Build a Smarter World

M26 MQTT

Application Note

GSM/GPRS Module Series

Rev. M26_MQTT_Application_Note_V1.0

Date: 2017-06-12

www.quectel.com

Our aim is to provide customers with timely and comprehensive service. For any assistance, please contact our company headquarters:

Quectel Wireless Solutions Co., Ltd.
Office 501, Building 13, No.99, Tianzhou Road, Shanghai, China, 200233
Tel: +86 21 5108 6236
Email: info@quectel.com

Or our local office. For more information, please visit:
http://www.quectel.com/support/salesupport.aspx

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/techsupport.aspx
Or Email to: Support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

About the document

History

Revision	Date	Author	Description
1.0	2017-06-12	Louis GU/ Sherlock ZHAO	Initial

Contents

Table Index

1 Introduction

MQTT (Message Queuing Telemetry Transport) is a broker-based publish/subscribe messaging protocol designed to be open, simple, lightweight and easy to implement. It is designed for connections with remote locations where a "small code footprint" is required or the network bandwidth is limited. This document mainly introduces how to use the MQTT function of Quectel M26 modules through AT commands.

2 MQTT Data Interaction

This chapter gives the data interaction mechanism of MQTT function.

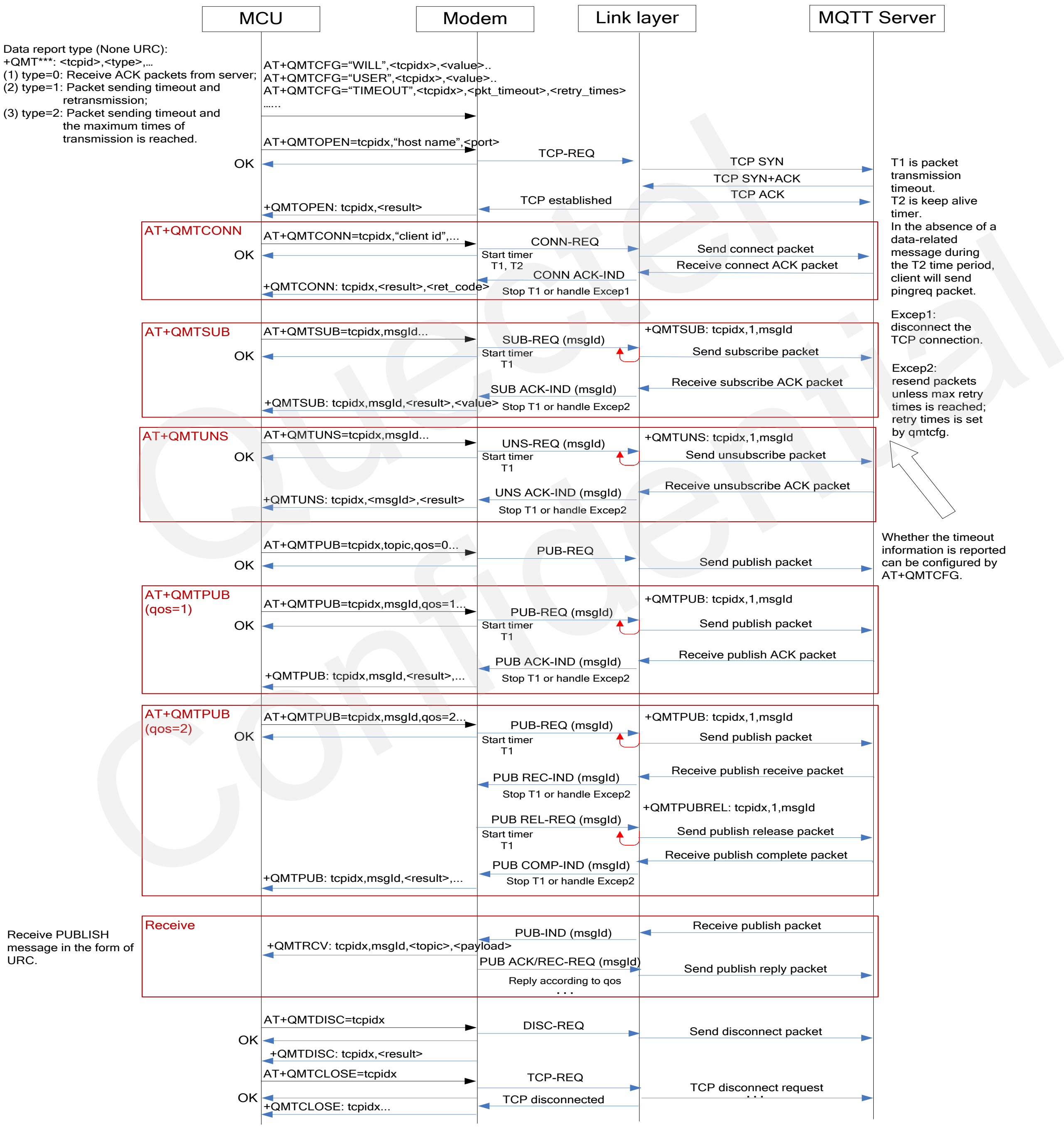
MCU
Modem
Link layer
MQTT Server
Data report type (None URC):
+QMT***: <tcpid>,<type>,...
(1) type=0: Receive ACK packets from server;
(2) type=1: Packet sending timeout and retransmission;
(3) type=2: Packet sending timeout and the maximum times of transmission is reached.
AT+QMTCFG="WILL",<tcpidx>,<value>..
AT+QMTCFG="USER",<tcpidx>,<value>..
AT+QMTCFG="TIMEOUT",<tcpidx>,<pkt_timeout>,<retry_times>
......
AT+QMTOPEN=tcpidx,"host name",<port>
OK
TCP-REQ
TCP SYN
TCP SYN+ACK
TCP ACK
TCP established
+QMTOPEN: tcpidx,<result>
T1 is packet transmission timeout.
T2 is keep alive timer.
In the absence of a data-related message during the T2 time period, client will send pingreq packet.
AT+QMTCONN
AT+QMTCONN=tcpidx,"client id",...
OK
CONN-REQ
Start timer T1, T2
Send connect packet
Receive connect ACK packet
CONN ACK-IND
+QMTCONN: tcpidx,<result>,<ret_code>
Stop T1 or handle Excep1
Excep1: disconnect the TCP connection.
AT+QMTSUB
AT+QMTSUB=tcpidx,msgId...
OK
SUB-REQ (msgId)
Start timer T1
+QMTSUB: tcpidx,1,msgId
Send subscribe packet
Receive subscribe ACK packet
SUB ACK-IND (msgId)
+QMTSUB: tcpidx,msgId,<result>,<value>
Stop T1 or handle Excep2
Excep2: resend packets unless max retry times is reached; retry times is set by qmtcfg.
AT+QMTUNS
AT+QMTUNS=tcpidx,msgId...
OK
UNS-REQ (msgId)
Start timer T1
+QMTUNS: tcpidx,1,msgId
Send unsubscribe packet
Receive unsubscribe ACK packet
UNS ACK-IND (msgId)
+QMTUNS: tcpidx,<msgId>,<result>
Stop T1 or handle Excep2
Whether the timeout information is reported can be configured by AT+QMTCFG.
AT+QMTPUB=tcpidx,topic,qos=0...
OK
PUB-REQ
Send publish packet
AT+QMTPUB (qos=1)
AT+QMTPUB=tcpidx,msgId,qos=1...
OK
PUB-REQ (msgId)
Start timer T1
+QMTPUB: tcpidx,1,msgId
Send publish packet
Receive publish ACK packet
PUB ACK-IND (msgId)
+QMTPUB: tcpidx,msgId,<result>,...
Stop T1 or handle Excep2
AT+QMTPUB (qos=2)
AT+QMTPUB=tcpidx,msgId,qos=2...
OK
PUB-REQ (msgId)
Start timer T1
+QMTPUB: tcpidx,1,msgId
Send publish packet
Receive publish receive packet
PUB REC-IND (msgId)
Stop T1 or handle Excep2
+QMTPUBREL: tcpidx,1,msgId
PUB REL-REQ (msgId)
Start timer T1
Send publish release packet
Receive publish complete packet
PUB COMP-IND (msgId)
+QMTPUB: tcpidx,msgId,<result>,...
Stop T1 or handle Excep2
Receive
Receive PUBLISH message in the form of URC.
PUB-IND (msgId)
Receive publish packet
+QMTRCV: tcpidx,msgId,<topic>,<payload>
PUB ACK/REC-REQ (msgId)
Send publish reply packet
Reply according to qos
. . .
AT+QMTDISC=tcpidx
DISC-REQ
Send disconnect packet
OK
+QMTDISC: tcpidx,<result>
AT+QMTCLOSE=tcpidx
TCP-REQ
TCP disconnect request
. . .
OK
TCP disconnected
+QMTCLOSE: tcpidx...

Figure 1: MQTT Data Interaction Diagram

3 MQTT AT Commands

This chapter presents the AT commands for operating MQTT function.

3.1. AT Command Syntax

Table 1: Types of AT Commands and Responses

Test Command	**AT+<*x*>=?**	This command returns the list of parameters and value ranges set by the corresponding Write Command or internal processes.
Read Command	**AT+<*x*>?**	This command returns the currently set value of the parameter or parameters.
Write Command	**AT+<x>=<…>**	This command sets the user-definable parameter values.
Execution Command	**AT+<x>**	This command reads non-variable parameters affected by internal processes in the UE.

3.2. Description of AT Commands

3.2.1. AT+QMTCFG Configure Parameters of MQTT

The command is used to configure optional parameters of MQTT.

AT+QMTCFG Configure Optional Parameters of MQTT	
Test Command **AT+QMTCFG=?**	Response **+QMTCFG: "<config_type>"**,(list of supported **<tcpconnectID>**s),[,(list of supported **<value>**s)] **OK**
Write Command Configure Will information **AT+QMTCFG="WILL",<tcpconnectID>[,<will_fg>[,<will_qos>,<will_retain>,"<will_topic>","<will_msg>"]]**	Response **OK** If **<will_fg>**, **<will_qos>**, **<will_retain>**, **<will_topic>** and **<will_msg>** are omitted, query the "WILL" value. Response **+QMTCFG: <will_fg>[,<will_qos>,<will_retain>,<will_topic>,<will_msg>]** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure timeout of message delivery **AT+QMTCFG="TIMEOUT",<tcpconnectID>[,<pkt_timeout>[,<retry_times>][,<timeout_notice>]]**	Response **OK** If **<pkt_timeout>**, **<retry_times>**, **<timeout_notice>** are omitted, query the "TIMEOUT" value. Response **+QMTCFG: <pkt_timeout>,<retry_times>,<timeout_notice>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**

Write Command Configure session type **AT+QMTCFG="SESSION",<tcpconnectID>[,<clean_session>]**	Response **OK** If **<clean_session>** is omitted, query the "SESSION" value. Response **+QMTCFG: <clean_session>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure keep alive time **AT+QMTCFG="KEEPALIVE",<tcpconnectID>[,<keep-alive time>]**	Response **OK** If **<keep-alive time>** is omitted, query the "KEEPALIVE" value. Response **+QMTCFG: <keep-alive time>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure SSL secure connection **AT+QMTCFG="SSL",<tcpconnectID>[,<sslenable>[,<ctxindex>]]**	Response **OK** If **<sslenable>** is omitted, query the "SSL" value. Response **+QMTCFG: <sslenable>[,<ctxindex>]** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	300ms

Parameter

<config_type>	The type of configuration. It can be any of the following types: "WILL" "TIMEOUT" "SESSION" "KEEPALIVE" "SSL"

<tcpconnectID> MQTT socket identifier. The range is 0-5.

<value> Configuration value of MQTT optional parameter.

<will_fg> Configure the Will flag
0 Ignore the Will flag configuration
1 Require the Will flag configuration

<will_qos> Quality of service for message delivery
0 At most once
1 At least once
2 Exactly once

<will_retain> The Will retain flag is only used on PUBLISH messages.
0 When a client sends a PUBLISH message to a server, the server will not hold on to the message after it has been delivered to the current subscribers
1 When a client sends a PUBLISH message to a server, the server should hold on to the message after it has been delivered to the current subscribers

<will_topic> Will topic string

<will_msg> The Will message defines the content of the message that is published to the will topic if the client is unexpectedly disconnected. It can be a zero-length message.

<pkt_timeout> Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.

<retry_times> Retry times when packet delivery times out. The range is 0-10. The default value is 3.

<timeout_notice> 0 Not report timeout message when transmitting packet
1 Report timeout message when transmitting packet

<clean_session> Configure the session type
0 The server must store the subscriptions of the client after it disconnects.
1 The server must discard any previously maintained information about the client and treat the connection as "clean".

<keep-alive time> Keep alive time. The range is 0-3600. The default value is 120. Unit: second. It defines the maximum time interval between messages received from a client. If the server does not receive a message from the client within 1.5 times of the keep alive time period, it disconnects the client as if the client has sent a DISCONNECT message.
0 The client is not disconnected

<sslenable> Configure the MQTT SSL mode
0 Use normal TCP connection for MQTT
1 Use SSL TCP secure connection for MQTT

<ctxindex> SSL context index. The range is 0-5.

NOTES

1. If **<will_flag>**=1, then **<will_qos>**, **<will_retain>**, **<will_topic>** and **<will_msg>** must be present. Otherwise they will be omitted.
2. **<clean_session>**=0 is only effective when server supports the operation.
3. If MQTT connection is configured to SSL mode, **<ctxindex>** must be present. Also, customers need

to use **AT+QSSLCFG** command to configure the SSL version, cipher suite, secure level, CA certificate, client certificate, client key and ignorance of RTC time, which will be used in MQTT SSL handshake procedure.

4. Care must be taken to ensure message delivery does not time out while it is still being sent.

3.2.2. AT+QMTOPEN Open a Network for MQTT Client

The command is used to open a network for MQTT client.

AT+QMTOPEN Open a Network for MQTT Client	
Test Command **AT+QMTOPEN=?**	Response **+QMTOPEN:** (list of supported **<tcpconnectID>**s),**"<host_name>"**,(list of supported **<port>**s) **OK**
Read Command **AT+QMTOPEN?**	Response **[+QMTOPEN: <tcpconnectID>,"<host_name>",<port>]** **OK**
Write Command **AT+QMTOPEN=<tcpconnectID>,"<host_name>",<port>**	Response **OK** **+QMTOPEN: <tcpconnectID>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	75s, determined by network

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<host_name>	The address of the server. It could be an IP address or a domain name. The maximum size is 100 bytes.
<port>	The port of the server. The range is 1-65535.
<result>	Result of the command execution -1 Failed to open network 0 Opened network successfully 1 Wrong parameter 2 MQTT identifier is occupied 3 Failed to activate PDP

4	Failed to parse domain name
5	Network disconnection error

3.2.3. AT+QMTCLOSE Close a Network for MQTT Client

The command is used to close a network for MQTT client.

AT+QMTCLOSE Close a Network for MQTT Client	
Test Command **AT+QMTCLOSE=?**	Response **+QMTCLOSE:** (list of supported **<tcpconnectID>**s) **OK**
Write Command **AT+QMTCLOSE=<tcpconnectID>**	Response **OK** **+QMTCLOSE: <tcpconnectID>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	300ms

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<result>	Result of the command execution -1 Failed to close network 0 Closed network successfully

3.2.4. AT+QMTCONN Connect a Client to MQTT Server

The command is used when a client requests a connection to MQTT server. When a TCP/IP socket connection is established from a client to a server, a protocol level session must be created using a CONNECT flow.

AT+QMTCONN Connect a Client to MQTT Server	
Test Command **AT+QMTCONN=?**	Response **+QMTCONN:** (list of supported **<tcpconnectID>**s), **"<clientID>"[,"<username>"[,"<password>"]]** **OK**

Read Command **AT+QMTCONN?**	Response **[+QMTCONN: <tcpconnectID>,<state>]** **OK**
Write Command **AT+QMTCONN=<tcpconnectID>,“<clientID>”[,“<username>”[,“<password>”]]**	Response **OK** **+QMTCONN: <tcpconnectID>,<result>[,<ret_code>]** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	**<pkt_timeout>** (default 5s), determined by network

Parameter

<tcpconnectID> MQTT socket identifier. The range is 0-5.

<clientID> The client identifier string.

<username> User name of the client. It can be used for authentication.

<password> Password corresponding to the user name of the client. It can be used for authentication.

<result> Result of the command execution
0 Sent packet successfully and received ACK from server
1 Packet retransmission
2 Failed to send packet

<state> MQTT connection state
1 MQTT is initial
2 MQTT is connecting
3 MQTT is connected
4 MQTT is disconnecting

<ret_code> Connect return code
0 Connection Accepted
1 Connection Refused: Unacceptable Protocol Version
2 Connection Refused: Identifier Rejected
3 Connection Refused: Server Unavailable
4 Connection Refused: Bad User Name or Password
5 Connection Refused: Not Authorized

NOTE

If a client with the same Client ID is already connected to the server, the "older" client must be disconnected by the server before completing the CONNECT flow of the new client.

3.2.5. AT+QMTDISC Disconnect a Client from MQTT Server

The command is used when a client requests a disconnection from MQTT server. A DISCONNECT message is sent from the client to the server to indicate that it is about to close its TCP/IP connection.

AT+QMTDISC Disconnect a Client from MQTT Server	
Test Command **AT+QMTDISC=?**	Response **+QMTDISC:** (list of supported **<tcpconnectID>**s) **OK**
Write Command **AT+QMTDISC=<tcpconnectID>**	Response **OK** **+QMTDISC: <tcpconnectID>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	300ms

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<result>	Result of the command execution -1 Failed to close connection 0 Closed connection successfully

3.2.6. AT+QMTSUB Subscribe to Topics

The command is used to subscribe to one or more topics. A SUBSCRIBE message is sent by a client to register an interest in one or more topic names with the server. Messages published to these topics are delivered from the server to the client as PUBLISH messages.

AT+QMTSUB Subscribe to Topics	
Test Command **AT+QMTSUB=?**	Response **+QMTSUB:** (list of supported **<tcpconnectID>**s),(list of supported **<msgID>**s),**"<topic>"**,(list of supported **<qos>**s) **OK**
Write Command **AT+QMTSUB=<tcpconnectID>,<msgID>,"<topic1>",<qos1>[,"<topic2>",<qos2>…]**	Response **OK** **+QMTSUB: <tcpconnectID>,<msgID>,<result>[,<value>]**

	If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	**<pkt_timeout>** * **<retry_times>** (default 15s), determined by network

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<msgID>	Message identifier of packet. The range is 1-65535.
<topic>	Topic that the client wants to subscribe to or unsubscribe from
<qos>	The QoS level at which the client wants to publish the messages. 0 At most once 1 At least once 2 Exactly once
<result>	Result of the command execution 0 Sent packet successfully and received ACK from server 1 Packet retransmission 2 Failed to send packet
<value>	If **<result>** is 0, it is a vector of granted QoS levels. If **<result>** is 1, it means the times of packet retransmission. If **<result>** is 2, it will not be presented.

NOTE

The **<msgID>** is only present in messages where the QoS bits in the fixed header indicate QoS levels 1 or 2. It must be unique amongst the set of "in flight" messages in a particular direction of communication. It typically increases by exactly one from one message to the next, but is not required to do so.

3.2.7. AT+QMTUNS Unsubscribe from Topics

The command is used to unsubscribe from one or more topics. An UNSUBSCRIBE message is sent by the client to the server to unsubscribe from named topics.

AT+QMTUNS Unsubscribe from Topics	
Test Command **AT+QMTUNS=?**	Response **+QMTUNS:** (list of supported **<tcpconnectID>**s),(list of supported **<msgID>**s),**"<topic>"** **OK**
Write Command **AT+QMTUNS=<tcpconnectID>,<ms**	Response **OK**

gID>,“<topic1>”[,“<topic2>”…]	**+QMTUNS: <tcpconnectID>,<msgID>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	**<pkt_timeout>** * **<retry_times>** (default 15s), determined by network

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<msgID>	Message identifier of packet. The range is 1-65535.
<topic>	Topic that the client wants to subscribe to or unsubscribe from
<result>	Result of the command execution 0 Sent packet successfully and received ACK from server 1 Packet retransmission 2 Failed to send packet

3.2.8. AT+QMTPUB Publish Messages

The command is used to publish messages by a client to a server for distribution to interested subscribers. Each PUBLISH message is associated with a topic name. If a client subscribes to one or more topics, any message published to those topics are sent by the server to the client as a PUBLISH message.

AT+QMTPUB Publish Messages	
Test Command **AT+QMTPUB=?**	Response **+QMTPUB:** (list of supported **<tcpconnectID>**s),(list of supported **<msgID>**s),(list of supported **<qos>**s),(list of supported **<retain>**s),**“<topic>”**,**“<msg>”** **OK**
Write Command **AT+QMTPUB=<tcpconnectID>,<msgID>,<qos>,<retain>,“<topic>”** After “>” is responded, input the data to be sent. Tap “CTRL+Z” to send, and tap “ESC” to cancel the operation.	Response **OK** **+QMTPUB: <tcpconnectID>,<msgID>,<result>[,<value>]** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	**<pkt_timeout>** * **<retry_times>** (default 15s), determined by network

Parameter

<tcpconnectID> MQTT socket identifier. The range is 0-5.

<msgID> Message identifier of packet. The range is 0-65535. It will be 0 only when **<qos>**=0.

<qos> The QoS level at which the client wants to publish the messages.
0 At most once
1 At least once
2 Exactly once

<retain> The server will retain the message after it has been delivered to the current subscribers or not.
0 The server will not retain the message after it has been delivered to the current subscribers
1 The server will retain the message after it has been delivered to the current subscribers

<topic> Topic that needs to be published

<result> Result of the command execution
0 Sent packet successfully and received ACK from server (message that published when **<qos>**=0 does not require ACK)
1 Packet retransmission
2 Failed to send packet

<value> If **<result>** is 1, it means the times of packet retransmission.
If **<result>** is 0 or 2, it will not be presented.

NOTES

1. If this command is executed successfully and gets **OK** back, the client can continue to publish new packet. The maximum quantity of transmitting packet should not be greater than that of in-flight windows (5); otherwise, **+CME ERROR: 8512** will be returned. For more details, please refer to ***Chapter 6.2***.
2. After executing this command, the client will be ready to send data, which will be sent as payload. The maximum length of the input data is 1548 bytes at a time and tap "Ctrl+Z" to send the data.
3. PUBLISH messages can be sent either from a publisher to the server, or from the server to a subscriber. When server publishes messages to a subscriber, **+QMTRECV: <tcpconnectID>,<msgID>,<topic>,<payload>** will be returned to notify the host to read the received data that is reported from MQTT server. For more details, please refer to ***Chapter 4.1.2***.

4 MQTT Related Notifications

This chapter gives MQTT related notifications and their descriptions.

Table 1: MQTT Related Notifications

Index	Notification Display	Description
[1]	**+QMTSTAT: <tcpconnectID>,<err_code>**	When the state of MQTT link layer is changed, the client will close the MQTT connection and report the URC.
[2]	**+QMTRECV: <tcpconnectID>,<msgID>,<topic><payload>**	The client has received the packet data from MQTT server.

4.1.1. Description of URC "+QMTSTAT"

Table 2: Error Codes of the URC

Code of <err>	Description	How to do
1	Connection is closed or reset by peer.	Execute **AT+QMTOPEN** command and reopen MQTT connection.
2	Sending PINGREQ packet timed out or failed.	Deactivate PDP first, and then active PDP and reopen MQTT connection.
3	Sending CONNECT packet timed out or failed	1. Check inputted user name and password are correct or not. 2. Make sure the client ID is not used. 3. Reopen MQTT connection and try to send CONNECT packet to server again.
4	Receiving CONNACK packet timed out or failed	1. Check inputted user name and password are correct or not. 2. Make sure the client ID is not used. 3. Reopen MQTT connection and try to send CONNECT packet to server again.
5	Client sends DISCONNECT packet to sever but server is initiative to close MQTT	This is a normal process.

	connection.	
6	Client is initiative to close MQTT connection due to packet sending failure all the time.	1. Make sure the data is correct. 2. Try to reopen MQTT connection since there may be network congestion or error.
7	The link is not alive or the server is unavailable	Make sure the link is alive or the server is available currently.
8-255	Reserved for future use	

4.1.2. Description of “+QMTRECV”

The format of data report is “**+QMTRECV:**”. It is mainly used to notify the host to read the received MQTT packet data that is reported from MQTT server.

Notify the Host to Read MQTT Packet Data	
+MQTRECV: <tcpconnectID>,<msgID>,<topic>,<payload>	Notify the host to read the received data that is reported from MQTT server.
Reference	Notify the host to read the received data that is reported from MQTT server.

Parameter

<tcpconnectID>	MQTT socket identifier. The range is 0-5.
<msgID>	The message identifier of packet
<topic>	The topic that received from MQTT server
<payload>	The payload that relates to the topic name

5 Examples

This chapter gives the examples to explain how to use MQTT commands.

5.1. Example of MQTT Operation without SSL

```
AT+QMTOPEN=?
+QMTOPEN: <tcpconnectID>,“<host_name>”,<port>

OK
//Open a network for MQTT client.
AT+QMTOPEN=0, “220.180.239.212”,1883
OK

+QMTOPEN: 0,0                    //Opened the MQTT client network successfully.

AT+QMTOPEN?
+QMTOPEN: 0,“220.180.239.212”,1883

OK

AT+QMTCONN=?
+QMTCONN: <tcpconnectID>,“<clientID>” [,“<username>”[,“<password>”]]

OK

//Connect a client to MQTT server.
AT+QMTCONN=0,“clientExample”,“username”,“password”
OK

+QMTCONN: 0,0,0                  //Connected the client to MQTT server successfully.

AT+QMTSUB=?
+QMTSUB : <tcpconnectID>,<msgID>,“<topic>”,<qos>

OK

//Subscribe to topics.
```

```
AT+QMTSUB=0,1,“topic/example”,2
OK

+QMTSUB: 0,1,0,2

AT+QMTSUB=0,1,“topic/pub”,0
OK

+QMTSUB: 0,1,0,0

//If a client subscribes to a topic and other devices publish the same topic to the server, the module will
report the following information.
+QMTRECV: 0,0, “topic/example”, “This is the payload related to topic”

//Unsubscribe from topics.
AT+QMTUNS=0,2, “topic/example”
OK

+QMTUNS: 0,2,0

AT+QMTPUB=?
+QMTPUB : <tcpconnectID>,<msgID>,<qos>,<retain>,“<topic>”

OK

//Publish messages.
AT+QMTPUB=0,0,0,0,“topic/pub”
>This is test data, hello MQTT.      //After receiving ‘>’, input data “This is test data, hello MQTT.” and
                                      send it. The maximum length of the data is 1548 bytes and the data
                                      that beyond 1548 bytes will be omitted. After inputting data, tap
                                      “Ctrl+Z” to send.
OK

+QMTPUB: 0,0,0

//If a client subscribes to a topic named “topic/pub” and other devices publish the same topic to the server,
the module will report the following information.
+QMTRECV: 0,0, “topic/pub”,This is test data, hello MQTT.

//Disconnect a client from MQTT server.
AT+QMTDISC=0
OK

+QMTDISC: 0,0                        //Closed disconnection successfully.
```

5.2. Example of MQTT Operation with SSL

```
//Configure MQTT session into SSL mode.
AT+QMTCFG=”SSL”, 0, 1, 2
OK

//If SSL authentication mode is “server authentication”, store CA certificate to RAM.
AT+QSECWRITE="RAM:cacert.pem",1758,100
CONNECT
<Input the cacert.pem data, the size is 1758 bytes>
+QSECWRITE: 1758,384a

OK

//If SSL authentication mode is “server authentication”, store CC certificate to RAM.
AT+QSECWRITE="RAM:client.pem",1220,100
CONNECT
<Input the client.pem data, the size is 1220 bytes>
+QSECWRITE: 1220,2d53

OK

//If SSL authentication mode is “server authentication”, store CK certificate to RAM.
AT+QSECWRITE="RAM:user_key.pem",1679,100
CONNECT
<Input the client.pem data, the size is 1679 bytes>
+QSECWRITE: 1679,335f

OK

//Configure CA certificate.
AT+QSSLCFG="cacert",2,"RAM:cacert.pem"
OK

//Configure CC certificate.
AT+QSSLCFG="clientcert",2,"RAM:client.pem"
OK

//Configure CK certificate.
AT+QSSLCFG="clientkey",2,"RAM:user_key.pem"
OK

//Configure SSL parameters.
AT+QSSLCFG="seclevel”,2,2                //SSL authentication mode: server authentication
```

```
OK
AT+QSSLCFG="sslversion”,2,4                //SSL authentication version
OK
AT+QSSLCFG="ciphersuite”,2,”0xFFFF”   //Cipher suite
OK
AT+Qsslcfg="ignorertctime",1                //Ignore the time of authentication.
OK

//Start MQTT SSL connection.
AT+QMTOPEN=0, "a1zgnxur10j8ux.iot.us-east-1.amazonaws.com",”8883”
OK

+QMTOPEN: 0,0

//Connect a client to MQTT server.
AT+QMTCONN=0,"M26_0206"
OK

+QMTCONN: 0,0,0

//Subscribe to topics.
AT+QMTSUB=0,1,"$aws/things/M26_0206/shadow/update/accepted",1
OK

+QMTSUB: 0,1,0,1

//Publish messages.
AT+QMTPUB=0,1,1,0,"$aws/things/M26_0206/shadow/update/accepted"
>This is publish data from client
OK

+QMTPUB: 0,1,0

//If a client subscribes to a topic named “$aws/things/M26_0206/shadow/update/accepted” and other
devices publish the same topic to the server, the module will report the following information.
+QMTRECV: 0,1,"$aws/things/M26_0206/shadow/update/accepted",This is publish data from
client

//Disconnect a client from MQTT server.
AT+QMTDISC=0
OK

+QMTDISC: 0,0
```

6 Appendix

6.1. References

Table 3: Related Document

SN	Document Name	Remarks
[1]	MQTT V3.1 Protocol Specification	MQTT protocol specification version 3.1

Table 4: Terms and Abbreviations

Abbreviation	Description
ACK	Acknowledgement
MQTT	Message Queuing Telemetry Transport
QoS	Quality of Service
RAM	Random Access Memory
SSL	Secure Sockets Layer
TCP	Transmission Control Protocol
URC	Unsolicited Result Code

6.2. Summary of Error Codes

<err> indicates an error related to mobile equipment or network. The details about <err> are described in the following table.

Table 5: Summary of Error Codes

Code of <err>	Meaning
8503	MQTT link error
8504	MQTT illegal packet
8505	MQTT illegal character
8506	MQTT illegal UTF8
8507	MQTT invalid parameter
8508	MQTT packet too large
8509	MQTT buffer overflow
8510	MQTT out of memory
8511	MQTT memory error
8600	MQTT unknown error
8512	MQTT maximum quantity of transmitting packet greater than that of in-flight windows