EC2x&EG9x&EM05 SSL AT Commands Manual

LTE Module Series

Rev. EC2x&EG9x&EM05_SSL_AT_Commands_Manual_V1.0

Date: 2017-11-22

Status: Released

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

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

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

For technical support, or to report documentation errors, please visit:
http://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

History

Revision	Date	Author	Description
1.0	2017-11-22	Duke XIN/ Jessica GENG	Initial

Contents

Table Index

1 Introduction

This document describes how to use the SSL functionality of Quectel EC2x&EG9x&EM05 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 it can prevent data from eavesdropping, tampering, or forging during the communication process. The SSL function meets these demands.

This document is applicable to following Quectel modules.

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

1.1. SSL Version and Cipher Suite

The following SSL versions are supported.

Table 1: SSL Versions

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

The following table shows SSL cipher suites supported by Quectel EC2x&EG9x&EM05 modules. For detailed description of cipher suites, please refer to *RFC 2246-The TLS Protocol Version 1.0*.

Table 2: 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 cipher suites 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. Please refer to *Quectel_EC2x&EG9x&EM05_TCP(IP)_AT_Commands_Manual* for details.

Step 2: Activate the PDP context by AT+QIACT, then the assigned IP address can be queried by AT+QIACT?. Please refer to *Quectel_EC2x&EG9x&EM05_TCP(IP)_AT_Commands_Manual* for details.

Step 3: Configure the SSL version, cipher suite, path of trusted CA certificate and the security level for the specified SSL context by AT+QSSLCFG.

Step 4: Open SSL client connection by AT+QSSLOPEN. <sslctxID> 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 under each access mode, please refer to ***Chapter 1.3***.

Step 6: Close SSL connection by AT+QSSLCLOSE.

Step 7: Deactivate the PDP context by AT+QIDEACT.

1.3. Description of Data Access Modes

The SSL connection supports the following three kinds of 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 parameter <access_mode>. After the SSL connection is established, customers can switch the access mode via AT+QISWTMD.

1. In buffer access mode, data can be sent via AT+QSSLSEND command, and if the module has received data from the Internet, it will report a URC as "+QSSLURC: "recv",<clientID>". Customers can retrieve data via AT+QSSLRECV command.

2. In direct push mode, data can be sent via AT+QSSLSEND command, 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 following format: "+QSSLURC: "recv",<clientID>,<currectrecvlength><CR><LF><data>".

3. In transparent access mode, the corresponding port enters into 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. Customers can use "+++" or DTR (AT&D1 should be set first) to switch to buffer access mode. In transparent access mode, if SSL connection encounters any abnormal disconnection, the module will report "NO CARRIER".

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

   1) Do not input any character within 1s 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 1s after "+++" has been inputted.
   4) Use "+++" or DTR (AT&D1 should be set first) to make the module exit from transparent access mode, and wait until "OK" is returned.

5. There are two methods to return back to transparent access mode:

   1) By AT+QISWTMD. Specify the <access_mode> as 2 when executing this command. If entering into transparent access mode successfully, "CONNECT" will be returned.
   2) By ATO. ATO will change the access mode of connection that exits from transparent access mode lately. If entering transparent access mode successfully, "CONNECT" will be returned. If there is no connection entering into transparent access mode before, ATO will return "NO CARRIER".

1.4. Validity Check for 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 for certification is required (<ignoreltime> set as 0 when executing AT+QSSLCFG), in order to avoid failure of certificate validity check, AT+CCLK command should be used to configure the module time within the validity time period of the certificate.

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. Description of AT Commands

2.1.1. AT+QSSLCFG Configure Parameters of an SSL Context

The command can be used to configure the SSL version, cipher suites, security level, CA certificate, client certificate and client key. These parameters will be used in the handshake procedure.

<sslctxID> 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",(0-5),(0-4)** **+QSSLCFG: "ciphersuite",(0-5),(0X0035,0X002F,0X0005,0X0004,0X000A,0X003D,0XC002,0XC003,0XC004,0XC005,0XC007,0XC008,0XC009,0XC00A,0XC011,0XC012,0XC013,0XC014,0XC00C,0XC00D,0XC00E,0XC00F,0XC023,0XC024,0XC025,0XC026,0XC027,0XC028,0XC029,0XC02A,0XC02F,0XFFFF)** **+QSSLCFG: "cacert",(0-5),<cacertpath>** **+QSSLCFG: "clientcert",(0-5),<client_cert_path>** **+QSSLCFG: "clientkey",(0-5),<client_key_path>** **+QSSLCFG: "seclevel",(0-5),(0-2)** **+QSSLCFG: "ignorelocaltime",(0-5),(0,1)** **+QSSLCFG: "negotiatetime",(0-5),(10-300)** **+QSSLCFG: "sni",(0-5),(0,1)** **OK**
Write Command Configure the SSL version for the specified SSL context: **AT+QSSLCFG="sslversion",<sslctxID**	Response If <sslversion> is omitted, query the SSL version for the specified SSL context, and response: **+QSSLCFG: "sslversion",<sslctxID>,<sslversion>**

>[,<sslversion>]	**OK** If <sslversion> is not omitted, set the SSL version for the specified SSL context, and response: **OK** Or **ERROR**
Write Command Configure the SSL cipher suites for the specified SSL context: **AT+QSSLCFG="ciphersuite",<sslctxID>[,<ciphersuites>]**	Response If <ciphersuites> is omitted, query the SSL cipher suites for the specified SSL context, and response: **+QSSLCFG: "ciphersuite",<sslctxID>,<ciphersuites>** **OK** If <ciphersuites> is not omitted, set the SSL cipher suite for the specified SSL context, and response: **OK** Or **ERROR**
Write Command Configure the path of trusted CA certificate for the specified SSL context: **AT+QSSLCFG="cacert",<sslctxID>[,<cacertpath>]**	Response If <cacertpath> is omitted, query the path of trusted CA certificate for the specified SSL context, and response: **+QSSLCFG: "cacert",<sslctxID>,<cacertpath>** **OK** If <cacertpath> is not omitted, set the path of trusted CA certificate for the specified SSL context, and response: **OK** Or **ERROR**
Write Command Configure the path of client certificate for the specified SSL context: **AT+QSSLCFG="clientcert",<sslctxID>[,<client_cert_path>]**	Response If <client_cert_path> is omitted, query the path of client certificate for the specified SSL context, and response: **+QSSLCFG: "clientcert",<sslctxID>,<client_cert_path>** **OK** If <client_cert_path> is not omitted, set the path of client certificate for the specified SSL context, and response: **OK** Or **ERROR**

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

	OK If <negotiate_time> is not omitted, set the maximum timeout in SSL negotiation stage for the specified SSL context, and response: **OK** Or **ERROR**
Write Command Configure Server Name Indication feature for the specified SSL context: **AT+QSSLCFG="sni",<sslctxID>[,<sni>]**	Response If <sni> is omitted, query whether the Server Name Indication feature is enabled for the specified SSL context, and response: **+QSSLCFG: "sni",<sslctxID>,<sni>** **OK** If <sni> is not omitted, disable/enable Server Name Indication feature for the specified SSL context, and response: **OK** Or **ERROR**

Parameter

<sslctxID>	Numeric type. SSL context ID. The range is 0-5.
<sslversion>	Numeric type. SSL Version. 0 SSL3.0 1 TLS1.0 2 TLS1.1 3 TLS1.2 4 All
<ciphersuites>	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 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
<ignoreltime>	Numeric format. How to deal with expired certificate.	
	0	Care about validity check for certification
	1	Ignore validity check for certification
<cacertpath>	String format. The path of the trusted CA certificate.	
<client_cert_path>	String format. The path of the client certificate.	
<client_key_path>	String format. The path of the client private key.	
<seclevel>	Numeric format. The authentication mode.	
	0	No authentication
	1	Manage server authentication
	2	Manage server and client authentication if requested by the remote server
<negotiate_time>	Numeric format. Indicates maximum timeout used in SSL negotiation stage. Range: 10-300. The default value is 300. Unit: second.	
<sni>	Numeric format. Disable/enable Server Name Indication feature	
	0	Disable
	1	Enable

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

The command is used to set up an SSL connection. 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, AT+QIACT command should be executed 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 command 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:** **(1-16),(0-5),(0-11),<serveraddr>,<server_port>[,(0-2)]** **OK**
Write Command **AT+QSSLOPEN=<pdpctxID>,<sslctxID>,<clientID>,<serveraddr>,<server_port>[,<access_mode>]**	Response If the <access_mode> is transparent access mode and the SSL connection is successfully set up, response: **CONNECT** If there is any error, response: **ERROR** Error description can be got via AT+QIGETERROR. If the <access_mode> is buffer access mode or direct push mode, response: **OK** **+QSSLOPEN: <clientID>,<err>** <err> is 0 when SSL socket is opened successfully, otherwise <err> is not 0. If there is any error, response: **ERROR** Error description can be got via AT+QIGETERROR.
Maximum Response Time	Maximum network response time of 150s, plus configured time of <negotiate_time>.

Parameter

<pdpctxID>	Numeric type. PDP context ID. The range is 1-16.
<sslctxID>	Numeric type. SSL context ID. The range is 0-5.
<clientID>	Numeric type. Socket index. The range is 0-11.

<serveraddr>	String type. The address of remote server.
<server_port>	Numeric type. The listening port of remote server.
<access_mode>	Numeric 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. Please refer to ***Chapter 5***.

2.1.3. 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: (0-11)[,(1-1460)]** **OK**
Write Command **AT+QSSLSEND=<clientID>** 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 connection has been established and sending is successful, response: **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, response: **ERROR**
Write Command **AT+QSSLSEND=<clientID>,<sendlen>** After response ">", input data until the data length is equal to <sendlen>.	Response **>** **<input data with specified length>** If connection has been established and sending is successful, response: **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, response: **ERROR**

Parameter

<clientID>	Numeric type. Socket index. The range is 0-11.
<sendlen>	Numeric type. The length of sending data. The range is 1-1460. Unit: byte.

2.1.4. 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. Customers can read the data from buffer by AT+QSSLRECV command.

AT+QSSLRECV Receive Data via SSL Connection	
Test Command **AT+QSSLRECV=?**	Response **+QSSLRECV: (0-11),(1-1500)** **OK**
Write Command **AT+QSSLRECV=<clientID>,<readlen>**	Response If the specified connection has received data, response: **+QSSLRECV: <havereadlen><CR><LF><data>** **OK** If the buffer is empty, response: **+QSSLRECV: 0** **OK** If the parameters are incorrect or the connection cannot be established, response: **ERROR**

Parameter

<clientID>	Numeric type. Socket index. The range is 0-11.
<readlen>	Numeric type. The length of data to be retrieved. The range is 1-1500. Unit: byte.
<havereadlen>	Numeric type. The actual data length obtained by AT+QSSLRECV. Unit: byte.

<data>	The retrieved data.

2.1.5. AT+QSSLCLOSE Close an SSL Connection

The command is used to close 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: (0-11)[,(0-65535)]** **OK**
Write Command **AT+QSSLCLOSE=<clientID>[,<close_timeout>]**	Response If successfully closed, response: **OK** If failed to close, response: **ERROR**

Parameter

<clientID>	Numeric type. Socket index. The range is 0-11.
<close_timeout>	Numeric type. The timeout value of AT+QSSLCLOSE. The range is 0-65535, and the default value is 10. Unit: second. 0 means close immediately.

2.1.6. AT+QSSLSTATE Query the State of SSL Connection

The command is used to query the socket connection status. It can only query the status of SSL connection.

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>,<pdpctxID>,<serverID>,<access_mode>,<AT_port>,<sslctxID>** **OK**
Execution Command	Response

AT+QSSLSTATE	List of **(+QSSLSTATE:** **<clientID>,"SSLClient",<IP_address>,<remote_port>,<local_port>,<socket_state>,<pdpctxID>,<serverID>,<access_mode>,<AT_port>,<sslctxID>)** **OK**

Parameter

<clientID>	Numeric type. Socket index. The range is 0-11.		
<IP_address>	String type. The address of remote server.		
<remote_port>	Numeric type. The port of remote server.		
<local_port>	Numeric type. The local port.		
<socket_state>	Numeric 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
<pdpctxID>	Numeric type. PDP context ID. The range is 1-16.		
<serverID>	Numeric type. Reserved.		
<access_mode>	Numeric 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.		
<sslctxID>	Numeric type. SSL context ID. The range is 0-5.		

2.2. Description of URC

2.2.1. Notify Received Data

Notify received data which comes from peer.

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>,<currentrecvlength><CR><LF><data>	The URC of SSL data incoming in direct push mode.

Parameter

<clientID>	Integer type. Socket index. The range is 0-11.
<currentrecvlength>	Integer type. The length of actual received data.
<data>	The received data.

2.2.2. Notify Abnormal Close

Notify that the connection has been disconnected. Lots of reasons can cause this phenomenon, 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”.

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

Parameter

<clientID>	Integer type. Socket index. The range is 0-11.

3 Examples

3.1. Configure and Activate a PDP Context

3.1.1. Configure a PDP Context

```
AT+QICSGP=1,1,"UINET","","",1        //Configure context 1. APN is “UINET” for China Unicom.
OK
```

3.1.2. Activate a PDP Context

```
AT+QIACT=1                           //Activate context 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
OK
AT+QSSLCFG="ciphersuite",1,0X0035
OK
AT+QSSLCFG="seclevel",1,1
OK
AT+QSSLCFG="cacert",1,"RAM:cacert.pem"
OK
```

3.3. SSL Client Works in Buffer Access Mode

3.3.1. Set up an SSL Connection and Enter into 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

```
AT+QSSLSEND=4                     //Send changeable 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
```

3.3.3. Receive Data in Buffer Access Mode

```
+QSSLURC: "recv",4                //The <clientID> 4 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 a connection whose <clientID> is 4. Depending on the
                                    network, the maximum response time is 10s.
OK
```

3.4. SSL Client Works in Direct Push Mode

3.4.1. Set up an SSL Connection and Enter into 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 changeable 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> is 4. Depending on the
                                    network, the maximum response time is 10s.
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 into 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 (AT&D1 should be set first) to exit from 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 (AT&D1 should be set first) to exit from 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 whose <connectID> is 4. Depending on the network, the maximum response time is 10s.
OK
```

4 Error Handling

4.1. Open SSL Connection Fails

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

1. Query the status of the specified PDP context: use AT+QIACT? command to check whether the specified PDP context has been activated.

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

3. Please check the SSL configuration by AT+QSSLCFG command, especially the SSL version and cipher suite, so as to make sure they are supported on server side. If <seclevel> has been configured as 1 or 2, customers must upload trusted CA certificate to the module by FILE AT command. If the server side has configured “SSLVerifyClient required”, then the customer must upload the client certificate and client private key to the module by FILE AT commands. For details about certificate validity check, please refer to ***Chapter 1.4***. And for more details about related FILE AT commands, please refer to *Quectel_EC2x&EG9x&EM05_FILE_AT_Commands_Manual*.

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 command just returns error code of the last SSL AT command.

Table 3: 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 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_EC2x&EG9x&EM05_TCP(IP)_AT_Commands_Manual	Introduction about EC2x&EG9x&EM05 TCP/IP AT commands
[4]	Quectel_EC2x&EG9x&EM05_FILE_AT_Commands_Manual	Introduction about EC2x&EG9x&EM05 FILE AT commands

Table 5: Terms and Abbreviations

Abbreviation	Description
DNS	Domain Name Server
DTR	Data Terminal Ready
PDP	Packet Data Protocol
SSL	Security Socket Layer