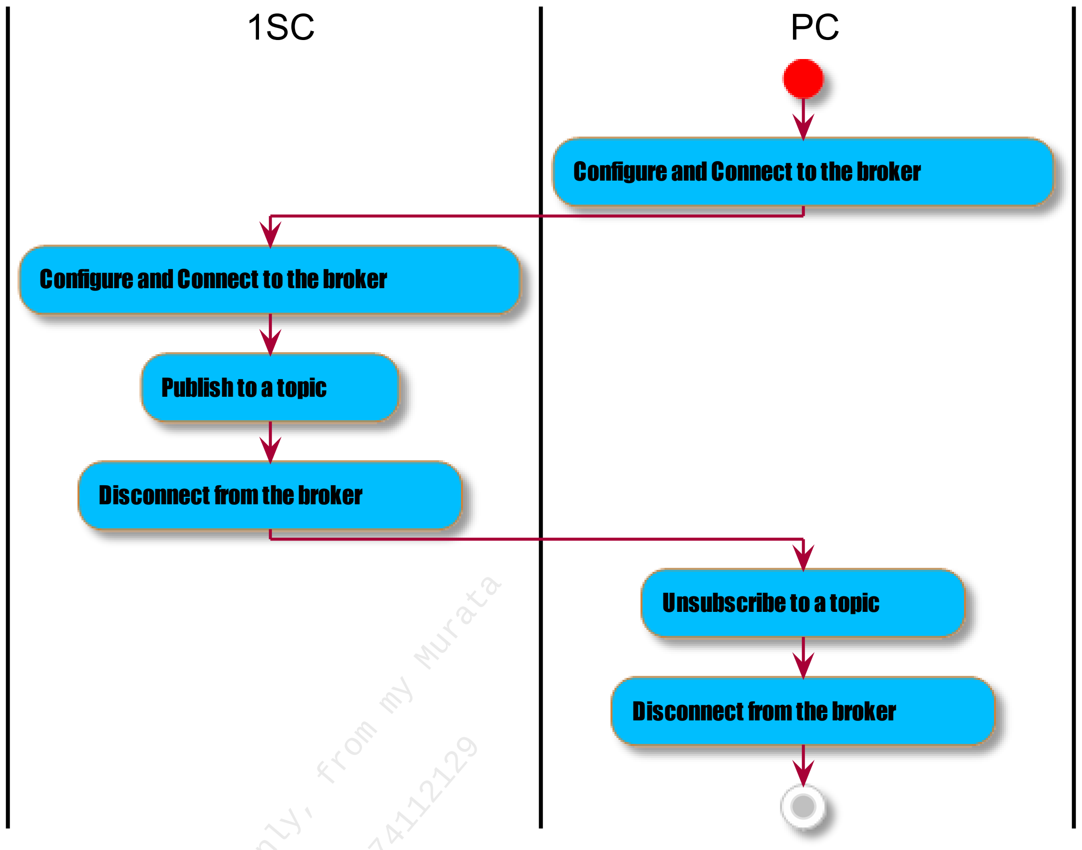

1SC
PC
Configure and Connect to the broker
Configure and Connect to the broker
Publish to a topic
Disconnect from the broker
Unsubscribe to a topic
Disconnect from the broker

Figure 7 Publish to a topic

confidential, reference only, from my Murata
jang @ codezoo.co.kr 74112129
Wednesday, August 16, 2023
Change may apply without notification

The following are the detailed steps for this example:

- Subscribe to a topic on the broker from a PC

```
mosquitto_sub -d -h broker.hivemq.com -p 1883 -t TopicName
```

- Connect to the broker

  See 3.1.1 for the details

- Publish the message to a topic on the broker

```
AT%MQTTCMD="publish",1,0,0,"TopicName",5
11111

<0> - Publication QoS level 2
<5> - Enter 5 characters to publish.
<11111> - Data payload

%MQTTCMD: 1
OK
```

```
<1> - message ID
```

```
Note: make sure there is no <CR> in the AT%MQTTCMD="publish" command
above. This multi-line command should be Linux file format (LF-based file).
One way of converting it is to copy this command to a Notepad++ editor and
remove all the <CR> at the end of each line and only leave the <LF>.
```

- Verify that the message is received by PC:

```
Client (null) received PUBLISH (d0, q0, r0, m0, 'TopicName', ... (5 bytes))
11111
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

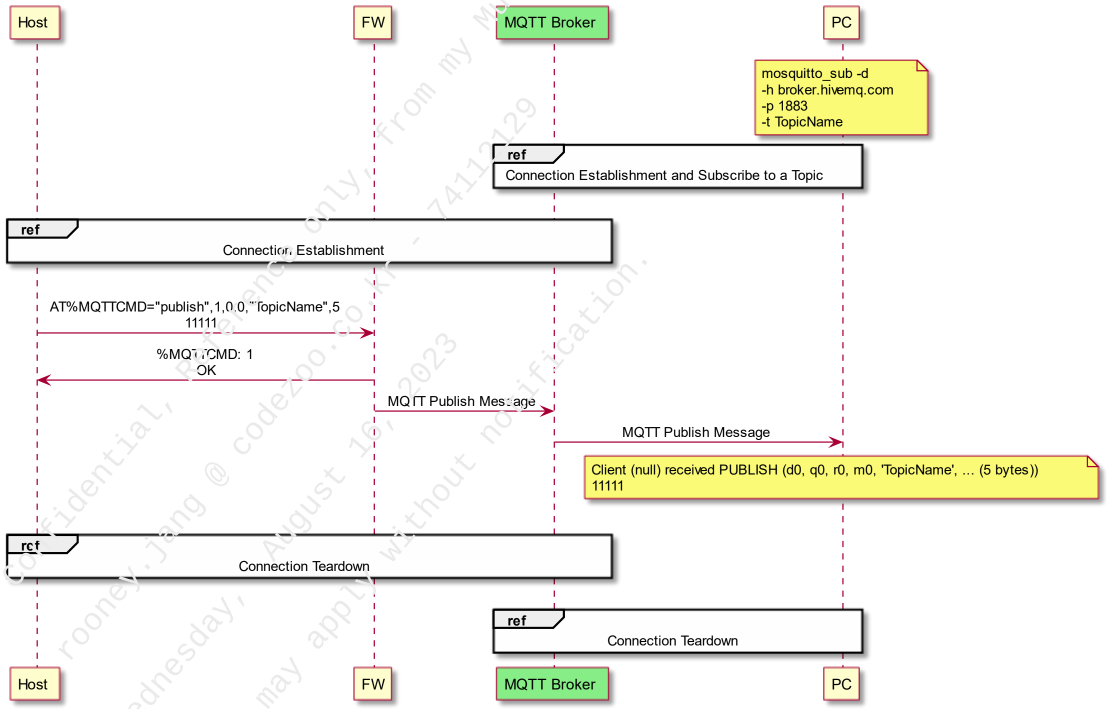


Figure 8 MSC - Publication at QOS=0

Confidential, reference only, from my Murata
rooney.jang @ code200.co.kr - 74112129
Wednesday, August 16, 2023
Change may apply without notification.

3.1.6 Publish to a topic with QOS=1

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from a PC host, then publish a message to same topic from a 1SC host and verify that the message is received by PC host.

In this example, both QoS of the subscribing and publishing are 1. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool

mosquitto_sub is used to subscribe to a topic; see https://mosquitto.org/man/mosquitto_sub-1.html for the detailed instruction.

The procedure for this example is same as in Figure 7.

The following are the detailed steps for this example:

- Subscribe to a topic on the broker from a PC

  ```
  mosquitto_sub -d -h broker.hivemq.com -p 1883 -q 1 -t TopicName
  ```

- Connect to the broker

  See 3.1.1 for the details

- Publish the message to a topic on the broker

  ```
  AT%MQTTCMD="publish",1,1,0,"TopicName",5
  11111

  <2> - Publication QoS level 2
  <5> - Enter 5 characters to publish.
  <11111> - Data payload

  %MQTTCMD: 1
  OK

  <1> - message ID
  ```

  Note: make sure there is no <CR> in the AT%MQTTCMD="publish" command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the <LF>.

- Verify that the message is received by PC:

  ```
  Client (null) received PUBLISH (d0, q1, r0, m51, 'TopicName', ... (5 bytes))
  Client (null) sending PUBACK (m51, rc0)
  1111111111
  ```

- Receive an Outgoing Publication Procedure Confirmation Status URC as below,

  ```
  %MQTTEVU:"PUBCONF",1,1,0
  OK

  <1> - message ID
  <0> - success
  ```

- Teardown connection:

  See 3.1.1 for the details

Confidential reference only, from my Murata
rooney.jeong @ [illegible].co.kr - 74112129
Wednesday, August [illegible] 2023
change may apply without notification.

For the detailed process description, refer to the message sequence chart diagram below:

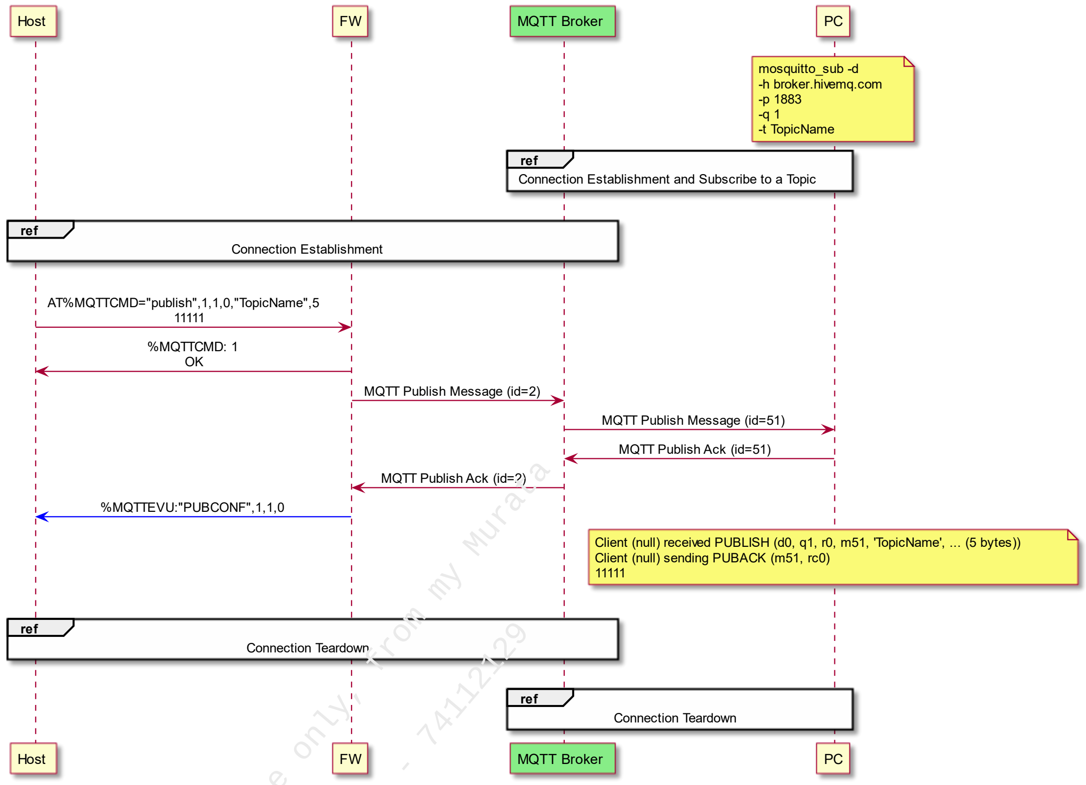


Figure 9 MSC - Publication at QOS=1

3.1.7 Publish to a topic with QOS=2

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic from a PC host, then publish a message to same topic from a 1SC host and verify that the message is received by PC host.

In this example, both QoS of the subscribing and publishing are 2. HiveMQ open-source public broker is used; see their website https://www.hivemq.com/public-mqtt-broker/ for more details. An open-source tool **mosquitto_sub** is used to subscribe to a topic; see https://mosquitto.org/man/mosquitto_sub-1.html for the detailed instruction.

The procedure for this example is same as in Figure 7.

The following are the detailed steps for this example:

- Subscribe to a topic on the broker from a PC

```
mosquitto_sub -d -h broker.hivemq.com -p 1883 -q 2 -t TopicName
```

- Connect to the broker

  See 3.1.1 for the details

- Publish the message to a topic on the broker

```
AT%MQTTCMD="publish",1,2,0,"TopicName",5
11111

<2> - Publication QoS level 2
<5> - Enter 5 characters to publish.
<11111> - Data payload

%MQTTCMD: 1
OK

<1> - message ID
```

Note: make sure there is no <CR> in the `AT%MQTTCMD="publish"` command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the <LF>.

- Verify that the message is received by PC:

```
Client (null) received PUBLISH (d0, q2, r0, m51, 'TopicName', ... (5 bytes))
Client (null) sending PUBREC (m51, rc0)
Client (null) received PUBREL (Mid: 51)
Client (null) sending PUBCOMP (m51)
11111
```

- Receive an Outgoing Publication Procedure Confirmation Status URC as below,

```
%MQTTEVU:"PUBCONF",1,1,0
OK

<1> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential, Reference only for my Murata
rodney.liang @ codezoo.co.jp - 74.119
Wednesday, August 16, 2023
Changes may apply without notification.

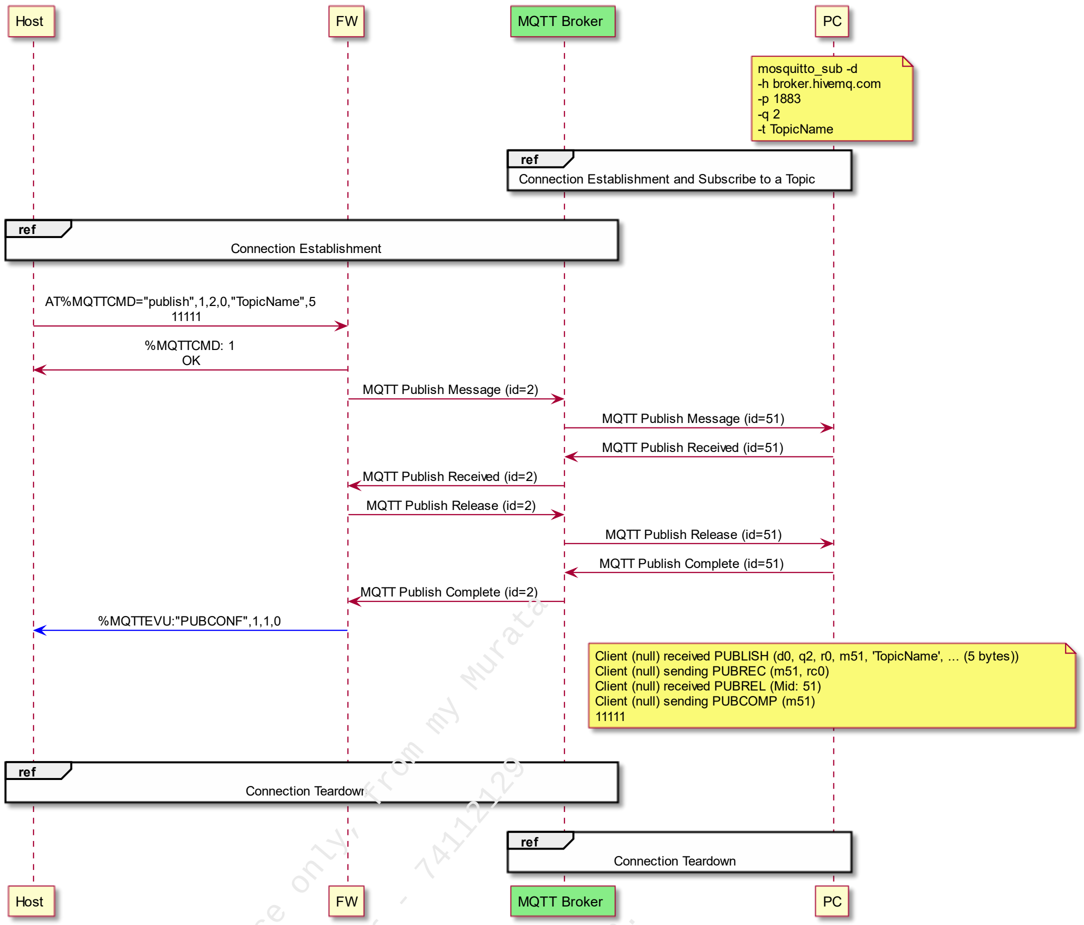

Host
FW
MQTT Broker
PC
mosquitto_sub -d
-h broker.hivemq.com
-p 1883
-q 2
-t TopicName
ref
Connection Establishment and Subscribe to a Topic
ref
Connection Establishment
AT%MQTTCMD="publish",1,2,0,"TopicName",5
11111
%MQTTCMD: 1
OK
MQTT Publish Message (id=2)
MQTT Publish Message (id=51)
MQTT Publish Received (id=51)
MQTT Publish Received (id=2)
MQTT Publish Release (id=2)
MQTT Publish Release (id=51)
MQTT Publish Complete (id=51)
MQTT Publish Complete (id=2)
%MQTTEVU:"PUBCONF",1,1,0
Client (null) received PUBLISH (d0, q2, r0, m51, 'TopicName', ... (5 bytes))
Client (null) sending PUBREC (m51, rc0)
Client (null) received PUBREL (Mid: 51)
Client (null) sending PUBCOMP (m51)
11111
ref
Connection Teardown
ref
Connection Teardown

Figure 10 MSC - Publication at QOS=2

3.1.8 Subscribe and Publish to a topic with QOS=0

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic first, then receive and publish the message to this topic after the subscription. In this example, Both QoS of subscribing and publishing are 0. In this example, HiveMQ open source public broker is used, and for more details see their website at https://www.hivemq.com/public-mqtt-broker/.

The following is a simple illustration of the procedure for this example:

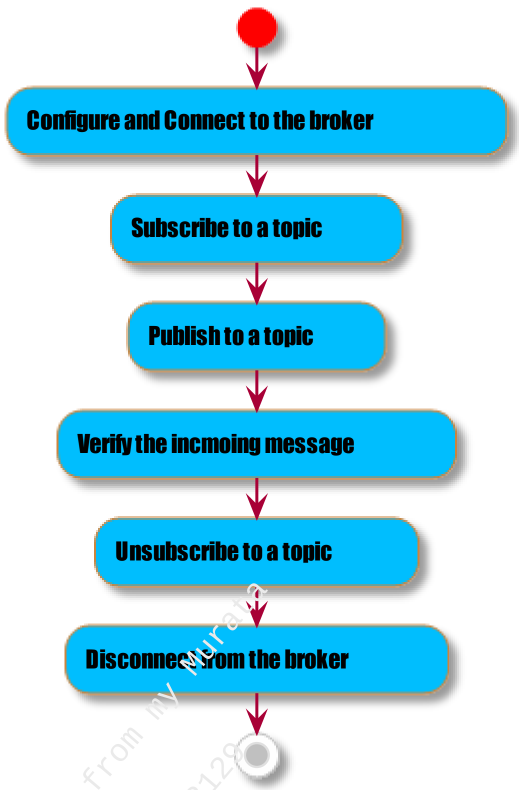

Configure and Connect to the broker
Subscribe to a topic
Publish to a topic
Verify the incmoing message
Unsubscribe to a topic
Disconnect from the broker

Figure 11 Subscribe and Publish to a topic

The following are the detailed steps for this example:

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

```
AT%MQTTCMD="subscribe",1,0,"TopicName"

<0> - QoS level is 0

%MQTTCMD: 1
OK
<1> - message ID
```

  Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0

<1> - message ID
<0> - success
```

- Publish the message to a topic on the broker

Confidential Reference only, from my Murata rodney.wang @ codezoo.co.kr - 20112128 Wednesday, August 16, 2023 Change may apply without notification.

```
AT%MQTTCMD="publish",1,0,0,"TopicName",5
11111

<0> - Publication QoS level 0
<5> - Enter 5 characters to publish.
<11111> - Data payload

%MQTTCMD: 2
OK

<2> - message ID
```

Note: make sure there is no <CR> in the `AT%MQTTCMD="publish"` command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the <LF>.

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,0,"TopicName",5
11111

<0> - message ID. It may be zero (undefined) for QoS=0
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 3
OK

%MQTTEVU:"UNSCONF",1,3,0

<3> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential, Reference only, from my murata
romey.yang @ codezoo.co.kr - 74412129
Wednesday, August 16, 2023
change may apply without notification.

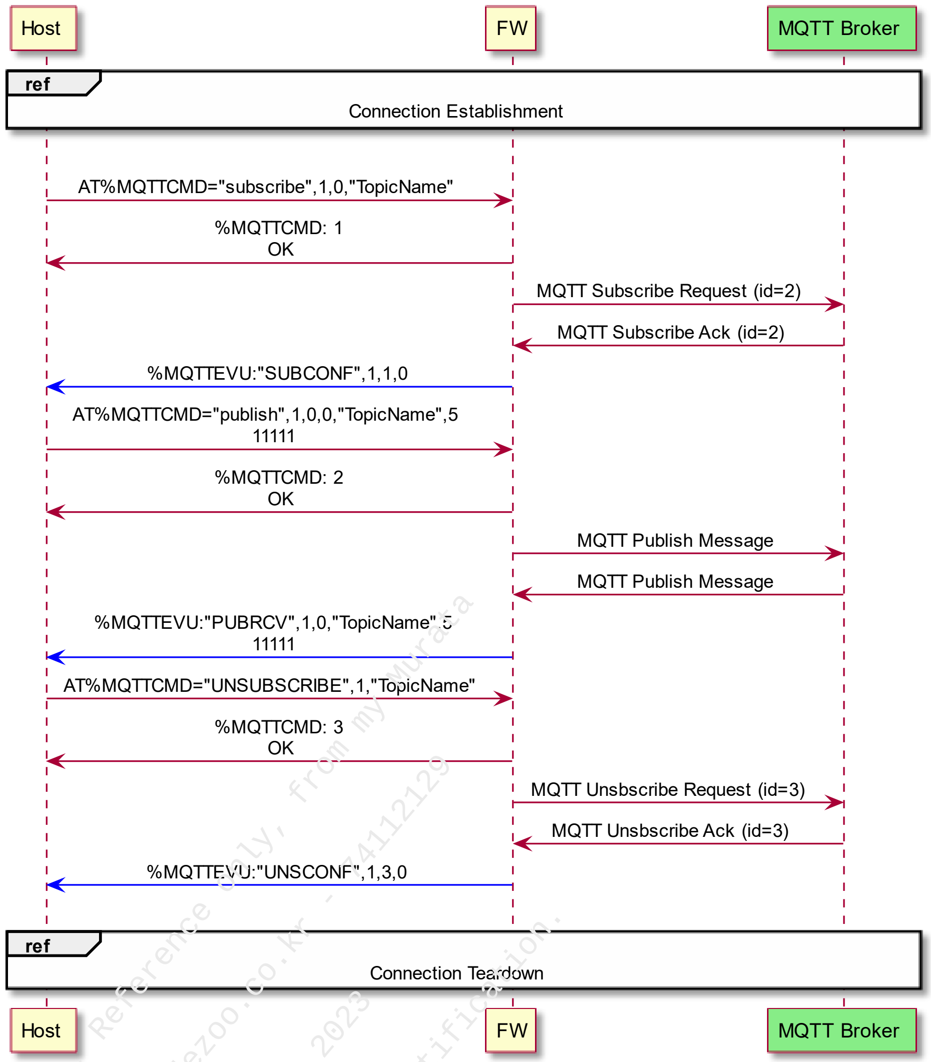

Host
FW
MQTT Broker
ref
Connection Establishment
AT%MQTTCMD="subscribe",1,0,"TopicName"
%MQTTCMD: 1
OK
MQTT Subscribe Request (id=2)
MQTT Subscribe Ack (id=2)
%MQTTEVU:"SUBCONF",1,1,0
AT%MQTTCMD="publish",1,0,0,"TopicName",5
11111
%MQTTCMD: 2
OK
MQTT Publish Message
MQTT Publish Message
%MQTTEVU:"PUBRCV",1,0,"TopicName",5
11111
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 3
OK
MQTT Unsbscribe Request (id=3)
MQTT Unsbscribe Ack (id=3)
%MQTTEVU:"UNSCONF",1,3,0
ref
Connection Teardown
Host
FW
MQTT Broker

Figure 12 MSC – Subscribe and Publish with QOS=0

3.1.9 Subscribe and Publish to a topic with QOS=1

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic first, then receive and publish the message to this topic after the subscription. In this example, both QoS of the subscribing and publishing are 1. In this example, HiveMQ open source public broker is used, and for more details see their website at https://www.hivemq.com/public-mqtt-broker/.

The procedure for this example is same as in Figure 11.

The following are the detailed steps for this example:

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

```
AT%MQTTCMD="subscribe",1,1,"TopicName"
```

```
%MQTTCMD: 1
OK

<1> - Subscription QoS level 1
```

Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0

<1> - message ID
<0> - success
```

- Publish the message to a topic on the broker

```
AT%MQTTCMD="publish",1,1,0,"TopicName",5
11111

<1> - Publication QoS level 1
<5> - Enter 5 characters to publish.
<11111> - Data payload

%MQTTCMD:2
OK

<2> - message ID
```

Note: make sure there is no <CR> in the `AT%MQTTCMD="publish"` command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the <LF>.

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,51,"TopicName",5
11111

<51> - message ID. It may be zero (undefined) for QoS=0
```

- Receive an Outgoing Publication Procedure Confirmation Status URC as below,

```
%MQTTEVU:"PUBCONF",1,2,0
OK

<2> - message ID
<0> - success
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 3
OK
```

Confidential, Reference only, from my Murata
roomy.jiang@codezoo.com 74112129
Wednesday, August 16, 2023
Changes may apply without notification

```
%MQTTEVU:"UNSCONF",1,3,0

<3> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

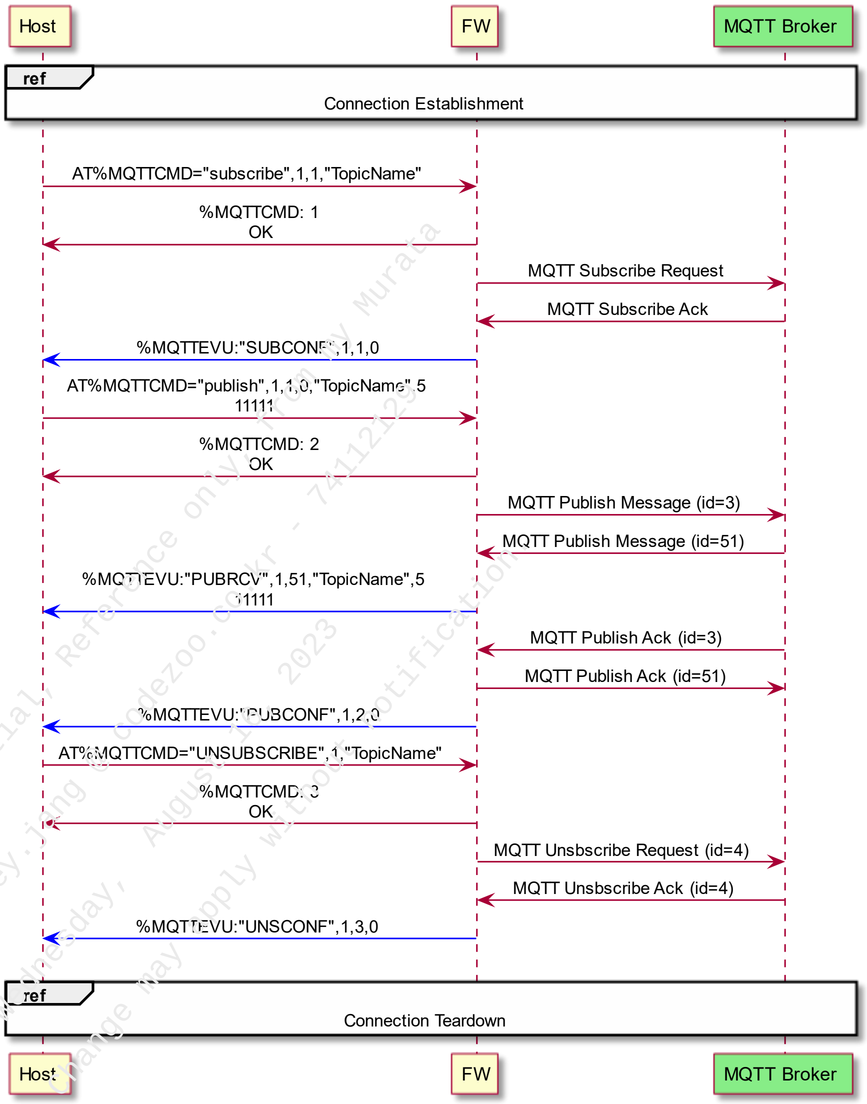


Figure 13 MSC – Subscribe and Publish with QOS=1

3.1.10 Subscribe and Publish to a topic with QOS=2

This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is to subscribe to a topic first, then receive and publish the message to this topic after the subscription. In this example, both QoS of the subscribing and publishing are 2. In this example, HiveMQ open source public broker is used, and for more details see their website at https://www.hivemq.com/public-mqtt-broker/.

The procedure for this example is same as in Figure 11.

The following are the detailed steps for this example:

- Connect to the broker

  See 3.1.1 for the details

- Subscribe to a topic on the broker

```
AT%MQTTCMD="subscribe",1,2,"TopicName"
%MQTTCMD: 1
OK

<2> - Subscription QoS level 2
```

  Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0

<1> - message ID
<0> - success
```

- Publish the message to a topic on the broker

```
AT%MQTTCMD="publish",1,2,0,"TopicName",5
11111

<2> - Publication QoS level 2
<5> - Enter 5 characters to publish.
<11111> - Data payload

%MQTTCMD:2
OK

<2> - message ID
```

Note: make sure there is no <CR> in the `AT%MQTTCMD="publish"` command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the <LF>.

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,51,"TopicName",5
11111

<51> - message ID. It may be zero (undefined) for QoS=
```

- Receive an Outgoing Publication Procedure Confirmation Status URC as below,

Confidential reference only from my Murata
Yooney.jang @ codezoo.co.kr 741512129
Changed Wednesday, August 16, 2023
May apply without notification.

```
%MQTTEVU:"PUBCONF",1,2,0
OK

<2> - message ID
<0> - success
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 3
OK

%MQTTEVU:"UNSCONF",1,3,0

<3> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

For the detailed process description, refer to the message sequence chart diagram below:

Confidential, Reference only, from my Murata
rooney.jang @ codezoo.co.kr - 74112123
Wednesday, August 16, 2023
Change may apply without notification.

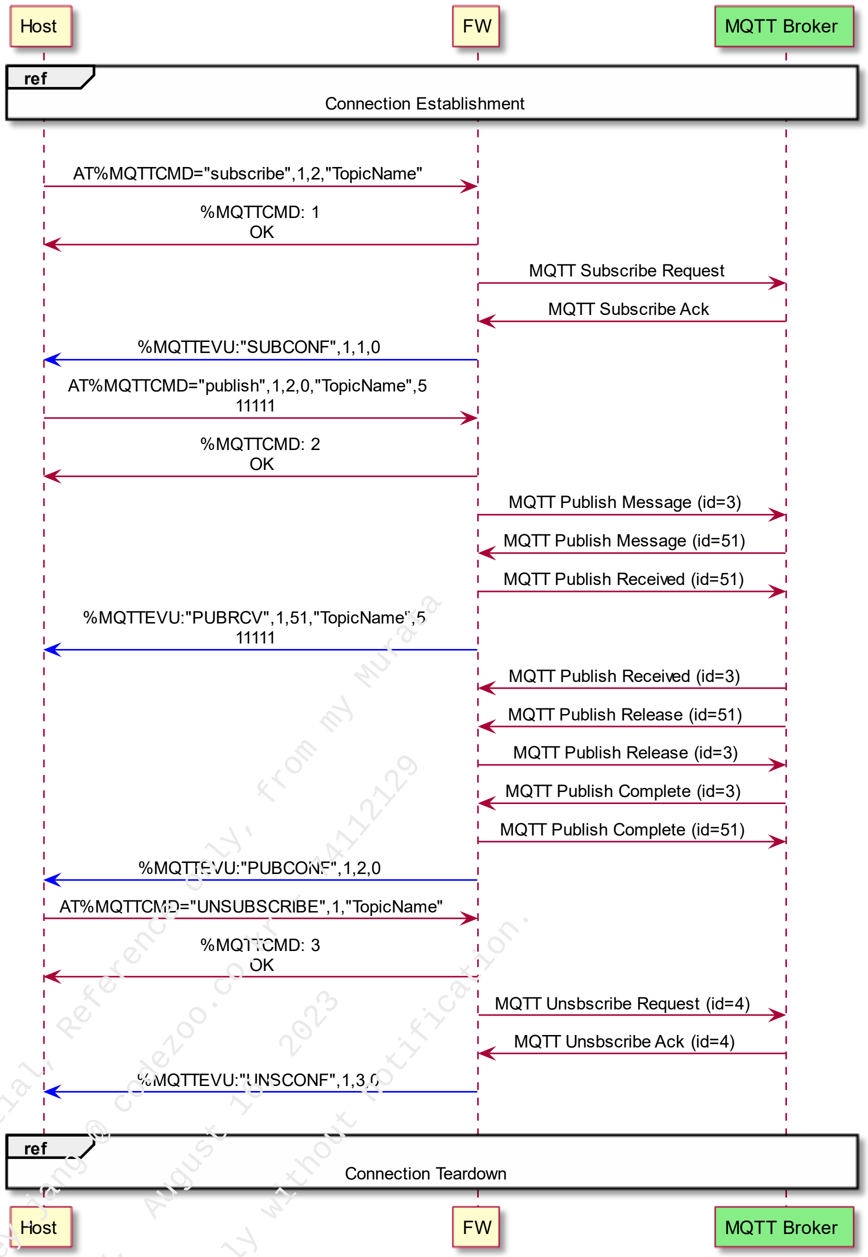

Host
FW
MQTT Broker
ref
Connection Establishment
AT%MQTTCMD="subscribe",1,2,"TopicName"
%MQTTCMD: 1
OK
MQTT Subscribe Request
MQTT Subscribe Ack
%MQTTEVU:"SUBCONF",1,1,0
AT%MQTTCMD="publish",1,2,0,"TopicName",5
11111
%MQTTCMD: 2
OK
MQTT Publish Message (id=3)
MQTT Publish Message (id=51)
MQTT Publish Received (id=51)
%MQTTEVU:"PUBRCV",1,51,"TopicName",5
11111
MQTT Publish Received (id=3)
MQTT Publish Release (id=51)
MQTT Publish Release (id=3)
MQTT Publish Complete (id=3)
MQTT Publish Complete (id=51)
%MQTTEVU:"PUBCONF",1,2,0
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 3
OK
MQTT Unsbscribe Request (id=4)
MQTT Unsbscribe Ack (id=4)
%MQTTEVU:"UNSCONF",1,3,0
ref
Connection Teardown
Host
FW
MQTT Broker

Figure 14 MSC – Subscribe and Publish with QOS=2

3.1.11 Subscribe and Publish to a topic to/from a file with QOS=0

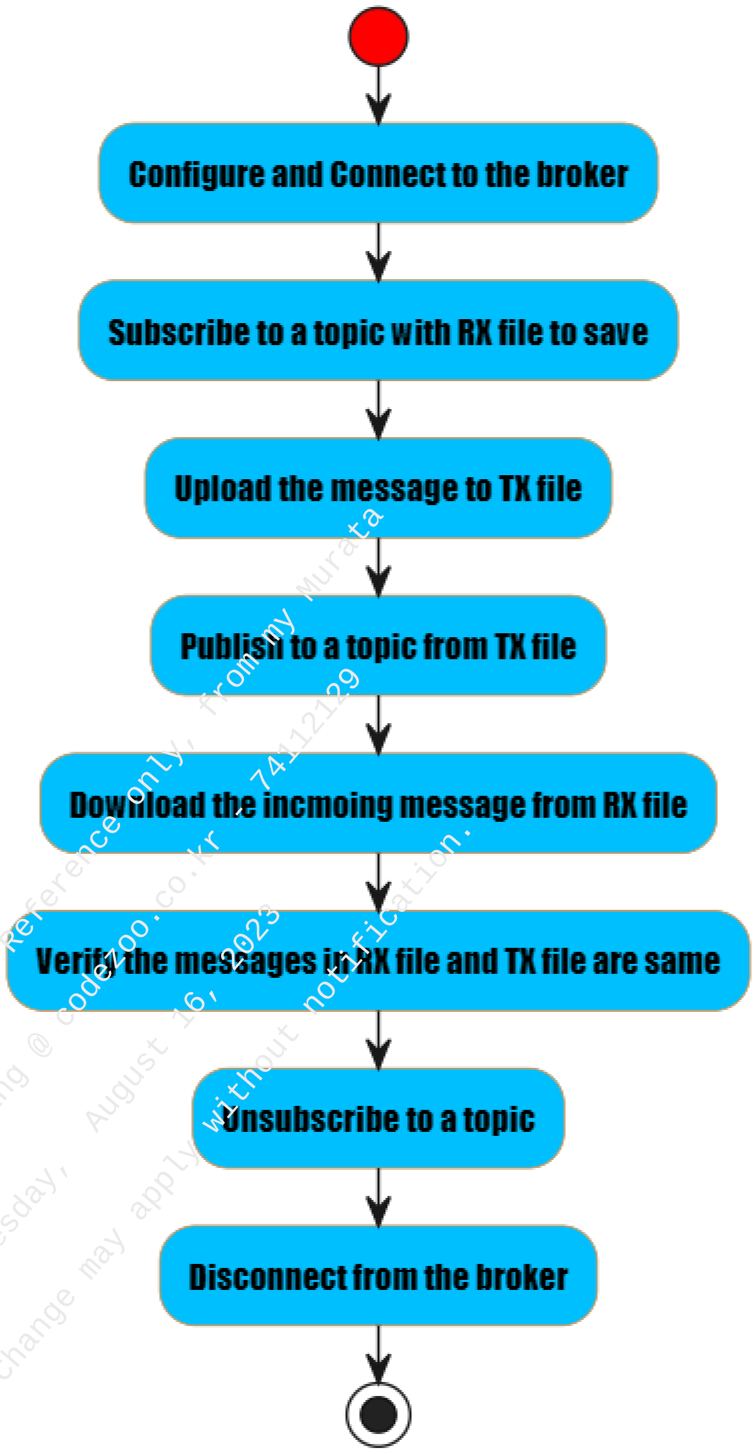
This example shows a MQTT scenario running over a non-secure connection. The procedure for this example is as below:

- Subscribe to a topic and specify that incoming message is stored into a file ("b:/RxFile")
- Upload a message to a file in NV store ("b:/TxFile")
- Publish the message to the same topic from this file ("b:/TxFile")
- Receive the message published back to the device and stored into the file ("b:/RxFile")
- Download the message from the stored file ("b:/RxFile")
- Verify that the messages transmitted and received are the same

In this example, QoS of subscribing and publishing are both 0. HiveMQ open-source public broker is used (for more details see their website at https://www.hivemq.com/public-mqtt-broker/).

Confidential, Reference only, from my Murata
rooney.yang @ codezoo.com 4112129
Wednesday, August 16, 2023
Chang may apply without notification.

The following is a simple illustration of the procedure for this example:


Configure and Connect to the broker
Subscribe to a topic with RX file to save
Upload the message to TX file
Publish to a topic from TX file
Download the incmoing message from RX file
Verify the messages in RX file and TX file are same
Unsubscribe to a topic
Disconnect from the broker

Confidential, Reference only, from my Murata rooney.jang @ codezoo.co.kr - 74112129 Wednesday, August 16, 2023 change may apply without notification.

Figure 15 Subscribe and Publish to a topic to/from a file

The following are the detailed steps for this example:

- Connect to the broker

See 3.1.1 for the details

- Subscribe to a topic on the broker and store received data in RxFile.

```
AT%MQTTCMD="subscribe",1,0,"TopicName", "RxFile"

<0> - QoS level is 0

%MQTTCMD: 1
OK
<1> - message ID
```

Receive a subscribe procedure confirmation status URC as below,

```
%MQTTEVU:"SUBCONF",1,1,0

<1> - message ID
<0> - success
```

- Upload the message to a TX file

```
AT%FILECMD="PUT","TxFile",1,5,"3768048439"
OK

<1> - "inband", usage of AT%FILEDATA is expected
<5> - the length of the file to be transferred
<3768048439> - CRC32 value in decimal encoding of the file to be transferred

AT%FILEDATA="WRITE",0,10,"3131313131"
%FILEDATA:10

OK

<10> - length of transmitted data in ASCII string length units
<3131313131> - The file chunk data, in HEX format
```

- Publish the message to a topic on the broker from a file

```
AT%MQTTCMD="publish",1,0,0,"TopicName",0, "b:/TxFile"

<0> - QoS level is 0

%MQTTCMD: 1
OK

<1> - message ID
```

- Receive an Incoming Publication Message Received URC as below,

```
%MQTTEVU:"PUBRCV",1,0,"TopicName",0,5,"RxFile"
```

Confidential, Reference only, from Murata
rooney.jang@codezoo.co.kr, 20112129
Wednesday, August 16, 2023
Changes may apply without notification

```
<0> - message ID. It may be zero (undefined) for QoS=0
<5> - data size in bytes stored into file
```

- Download the message from a RX file

```
AT%FILECMD="GET","b:/RxFile",1
<1> - "inband", usage of AT%FILEDATA is expected

%FILECMD: 5,3768048439

OK

<5> - the length of the file to be transferred
<3768048439> - CRC32 value in decimal encoding of the file to be transferred

AT%FILEDATA="READ",10
<10> - the maximal length of data in bytes which requested to be read in this
transaction

%FILEDATA:0,10,"3131313131"

OK

<0> - no more data to read
<10> - the actual received data length in ASCII string length units
<3131313131> - the read data, in HEX format
```

- Verify the messages in RX file and TX file are same

```
The message "3131313131" in RxFile is the same as the message the host app
uploaded.
```

- Unsubscribe from the broker

```
AT%MQTTCMD="UNSUBSCRIBE",1,"TopicName"
%MQTTCMD: 2
OK

%MQTTEVU:"UNSCONF",1,2,0

<2> - message ID
<0> - success
```

- Teardown connection:

  See 3.1.1 for the details

3.2 Secure Connection examples

This section provides a few examples of MQTT service usage over secure connection like TLS.

3.2.1 Configure and connect to a broker

This example shows a MQTT scenario running over a secure connection. The procedure for this example is to

Confidential, Reference only from my Murata
rodney.jang@codezoo.kr 7412129
Wednesday, August 16, 2023
change may apply without notification.

configure and connect to a public MQTT broker. In this example, emqx open-source public broker is used, for more details see their website at https://www.emqx.io.

The following is a simple illustration of the procedure for this example:

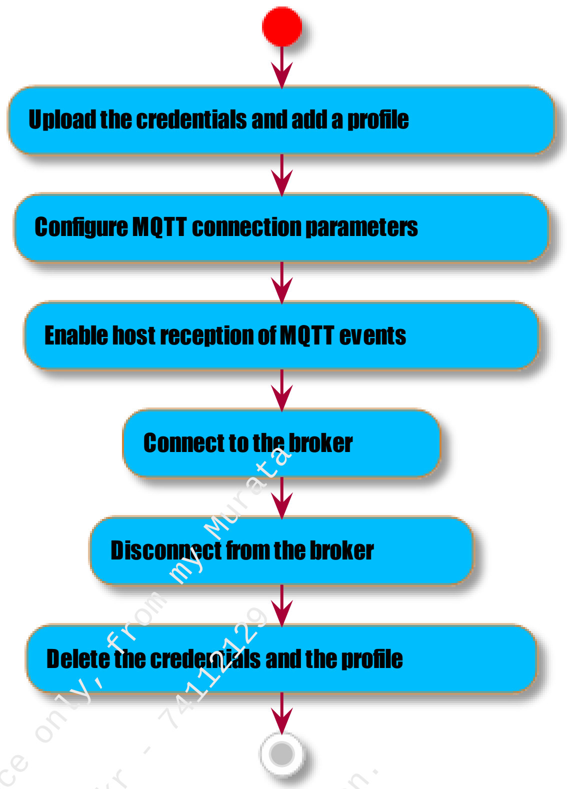

Upload the credentials and add a profile
Configure MQTT connection parameters
Enable host reception of MQTT events
Connect to the broker
Disconnect from the broker
Delete the credentials and the profile

Figure 16 Configure and connect to a broker over a secure connection

The following are the detailed steps for this example:

- Upload the credentials to NV and Add a single TLS/DTLS profile into configuration file

```
AT%CERTCMD="WRITE","broker.emqx.io-ca.crt",0,
"-----BEGIN CERTIFICATE-----
MIIF3jCCA8agAwIBAgIQAf1tMPyjylGoG7xkDjUDLTANBgkqhkiG9w0BAQwFADCB
…
L6KCq9NjRHDEjf8tM7qtj3u1cIiuPhnPQCjY/MiQu12ZIvVS5ljFH4gxQ+6IHdfG
jjxDah2nGN59PRbxYvnKkKj9
-----END CERTIFICATE-----"
OK

AT%CERTCMD="WRITE","client.cer",0,
"-----BEGIN CERTIFICATE-----
MIIDWTCCAkGgAwIBAgIUeUZ3uhjpS4AF3XhWUUSlkLzZzAowDQYJKoZIhvcNAQEL
…
40RPr/lZfKqq0GLo/qISrJHtV+YCFR49eVPWej2cQYNxU7bn7Y6aqLgTKBDQ0yjK
0p15A68LLFDygb5SGrvNtwqjdD7GhTWgJuJQRDHeKecv0U2mBggYVM+xqy4V
-----END CERTIFICATE-----"
OK
```

Confidential Reference only, from my Murata - 74112129
cooky.jang@codezoo.co.kr - Wednesday, August 16, 2023
Changes may apply without notification.

```
AT%CERTCMD="WRITE","privkey.cer",1,
"-----BEGIN RSA PRIVATE KEY-----
MIIEpAIBAAKCAQEAtBz1QeNDQx1GC3uzamE5WtYSjIKBUpvteqO2m1pl9RP2/zSM
…
ET2UQeOFb+90fcbi+hg0bhEPDtLpqkfojWRZTv4x2FGS7x4c5a+Q/o2QZs8sbcY1
0vi9IXZa0XpGUGPJgzKVAqFjyUk5vIHdN7o2SSioAlotjBu1es5pQw==
-----END RSA PRIVATE KEY-----"
OK
```

Note: make sure there is no <CR> in the `AT%CERTCMD="WRITE"` command above. This multi-line command should be Linux file format (LF-based file). One way of converting it is to copy this command to a Notepad++ editor and remove all the <CR> at the end of each line and only leave the<LF>.

```
AT%CERTCFG="ADD",1,"broker.emqx.io-ca.crt",,"client.cer","privkey.cer"
OK

<1> - Profile ID
```

- Configure MQTT connection parameters

```
AT%MQTTCFG="nodes",1,"ClientName","broker.emqx.io"
OK

AT%MQTTCFG="IP",1,,0,8883
OK

AT%MQTTCFG="TLS",1,0,1
OK

<1> - Profile ID

AT%MQTTCFG="PROTOCOL",1,0,60,1
OK

<1> - clean session
```

- Enable all MQTT events

```
AT%MQTTEV="all",1
OK
```

- Connect to the broker:

```
AT%MQTTCMD="connect",1
OK

<1> - connection id
```

Confidential, Reference only from Murata
pooney.wang@codepoo.co.kr - 74412129
Wednesday, August 16, 2023
Changes may apply without notification.

Receive a connect procedure confirmation status URC

```
%MQTTEVU:"CONCONF",1,0

<0> - success
```

- Teardown connection:

```
AT%MQTTCMD="disconnect",1
OK
```

Receive a Disconnect Procedure Confirmation Status URC

```
%MQTTEVU:"DISCONF",1,0

<0> - success
```

- Delete the credentials and the profile

```
AT%CERTCMD="DELETE","broker.emqx.io-ca.crt"
OK
AT%CERTCMD="DELETE","client.cer"
OK
AT%CERTCMD="DELETE","./privkey.cer"
OK

AT%CERTCFG="DELETE",1
OK
```

For detailed procedure description, refer to the Figure 2 as in the non-secure connection example except that the credentials need to be uploaded and TLS connection needs to be established.

Confidential reference only, from by Murata
rooney.jang @ codezoo.co.kr - 74112229
Wednesday, August 16, 2023
Change may apply without notification.

4. PRECONDITION TO USE OUR PRODUCTS

PLEASE READ THIS NOTICE BEFORE USING OUR PRODUCTS.

Please make sure that your product has been evaluated and confirmed from the aspect of the fitness for the specifications of our product when our product is mounted to your product.
All the items and parameters in this product specification/datasheet/catalog have been prescribed on the premise that our product is used for the purpose, under the condition and in the environment specified in this specification. You are requested not to use our product deviating from the condition and the environment specified in this specification.
Please note that the only warranty that we provide regarding the products is its conformance to the specifications provided herein. Accordingly, we shall not be responsible for any defects in products or equipment incorporating such products, which are caused under the conditions other than those specified in this specification.
WE HEREBY DISCLAIMS ALL OTHER WARRANTIES REGARDING THE PRODUCTS, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION ANY WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE, THAT THEY ARE DEFECT-FREE, OR AGAINST INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS.
The product shall not be used in any application listed below which requires especially high reliability for the prevention of such defect as may directly cause damage to the third party's life, body or property. You acknowledge and agree that, if you use our products in such applications, we will not be responsible for any failure to meet such requirements. Furthermore, YOU AGREE TO INDEMNIFY AND DEFEND US AND OUR AFFILIATES AGAINST ALL CLAIMS, DAMAGES, COSTS, AND EXPENSES THAT MAY BE INCURRED, INCLUDING WITHOUT LIMITATION, ATTORNEY FEES AND COSTS, DUE TO THE USE OF OUR PRODUCTS IN SUCH APPLICATIONS.

- Aircraft equipment. - Aerospace equipment - Undersea equipment.
- Power plant control equipment - Medical equipment.
- Transportation equipment (vehicles, trains, ships, elevator, etc.).
- Traffic signal equipment. - Disaster prevention / crime prevention equipment.
- Burning / explosion control equipment
- Application of similar complexity and/ or reliability requirements to the applications listed in the above.

We expressly prohibit you from analyzing, breaking, reverse-engineering, remodeling altering, and reproducing our product. Our product cannot be used for the product which is prohibited from being manufactured, used, and sold by the regulations and laws in the world.
We do not warrant or represent that any license, either express or implied, is granted under any our patent right, copyright, mask work right, or our other intellectual property right relating to any combination, machine, or process in which our products or services are used. Information provided by us regarding third-party products or services does not constitute a license from us to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from us under our patents or other intellectual property.
Please do not use our products, our technical information and other data provided by us for the purpose of developing of mass-destruction weapons and the purpose of military use.
Moreover, you must comply with "foreign exchange and foreign trade law", the "U.S. export administration regulations", etc.
Please note that we may discontinue the manufacture of our products, due to reasons such as end of supply of materials and/or components from our suppliers.
By signing on specification sheet or approval sheet, you acknowledge that you are the legal representative for your company and that you understand and accept the validity of the contents herein. When you are not able to return the signed version of specification sheet or approval sheet within 30 days from receiving date of specification sheet or approval sheet, it shall be deemed to be your consent on the content of specification sheet or approval sheet. Customer acknowledges that engineering samples may deviate from specifications and may contain defects due to their development status. We reject any liability or product warranty for engineering samples. In particular we disclaim liability for damages caused by
- the use of the engineering sample other than for evaluation purposes, particularly the installation or integration in the product to be sold by you,
-deviation or lapse in function of engineering sample,
-improper use of engineering samples.
We disclaim any liability for consequential and incidental damages.
If you can't agree the above contents, you should inquire our sales.

Confidential, Reference only, from Murata
Yooeun Jang@codegoo.co.kr_7411229
Wednesday, August 16, 2023
change may apply without notification