EC2x&EG9x&EM05 MQTT Application Note

LTE Module Series

Rev. EC2x&EG9x&EM05_MQTT_Application_Note_V1.1

Date: 2018-12-12

Status: Released

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

Quectel Wireless Solutions Co., Ltd.
7th Floor, Hongye Building, No.1801 Hongmei Road, Xuhui District, Shanghai 200233, China
Tel: +86 21 5108 6236
Email: info@quectel.com

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

For technical support, or to report documentation errors, please visit:
http://www.quectel.com/support/technical.htm
Or email to: support@quectel.com

GENERAL NOTES

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2018-08-21	Chavis CHEN	Initial
1.1	2018-12-12	Slark WANG	Added Write Command AT+QMTCFG="qmtping" and its related information

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 EC2x&EG9x&EM05 modules through AT commands.

This document is applicable to following Quectel modules.

- EC2x (including EC25, EC21, EC20 R2.0 and EC20 R2.1)
- EG9x (including EG91 and EG95)
- EM05

2 MQTT Data Interaction

This chapter gives the data interaction mechanism of MQTT function.

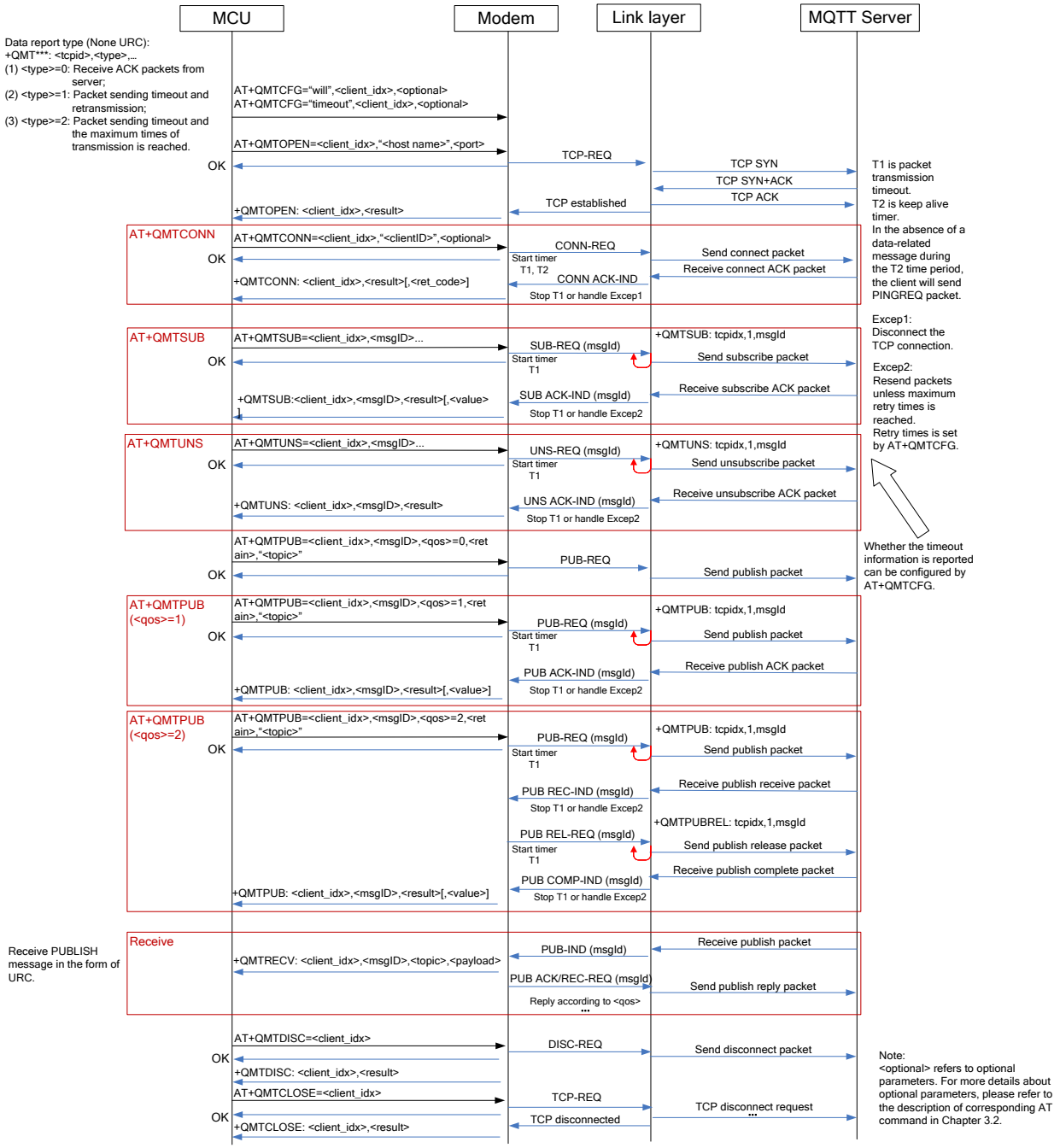

Figure 1: MQTT Data Interaction Diagram

3 MQTT Related 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 MQTT Related AT Commands

3.2.1. AT+QMTCFG Configure Optional 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: "version",(0-5),(3,4)** **+QMTCFG: "pdpcid",(0-5),(1-16)** **+QMTCFG: "ssl",(0-5),(0,1),(0-5)** **+QMTCFG: "keepalive",(0-5),(0-3600)** **+QMTCFG: "session",(0-5),(0,1)** **+QMTCFG: "timeout",(0-5),(1-60),(1-10),(0,1)** **+QMTCFG: "will",(0-5),(0,1),(0-2),(0,1),"willtopic","willmessage"**

	+QMTCFG: "recv/mode",(0-5),(0,1),(0,1) **+QMTCFG: "aliauth",(0-5),"productkey","devicename","devicesecret"** **+QMTCFG: "qmtping",(0-5),(5-60)** **OK**
Write Command **Configure the MQTT protocol version** **AT+QMTCFG="version",<client_idx>[,<vsn>]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<vsn>** is omitted, query the MQTT protocol version. Response **+QMTCFG: "version",<vsn>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure the PDP to be used by the MQTT client **AT+QMTCFG="pdpcid",<client_idx>[,<cid>]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<cid>** is omitted, query the PDP to be used by the MQTT client. Response **+QMTCFG: "pdpcid",<cid>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure Will information **AT+QMTCFG="will",<client_idx>[,<will_fg>[,<will_qos>,<will_retain>,"<will_topic>","<will_msg>"]]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<will_fg>**, **<will_qos>**, **<will_retain>,"<will_topic>"** and **<will_msg>** are omitted, query the Will information. Response **+QMTCFG: "will",<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",<client_idx>[,<pkt_timeout>[,<retry_times>][,<timeout_notice>]]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<pkt_timeout>**, **<retry_times>** and **<timeout_notice>** are omitted, query the timeout of message delivery. Response **+QMTCFG: "timeout",<pkt_timeout>,<retry_times>,<timeout_notice>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure the session type **AT+QMTCFG="session",<client_idx>[,<clean_session>]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<clean_session>** is omitted, query the session type. Response **+QMTCFG: "session",<clean_session>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure the keep-alive time **AT+QMTCFG="keepalive",<client_idx>[,<keep-alive time>]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<keep-alive time>** is omitted, query the keep-alive time. Response **+QMTCFG: "keepalive",<keep-alive time>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**

Write Command Configure the MQTT SSL mode and SSL context index **AT+QMTCFG="ssl",<client_idx>[,<sslenable>[,<sslctx_idx>]]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<sslenable>** and **<sslctx_idx>** are omitted, query the MQTT SSL mode and SSL context index. Response **+QMTCFG: "ssl",<sslenable>[,<sslctx_idx>]** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure receiving mode when data is received from server **AT+QMTCFG="recv/mode",<client_idx>,<msg_recv_mode>[,<msg_len_enable>]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<msg_recv_mode>** and **<msg_len_enable>** are omitted, query the MQTT message receiving mode. Response **+QMTCFG: "recv/mode",<msg_recv_mode>[,<msg_len_enable>]** **OK**
Write Command Configure Alibaba device information for Alibaba Cloud **AT+QMTCFG="aliauth",<client_idx>[,"<product_key>","<device_name>","<device_secret>"]**	If MQTT connection is not established, the response is: **OK** Else the response is: **ERROR** If **<product_key>**, **<device_name>** and **<device_secret>** are omitted, query the device information. Response **[+QMTCFG: "aliauth","<product_key>","<device_name>","<device_secret>"]** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Write Command Configure the interval for "**+QMTSTAT**" report after "**+QMTPING**" is reported	If MQTT connection is not established, the response is: **OK** Else the response is:

AT+QMTCFG="qmtping"[,<qmtping_interval>]	**ERROR** If **<qmtping_interval>** is omitted, query the interval. Response **+QMTCFG: "qmtping",<qmtping_interval>** **OK** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	300ms

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<vsn>	Integer type. MQTT protocol version 3 MQTT protocol v3.1 4 MQTT protocol v3.11
<cid>	Integer type. The PDP to be used by the MQTT client. The range is 1-16. The default value is 1.
<will_fg>	Integer type. Configure the Will flag 0 Ignore the Will flag configuration 1 Require the Will flag configuration
<will_qos>	Integer type. Quality of service for message delivery 0 At most once 1 At least once 2 Exactly once
<will_retain>	Integer type. 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>	String type. Will topic string
<will_msg>	String type. 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>	Integer type. Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.
<retry_times>	Integer type. Retry times when packet delivery times out. The range is 0-10. The default value is 3.
<timeout_notice>	Integer type. 0 Not report timeout message when transmitting packet 1 Report timeout message when transmitting packet

<clean_session>	Integer type. 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>	Integer type. 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>	Integer type. Configure the MQTT SSL mode 0 Use normal TCP connection for MQTT 1 Use SSL TCP secure connection for MQTT
<sslctx_idx>	Integer type. SSL context index. The range is 0-5.
<msg_recv_mode>	Integer type. Configure the MQTT message receiving mode. 0 MQTT message received from server will be contained in URC. 1 MQTT message received from server will not be contained in URC.
<msg_len_enable>	0 Length of MQTT message received from server will not be contained in URC. 1 Length of MQTT message received from server will be contained in URC.
<product_key>	String type. Product key issued by Alibaba Cloud
<device_name>	String type. Device name issued by Alibaba Cloud
<device_secret>	String type. Device secret key issued by Alibaba Cloud
<qmtping_interval>	Integer type. The interval for “**+QMTSTAT**” report after “**+QMTPING**” is reported . The default value is 5. Unit: second.

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 the server supports the operation.
3. If MQTT connection is configured to SSL mode, **<sslctx_idx>** 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.
5. **AT+QMTCFG=“aliauth”** command is only used for Alibaba Cloud. If it is configured, the parameters **<username>** and **<password>** in command **AT+QMTCONN** can be omitted.

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 **<client_idx>s**),**"<host_name>"**,(list of supported **<port>s**) **OK**
Read Command **AT+QMTOPEN?**	Response **[+QMTOPEN: <client_idx>,"<host_name>",<port>]** **OK** or **OK**
Write Command **AT+QMTOPEN=<client_idx>,"<host_name>",<port>**	Response **OK** **+QMTOPEN: <client_idx>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	120s, determined by network

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<host_name>	String type. The address of the server. It could be an IP address or a domain name. The maximum size is 100 bytes.
<port>	Integer type. The port of the server. The range is 1-65535.
<result>	Integer type. Result of the command execution -1 Failed to open network 0 Network opened successfully 1 Wrong parameter 2 MQTT identifier is occupied 3 Failed to activate PDP 4 Failed to parse domain name 5 Network connection 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 **<client_idx>**s) **OK**
Write Command **AT+QMTCLOSE=<client_idx>**	Response **OK** **+QMTCLOSE: <client_idx>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	30s

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<result>	Integer type. Result of the command execution -1 Failed to close network 0 Network closed 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 **<client_idx>**s),**“<clientID>”[,“<username>”[,“<password>”]]** **OK**
Read Command **AT+QMTCONN?**	Response **+QMTCONN: <client_idx>,<state>** **OK**

	or **OK**
Write Command **AT+QMTCONN=<client_idx>,"<clientID>"[,"<username>"[,"<password>"]]**	Response **OK** **+QMTCONN: <client_idx>,<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

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<clientID>	String type. The client identifier string.
<username>	String type. User name of the client. It can be used for authentication.
<password>	String type. Password corresponding to the user name of the client. It can be used for authentication.
<result>	Integer type. Result of the command execution 0 Packet sent successfully and ACK received from server 1 Packet retransmission 2 Failed to send packet
<state>	Integer type. MQTT connection state 1 MQTT is initial 2 MQTT is connecting 3 MQTT is connected 4 MQTT is disconnecting
<ret_code>	Integer type. Connection status 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
<pkt_timeout>	Integer type. Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.

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 **<client_idx>**s) **OK**
Write Command **AT+QMTDISC=<client_idx>**	Response **OK** **+QMTDISC: <client_idx>,<result>** If there is an error related to ME functionality: **+CME ERROR: <err>**
Maximum Response Time	30s

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.	
<result>	Integer type. Result of the command execution	
	-1	Failed to close connection
	0	Connection closed 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 **<client_idx>s**),(list of supported **<msgID>**s),**“<topic>”,**(list of supported **<qos>s**) **OK**
Write Command **AT+QMTSUB=<client_idx>,<msgID>,“<topic1>”,<qos1>[,“<topic2>”,<q**	Response **OK**

os2>…]	**+QMTSUB: <client_idx>,<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

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<msgID>	Integer type. Message identifier of packet. The range is 1-65535.
<topic>	String type. Topic that the client wants to subscribe to or unsubscribe from
<qos>	Integer type. The QoS level at which the client wants to publish the messages. 0 At most once 1 At least once 2 Exactly once
<result>	Integer type. Result of the command execution 0 Sent packet successfully and received ACK from server 1 Packet retransmission 2 Failed to send packet
<value>	Integer type. 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.
<pkt_timeout>	Integer type. Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.
<retry_times>	Integer type. Retry times when packet delivery times out. The range is 0-10. The default value is 3.

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 “inflight” 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 **<client_idx>**s),(list of supported **<msgID>**s),**"<topic>"** **OK**
Write Command **AT+QMTUNS=<client_idx>,<msgID>,"<topic1>"[,"<topic2>"…]**	Response **OK** **+QMTUNS: <client_idx>,<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

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<msgID>	Integer type. Message identifier of packet. The range is 1-65535.
<topic>	String type. Topic that the client wants to subscribe to or unsubscribe from.
<result>	Integer type. Result of the command execution 0 Sent packet successfully and received ACK from server 1 Packet retransmission 2 Failed to send packet
<value>	Integer type. 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.
<pkt_timeout>	Integer type. Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.
<retry_times>	Integer type. Retry times when packet delivery times out. The range is 0-10. The default value is 3.

3.2.8. AT+QMTPUBEX Publish Messages

The command is used to publish messages with fixed length 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+QMTPUBEX Publish Messages	
Test Command **AT+QMTPUBEX=?**	Response **+QMTPUBEX:** (list of supported **<client_idx>**s),(list of supported **<msgID>**s),(list of supported **<qos>**s),(list of supported **<retain>**s),**"<topic>",<msg_length>** **OK**
Write Command **AT+QMTPUBEX=<client_idx>,<msgID>,<qos>,<retain>,"<topic>",<msg_length>** After "**>**" is responded, input the data to be sent. When the actual size of data is greater than **<msg_length>**, the first **<msg_length>** byte(s) data will be sent out.	Response **OK** **+QMTPUBEX: <client_idx>,<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

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<msgID>	Integer type. Message identifier of packet. The range is 0-65535. It will be 0 only when **<qos>**=0.
<qos>	Integer type. The QoS level at which the client wants to publish the messages. 0 At most once 1 At least once 2 Exactly once
<retain>	Integer type. Whether or not the server will retain the message after it has been delivered to the current subscribers. 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>	String type. Topic that needs to be published
<msg_length>	Integer type. Length of message to be published
<result>	Integer type. Result of the command execution 0 Packet sent successfully and ACK received from server (message that published when **<qos>**=0 does not require ACK) 1 Packet retransmission 2 Failed to send packet
<value>	Integer type.

	If **<result>** is 1, it means the times of packet retransmission. If **<result>** is 0 or 2, it will not be presented.
<pkt_timeout>	Integer type. Timeout of the packet delivery. The range is 1-60. The default value is 5. Unit: second.
<retry_times>	Integer type. Retry times when packet delivery times out. The range is 0-10. The default value is 3.

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 inflight windows (5); otherwise, **ERROR** will be returned.
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 1500 bytes at a time.
3. PUBLISH messages can be sent either from a publisher to the server, or from the server to a subscriber. When a server publishes messages to a subscriber, the following URC will be returned to notify the host to read the received data that is reported from MQTT server:
   +QMTRECV: <client_idx>,<msgID>,"<topic>",[< payload_length>,]"<payload>"
   For more details about the URC description, please refer to ***Chapter 4.2***.

3.2.9. AT+QMTRECV Read Messages from Buffers

The command is used to read messages from storage buffer where the messages are stored in when they are reported by the server.

AT+QMTRECV Read Messages from Buffers	
Test Command **AT+QMTRECV=?**	Response **OK**
Read Command **AT+QMTRECV?**	Response **+QMTRECV: <client_idx>,<store_status_0>,<store_status_1>,<store_status_2>,<store_status_3>,<store_status_4>** **OK** If there is no MQTT connection, response: **OK**
Write Command **AT+QMTRECV=<client_idx>[,<recv_id>]**	Response (List of **+QMTRECV: <client_idx>,<msg_id>,"<topic>",[<payload_len>,]"<payload>"**) **OK**

	If there is no message received, response: **OK** If there is no MQTT connection, response: **ERROR**

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<store_status>	Integer type. Indicate whether there is a message stored in the buffer. 0 means no, and 1 means yes. The maximum quantity of message that can be stored in the buffer is 5. Therefore, URC reports maximally 5 messages simultaneously.
<recv_id>	Integer type. Indicate the serial number of every single message received. The range is 0-4.
<msg_id>	Integer type. Message identifier of packet. The range is 0-65535. It will be 0 only when **<qos>**=0.
<topic>	String type. Topic that needs to be published.
<payload_len>	Integer type. The length of payload.
<payload>	String type. The payload that relates to the topic name.

4 MQTT Related URCs

This chapter gives MQTT related URCs and their descriptions.

Table 2: MQTT Related URCs

SN	URC Format	Description
[1]	**+QMTSTAT: <client_idx>,<err_code>**	When the state of MQTT link layer is changed, the client will close the MQTT connection and report the URC.
[2]	**+QMTRECV: <client_idx>,<msgID>,"<topic>",[<payload_len>,]"<payload>"**	Reported when the client has received the packet data from MQTT server.
[3]	**+QMTRECV: <client_idx>,<recv_id>**	Reported when the message that received from MQTT server has been stored in buffer.
[4]	**+QMTPING: <client_idx>,<result>**	When the state of MQTT link layer is changed, the client will close the MQTT connection and report the URC.

4.1. "+QMTSTAT" URC to Indicate State Change in MQTT Link Layer

The URC begins with "**+QMTSTAT:**". It will be reported when there is a change in the state of MQTT link layer.

"+QMTSTAT" URC to Indicate State Change in MQTT Link Layer	
+QMTSTAT: <client_idx>,<err_code>	When the state of MQTT link layer is changed, the client will close the MQTT connection and report the URC.

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<err_code>	Integer type. An error code. Please refer to the table below for details.

Table 3: Error Codes of the URC

<err_code>	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 whether the inputted user name and password are correct. 2. Make sure the client ID is not used. 3. Reopen MQTT connection and try to send CONNECT packet to server again.
4	Receiving CONNECK packet timed out or failed.	1. Check whether the inputted user name and password are correct. 2. Make sure the client ID is not used. 3. Reopen MQTT connection and try to send CONNECT packet to server again.
5	The client sends DISCONNECT packet to sever and the server is initiative to close MQTT connection.	This is a normal process.
6	The 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 an 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.	

4.2. "+QMTRECV" URC to Notify the Host to Read MQTT Packet Data

The URC begins with "**+QMTRECV:**". It is mainly used to notify the host to read the received MQTT packet data that is reported from MQTT server.

"+QMTRECV" URC to Notify the Host to Read MQTT Packet Data	
+QMTRECV: <client_idx>,<msgID>,<topic>,[<payload_len>,]"<payload>"	Notify the host to read the received data that is reported from MQTT server.
+QMTRECV: <client_idx>,<recv_id>	Reported when the message that received from MQTT server has been stored in buffer.

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<msgID>	Integer type. The message identifier of packet
<topic>	String type. The topic that received from MQTT server
<payload_len>	Integer type. The length of payload.
<payload>	String type. The payload that relates to the topic name
<recv_id>	Integer type. Indicate the serial number of every single message received. The range is 0-4.

4.3. "+QMTPING" URC to Indicate PING State of Keep-alive in MQTT

The URC begins with "**+QMTPING:**". It will be reported when server does not receive a message from the client within 1.5 times of the keep-alive time period and it will disconnect the client as if the client has sent a DISCONNECT message.

"+QMTPING" URC to Indicate PING State of Keep-alive in MQTT	
+QMTPING: <client_idx>,<result>	When the state of MQTT link layer is changed, the client will close the MQTT connection and report the URC.

Parameter

<client_idx>	Integer type. MQTT client identifier. The range is 0-5.
<result>	Integer type. Result of PING state.
.	1 FAIL

5 Examples

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

5.1. Example of MQTT Operation without SSL

```
//Configure receiving mode.
AT+QMTCFG="recv/mode",0,0,1
OK

//Configure Alibaba device information for Alibaba cloud.
AT+QMTCFG="aliauth",0,"oyjtmPl5a5j","MQTT_TEST","wN9Y6pZSlly7Exa5qVzcmigEGO4kAazZ"
OK
AT+QMTOPEN=?
+QMTOPEN: (0-5),"hostname",(1-65536)
OK

//Open a network for MQTT client.
AT+QMTOPEN=0,"iot-as-mqtt.cn-shanghai.aliyuncs.com",1883
OK

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

AT+QMTOPEN?
+QMTOPEN: 0,"iot-as-mqtt.cn-shanghai.aliyuncs.com",1883

OK
AT+QMTCONN=?
+QMTCONN: (0-5),"clientid","username","password"
OK

//Connect a client to MQTT server.
//If Alibaba Cloud is connected, customers can use AT+QMTCFG="aliauth" command to configure the device information in advance, and do not need to provide username/password here anymore.
AT+QMTCONN=0,"clientExample"
OK

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

```
AT+QMTSUB=?
+QMTSUB: (0-5), <msgid>,list of [“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”,36,“This is the payload related to topic”

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

+QMTUNS: 0,2,0

AT+QMTPUBEX=?
+QMTPUBEX: (0-5),<msgid>,(0-2),(0,1),“topic”,“length”
OK

//Publish messages. After receiving ‘>’, input data “This is test data, hello MQTT.” and then send it. The maximum length of the data is 1500 bytes and the data that beyond 1500 bytes will be omitted.
AT+QMTPUBEX=0,0,0,0,“topic/pub”,30
> This is test data, hello MQTT.
OK

+QMTPUBEX: 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”,30,“This is test data, hello MQTT.”

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

```
+QMTDISC: 0,0                    //Connection closed successfully.
```

5.2. Example of MQTT Operation with SSL

```
//Configure receiving mode.
AT+QMTCFG="recv/mode",0,0,1
OK

//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+QFUPL="RAM:cacert.pem",1758,100
CONNECT
<Input the cacert.pem data, the size is 1758 bytes>
+QFUPL: 1758,384a

OK

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

OK

//If SSL authentication mode is "server authentication", store CK certificate to RAM.
AT+QFUPL="RAM:user_key.pem",1679,100
CONNECT
<Input the client.pem data, the size is 1679 bytes>
+QFUPL: 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="ignorelocaltime",2,1           //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 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+QMTPUBEX=0,1,1,0,"$aws/things/M26_0206/shadow/update/accepted",32
>This is publish data from client
OK

+QMTPUBEX: 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",32,"This is publish data from
```

```
client”

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

+QMTDISC: 0,0
```

6 Appendix A References

Table 4: Related Documents

SN	Document Name	Remarks
[1]	MQTT V3.1 Protocol Specification	MQTT protocol specification version 3.1
[2]	MQTT V3.1.1 Protocol Specification	MQTT protocol specification version 3.1.1

Table 5: 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