BG95&BG77 SSL Application Note

LPWA Module Series

Rev. BG95&BG77_SSL_Application_Note_V1.0

Date: 2019-10-12

Status: Released

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

Quectel Wireless Solutions Co., Ltd.
Building 5, Shanghai Business Park Phase III (Area B), No.1016 Tianlin Road, Minhang District, Shanghai, China 200233
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

COPYRIGHT

About the Document

History

Revision	Date	Author	Description
1.0	2019-10-12	Terrence YANG	Initial

Contents

Table Index

1 Introduction

This document describes how to use the SSL functionality of Quectel BG95 and BG77 modules. In some cases, in order to ensure communication privacy, the communication between the server and the client should be in an encrypted way to prevent data from eavesdropping, tampering or forging during the communication process. The SSL function meets these demands.

1.1. SSL Versions and Cipher Suites

The following are SSL versions supported by BG95 and BG77 modules.

Table 1: SSL Versions

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

The following table shows SSL cipher suites supported by BG95 and BG77 modules. For detailed description of cipher suites, please refer to *RFC 2246-The TLS Protocol Version 1.0*.

Table 2: Supported SSL Cipher Suites

Code of Cipher Suites	Name 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** command. Please refer to ***document [3]*** for details of the command.

Step 2: Activate the PDP context by **AT+QIACT** command, then the assigned IP address can be queried by **AT+QIACT?** command. Please refer to ***document [3]*** for details of the command.

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

Step 4: Open SSL client connection by **AT+QSSLOPEN** command. **<sslctxID>** is used to specify the SSL context, and **<access_mode>** is used to specify the 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 different data access modes, please refer to ***Chapter 1.3***.

Step 6: Close SSL connection by **AT+QSSLCLOSE** command.

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

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** command, the data access mode can be specified by the parameter **<access_mode>**. After the SSL connection has been established, **AT+QISWTMD** command can be used to switch the data access mode.

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

2. In direct push mode, data are 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>,<currentrecvlength><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 the Internet will be outputted to COM port directly. **+++** or DTR (**AT&D1** should be set first) can be used to switch the data access mode to buffer access mode. In transparent access mode, if any abnormal SSL disconnection happens, 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 (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 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** command. Specify the **<access_mode>** as 2 when executing this command. If entering into transparent access mode successfully, **CONNECT** will be returned.
   2) By **ATO** command. **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** command will return **NO CARRIER**.

1.4. Certificate Validity Check

To check whether a certificate is in the validity period, the certificate must be parsed. 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 certificate validity check is required (**<ignore_localetime>** is set as 0 when executing **AT+QSSLCFG** command), in order to avoid validity check failure, **AT+CCLK** command should be used to configure the module time within the validity time period of the certificate.

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),<clientcertpath>** **+QSSLCFG: "clientkey",(0-5),<clientkeypath>** **+QSSLCFG: "seclevel",(0-5),(0-2)** **+QSSLCFG: "session",(0-5),(0-1)** **+QSSLCFG: "sni",(0-5),(0-1)** **+QSSLCFG: "checkhost",(0-5),(0-1)** **+QSSLCFG: "ignorelocaltime",(0-5),(0,1)** **+QSSLCFG: "negotiatetime",(0-5),(10-300)** **+QSSLCFG: "dtls",(0-5),(0-1)** **+QSSLCFG: "dtlsversion",(0-5),(0-1)** **OK**
Write Command Configure the SSL version for a specified	Response If **<sslversion>** is omitted, query the SSL version for the

SSL context: **AT+QSSLCFG="sslversion",<sslctxID>[,<sslversion>]**	specified SSL context: **+QSSLCFG: "sslversion",<sslctxID>,<sslversion>** **OK** If **<sslversion>** is present, set the SSL version for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the SSL cipher suites for a specified SSL context: **AT+QSSLCFG="ciphersuite",<sslctxID>[,<ciphersuites>]**	Response If **<ciphersuites>** is omitted, query the SSL cipher suites for the specified SSL context: **+QSSLCFG: "ciphersuite",<sslctxID>,<ciphersuites>** **OK** If **<ciphersuites>** is present, set the SSL cipher suite for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of trusted CA certificate for a 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: **+QSSLCFG: "cacert",<sslctxID>,<cacertpath>** **OK** If **<cacertpath>** is present, set the path of trusted CA certificate for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the path of client certificate for a specified SSL context: **AT+QSSLCFG="clientcert",<sslctxID>[,<clientcertpath>]**	Response If **<clientcertpath>** is omitted, query the path of client certificate for the specified SSL context: **+QSSLCFG: "clientcert",<sslctxID>,<clientcertpath>** **OK** If **<clientcertpath>** is present, set the path of client certificate for the specified SSL context: **OK**

	Or **ERROR**
Write Command Configure the path of client private key for a specified SSL context: **AT+QSSLCFG="clientkey",<sslctxID>[,<clientkeypath>]**	Response If **<clientkeypath>** is omitted, query the path of client private key for the specified SSL context: **+QSSLCFG: "clientkey",<sslctxID>,<clientkeypath>** **OK** If **<clientkeypath>** is present, set the path of client private key for the specified SSL context: **OK** Or **ERROR**
Write Command Configure the authentication mode for a specified SSL context: **AT+QSSLCFG="seclevel",<sslctxID>[,<seclevel>]**	Response If **<seclevel>** is omitted, query the authentication mode for the specified SSL context: **+QSSLCFG: "seclevel",<sslctxID>,<seclevel>** **OK** If **<seclevel>** is present, set the authentication mode for the specified SSL context: **OK** Or **ERROR**
Write Command Configure SSL Resumption feature for a specified SSL context: **AT+QSSLCFG="session",<sslctxID>[,<session>]**	Response If **<session>** is omitted, query whether SSL Resumption feature is enabled for the specified SSL context: **+QSSLCFG: "session",<sslctxID>,<session>** **OK** If **<session>** is present, set whether or not to enable SSL Resumption feature for the specified SSL context: **OK** Or **ERROR**
Write Command Configure Server Name Indication feature for a specified SSL context: **AT+QSSLCFG="sni",<sslctxID>[,<sni>]**	Response If **<sni>** is omitted, query whether Server Name Indication feature is enabled for the specified SSL context: **+QSSLCFG: "sni",<sslctxID>,<sni>** **OK**

	If **<sni>** is present, set whether or not to enable Server Name Indication feature for the specified SSL context: **OK** Or **ERROR**
Write Command Configure hostname validation feature for a specified SSL context: **AT+QSSLCFG="checkhost",<sslctxID>[,<checkhost>]**	Response If **<checkhost>** is omitted, query whether hostname validation feature is enabled for the specified SSL context: **+QSSLCFG: "checkhost",<sslctxID>,<checkhost>** **OK** If **<checkhost>** is present, set whether or not to enable hostname validation feature for the specified SSL context: **OK** Or **ERROR**
Write Command Configure whether to ignore certificate validity check for a specified SSL context: **AT+QSSLCFG="ignorelocaltime",<sslctxID>[,<ignore_localtime>]**	Response If **<ignore_localtime>** is omitted, query whether the certificate validity check is ignored for the specified SSL context: **+QSSLCFG: "ignorelocaltime",<sslctxID>,<ignore_localtime>** **OK** If **<ignore_localtime>** is present, 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 a 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: **+QSSLCFG: "negotiatetime",<sslctxID>,<negotiate_time>** **OK** If **<negotiate_time>** is present, set the maximum timeout in SSL negotiation stage for the specified SSL context: **OK** Or

	ERROR
Write Command Enable/disable DTLS feature for a specified SSL context: **AT+QSSLCFG="dtls",<sslctxID>[,<dtls_enable>]**	Response If **<dtls_enable>** is omitted, query the DTLS feature for the specified SSL context: **+QSSLCFG: "dtls",<sslctxID>,<dtls_enable>** **OK** If **<dtls_enable>** is present, enable/disable DTLS feature for the specified SSL context: **OK** Or **ERROR**
Write Command Configure DTLS version for a specified SSL context: **AT+QSSLCFG="dtlsversion",<sslctxID>[,<dtlsversion>]**	Response If **<dtlsversion>** is omitted, query the DTLS version for the specified SSL context: **+QSSLCFG: "dtlsversion",<sslctxID>,<dtlsversion>** **OK** If **<dtlsversion>** is present, set the DTLS vision for the specified SSL context: **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
<ignore_localtime>	Numeric format. How to deal with certificate validity check.	
	0	Does not ignore validity check for certificate
	1	Ignore validity check for certificate
<cacertpath>	String format. The path of the trusted CA certificate.	
<clientcertpath>	String format. The path of the client certificate.	
<clientkeypath>	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
<session>	Numeric format. Whether to enable SSL Resumption feature.	
	0	Disable SSL Resumption
	1	Enable SSL Resumption
<sni>	Numeric format. Whether to enable Server Name Indication feature, currently, the only server names supported are DNS hostnames.	
	0	Disable Server Name Indication
	1	Enable Server Name Indication
<checkhost>	Numeric format. Whether to enable hostname validation feature (Subject	

	CommonName(CN) matches the specified host name). 0 Disable hostname validation 1 Enable hostname validation
<negotiate_time>	Numeric format. Indicates maximum timeout used in SSL negotiation stage. The value rang is 10-300, and the default value is 300. Unit: second.
<dtls_enable>	Numeric format. SSL version. 0 Disable DTLS feature 1 Enable DTLS feature
<dtlsvision>	Numeric format. Whether enable DTLS feature. 0 DTLS1.0 1 DTLS1.2 2 All

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** command 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.

As 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: **CONNECT** If the **<access_mode>** is buffer access mode or direct push mode: **OK** **+QSSLOPEN: <clientID>,<err>**

	If there is any error: **ERROR** Error description can be got via **AT+QIGETERROR** command.
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***.
<negotiate_time>	Please refer to AT+QSSLCFG command for details.

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 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 the connection has been established but sending buffer is full: **SEND FAIL** If the connection has not been established, abnormally

	closed, or any parameter is incorrect: **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 the connection has been established and sending is successful: **SEND OK** If the connection has been established but sending buffer is full: **SEND FAIL** If the connection has not been established, abnormally closed, or the parameter is incorrect: **ERROR**

Parameter

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

2.1.4. AT+QSSLRECV Retrieve Data via SSL Connection

When an SSL connection is opened with **<access_mode>** specified as 0 (buffer access mode), the module will report URC **+QSSLURC: "recv",<clientID>** when it receives data from the Internet. The received data can be read from buffer with **AT+QSSLRECV** command.

AT+QSSLRECV Retrieve 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: **+QSSLRECV: <havereadlen><CR><LF><data>** **OK** When the buffer is empty:

	+QSSLRECV: 0 **OK** If any parameter is incorrect or the connection cannot be established: **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 and the default value is 1500. Unit: byte.
<havereadlen>	Numeric type. The actual data length obtained by **AT+QSSLRECV** command. 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 the SSL connection is successfully closed: **OK** If it is failed to close the connection: **ERROR**

Parameter

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

2.1.6. AT+QSSLSTATE Query the State of SSL Connections

The command is used to query the state of SSL connections.

AT+QSSLSTATE Query the State of SSL Connections	
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 **AT+QSSLSTATE**	Response List of **(+QSSLSTATE: <clientID>,"SSLClient",<IP_address>,<remote_port>,<local_port>,<socket_state>,<pdpctxID>,<serverID>,<access_mode>,<AT_port>,<sslctxID>)**s **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. +QSSLURC: "recv" URC Indicating Incoming Data

+QSSLURC: "recv" URC is used to indicate incoming data in buffer access mode and direct push mode.

+QSSLURC: "recv" URC Indicating Incoming Data	
+QSSLURC: "recv",<clientID>	Indicate SSL data incoming in buffer access mode. SSL data can be retrieved by **AT+QSSLRECV** command.
+QSSLURC: "recv",<clientID>,<currentrecvlength><CR><LF><data>	Indicate 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. +QSSLURC: "closed" URC Indicating Abnormal Close

+QSSLURC: "closed" URC is used to indicate that the SSL 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".

+QSSLURC: "closed" URC Indicating 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,"CMNBIOT","","",1        //Configure context 1. APN is "CMNBIOT".
OK
```

3.1.2. Activate a PDP Context

```
AT+QIACT=1                             //Activate context 1.
OK                                     //Activated the context successfully.

AT+QIACT?                              //Query the context state, protocol type and IP address of
                                       context 1..
+QIACT: 1,1,1,"100.142.162.0"

OK
```

3.1.3. Deactivate a PDP Context

```
AT+QIDEACT=1                           //Deactivate context 1.
OK                                     //Deactivated the context 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,"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 the SSL connection successfully.

AT+QSSLSTATE                      //Query the state 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 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
```

3.3.3. Retrieve Data in Buffer Access Mode

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

AT+QSSLRECV=4,1500          //Retrieve the data. The length of data to be retrieved is 1500 bytes.
+QSSLRECV: 18               //The retrieved data length is 18 bytes.
Test data from SSL

OK
```

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

OK
```

3.3.4. Close an SSL Connection

```
AT+QSSLCLOSE=4                  //Close the connection (<clientID>=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 Direct Push Mode

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

+QSSLOPEN: 4,0                  //Set up the 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. Retrieve Data in Direct Push Mode

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

3.4.4. Close an SSL Connection

```
AT+QSSLCLOSE=4                    //Close the connection (<clientID>=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 transparent access mode.
                                               //The 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 result code NO CARRIER indicates that the server has stopped the SSL connection.
```

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

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

3.5.3. Close an SSL Connection

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

OK
```

4 Error Handling

4.1. Failed to Open SSL Connection

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

1. Query the status of the specified PDP context by **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, then the trusted CA certificate must be uploaded to the module by **AT+QFUPL** command. If the server has configured "SSLVerifyClient required", then the client certificate and client private key must be uploaded to the module by **AT+QFUPL** command. For details about certificate validity check, please refer to ***Chapter 1.4***. And for more details about related FILE AT commands, please refer to ***document [4]***.

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** command. 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	GSM 07.10 multiplexing protocol
[3]	Quectel_BG95&BG77_TCP(IP)_Application_Note	TCP/IP Application Note for BG95 and BG77 modules
[4]	Quectel_BG95&BG77_FILE_Application_Note	FILE Application Note for BG95 and BG77 modules

Table 5: Terms and Abbreviations

Abbreviation	Description
DNS	Domain Name Server
DTR	Data Terminal Ready
PDP	Packet Data Protocol
SSL	Security Socket Layer
DTLS	Datagram Transport Layer Security
SNI	Server Name Indication