EC2x&EG9x&EG2x-G&EM05 Series

SSL Application Note

LTE Standard Module Series

Version: 1.1

Date: 2020-08-30

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.

DISCLAIMER

WHILE QUECTEL HAS MADE EFFORTS TO ENSURE THAT THE FUNCTIONS AND FEATURES UNDER DEVELOPMENT ARE FREE FROM ERRORS, IT IS POSSIBLE THAT THESE FUNCTIONS AND FEATURES COULD CONTAIN ERRORS, INACCURACIES AND OMISSIONS. UNLESS OTHERWISE PROVIDED BY VALID AGREEMENT, QUECTEL MAKES NO WARRANTIES OF ANY KIND, IMPLIED OR EXPRESS, WITH RESPECT TO THE USE OF FEATURES AND FUNCTIONS UNDER DEVELOPMENT. TO THE MAXIMUM EXTENT PERMITTED BY LAW, QUECTEL EXCLUDES ALL LIABILITY FOR ANY LOSS OR DAMAGE SUFFERED IN CONNECTION WITH THE USE OF THE FUNCTIONS AND FEATURES UNDER DEVELOPMENT, REGARDLESS OF WHETHER SUCH LOSS OR DAMAGE MAY HAVE BEEN FORESEEABLE.

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2017-11-22	Duke XIN/ Jessica GENG	Initial
1.1	2020-08-30	Luffy LIU	1. Added the following AT+QSSLCFG commands (Chapter 2.2.1): (1) AT+QSSLCFG="closetimemode" (2) AT+QSSLCFG="cacertex" (3) AT+QSSLCFG="ignoremulticertchainverify" (4) AT+QSSLCFG="ignoreinvalidcertsign" (5) AT+QSSLCFG="psk" (6) AT+QSSLCFG="dtls" (7) AT+QSSLCFG="dtlsversion" (8) AT+QSSLCFG="session_cache" (9) AT+QSSLCFG="alpn" (10)AT+QSSLCFG="renegotiation" 2. Updated the description of <close_timeout> for AT+QSSLCLOSE (Chapter 2.2.5). 3. Updated the example of sending data in buffer access mode (Chapter 3.3.2).

Contents

Table Index

1 Introduction

Quectel EC2x series, EG9x series, EG2x-G and EM05 series modules support SSL function. The SSL function is to ensure the privacy of communication. In some cases, the communication between the server and the client should be encrypted to prevent data from being eavesdropped, tampered with or forged during the communication process.

This document introduces how to use the SSL function of the following Quectel modules through AT commands.

Table 1: Applicable Modules

Module Series	Model
EC2x series	EC21 series
	EC25 series
	EC20 R2.1
EG9x series	EG91 series
	EG95 series
EG2x-G	EG21-G
	EG25-G
EM05 series	EM05 series

1.1. SSL Version and Cipher Suite

The following SSL versions are supported.

Table 2: SSL Versions

SSL Versions
SSL3.0
TLS1.2
TLS1.1
TLS1.0

The following table shows SSL cipher suites supported by Quectel EC2x series, EG9x series, EG2x-G, EM05 series modules, and all the SSL cipher suites are supported by default. For detailed description of cipher suites, see *RFC 2246-The TLS Protocol Version 1.0*.

Table 3: Supported SSL Cipher Suites

Codes of Cipher Suites	Names of 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
0XC002	TLS_ECDH_ECDSA_WITH_RC4_128_SHA
0XC003	TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
0XC004	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
0XC005	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA

0XC007	TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
0XC008	TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
0XC009	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
0XC00A	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
0XC011	TLS_ECDHE_RSA_WITH_RC4_128_SHA
0XC012	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
0XC013	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
0XC014	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
0xC00C	TLS_ECDH_RSA_WITH_RC4_128_SHA
0XC00D	TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
0XC00E	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
0XC00F	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA
0XC023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
0xC024	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
0xC025	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256
0xC026	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384
0XC027	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256
0XC028	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384
0xC029	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256
0XC02A	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384
0XC02F	TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
0XFFFF	Support all the cipher suites listed above

1.2. The Process of Using SSL Function

Step 1: Configure **<APN>**, **<username>**, **<password>** and other parameters of a PDP context by **AT+QICSGP**. See *Quectel_LTE_Standard_TCP(IP)_Application_Note* for details.
Step 2: Activate the PDP context by **AT+QIACT**, then query the assigned IP address by **AT+QIACT?**. See *Quectel_LTE_Standard_TCP(IP)_Application_Note* for details.
Step 3: Configure the SSL version, cipher suite, path of trusted CA certificate, authentication mode, the path of the client certificate and private key, etc. for the specified SSL context by **AT+QSSLCFG**.
Step 4: Open an SSL socket to connect a remote server by **AT+QSSLOPEN**. **<SSL_ctxID>** is used to specify SSL context, and **<access_mode>** is used to specify data access mode.
Step 5: After the SSL connection has been established, data will be sent or received via the connection. For details about how to send and receive data in each access mode, see ***Chapter 1.3***.
Step 6: Close SSL connection by **AT+QSSLCLOSE**.
Step 7: Deactivate the PDP context by **AT+QIDEACT**. See *Quectel_LTE_Standard_TCP(IP)_Application_Note* for details.

1.3. Description of Data Access Modes

The SSL connection supports the following three data access modes:

- Buffer access mode
- Direct push mode
- Transparent access mode

When opening an SSL connection via **AT+QSSLOPEN**, the data access mode can be specified by the **<access_mode>**. After the SSL connection has been established, **AT+QISWTMD** can be used to switch the data access mode. For details of **AT+QISWTMD**, see *Quectel_LTE_Standard_TCP(IP)_Application_Note* for details.

1. In buffer access mode, data can be sent via **AT+QSSLSEND**, and if the module has received data from the Internet, a URC **+QSSLURC: "recv",<clientID>** will be reported. In such case, data can be retrieved via **AT+QSSLRECV**.

2. In direct push mode, data can be sent via **AT+QSSLSEND**, and if the module has received data from the Internet, the data will be outputted directly via UART1/USB modem/USB AT port in the format of **+QSSLURC: "recv",<clientID>,<currectrecvlength><CR><LF><data>**.

3. In transparent access mode, the corresponding port enters exclusive mode. The data received from COM port will be sent to the Internet directly, and the received data from Internet will be outputted to COM port directly. Use **+++** or DTR (set **AT&D1** first) to exit transparent access mode. In transparent access mode, if any abnormal SSL disconnection happens, the module will report **NO CARRIER**. For

details of **AT&D**, see *Quectel_EC2x&EG9x&EG2x-G&EM05_Series_AT_Commands_Manual*.

- **Exit transparent access mode**

To exit transparent access mode, **+++** or DTR (set **AT&D1** first) can be used. To prevent the **+++** from being misinterpreted as data, follow the following sequence:

1) Do not input any character within 1 s (at least or longer) before inputting **+++**.
2) Input **+++** within 1s, and no other characters can be inputted during the time.
3) Do not input any character within 1 s after **+++** has been inputted.
4) Use **+++** or DTR (set **AT&D1** first) to make the module exit transparent access mode, and wait until **OK** is returned.

- **Return to transparent access mode**

1) By **AT+QISWTMD**. Specify the **<access_mode>** as 2 when executing this command. If entering transparent access mode successfully, **CONNECT** will be returned.
2) By **ATO**. **ATO** will change the access mode of connection that exits transparent access mode lately. If entering transparent access mode successfully, **CONNECT** will be returned. If there is no connection entering transparent access mode before, **ATO** will return **NO CARRIER**. For details of **ATO**, see *Quectel_EC2x&EG9x&EG2x-G&EM05_Series_AT_Commands_Manual*.

1.4. Validity Check of Certificate

To check whether a certificate is in the validity period, the certificate must be parsed, and compare the local time with the “Not before” and “Not after” of the certificate. If the local time is earlier than the time of “Not before” or later than the time of “Not after”, the certificate will be considered expired.

When validity check of certificate is required (set **<ignore_ltime>** as 0 when executing **AT+QSSLCFG**), in order to avoid failure of certificate validity check, execute **AT+CCLK** to configure the module time within the validity time period of the certificate. For details of **AT+CCLK**, see *Quectel_EC2x&EG9x&EG2x-G&EM05_Series_AT_Commands_Manual*.

1.5. Server Name Indication

SNI (Server Name Indication) is desirable for clients to provide Server Host Name information to facilitate secure connections to servers that host multiple 'virtual' servers at a single underlying network address. And this feature is only applicable for TLS protocol.

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 omitted, 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 4: 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 suite, path of trusted CA certificate, authentication mode, the path of the client certificate and private key, etc. for the specified SSL context. These parameters will be used in the handshake procedure.

<SSL_ctxID> is the index of the SSL context. The module supports 6 SSL contexts at most. On the basis of one SSL context, several SSL connections can be established. The settings such as the SSL version and the cipher suite are stored in the SSL context, and they will be applied to the new SSL connections associated with the SSL context.

AT+QSSLCFG Configure Parameters of an SSL Context	
Test Command **AT+QSSLCFG=?**	Response **+QSSLCFG: "sslversion",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<SSL_version>**s**)** **+QSSLCFG: "ciphersuite",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<cipher_suites>**s**)** **+QSSLCFG: "cacert",(**range of supported **<SSL_ctxID>**s**),<cacertpath>** **+QSSLCFG: "cacertex",(**range of supported **<SSL_ctxID>**s**),<cacertpath>** **+QSSLCFG: "clientcert",(**range of supported **<SSL_ctxID>**s**),<client_cert_path>** **+QSSLCFG: "clientkey",(**range of supported **<SSL_ctxID>**s**),<client_key_path>** **+QSSLCFG: "seclevel",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<seclevel>**s**)** **+QSSLCFG: "ignorelocaltime",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<ignore_ltime>**s**)** **+QSSLCFG: "negotiatetime",(**range of supported **<SSL_ctxID>**s**),(**range of supported **<negotiate_time>**s**)** **+QSSLCFG: "sni",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<SNI>**s**)** **+QSSLCFG: "closetimemode",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<close_time_mode>**s**)** **+QSSLCFG: "ignoremulticertchainverify",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<ignore_multicertchain_verify>**s**)** **+QSSLCFG: "ignoreinvalidcertsign",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<ignore_invalid_certsign>**s**)**

	+QSSLCFG: "psk",(range of supported **<SSL_ctxID>**s**),<identity>,<key>** **+QSSLCFG: "dtls",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<DTLS_enable>**s**)** **+QSSLCFG: "dtlsversion",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<DTLS_ver>**s**)** **+QSSLCFG:"session_cache",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<session_cache_enable>**s**)** **+QSSLCFG: "alpn",(**range of supported **<SSL_ctxID>**s**),<ALPN_name>** **+QSSLCFG: "renegotiation",(**range of supported **<SSL_ctxID>**s**),(**list of supported **<renegotiation_enable>**s**)** **OK**
Write Command Configure the SSL version for the specified SSL context: **AT+QSSLCFG="sslversion",<SSL_ctxID>[,<SSL_version>]**	Response If the optional parameter is omitted, query the SSL version for the specified SSL context: **+QSSLCFG: "sslversion",<SSL_ctxID>,<SSL_version>** **OK** If the optional parameter 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",<SSL_ctxID>[,<cipher_suites>]**	Response If the optional parameter is omitted, query the SSL cipher suites for the specified SSL context: **+QSSLCFG: "ciphersuite",<SSL_ctxID>,<cipher_suites>** **OK** If the optional parameter is specified, set the SSL cipher suite for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of trusted CA certificate for the specified SSL context: **AT+QSSLCFG="cacert",<SSL_ctxID>[,**	Response If the optional parameter is omitted, query the path of trusted CA certificate for the specified SSL context: **+QSSLCFG: "cacert",<SSL_ctxID>,<cacertpath>**

<cacertpath>]	**OK** If the optional parameter is specified, set the path of trusted CA certificate for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of trusted CA certificate for the specified SSL context: **AT+QSSLCFG="clientcertex"[,<SSL_ctxID>[,<client_cert_path>]]**	Response If the all optional parameter are omitted, query the path of trusted CA certificate for all SSL context: **+QSSLCFG: "cacertex",0,<cacertpath>** **…..** **+QSSLCFG: "cacertex",5,<cacertpath>** **OK** If only **<client_cert_path>** is omitted, query the path of trusted CA certificate for the specified SSL context: **+QSSLCFG: "cacertex",<SSL_ctxID>,<cacertpath>** **OK** If all optional parameter is specified, set the path of trusted 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",<SSL_ctxID>[,<client_cert_path>]**	Response If the optional parameter is omitted, query the path of client certificate for the specified SSL context: **+QSSLCFG: "clientcert",<SSL_ctxID>,<client_cert_path>** **OK** If the optional parameter 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:	Response If the optional parameter is omitted, query the path of client private key for the specified SSL context:

AT+QSSLCFG="clientkey",<SSL_ctxID>[,<client_key_path>]	**+QSSLCFG: "clientkey",<SSL_ctxID>,<client_key_path>** **OK** If the optional parameter is specified, set the path of client private key for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the authentication mode for the specified SSL context: **AT+QSSLCFG="seclevel",<SSL_ctxID>[,<seclevel>]**	Response If the optional parameter is omitted, query the authentication mode for the specified SSL context: **+QSSLCFG: "seclevel",<SSL_ctxID>,<seclevel>** **OK** If the optional parameter is specified, set the authentication mode for the specified SSL context: **OK** Or **ERROR**
Write Command Configure whether to ignore certificate validity check for the specified SSL context: **AT+QSSLCFG="ignorelocaltime",<SSL_ctxID>[,<ignore_ltime>]**	Response If the optional parameter is omitted, query whether the certificate validity check is ignored for the specified SSL context: **+QSSLCFG: "ignorelocaltime",<SSL_ctxID>,<ignore_ltime>** **OK** If the optional parameter is specified, set whether or not to ignore certificate validity check for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the maximum timeout in SSL negotiation stage for the specified SSL context: **AT+QSSLCFG="negotiatetime",<SSL_ctxID>[,<negotiate_time>]**	Response If the optional parameter is omitted, query the maximum timeout in SSL negotiation stage for the specified SSL context: **+QSSLCFG: "negotiatetime",<SSL_ctxID>,<negotiate_time>** **OK**

	If the optional parameter is specified, set the maximum timeout in SSL negotiation stage for the specified SSL context: **OK** Or **ERROR**
Write Command Configure Server Name Indication feature for the specified SSL context: **AT+QSSLCFG="sni",<SSL_ctxID>[,<SNI>]**	Response If the optional parameter is omitted, query whether the Server Name Indication feature is enabled for the specified SSL context: **+QSSLCFG: "sni",<SSL_ctxID>,<SNI>** **OK** If the optional parameter is specified , disable/enable Server Name Indication feature for the specified SSL context: **OK** Or **ERROR**
Write Command Enable/disable the SSL close linger time for the specified SSL context: **AT+QSSLCFG="closetimemode",<SSL_ctxID>[,<close_time_mode>]**	Response If the optional parameter is omitted, query whether the close linger time is enabled for the specified SSL context: **+QSSLCFG: "closetimemode",<SSL_ctxID>,<close_time_mode>** **OK** If the optional parameter is specified, enable/disable the SSL close linger time for the specified SSL context: **OK** Or **ERROR**
Write Command Configure whether to ignore multiple level certificate chain verification for the specified SSL context: **AT+QSSLCFG="ignoremulticertchainverify",<SSL_ctxID>[,<ignore_multicertchain_verify>]**	Response If the optional parameter is omitted, query whether the multiple level certificate chain verification is ignored for the specified SSL context: **+QSSLCFG: "ignoremulticertchainverify",<SSL_ctxID>,<ignore_multicertchain_verify>** **OK** If the optional parameter is specified, set whether or not to ignore multiple level certificate chain verification for the specified SSL context:

	OK Or **ERROR**
Write Command Configure whether to ignore the invalid certificate signature for the specified SSL context: **AT+QSSLCFG="ignoreinvalidcertsign",<SSL_ctxID>[,<ignore_invalid_certsign>]**	Response If the optional parameter is omitted, query whether the invalid certificate signature is ignored for the specified SSL context: **+QSSLCFG: "ignoremulticertchainverify",<SSL_ctxID>,<ignore_invalid_certsign>** **OK** If the optional parameter is specified, set whether or not to ignore the invalid certificate signature for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the PSK which used in handshake for the specified SSL context : **AT+QSSLCFG="psk",<SSL_ctxID>[,<identity>,<key>]**	Response If the optional parameters are omitted, query the current configuration for the specified SSL context: **+QSSLCFG:"psk",<SSL_ctxID>,<identity>,<key>** **OK** If the optional parameters are specified, set the PSK used in handshake for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the DTLS function for the specified SSL context: **AT+QSSLCFG="dtls",<SSL_ctxID>[,<DTLS_enable>**	Response If the optional parameter is omitted, query whether the DTLS function is enabled for the specified SSL context: **+QSSLCFG: "dtls",<SSL_ctxID>,<DTLS_enable>** **OK** If the optional parameter is specified, enable/disable the DTLS function for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the version of DTLS for the	Response If the optional parameter is omitted, query the current DTLS

specified SSL context: **AT+QSSLCFG="dtlsversion",<SSL_ctxID>[,<DTLS_ver>]**	version for the specified SSL context: **+QSSLCFG: "dtlsversion",<SSL_ctxID>,<DTLS_ver>** **OK** If the optional parameter is specified, set the DTLS version for the specified SSL context: **OK** Or **ERROR**
Write Command Enable/Disable SSL session resumption function for the specified SSL context: **AT+QSSLCFG="session_cache",<SSL_ctxID>[,<session_cache_enable>]**	Response If the optional parameter is omitted, query whether the SSL session resumption function is enabled for the specified SSL context: **+QSSLCFG: "session_cache",<SSL_ctxID>,<session_cache_enable>** **OK** If the optional parameter is specified, enabled/disabled the SSL session resumption function: **OK** Or **ERROR**
Write Command Configure the ALPN information: **AT+QSSLCFG="alpn",<SSL_ctxID>[,<ALPN_name>]**	Response If the optional parameter is omitted, query the ALPN information for the specified SSL context: **+QSSLCFG: "alpn",<SSL_ctxID>,<ALPN_name>** **OK** If the optional parameter is specified, configure the ALPN information: **OK** Or **ERROR**
Write Command Enable/disable TLS renegotiation function for the specified SSL context: **AT+QSSLCFG="renegotiation",<SSL_ctxID>[,<renegotiation_enable>]**	Response If the optional parameter is omitted, query whether the TLS renegotiation function is enabled for the specified SSL context: **+QSSLCFG: "renegotiation<SSL_ctxID>,<renegotiation_enable>** **OK**

	If the optional parameter is specified, enabled/disable the TLS renegotiation function for the specified SSL context: **OK** Or **ERROR**
Maximum Response Time	300 ms
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<SSL_ctxID> Integer type. SSL context ID. Range: 0–5.

<SSL_version> Integer type. SSL version.

0	SSL3.0
1	TLS1.0
2	TLS1.1
3	TLS1.2
4	All

<cipher_suites> Numeric type in HEX format. 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
0XC002	TLS_ECDH_ECDSA_WITH_RC4_128_SHA
0XC003	TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA
0XC004	TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA
0XC005	TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA
0XC007	TLS_ECDHE_ECDSA_WITH_RC4_128_SHA
0XC008	TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA
0XC009	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA
0XC00A	TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA
0XC011	TLS_ECDHE_RSA_WITH_RC4_128_SHA
0XC012	TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA
0XC013	TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA
0XC014	TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA
0xC00C	TLS_ECDH_RSA_WITH_RC4_128_SHA
0XC00D	TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA
0XC00E	TLS_ECDH_RSA_WITH_AES_128_CBC_SHA
0XC00F	TLS_ECDH_RSA_WITH_AES_256_CBC_SHA
0XC023	TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256

	0xC024 TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384 0xC025 TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256 0xC026 TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384 0XC027 TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256 0XC028 TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384 0xC029 TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256 0XC02A TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384 0XC02F TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256 0XFFFF Support all cipher suites
<ignore_ltime>	Integer type. How to deal with expired certificate. 0 Care about validity check for certification 1 Ignore validity check for certification
<cacertpath>	String type. The path of the trusted CA certificate.
<client_cert_path>	String type. The path of the client certificate.
<client_key_path>	String type. The path of the client private key.
<seclevel>	Integer type. The authentication mode. 0 No authentication 1 Perform server authentication 2 Perform server and client authentication if requested by the remote server
<negotiate_time>	Integer type. Indicates maximum timeout used in SSL negotiation stage. Range: 10–300. Default: 300. Unit: second.
<SNI>	Integer type. Disable/enable Server Name Indication feature 0 Disable 1 Enable
<close_time_mode>	Integer type. Enable/disable the SSL close linger time. 0 Disable, and the unit of SSL close linger time is s. 1 Enable, and the unit of SSL close linger time is ms.
<ignore_multicertchain_verify>	Integer type. Indicates whether or not to ignore the multiple level certificate chains verification. 0 Not to ignore 1 Ignore
<ignore_invalid_certsign>	Integer type. Indicates whether or not to ignore the invalid certificate signature. 0 Not to ignore 1 Ignore
<identity>	String type. Identity of PSK. The length is 0–255
<key>	String type. Key of PSK. The length is 0–255
<DTLS_enable>	Integer type. Enable/disable the DTLS function. 0 Disable 1 Enable
<DTLS_ver>	Integer type. Indicates the DTLS version. This parameter only takes effect when **<DTLS_enable>**=1. 0 DTLSv1

1 DTLSv1.2

<session_cache_enable> Integer type. Enable/disable the SSL session resumption function.
0 Disable
1 Enable

<ALPN_name> String type. ALPN means Application Layer Protocol Negotiation. It configures TLS extension ALPN protocol name, and if the content of this parameter is null (only double quotes are specified), TLS does not contain ALPN extension content.

<renegotiation_enable> Integer type. Enable/disable the TLS renegotiation function.
0 Disable
1 Enable

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

The command sets up an SSL connection, that is, opens an SSL socket to connect a remote server. During the negotiation between the module and the Internet, parameters configured by **AT+QSSLCFG** will be used in the handshake procedure. After shaking hands with the Internet successfully, the module can send or receive data via this SSL connection. Also the module can set up several SSL connections based on one SSL context.

According to steps mentioned in ***Chapter 1.2***, before executing **AT+QSSLOPEN**, execute **AT+QIACT** first to activate the PDP context.

It is suggested to wait for a specific period of time (refer to the Maximum Response Time below) for **+QSSLOPEN: <connectID>,<err>** URC to be outputted. If the URC response cannot be received during the time, **AT+QSSLCLOSE** can be used to close the SSL connection.

AT+QSSLOPEN Open an SSL Socket to Connect a Remote Server	
Test Command **AT+QSSLOPEN=?**	Response **+QSSLOPEN: (**range of supported **<PDP_ctxID>**s**),(**range of supported **<SSL_ctxID>**s**),(**range of supported **<clientID>**s**),<serveraddr>,<server_port>[,(**range of supported **<access_mode>**s**)]** **OK**
Write Command **AT+QSSLOPEN=<PDP_ctxID>,<SSL_ctxID>,<clientID>,<serveraddr>,<server_port>[,<access_mode>]**	Response If the **<access_mode>**=2 and the SSL connection is successfully set up: **CONNECT** If there is any error: **ERROR** Error description can be got via **AT+QIGETERROR**.

	If the **<access_mode>**=0/1: **OK** **+QSSLOPEN: <clientID>,<err>** **<err>** is 0 when SSL socket is opened successfully, otherwise **<err>** is not 0. If there is any error: **ERROR** Error description can be got via **AT+QIGETERROR**.
Maximum Response Time	Maximum network response time of 150 s, plus configured time of **<negotiate_time>**.
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<PDP_ctxID>	Integer type. PDP context ID. Range: 1–16.
<SSL_ctxID>	Integer type. SSL context ID. Range: 0–5.
<clientID>	Integer type. Socket index. Range: 0–11.
<serveraddr>	String type. The address of remote server.
<server_port>	Integer type. The listening port of remote server.
<access_mode>	Integer type. The access mode of SSL connection. 0 Buffer access mode 1 Direct push mode 2 Transparent mode
<err>	Integer type. The error code of the operation. See ***Chapter 5***.
<negotiate_time>	Integer type. Indicates maximum timeout used in SSL negotiation stage. Range: 10–300. Default: 300. Unit: second.

2.2.3. AT+QSSLSEND Send Data via SSL Connection

After the connection is established, the module can send data through the SSL connection.

AT+QSSLSEND Send Data via SSL Connection	
Test Command **AT+QSSLSEND=?**	Response **+QSSLSEND: (**range of supported **<clientID>**s**)[,(**range of supported **<sendlen>**s**)]** **OK**

Write Command Send variable-length data **AT+QSSLSEND=<clientID>**	Response **>** After the above response, input the data to be sent. Tap **CTRL+Z** to send, and tap **ESC** to cancel the operation. If the connection has been established and sending is successful:**SEND OK** If connection has been established but sending buffer is full: **SEND FAIL** If connection cannot be established, abnormally closed, or the parameter is incorrect, response: **ERROR**
Write Command Send fixed-length data **AT+QSSLSEND=<clientID>,<sendlen>**	Response **>** After the above response, input the data until the data length equals **<sendlen>**. If connection has been established and sending is successful: **SEND OK** If connection has been established but sending buffer is full, response: **SEND FAIL** If connection cannot be established, abnormally closed, or the parameter is incorrect: **ERROR**
Maximum Response Time	300 ms
Characteristics	The command takes effect immediately.

Parameter

<clientID>	Integer type. Socket index. Range: 0–11.
<sendlen>	Integer type. The length of sending data. Range: 1–1460. Unit: byte.

NOTE

When sending variable-length data, the maximum length is 1460 bytes.

2.2.4. AT+QSSLRECV Receive Data via SSL Connection

When an SSL connection is opened with **<access_mode>** specified as 0, the module will report URC as **+QSSLURC: "recv",<clientID>** when it receives data from the Internet. You can read the data from buffer by **AT+QSSLRECV**.

AT+QSSLRECV Receive Data via SSL Connection	
Test Command **AT+QSSLRECV=?**	Response **+QSSLRECV: (**range of supported **<clientID>**s**),(**range of supported **<readlen>**s**)** **OK**
Write Command **AT+QSSLRECV=<clientID>,<readlen>**	Response If the specified connection has received data: **+QSSLRECV: <have_readlen><CR><LF><data>** **OK** If the buffer is empty: **+QSSLRECV: 0** **OK** If the parameters are incorrect or the connection cannot be established: **ERROR**
Maximum Response Time	300 ms
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<clientID>	Integer type. Socket index. Range: 0–11.
<readlen>	Integer type. The length of data to be retrieved. Range: 1–1500. Unit: byte.
<have_readlen>	Integer type. The actual data length obtained by **AT+QSSLRECV**. Unit: byte.
<data>	The retrieved data. Unit: byte.

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 **<clientID>**s**),(**range of supported **<close_timeout>**s**)** **OK**
Write Command **AT+QSSLCLOSE=<clientID>[,<close_timeout>]**	Response If the SSL connection is successfully closed: **OK** If it is failed to close the connection: **ERROR**
Maximum Response Time	Determined by parameter **<close_timeout>**
Characteristics	The command takes effect immediately. The configurations will not be saved.

Parameter

<clientID>	Integer type. Socket index. Range: 0–11.
<close_timeout>	Integer type. The timeout of executing **AT+QSSLCLOSE**. Range: 0–65535. Default: 10. 0 means close the command execution immediately. The unit of **<close_timeout>** depends on the configuration of **AT+QSSLCFG="closetimemode"**, if **<close_time_mode>**=0, the unit of **<close_timeout>** is s; if **<close_time_mode>**=1, the unit of **<close_timeout>** is ms.

2.2.6. AT+QSSLSTATE Query the State of SSL Connection

The command queries the socket connection status and can only query the SSL connection status.

AT+QSSLSTATE Query the State of SSL Connection	
Test Command **AT+QSSLSTATE=?**	Response **OK**
Write Command **AT+QSSLSTATE=<clientID>**	Response **+QSSLSTATE: <clientID>,"SSLClient",<IP_address>,<remote_port>,<local_port>,<socket_state>,<PDP_ctxID>,<serverID>,<access_mode>,<AT_port>,<SSL_ctxID>**

	 OK
Execution Command **AT+QSSLSTATE**	Response List of **(+QSSLSTATE: <clientID>,"SSLClient",<IP_address>,<remote_port>,<local_port>,<socket_state>,<PDP_ctxID>,<serverID>,<access_mode>,<AT_port>,<SSL_ctxID>)** **OK**
Maximum Response Time	300 ms
Characteristics	/

Parameter

<clientID> Integer type. Socket index. Range: 0–11.
<IP_address> String type. The address of remote server.
<remote_port> Integer type. The port of remote server. Range: 0–65535.
<local_port> Integer type. The local port. Range: 0–65535.
<socket_state> Integer type. The state of SSL connection.

0	"Initial"	Connection has not been established
1	"Opening"	Client is connecting
2	"Connected"	Client connection has been established
4	"Closing"	Connection is closing

<PDP_ctxID> Integer type. PDP context ID. Range: 1–16.
<serverID> Integer type. Reserved.
<access_mode> Integer type. The access mode of SSL connection.

0 Buffer access mode
1 Direct push mode
2 Transparent access mode

<AT_port> String type. COM port.
<SSL_ctxID> Integer type. SSL context ID. Range: 0–5.

2.3. Description of URCs

2.3.1. +QSSLURC: "recv" Notify Received Data

The URC notifies received data which comes from peer.

+QSSLURC: "recv" Notify Received Data	
+QSSLURC: "recv",<clientID>	The URC of SSL data incoming in buffer access mode. SSL

	data can be received by **AT+QSSLRECV**.
+QSSLURC: "recv",<clientID>,<current_recvlength><CR><LF><data>	The URC of SSL data incoming in direct push mode.

Parameter

<clientID>	Integer type. Socket index. Range: 0–11.
<current_recvlength>	Integer type. The length of actual received data.
<data>	The received data.

2.3.2. +QSSLURC: "closed" Notify Abnormal Disconnection

The URC notifies that the connection has been disconnected. Disconnection can be caused by many reasons, such as the Internet closes the connection or the state of GPRS PDP is deactivated. The SSL connection state based on the specified socket will be “closing”. In such case, **AT+QSSLCLOSE=<connectID>** must be executed to change the SSL connection state to “initial”.

+QSSLURC: "closed" Notify Abnormal Close	
+QSSLURC: "closed",<clientID>	The SSL connection based on the specified socket is closed.

Parameter

<clientID>	Integer type. Socket index. Range: 0–11.

3 Examples

3.1. Configure and Activate a PDP Context

3.1.1. Configure a PDP Context

```
AT+QICSGP=1,1,"UNINET","","",1       //Configure context as 1. APN is "UNINET" for China Unicom.
OK
```

3.1.2. Activate a PDP Context

```
AT+QIACT=1                           //Activate context as 1.
OK                                   //Activated successfully.
AT+QIACT?                            //Query the state of context.
+QIACT: 1,1,1,"10.7.157.1"

OK
```

3.1.3. Deactivate a PDP Context

```
AT+QIDEACT=1                         //Deactivate context 1.
OK                                   //Deactivated successfully.
```

3.2. Configure an SSL Context

```
AT+QSSLCFG="sslversion",1,1              //Set SSL context ID and SSL version as 1.
OK
AT+QSSLCFG="ciphersuite",1,0X0035         //Set SSL context ID as 1 and SSL cipher suites as
                                            TLS_RSA_WITH_AES_256_CBC_SHA.
OK
AT+QSSLCFG="seclevel",1,1                  //Set SSL context ID as 1 and authentication mode as
                                            perform server authentication.
OK
AT+QSSLCFG="cacert",1,"RAM:cacert.pem" //Set SSL context ID as 1 and the path of the trusted CA
```

```
                                          certificate as RAM:cacert.pem.
OK
```

3.3. SSL Client Works in Buffer Access Mode

3.3.1. Set up an SSL Connection and Enter Buffer Access Mode

```
AT+QSSLOPEN=1,1,4,"220.180.239.212",8010,0
OK

+QSSLOPEN: 4,0                    //Set up an SSL connection successfully.
AT+QSSLSTATE                      //Query the status of all SSL connections.
+QSSLSTATE: 4,"SSLClient","220.180.239.212",8010,65344,2,1,4,0,"usbmodem",1

OK
```

3.3.2. Send Data in Buffer Access Mode

3.3.2.1. Send Variable-Length Data

```
AT+QSSLSEND=4                      //Send variable-length data.
>
Test data from SSL
<CTRL+Z>
SEND OK
```

3.3.2.2. Send Fixed-Length Data

```
AT+QSSLSEND=4,18                   //Send fixed-length data with the data length of 18 bytes.
>
Test data from SSL
SEND OK
```

3.3.3. Receive Data in Buffer Access Mode

```
+QSSLURC: "recv",4                //The socket 4 (<clientID>=4) has received data.

AT+QSSLRECV=4,1500                //Read data. The length of data to be retrieved is 1500 bytes.
```

```
+QSSLRECV: 18                          //The actual received data length is 18 bytes.
Test data from SSL

OK
AT+QSSLRECV=4,1500
+QSSLRECV: 0                           //No data in buffer.

OK
```

3.3.4. Close an SSL Connection

```
AT+QSSLCLOSE=4                         //Close an SSL connection (<clientID>=4). Depending on the
                                         network, the maximum response time is 10 s.
OK
```

3.4. SSL Client Works in Direct Push Mode

3.4.1. Set up an SSL Connection and Enter Direct Push Mode

```
AT+QSSLOPEN=1,1,4,"220.180.239.212",8011,1
OK

+QSSLOPEN: 4,0                         //Set up SSL connection successfully.
AT+QSSLSTATE                           //Query the status of all SSL connections.
+QSSLSTATE: 4,"SSLClient","220.180.239.212",8011,65047,2,1,4,1,"usbmodem",1

OK
```

3.4.2. Send Data in Direct Push Mode

```
AT+QSSLSEND=4                          //Send variable-length data.
>
Test data from SSL
<CTRL+Z>
SEND OK
AT+QSSLSEND=4,18                       //Send fixed-length data and the data length is 18 bytes.
>
Test data from SSL
SEND OK
```

3.4.3. Receive Data in Direct Push Mode

```
+QSSLURC: "recv",4,18
Test data from SSL
```

3.4.4. Close an SSL Connection

```
AT+QSSLCLOSE=4                        //Close a connection whose (<clientID>=4). Depending on the
                                        network, the maximum response time is 10 s.
OK
```

3.5. SSL Client Works in Transparent Access Mode

3.5.1. Set up an SSL Connection and Send Data in Transparent Access Mode

```
AT+QSSLOPEN=1,1,4,"220.180.239.212",8011,2    //Set up an SSL connection.
CONNECT                                       //Enter transparent access mode.
                               //Client is sending data from COM port to the Internet directly. (The data
                                 is not visible in the example.)
OK                             //Use +++ or DTR (set AT&D1 first) to exit transparent access mode.
                                 The NO CARRIER result code indicates that the server has stopped
                                 the SSL connection.
```

3.5.2. Set up an SSL Connection and Receive Data in Transparent Access Mode

```
AT+QSSLOPEN=1,1,4,"220.180.239.212",8011,2     //Set up an SSL connection.
CONNECT
<Received data>               //Client is reading the data.
OK                            //Use +++ or DTR (set AT&D1 first) to exit transparent access mode.
                                The NO CARRIER result code indicates that the server has stopped
                                the SSL connection.
```

3.5.3. Close an SSL Connection

```
AT+QSSLCLOSE=4              //Close a connection (<clientID>=4). Depending on the network, the
                              maximum response time is 10 s.
OK
```

4 Error Handling

4.1. Open SSL Connection Fails

If it is failed to open SSL connection, please check the following:

1. Query the status of the specified PDP context by **AT+QIACT?** to check whether the specified PDP context has been activated.

2. Since an invalid DNS server address cannot convert domain name to IP address, if the address of server is a domain name, please check whether the address of DNS server is valid by **AT+QIDNSCFG=<contextID>**.

3. Check the SSL configuration by **AT+QSSLCFG**, especially the SSL version and cipher suite to ensure that they are supported on server side. If **<seclevel>** has been configured as 1 or 2, then the trusted CA certificate has to be uploaded to the module with **AT+QFUPL**. If the server side has configured "SSLVerifyClient required", then the client certificate and client private key have to be uploaded to the module with **AT+QFUPL**. For details about certificate validity check, see ***Chapter 1.4***. And for more details of **AT+QFUPL**, see *Quectel_LTE_Standard_FILE_Application_Note*.

5 Summary of Error Codes

If an **ERROR** is returned after executing SSL AT commands, the details of error can be queried by **AT+QIGETERROR**. Please note that **AT+QIGETERROR** just returns error code of the last SSL AT command.

Table 5: Summary of Error Codes

<err>	Meaning
0	Operation successful
550	Unknown error
551	Operation blocked
552	Invalid parameter
553	Memory not enough
554	Create socket failed
555	Operation not supported
556	Socket bind failed
557	Socket listen failed
558	Socket write failed
559	Socket read failed
560	Socket accept failed
561	Open PDP context failed
562	Close PDP context failed
563	Socket identity has been used
564	DNS busy

565	DNS parse failed
566	Socket connection failed
567	Socket has been closed
568	Operation busy
569	Operation timeout
570	PDP context break down
571	Cancel send
572	Operation not allowed
573	APN not configured
574	Port busy

6 Appendix A References

Table 6: Related Documents

SN	Document Name	Remark
[1]	RFC 2246-The TLS Protocol Version 1.0	Transport Layer Security (TLS) protocol. It provides communications privacy over the Internet.
[2]	GSM 07.07	Digital cellular telecommunications (Phase 2+); AT command set for GSM Mobile Equipment (ME)
[3]	GSM 07.10	Support GSM 07.10 multiplexing protocol
[4]	Quectel_LTE_Standard_TCP(IP)_Application_Note	TCP/IP Application Note applicable for EC2x series, EG9x series, EG2x-G and EM05 series modules
[5]	Quectel_LTE_Standard_FILE_Application_Note	FILE application note applicable for EC2x series, EG9x series, EG2x-G and EM05 series modules
[6]	Quectel_EC2x&EG9x&EG2x-G&EM05_Series_AT_Commands_Manual	AT commands manual applicable for EC2x series, EG9x series, EG2x-G and EM05 series modules

Table 7: Terms and Abbreviations

Abbreviation	Description
ALPN	Application Layer Protocol Negotiation
APN	Access Point Name
CA	Certificate Authority
DNS	Domain Name Server
DTR	Data Terminal Ready
DTLS	Datagram Transport Layer Security

PDP	Packet Data Protocol
SNI	Server Name Indication
SSL	Security Socket Layer
TCP/IP	Transmission Control Protocol/Internet Protocol
TLS	Transport Layer Security
UART	Universal Asynchronous Receiver/Transmitter
URC	Unsolicited Result Code
USB	Universal Serial Bus