GSM SSL

Application Note

GSM/GPRS/GNSS Module Series

Rev. GSM_SSL_Application_Note_V3.3

Date: 2020-04-01

Status: Released

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.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang 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

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

Revision History

Version	Date	Author	Description
3.0	2013-10-24	Andy CHEN	Initial
3.1	2015-04-08	Andy CHEN	Added applicable modules
3.2	2018-11-09	Oven TAO/ Sandy YE	1. Updated the applicable modules of this document. 2. Added example about transparent mode of SSL function (Chapter 3.4). 3. Updated the descriptions of AT+QSSLOPEN and AT+QSSLCLOSE.
3.3	2020-04-01	Jaryoung LI/ Samuel LIN	Updated the description of the server root CA certificate (Chapter 2.2.7~2.2.9 and 3).

Contents

Table Index

1 Introduction

This document describes how to use the SSL function of Quectel GSM modules.

In some cases, in order to ensure communication privacy, the communication between the server and the client should be in an encrypted way so that data can be prevented from eavesdropping, tampering, or forging during the communication process. The SSL function meets these demands.

This document is applicable to all the Quectel GSM modules.

1.1. SSL Version and Cipher Suite

The following SSL versions are supported.

Table 1: SSL Versions

SSL Version
SSL3.0
TLS1.0
TLS1.1
TLS1.2

The following table shows SSL cipher suites supported by Quectel GSM modules. For detailed description of cipher suites, please refer to *RFC 2246-The TLS Protocol Version 1.0*.

Table 2: Supported SSL Cipher Suites

Cipher Suite Code	Cipher Suite Name
0X0035	TLS_RSA_WITH_AES_256_CBC_SHA
0X0005	TLS_RSA_WITH_RC4_128_SHA

0X0004	TLS_RSA_WITH_RC4_128_MD5
0X000A	TLS_RSA_WITH_3DES_EDE_CBC_SHA
0X002F	TLS_RSA_WITH_AES_128_CBC_SHA
0X003D	TLS_RSA_WITH_AES_256_CBC_SHA256

1.2. The Process of Using SSL Function

Step 1: Install certificate and key to file system by **AT+QSECWRITE**. **AT+QSECDEL** deletes the certificate and key, and **AT+QSECREAD** checks the checksum of certificate and key.

Step 2: Configure APN, username and password of the context by **AT+QIFGCNT** and **AT+QICSGP**. And start TCP/IP task by **AT+QIREGAPP**.

Step 3: Activate GPRS PDP context by **AT+QIACT**. After the context is activated, query the local IP address by **AT+QILOCIP**.

Step 4: Configure SSL version, cipher suite, server authentication, client authentication, server root CA certificate, client certificate and client key by **AT+QSSLCFG**.

Step 5: Establish an SSL connection by **AT+QSSLOPEN**. If the connection is successful, the response will be "**CONNECT**" or "**+QSSLOPEN: <ssid>,<connectcode>**".

Step 6: In non-transparent mode, data is sent by **AT+QSSLSEND**. If the module receives data from network, it will report an URC: "**+QSSLURC: "recv",<cid>,<ssid>**", and can read the received data by **AT+QSSLRECV**.

In transparent mode, data transmission and receiving are directly input and output from the serial COM port. **+++** or DTR can be used to exit from the data mode and enter command mode. If the connection is abnormal, module will automatically exit from the data mode.

Step 7: When data transmission is accomplished, close the SSL connection by **AT+QSSLCLOSE**.

Step 8: Deactivate GPRS PDP context by **AT+QIDEACT**.

1.3. SSL Function Coexists with Normal TCP/IP Session

SSL connection can coexist with normal TCP connection, which means one or several SSL and normal TCP connections can be established at the same time.

In the same foreground context, please establish SSL and normal TCP connections with different socket indexes. For example, establish a normal TCP connection with socket index one, and establish an SSL connection with socket index three.

The following steps show how SSL function works together with normal TCP session.

Step 1: Configure APN, username and password of the context by **AT+QIFGCNT** and **AT+QICSGP**.

Step 2: Enable multiple TCP/IP session by **AT+QIMUX=1**.

Step 3: Start TCP/IP task by **AT+QIREGAPP**.

Step 4: Activate GPRS PDP context by **AT+QIACT**. After the context is activated, query the local IP address by **AT+QILOCIP**.

Step 5: Configure the method of handling received TCP/IP data with buffer mode by **AT+QINDI=1**. Execute **AT+QIOPEN** to establish a normal TCP connection, and specify **<index>** as 1. After the normal TCP connection is established successfully, data can be sent by **AT+QISEND** and received by **AT+QIRD**. If It is necessary to close the connection, execute **AT+QICLOSE**.

Step 6: Establish an SSL connection by **AT+QSSLOPEN**, and specify **<ssid>** as 3. After the connection is established successfully, send data by **AT+QSSLSEND**. When module receives data from the peer, the URC "**+QSSLURC: "recv",<cid>,<ssid>"** will notify the host to read data. And host can execute the **AT+QSSLRECV** to read data. When data transmission is accomplished, close the SSL connection by **AT+QSSLCLOSE**.

Step 7: Deactivate GPRS PDP context by **AT+QIDEACT**.

NOTE

For detailed information of these AT commands, please refer to ***document [3]***.

2 Description of SSL AT Commands

2.1. AT Command Syntax

2.1.1. Definitions

- **<CR>** Carriage return character.
- **<LF>** Line feed character.
- **<...>** Parameter name. Angle brackets do not appear on command line.
  [...] Optional parameter of a command or an optional part of TA information response. Square brackets do not appear on command line. When an optional parameter is not given, the new value equals its previous value or its default setting, unless otherwise specified.
- **Underline** Default setting of a parameter.

2.1.2. AT Command Syntax

The **AT** or **at** prefix must be added at the beginning of each command line. Entering **<CR>** will terminate a command line. Commands are usually followed by a response that includes **<CR><LF><response><CR><LF>**. Throughout this document, only the response **<response>** will be presented, **<CR><LF>** are omitted intentionally.

Table 3: Type of AT Commands and Responses

Test Command	**AT+<cmd>=?**	This command returns the list of parameters and value ranges set by the corresponding Write Command or internal processes.
Read Command	**AT+<cmd>?**	This command returns the currently set value of the parameter or parameters.
Write Command	**AT+<cmd>=<p1>[,<p2>[,<p3>[...]]]**	This command sets the user-definable parameter values.
Execution Command	**AT+<cmd>**	This command reads non-variable parameters affected by internal processes in the module.

2.2. Description of AT Commands

2.2.1. AT+QSSLCFG Configure Parameters of an SSL Context

The command configures the SSL version, cipher suites, security level, server root CA certificate, client certificate, client key, whether to ignore RTC time, HTTP/HTTPS, and SMTP/SMTPS. These parameters will be used in the handshake procedure.

CTX refers to the abbreviation of the SSL (Secure Socket Layer) context, and **<ctxindex>** is the index of the SSL context. Quectel GSM module supports 6 SSL contexts at most. Several SSL connections can be established on the basis of one SSL context. The settings such as the SSL version and the cipher suite are stored in the SSL context, and will be applied to the new SSL connection associated with the SSL context.

AT+QSSLCFG Configure Parameters of an SSL Context	
Test Command **AT+QSSLCFG=?**	Response **+QSSLCFG: "type",(**range of supported **<ctxindex>**s**),"value"** **OK**
Write Command Query the setting of the context: **AT+QSSLCFG="ctxindex",<ctxindex>**	Response **+QSSLCFG: <ctxindex>,<sslversion>,<seclevel>,<ciphersuite>,<cacertname>,<clientcertname>,<clientkeyname>** **OK** Or **ERROR**
Write Command Configure the SSL version for the specified SSL context: **AT+QSSLCFG="sslversion",<ctxindex>[,<sslversion>]**	Response If **<sslversion>** is omitted, query the SSL version for the specified SSL context: **+QSSLCFG: "sslversion",<sslversion>** **OK** If **<sslversion>** is specified, set the SSL version for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the SSL cipher suites for the specified SSL context: **AT+QSSLCFG="ciphersuite",<ctxind**	Response If **<ciphersuite>** is omitted, query the SSL cipher suites for the specified SSL context: **+QSSLCFG: "ciphersuite",<ciphersuite>**

ex>[,<ciphersuite>]	 **OK** If **<ciphersuite>** is specified, set the SSL cipher suite for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the authentication mode for the specified SSL context: **AT+QSSLCFG="seclevel",<ctxindex>[,<seclevel>]**	Response If **<seclevel>** is omitted, query the authentication mode for the specified SSL context: **+QSSLCFG: "seclevel",<seclevel>** **OK** If **<seclevel>** is specified, set the authentication mode for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of trusted server root CA certificate for the specified SSL context: **AT+QSSLCFG="cacert",<ctxindex>[,<cacertname>]**	Response If **<cacertname>** is omitted, query the path of trusted server root CA certificate for the specified SSL context: **+QSSLCFG: "cacert",<cacertname>** **OK** If **<cacertname>** is specified, set the path of trusted server root CA certificate for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of client certificate for the specified SSL context: **AT+QSSLCFG="clientcert",<ctxindex>[,<clientcertname>]**	Response If **<clientcertname>** is omitted, query the path of client certificate for the specified SSL context: **+QSSLCFG: "clientcert",<clientcertname>** **OK** If **<clientcertname>** is specified, set the path of client certificate for the specified SSL context: **OK** Or **ERROR**

Write Command Configure the path of client private key for the specified SSL context: **AT+QSSLCFG="clientkey",<ctxindex>[,<clientkeyname>]**	Response If **<clientkeyname>** is omitted, query the path of client private key for the specified SSL context: **+QSSLCFG: "clientkey",<clientkeyname>** **OK** If **<clientkeyname>** is specified, set the path of client private key for the specified SSL context: **OK** Or **ERROR**
Write Command Configure whether to ignore the RTC time for the specified SSL context: **AT+QSSLCFG="ignorertctime"[,<ignorertctime>]**	Response If **<ignorertctime>** is omitted, query whether the RTC time is ignored for the specified SSL context: **+QSSLCFG: "ignorertctime",<ignorertctime>** **OK** If **<ignorertctime>** is specified, set whether to ignore the RTC time for the specified SSL context: **OK** Or **ERROR**
Write Command Enable/disable HTTPS function: **AT+QSSLCFG="https"[,<httpsenable>]**	Response If **<httpsenable>** is omitted, query whether to enable/disable HTTPS function: **+QSSLCFG: "https",<httpsenable>** **OK** If **<httpsenable>** is specified, set whether to enable/disable HTTPS function: **OK** Or **ERROR**
Write Command Configure the SSL context index for HTTPS: **AT+QSSLCFG="httpsctxi"[,<httpsctxindex>]**	Response If **<httpsctxindex>** is omitted, query the SSL context index for HTTPS: **+QSSLCFG: "httpsctxi",<httpsctxindex>** **OK** If **<httpsctxindex>** is specified, set the SSL context for

	HTTPS: **OK** Or **ERROR**
Write Command Configure the type of SMTP/SMTPS: **AT+QSSLCFG="smtpstyle"[,<smtpstyle>]**	Response If **<smtpstyle>** is omitted, query the type of SMTP/SMTPS: **+QSSLCFG: "smtpstyle",<smtpstyle>** **OK** If **<smtpstyle>** is specified, set the type of SMTP/SMTPS: **OK** Or **ERROR**
Write Command Configure the SSL context index for SMTPS: **AT+QSSLCFG="smtpsctxi"[,<smtpsctxindex>]**	Response If **<smtpsctxindex>** is omitted, query the SSL context index for SMTPS: **+QSSLCFG: "smtpsctxi",<smtpsctxindex>** **OK** If **<smtpsctxindex>** is specified, set the SSL context index for SMTPS: **OK** Or **ERROR**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<ctxindex>	SSL context index. Range: 0–5.	
<sslversion>	Numeric type. SSL Version.	
	0	SSL3.0
	1	TLS1.0
	2	TLS1.1
	3	TLS1.2
	4	All
<ciphersuite>	Numeric type. SSL cipher suites.	
	0X0035	TLS_RSA_WITH_AES_256_CBC_SHA
	0X002F	TLS_RSA_WITH_AES_128_CBC_SHA
	0X0005	TLS_RSA_WITH_RC4_128_SHA
	0X0004	TLS_RSA_WITH_RC4_128_MD5

0X000A TLS_RSA_WITH_3DES_EDE_CBC_SHA
0X003D TLS_RSA_WITH_AES_256_CBC_SHA256
0XFFFF All

<seclevel> Numeric type. Authentication mode.
0 No authentication
1 Manage server authentication
2 Manage server and client authentication if requested by the remote server

<cacertname> String type. The path of the trusted server root CA certificate.

<clientcertname> String type. The path of the client certificate.

<clientkeyname> String type. The path of the client private key.

<ignorertctime> Numeric type. Configure whether to ignore the RTC time.
0 Not ignore the RTC time
1 Ignore the RTC time

<httpsenable> Enable/disable the HTTPS function.
0 Disable HTTPS
1 Enable HTTPS

<httpsctxindex> SSL context for HTTPS.
The parameter is the index of SSL context. If the host does not configure it, its value is -1. Range: 0–5.

<smtpstyle> Type of SMTP/SMTPS.
0 No SSL
1 SSL
2 STARTTLS

<smtpsctxindex> SSL context for SMTPS.
The parameter is the index of SSL context. If the host does not configure it, its value is -1. Range: 0–5.

NOTES

1. The format of **<cacertname>**, **<clientcertname>** and **<clientkeyname>** can be as follows:
   "RAM:filename" File is uploaded to RAM
   "NVRAM:filename" File is uploaded to NVRAM. Support server root CA certificate, one client certificate and one client private key. The filename of server root CA certificate must be CA0, the filename of client certificate must be CC0, and the filename of client private key must be CK0.
   CA0 Identify a server root CA certificate
   CC0 Identify a client certificate
   CK0 Identify a client key
2. If no authentication is set, then no security data is needed. If server authentication has been set, server CA certificate needs to be configured. If server and client authentication has been set, client certificate, server CA certificate and client private key need to be configured.

2.2.2. AT+QSSLOPEN Open an SSL Socket to Connect a Remote Server

The command opens an SSL socket to connect a remote server. During the negotiation between the module and the peer, **AT+QSSLCFG** will be used for parameter configuration in handshake procedure. After shaking hands with the peer successfully, the module can send or receive data via this SSL connection. Also the module can establish several SSL connections based on one SSL context.

The host can configure a timeout for **AT+QSSLOPEN**. If module does not finish establishing an SSL connection until timeout period has expired, the URC "**+QSSLOPEN: <ssid>,<connectcode>**" will be reported. If the host does not configure timeout value, the default value of timeout is 90 seconds.

AT+QSSLOPEN Open an SSL Socket to Connect a Remote Server	
Test Command **AT+QSSLOPEN=?**	Response **+QSSLOPEN: <ssid>,<ctxindex>,<ipaddr/domainname>,<port>,<connectmode>[,<timeout>]** **OK**
Read Command **AT+QSSLOPEN?**	Response **OK**
Write Command **AT+QSSLOPEN=<ssid>,<ctxindex>,<ipaddr/domainname>,<port>,<connectmode>[,<timeout>]**	Response If **<connectmode>** is transparent mode and the SSL connection is established successfully: **CONNECT** If there is any error: **ERROR** If **<connectmode>** is not transparent mode: **+QSSLOPEN: <ssid>,<connectcode>** If the SSL connection is established unsuccessfully and the value of **<connectcode>** is not 0: **+QSSLOPEN: <ssid>,<connectcode>**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<ssid>	Numeric type. Secure socket identifier. Range: 0–5.
<ctxindex>	Numeric type. SSL context index. Range: 0–5.
<ipaddr/domainname>	String type. IP address of SSL server or URL.
<port>	Port of remote server.
<connectmode>	Transferring mode of SSL connection.

	0	Not transparent mode
	1	Transparent mode
<timeout>		Timeout time. Range: 10–300. Default: 90. Unit: second.
<connectcode>		The result of connection.
	0	Success
	-1	Error
	-2	Socket is occupied

2.2.3. AT+QSSLSEND Send Data through SSL Connection

After the SSL connection is established, the module can send data through the connection. If sending data successfully, the module will return **SEND OK**. If the process of sending data is blocked, the module will return **SEND FAIL**. And if some other errors occur, the module will return **ERROR**.

When receiving **SEND FAIL**, the host should delay some time for sending data. When receiving **ERROR**, the host should establish SSL connection again.

AT+QSSLSEND Send Data through SSL Connection	
Test Command **AT+QSSLSEND=?**	Response **+QSSLSEND: (**range of supported **<ssid>**s**)[,(**range of supported **<length>**s**)]** **OK**
Read Command **AT+QSSLSEND?**	Response **OK**
Write Command **AT+QSSLSEND=<ssid>[,<length>]** After response **>**, input the data to be sent. Tap **CTRL+Z** to send, and tap "ESC" to cancel the operation.	Response **>** **<input data>** **<CTRL+Z>** If SSL connection has been established and data sending is successful: **SEND OK** If the process of sending data is blocked: **SEND FAIL** If SSL connection is not established, disconnected, or if there is any error: **ERROR**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<ssid>	Numeric type. Secure socket identifier. Range: 0–5.
<length>	Numeric type. Indicate the length of sending data. Range: 1–1460.

2.2.4. AT+QSSLRECV Receive Data through SSL Connection

When module receives data from the peer host, it can read data from buffer. After receiving data, the module will buffer it and report "**+QSSLURC: "recv",<cid>,<ssid>**" to notify the host. Then the peer host can retrieve data by **AT+QSSLRECV**.

AT+QSSLRECV Receive Data through SSL Connection	
Test Command **AT+QSSLRECV=?**	Response **+QSSLRECV: (**range of supported **<cid>**s**),(**range of supported **<ssid>**s**),(**range of supported **<length>**s**)** **OK**
Write Command **AT+QSSLRECV=<cid>,<ssid>,<length>**	Response If data has been received through specified connection: **+QSSLRECV: <ipaddr>:<port>,TCP,<actual length><CR><LF><data>** **OK** If the buffer is empty: **OK** If there is any error: **ERROR**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<cid>	Numeric type. Foreground context No.. Range: 0–1.
<ssid>	Numeric type. Secure socket identifier. Range: 0–5.
<length>	Numeric type. The maximum length of data to be retrieved. Range: 1–1500.
<ipaddr>	IP address.
<port>	The port of remote server.
<actual length>	The actual data length obtained by **AT+QSSLRECV**.

NOTE

If the buffer is not empty, and the module receives data again, then it will not report URC "**+QSSLURC: "recv",<cid>,<ssid>**" until all the received data has been retrieved by **AT+QSSLRECV** from buffer.

2.2.5. AT+QSSLCLOSE Close an SSL Connection

The command closes an SSL connection. If all the SSL connections based on the same SSL context are closed, the module will release the SSL context.

AT+QSSLCLOSE Close an SSL Connection	
Test Command **AT+QSSLCLOSE=?**	Response **+QSSLCLOSE: (**range of supported **<ssid>**s**)[,(0,1)]** **OK**
Read Command **AT+QSSLCLOSE?**	Response **OK**
Write Command **AT+QSSLCLOSE=<ssid>**	Response If the SSL connection is closed successfully: **CLOSE OK** If there is any error: **ERROR**
Characteristics	Whether the command actual takes effect determined by network. The configurations will not be saved.

Parameter

<ssid>	Numeric type. Secure socket identifier. Range: 0–5.

2.2.6. AT+QSSLSTATE Query Socket Connection Status

This command queries the socket connection status. It can not only query the status of SSL connection, but also the status of the normal TCP/UDP connection.

AT+QSSLSTATE Query Socket Connection Status	
Test Command **AT+QSSLSTATE=?**	Response **OK**
Read Command **AT+QSSLSTATE?**	Response **OK**

Write Command **AT+QSSLSTATE**	Response If the socket connection status is queried successfully: **+QSSLSTATE: <state>** **+QSSLSTATE: <socketindex>,<connectiontype>,<ipaddr>,<port>,<socketstatus>,<sslconnectionflag>** **…** **OK** If there is any error: **ERROR**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<state>	A string parameter to indicate the status of the connection.	
	"IP INITIAL" The TCP/IP stack is in idle state.	
	"IP START"	The TCP/IP stack has been registered to.
	"IP CONFIG"	Module has been boot to activate GPRS/CSD context.
	"IP IND"	It is activating GPRS/CSD context.
	"IP GPRSACT"	GPRS/CSD context has been activated successfully.
	"IP STATUS"	The local IP address has been gotten by **AT+QILOCIP**.
	"IP PROCESSING"	Establish connection.
	"PDP DEACT"	GPRS/CSD context was deactivated due to unknown reason.
<socketindex>	Numeric type. Socket index. Range: 0–5.	
<connectiontype>	Connection type.	
	"TCP"	
	"UDP"	
<ipaddr>	IP address.	
<port>	Port number.	
<socketstatus>	Socket status.	
	"INITIAL"	
	"CONNECTING"	
	"CONNECTED"	
	"REMOTE CLOSING"	
	"CLOSING"	
	"CLOSED"	
<sslconnectionflag>	Judge whether the connection is normal TCP/UDP or TCP SSL.	
	0	Normal TCP/UDP connection
	1	TCP SSL connection

2.2.7. AT+QSECWRITE Add a Certificate or Key

The command adds a user certificate, user key or server root CA certificate to RAM or NVRAM. And the certificate and key will be stored in these storages in an encrypted way. After the certificate and key is stored in these storages, the host cannot read the data from these storages and can only query the data checksum. Please note that the certificate or key should not exist in the corresponding storage until it is added to RAM or NVRAM; if it already exists, the host should delete it and then add it to the corresponding storage.

AT+QSECWRITE Add a Certificate or Key	
Test Command **AT+QSECWRITE=?**	Response **+QSECWRITE: <filename>,<filesize>[,(**range of supported **<timeout>)**s**]** **OK**
Read Command **AT+QSECWRITE?**	Response **OK**
Write Command **AT+QSECWRITE=<filename>,<filesize> [,<timeout>]**	Response If the AT command format is right: **CONNECT** After module switches to data mode, the certificate or key data can be inputted. When the size of the inputted data reaches **<filesize>** (unit: byte) or module receives **+++** sequence from UART, the module will return to command mode and reply the following codes. **+QSECWRITE: <uploadsize>,<checksum>** **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	The command takes effect immediately. The storage mechanism depends on the parameter configuration.

Parameter

<filename>	The name of the file to be stored. The format can be as follows:	
	"RAM:filename"	File is uploaded to RAM
	"NVRAM:filename"	File is uploaded to NVRAM. Support server root CA certificate, one client certificate and one client private key. The filename of server root CA certificate must be *CA0*, the filename of client certificate must be *CC0*, and the filename of client private key

must be *CK0*.

CA0 Identify a server root CA certificate

CC0 Identify a client certificate

CK0 Identify a client key

<filesize> The size of the file to be uploaded. Unit: byte.
If the file is uploaded to the RAM, the maximum size is 32768. If the file is uploaded to NVRAM, the maximum size is 2017. The minimum size is 1.

<timeout> The time in seconds to wait for inputted data from UART.
Range: 3–200. Default: 100. Unit: second.

<uploadsize> The size of the actual uploaded data. Unit: byte.

<checksum> The checksum of the uploaded data.

2.2.8. AT+QSECREAD Query the Checksum of a Certificate or Key

The command queries the checksum of a certificate or key. If the checksum is not the same as the original one which is owned by the user, some mistakes will occur.

AT+QSECREAD Query the Checksum of a Certificate or Key	
Test Command **AT+QSECREAD=?**	Response **+QSECREAD: <filename>** **OK**
Write Command **AT+QSECREAD=<filename>**	Response **+QSECREAD: <good>,<checksum>** **OK** If there is any error: **+CME ERROR: <err>>**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<filename> The name of the file to be stored. The format can be as follows:

"RAM:filename" Query the checksum of file that is stored in RAM.

"NVRAM:filename" Query the checksum of file that is stored in NVRAM. Support server root CA certificate, one client certificate and one client private key. The filename of server root CA certificate must be CA0, the filename of client certificate must be *CC0*, and the filename of client private key must be CK0.

CA0 Identify a server root CA certificate

CC0 Identify a client certificate

	CK0	Identify a client key
<good>	Indicate the certificate or key is correct or not. When uploading the certificate or key by **AT+QSECWRITE**, the checksum of certificate or key will be stored at the same time. After executing **AT+QSECREAD**, checksum of the certificate or key will be calculated again. Then compare this checksum with the one stored by **AT+QSECWRITE**. If they are the same, the certificate or key is correct, otherwise it is wrong.	
	0	The certificate or key is wrong
	1	The certificate or key is correct
<checksum>	The checksum of the file.	

2.2.9. AT+QSECDEL Delete a Certificate or Key

The command deletes a certificate or key.

AT+QSECDEL Delete a Certificate or Key	
Test Command **AT+QSECDEL=?**	Response **+QSECDEL: <filename>** **OK**
Write Command **AT+QSECDEL=<filename>**	Response **OK** If there is any error: **+CME ERROR: <err>**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<filename> The name of the file to be stored. The format can be as follows:

"RAM:filename"	Delete a certificate or key that is stored in RAM	
"NVRAM:filename"	Delete a certificate or key that is stored in NVRAM. Server root CA certificate, one client certificate and one client private key are supported. The filename of server root CA certificate must be CA0, the filename of client certificate must be CC0, and the filename of client private key must be CK0.	
	CA0	Identify a server root CA certificate
	CC0	Identify a client certificate
	CK0	Identify a client key

2.2.10. Description of URC

The format of SSL URC is **"+QSSLURC:"** and it is mainly used to notify the host to read received data and disconnect the connections.

2.2.10.1. Notify Host to Read Data

The URC notifies host to read data from peer.

Notify Host to Read Data	
+QSSLURC: "recv",<cid>,<ssid>	This is a URC to notify the host to read SSL data.

Parameter

<cid>	Numeric type. Foreground context No. Range: 0–1.
<ssid>	Numeric type. Secure socket identifier. Range: 0–5.

NOTES

1. Module has a socket buffer which stores the received data. When module receives the data from the peer, it will put the data into the socket buffer. Only in the case that the socket buffer is empty, and the data from the peer arrivals, then module will report the URC "**+QSSLURC: "recv",<cid>,<ssid>**" to notify host to read. Host can use **AT+QSSLRECV** to read the data. When the socket buffer is not empty, and the data arrivals, then module will not report the URC.
2. **AT+QSSLRECV=<cid>,<ssid>,<length>** reads the data from the module's socket buffer. The maximum length to be read is 1500. If the data length in the buffer is less than 1500, this command will read all the data.

2.2.10.2. Notify Disconnection

The URC notifies host that the connection has been disconnected. The disconnection can be caused by a number of reasons such as peer closing the connection or GPRS PDP being deactivated. If this URC is reported, the module will close SSL connection automatically, and the host does not need to execute **AT+QSSLCLOSE** to close the SSL connection.

Notify Disconnection	
+QSSLURC: "closed",<ssid>	The SSL connection based on the specified socket is closed.

Parameter

<ssid>	Numeric type. Secure socket identifier. Range: 0–5.

3 Examples

3.1. SSL Function with Certificate and Key in RAM

This is an example about server authentication and client authentication, and the certificate and key are stored in RAM. It shows how to establish SSL connection and implement data sending and receiving between module and server.

```
//Step 1: Upload a certificate and key to RAM.
AT+QSECWRITE="RAM:ca_cert.pem",1614,100         //Upload the server root CA certificate to RAM.
CONNECT
<Input the ca_cert.pem data, the size is 1614 bytes>
+QSECWRITE: 1614,4039

OK
AT+QSECWRITE="RAM:client_cert.pem",1419,100     //Upload the client certificate to RAM.
CONNECT
<Input the client_cert.pem data, the size is 1419 bytes>
+QSECWRITE: 1419,618

OK
AT+QSECWRITE="RAM:client_key.pem",1679,100      //Upload the client private key to RAM.
CONNECT
<Input the client_key.pem data, the size is 1679 bytes>
+QSECWRITE: 1679,83a7

OK

//Step 2: Configure and activate the PDP context.
AT+QIFGCNT=0                                    //Set context 0 as foreground context.
OK
AT+QICSGP=1,"CMNET"                             //Set bear type as GPRS and APN as
                                                "CMNET", which does not have a username and
                                                password.
OK
AT+QIREGAPP                                     //Register to TCP/IP stack.
OK
```

```
AT+QIACT                                          //Activate GPRS PDP context.
OK
AT+QILOCIP                                        //Query local IP address.
10.1.83.188

//Step 3: Configure SSL version, cipher suite, server authentication and client authentication. And
certificate and private key are in RAM.
AT+QSSLCFG="ignorertctime",1                      //Ignore the RTC time.
OK
AT+QSSLCFG="sslversion",0,4                       //Configure SSL version.
OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"               //Configure cipher suite.
OK
AT+QSSLCFG="seclevel",0,2                         //Configure Server authentication and client
                                                    authentication.
OK
AT+QSECREAD="RAM:ca_cert.pem"                     //Check CA certificate is correct or not.
+QSECREAD: 1,4039

OK
AT+QSECREAD="RAM:client_cert.pem"                 //Check client certificate is correct or not.
+QSECREAD: 1,618

OK
AT+QSECREAD="RAM:client_key.pem"                  //Check client private key is correct or not.
+QSECREAD: 1,83a7

OK
AT+QSSLCFG="cacert",0,"RAM:ca_cert.pem"           //Configure CA certificate.
OK
AT+QSSLCFG="clientcert",0,"RAM:client_cert.pem" //Configure client certificate.
OK
AT+QSSLCFG="clientkey",0,"RAM:client_key.pem"     //Configure client key.
OK

//Step 4: Establish SSL connection, send and receive data.
AT+QSSLOPEN=1,0,"116.247.104.27",465,0            //Establish SSL connection with socket index as
                                                    1. The connection is based on context 0, and
                                                    is non-transparent mode.
OK
+QSSLOPEN: 1,0                                    //Establish SSL connection successfully.
AT+QSSLSEND=1,12                                  //Send 12 bytes data at a fixed length.

> <Input 12 bytes data>
SEND OK
```

```
AT+QSSLSEND=1                                  //Send data in any byte less than 1460.
> <input some bytes data>,<Ctrl+Z>          //After completing data input, tap CTRL+Z to send.
SEND OK

+QSSLURC: "recv",0,1                        //Notify the host to acquire the data from the server.
AT+QSSLRECV=0,1,1500                        //Read data and output the data to UART.
+QSSLRECV: 116.247.104.27:465,TCP,7
1234567

OK

//Step 5: Close SSL connection, delete the certificate and key from RAM.
AT+QSSLCLOSE=1                              //Close socket index 1.
CLOSE OK
AT+QSECDEL="RAM:ca_cert.pem"
OK
AT+QSECDEL="RAM:client_cert.pem"
OK
AT+QSECDEL="RAM:client_key.pem"
OK
AT+QIDEACT                                  //Deactivate GPRS PDP context.
DEACT OK
```

3.2. SSL Function with Certificate and Key in NVRAM

This is an example about server authentication and client authentication, and the certificate and key are stored in NVRAM. It shows how to establish SSL connection, implement data sending and receiving between module and server.

```
//Step 1: Upload a certificate and key to NVRAM.
AT+QSECWRITE="NVRAM:CA0",1614,100           //Upload the server root CA certificate to NVRAM.
CONNECT
<Input the CA0 data, the size is 1614 bytes>
+QSECWRITE: 1614,4039

OK
AT+QSECWRITE="NVRAM:CC0",1419,100           //Upload the client certificate to NVRAM.
CONNECT
<Input the CC0 data, the size is 1419 bytes>
+QSECWRITE: 1419,618

OK
```

```
AT+QSECWRITE="NVRAM:CK0",1679,100          //Upload the client private key to NVRAM.
CONNECT
<Input the CK0 data, the size is 1679 bytes>
+QSECWRITE: 1679,83a7

OK

//Step 2: Configure and activate the PDP context.
AT+QIFGCNT=0                                //Set context 0 as foreground context.
OK
AT+QICSGP=1,"CMNET"                        //Set bear type as GPRS and APN as
                                            "CMNET", which does not have a username and
                                            password.
OK
AT+QIREGAPP                                 //Register to TCP/IP stack.
OK
AT+QIACT                                   //Activate GPRS PDP context.
OK
AT+QILOCIP                                 //Query local IP address.
10.1.83.188

//Step 3: Configure SSL version, cipher suite, server authentication and client authentication. Certificate
and private key are in NVRAM.
AT+QSSLCFG="ignorertctime",1               //Ignore the RTC time.
OK
AT+QSSLCFG="sslversion",0,4                //Configure SSL version.
OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"        //Configure cipher suite.
OK
AT+QSSLCFG="seclevel",0,2                  //Configure server authentication and client
                                             authentication.
OK
AT+QSECREAD="NVRAM:CA0"                    //Check server root CA certificate is correct or not.
+QSECREAD: 1,4039

OK
AT+QSECREAD="NVRAM:CC0"                    //Check client certificate is correct or not.
+QSECREAD: 1,618

OK
AT+QSECREAD="NVRAM:CK0"                    //Check client private key is correct or not.
+QSECREAD: 1,83a7

OK
```

```
AT+QSSLCFG="cacert",0,"NVRAM:CA0"          //Configure server root CA certificate.
OK
AT+QSSLCFG="clientcert",0,"NVRAM:CC0"      //Configure client certificate.
OK
AT+QSSLCFG="clientkey",0, "NVRAM:CK0"       //Configure client key.
OK

//Step 4: Establish SSL connection, send and receive data.
AT+QSSLOPEN =1,0,"116.247.104.27",465,0   //Establish SSL connection with socket index as 1. The
                                            connection is based on context 0, and is
                                            non-transparent mode.
OK

+QSSLOPEN: 1,0                             //Establish SSL connection successfully.
AT+QSSLSEND=1,12                            //Send 12 bytes data at a fixed length.
> <Input 12 bytes data>
SEND OK
AT+QSSLSEND=1                              //Send data in any byte less than 1460.
> <Input some bytes data> ,<Ctrl+Z>        //After completing data input, tap CTRL+Z to send
                                             data.
SEND OK

+QSSLURC: "recv",0,1                        //Notify the host to acquire the data from the server.
AT+QSSLRECV=0,1,1500                        //Read data and output the data to UART.
+QSSLRECV: 116.247.104.27:465,TCP,7
1234567

OK

//Step 5: Close SSL connection.
AT+QSSLCLOSE=1                             //Close socket index 1.
CLOSE OK
AT+QIDEACT                                  //Deactivate GPRS PDP context.
DEACT OK
```

3.3. Example about SSL Function Coexisting with Normal TCP/IP Function

```
//Step 1: Configure and activate the PDP context.
AT+QIFGCNT=0                          //Set context 0 as foreground context.
OK
```

```
AT+QICSGP=1,"CMNET"                  //Set bear type as GPRS and APN as
                                     "CMNET", which does not have a username and password.
AT+QIMUX=1                           //Enable multiple TCP/IP session.
OK
AT+QIREGAPP                          //Register to TCP/IP stack.
OK
AT+QIACT                             //Activate GPRS PDP context.
OK
AT+QILOCIP                           //Query local IP address.
10.1.83.188

//Step 2: Establish normal TCP connection, send and receive data.
AT+QINDI=1                           //Set the method to handle the received TCP/IP data. Output a
                                     notification statement "+QIRDI: <id>,<sc>,<sid>"through UART
                                     to notify host to read the received TCP/IP data.
OK
AT+QIOPEN=1,"TCP","116.247.104.27",6021   //Establish normal TCP connection and specify the
                                           socket index as 1.
OK

1, CONNECT OK                             //Establish normal TCP connection successfully.
AT+QISEND=1,10                            //Send 10 bytes data at a fixed length.
><input 10 bytes data>
SEND OK

+QIRDI: 0,1,1                             //Module receives the data based on context 0 acting as
                                           the client, and the socket index is 1.
AT+QIRD=0,1,1,1024                        //Read the data from the module's socket buffer.
+QIRD: 116.247.104.27:6021,TCP,10         //The maximum length to retrieve is 1024. If the data
                                           length in the buffer is less than 1024, retrieve all the
                                           data from the buffer.
ABCDE12345
OK

//Step 3: Configure SSL version, cipher suite and no authentication mode. Establish SSL connection, send
and receive data.
AT+QSSLCFG="sslversion",0,4               //Configure SSL version.
OK
AT+QSSLCFG="seclevel",0,0                 //Configure Server authentication and client
                                            authentication.
OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"       //Configure cipher suite.
OK
AT+QSSLOPEN=3,0,"124.74.41.170",5115,0    //Establish SSL connection and the socket index is 3,
                                         and the connection is based on context 0.
OK

+QSSLOPEN: 3,0                           //Establish SSL connection successfully.
AT+QSSLSEND=3,12                         //Send 12 bytes data in the way of fixed length.
> <Input 12 bytes data>
SEND OK
AT+QSSLSEND=3
> <Input some bytes data>,<Ctrl+Z>       //After completing to input data, tap CTRL+Z to send
                                         data.
SEND OK

+QSSLURC: "recv",0,3                     //URC, notify the host to acquire the data from the
                                         server.
AT+QSSLRECV=0,3,1000                     //Read the data and output the data to UART.
+QSSLRECV: 124.74.41.170:5115,TCP,7
1234567

OK

//Step 4: Close normal TCP and SSL connection.
AT+QSSLCLOSE=3                           //Close SSL connection, the socket index is 3.
CLOSE OK
AT+QICLOSE=1                             //Close normal TCP connection, the socket index is 1.
1,CLOSE OK
AT+QIDEACT                               //Deactivate GPRS PDP context.
DEACT OK
```

3.4. Example about Transparent Mode of SSL Function

3.4.1. No Authentication

```
//Step 1: Configure and activate the PDP context.
AT+QIFGCNT=0                             //Set context 0 as foreground context.
OK
AT+QICSGP=1,"CMNET"                      //Set bear type as GPRS and APN as
                                         "CMNET", which does not have a username and
                                         password.
OK
AT+QIREGAPP                              //Register to TCP/IP stack.
OK
```

```
AT+QIACT                                          //Activate GPRS PDP context.
OK
AT+QILOCIP                                         //Query local IP address.
10.1.83.188

//Step 2: Configure SSL version, cipher suite and no authentication mode. Establish SSL connection, send
and receive data.
AT+QSSLCFG="sslversion",0,4                        //Configure SSL version.
OK
AT+QSSLCFG="seclevel",0,0                          //Configure server authentication and client
                                                  authentication

OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"                //Configure cipher suite.
OK
AT+QSSLOPEN=1,0,"220.180.239.212",8011,1          //Establish SSL connection with socket index as 1.
                                                 The  connection is based on context 0, and is
                                                 transparent mode.
OK

CONNECT                                           //Establish SSL connection successfully.
GET /processorder.php HTTP/1.1                    //Input the data.
HOST: 220.180.239.201:8011
Accept: */*
User-Agent: QUECTEL_MODULE
CONNECT: KEEP-ALIVE                               //Complete data input.

HTTP/1.1 200 OK                                   //SSL server responses data to the module.
Date: Wed, 07 Sep 2016 08:37:27 GMT
Server: Apache/2.4.4 (Win32) OpenSSL/1.0.1e PHP/5.5.33
X-Powered-By: PHP/5.5.33
Content-Length: 264
Content-Type: text/html

<html>
<head>
<title>Quectel's Auto Parts-Order Results</title>
</head>
<body>
<h1>Quectel's Auto Parts</h1>
<h2>Order Results</h2>

<p>Order processed at </p><p>Your order is as follows: </p> message<br /> apple<br /> orange<br
/></body>
```

```
</html>                                        //Server completes sending data.

//Step 3: Exit from or enter transparent mode.
+++                                            //Input +++.
OK                                             //Response OK.

ATO
CONNECT                                        //Enter transparent mode.

//Step 4: Close SSL connection.
+++                                            //Input +++
OK                                             //Response OK.

AT+QSSLCLOSE=1                                 //Close SSL connection and the socket index is 1.
CLOSE OK
AT+QIDEACT                                     //Deactivate GPRS PDP context.
DEACT OK
```

3.4.2. Server and Client Authentication

```
//Step 1: Upload a certificate and key to RAM.
AT+QSECWRITE="RAM:ca_cert.pem",1614,100        //Upload the server root CA certificate to RAM.
CONNECT
<Input the ca_cert.pem data, the size is 1614 bytes>
+QSECWRITE: 1614,4039

OK
AT+QSECWRITE="RAM:client_cert.pem",1419,100    //Upload the client certificate to RAM.
CONNECT
<Input the client_cert.pem data, the size is 1419 bytes>
+QSECWRITE: 1419,618

OK
AT+QSECWRITE="RAM:client_key.pem",1679,100     //Upload the client private key to RAM.
CONNECT
<Input the client_key.pem data, the size is 1679 bytes>
+QSECWRITE: 1679,83a7

OK

//Step 2: Configure and activate the PDP context.
AT+QIFGCNT=0                                   //Set context 0 as foreground context.
OK
AT+QICSGP=1,"CMNET"                            //Set bear type as GPRS and APN as "CMNET",
                                              which does not have a username and password.
OK
AT+QIREGAPP                                   //Register to TCP/IP stack.
OK
AT+QIACT                                      //Activate GPRS PDP context.
OK
AT+QILOCIP                                    //Query local IP address.
100.119.144.3

//Step 3: Configure SSL version, cipher suite, server authentication and client authentication. Certificate
and private key are in RAM.
AT+QSSLCFG="ignorertctime",1                  //Ignore the RTC time.
OK
AT+QSSLCFG="sslversion",0,4                   //Configure SSL version.
OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"           //Configure cipher suite.
OK
AT+QSSLCFG="seclevel",0,2                     //Configure Server authentication and client
                                               authentication.
OK
AT+QSECREAD="RAM:ca_cert.pem"                 //Check CA certificate is correct or not.
+QSECREAD: 1,4039

OK
AT+QSECREAD="RAM:client_cert.pem"             //Check client certificate is correct or not.
+QSECREAD: 1,618

OK
AT+QSECREAD="RAM:client_key.pem"              //Check client private key is correct or not.
+QSECREAD: 1,83a7

OK
AT+QSSLCFG="cacert",0,"RAM:ca_cert.pem"       //Configure server root CA certificate.
OK
AT+QSSLCFG="clientcert",0,"RAM:client_cert.pem" //Configure client certificate.
OK
AT+QSSLCFG="clientkey",0,"RAM:client_key.pem"   //Configure client key.
OK

//Step 4: Establish SSL connection, send data and receive data
AT+QSSLOPEN =1,0,"220.180.239.212",8011,1     //Establish SSL connection with socket index as 1.
                                              The connection is based on context 0, and is
                                              transparent mode.
OK
```

```
CONNECT                                        //Establish SSL connection successfully.
GET /processorder.php HTTP/1.1                 //Input the data.
HOST: 220.180.239.201:8011
Accept: */*
User-Agent: QUECTEL_MODULE
CONNECT: KEEP-ALIVE                            //Complete data input.

HTTP/1.1 200 OK                                //SSL server responses data to the module.
Date: Wed, 07 Sep 2016 08:37:27 GMT
Server: Apache/2.4.4 (Win32) OpenSSL/1.0.1e PHP/5.5.33
X-Powered-By: PHP/5.5.33
Content-Length: 264
Content-Type: text/html

<html>
<head>
<title>Quectel's Auto Parts-Order Results</title>
</head>
<body>
<h1>Quectel's Auto Parts</h1>
<h2>Order Results</h2>

<p>Order processed at </p><p>Your order is as follows: </p> message<br /> apple<br /> orange<br
/></body>
</html>                                     //Server completes sending data.

//Step 5: Exit from or enter transparent mode.
+++                                         //Input +++
OK                                          //Response OK

ATO
CONNECT                                     //Enter transparent mode.

//Step 6: Close SSL connection.
+++                                        //Input +++
OK                                         //Response OK
AT+QSSLCLOSE=1                             //Close SSL connection, and the socket index is 1.
CLOSE OK
AT+QIDEACT                                 //Deactivate GPRS PDP context.
DEACT OK
```

4 Error Handling

4.1. PDP Activation Failure

If PDP context is failed to be activated by **AT+QIACT**, please check the following aspects:

1. Query whether the PS domain is attached by **AT+CGATT?**. If not, execute **AT+CGATT=1** to attach PS domain.
2. Query the **AT+CGREG** status by **AT+CGREG?** and make sure the PS domain is registered
3. Query the PDP context parameters by **AT+QIREGAPP**, and make sure the APN of specified PDP context is set.
4. Make sure the specified PDP context ID is neither used by PPP nor activated by **AT+CGACT**.

If the result of checking is completed, but the result of executing **AT+QIACT** always fails, please reboot the module to solve this issue. After rebooting the module, please check the above procedures at least three times, at intervals of 10 minutes, to avoid frequent reboot of the module.

5 Appendix A Reference

Table 4: Related Documents

SN.	Document name	Remark
[1]	GSM 07.07	Digital cellular telecommunications (Phase 2+); AT command set for GSM Mobile Equipment (ME)
[2]	GSM 07.10	Support GSM 07.10 multiplexing protocol
[3]	Quectel_Mxx/MCxx_AT_Commands_Manual	AT commands manual for GSM modules
[4]	Quectel_GSM_TCP/IP_Application_Note	TCP/IP application note for GSM modules

Table 5: Terms and Abbreviations

Abbreviation	Description
CTX	SSL Context
GSM	Global System for Mobile Communications
ME	Mobile Equipment
MS	Mobile Station
NVRAM	Non-Volatile Random Access Memory
ID	Identification
IP	Internet Protocol
PDP	Packet Data Protocol
PPP	Point-to-point Protocol
RAM	Random Access Memory
SSL	Security Socket Layer

TA	Terminal Adapter
TCP	Transmission Control Protocol
URC	Unsolicited Result Code