GSM SSL TCP Application Note

GSM/GPRS Module Series

Rev. GSM_SSL_TCP_Application_Note_V3.1

Contents

Table Index

1 Introduction

This document describes how to use the SSL functionality of standard module.

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 being eavesdropped, tampered, or forged during the communication process. The SSL function meets these demands.

This document is applicable to M10, M26, M35 and M50 modules.

1.1. SSL Version and CipherSuite

So far, several SSL versions have been released. They are SSL2.0, SSL3.0, TLS1.0, TLS1.1, and TLS1.2. The following versions are supported by modules.

Table 1: SSL Version

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

The following table shows the names of the CipherSuites that module supports. Please refer to RFC 2246-The TLS Protocol Version 1.0 on the CipherSuite definitions for details.

Table 2: SSL CipherSuite

CipherSuite Code	CipherSuite Name
0X0035	TLS_RSA_WITH_AES_256_CBC_SHA
0X0005	TLS_RSA_WITH_RC4_128_SHA

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

1.2. The Procedure of Using SSL Function

Step 1: Install certificate and key to file system by command “AT+QSECWRITE”. “AT+QSECDEL” is used to delete the certificate and key, and “AT+QSECREAD” is used to check the checksum of certificate and key.

Step 2: Configure the APN, Username, Password of the context, and so on by command “AT+QICSGP”. The command “AT+QIREGAPP” is used to register to TCP/IP stack.

Step 3: Activate GPRS PDP context by command “AT+QIACT”. After the PDP context is activated, the command “AT+QILOCIP” is used to query the local IP address.

Step 4: Enable multiple TCP/IP session by command “AT+QIMUX=1”.

Step 5: Configure SSL version, CipherSuite, server authentication and client authentication, the CA certificate , client certificate and client key by command “AT+QSSLCFG”.

Step 6: Setup a SSL connection by command “AT+QSSLOPEN”. If connection is successful, the response will be “+QSSLOPEN: <ssid>,<connectcode>”.

Step 7: After the connection is established, send data by command “AT+QSSLSEND”. And when the host receives data from the peer, the URC “+QSSLURC: “recv”,<cid>,<ssid>” will notify the host to acquire data. The host should execute the command “AT+QSSLRECV” to read data continuously until all the data is read out.

Step 8: When data transmission is accomplished, close the SSL connection by command “AT+QSSLCLOSE”.

Step 9: Deactivate GPRS PDP context by command “AT+QIDEACT”.

NOTE

For detailed syntax information about AT commands of QICSGP, QIACT, QILOCIP, QIMUX and QIDEACT, please refer to *Mxx_AT_Commands_Manual*. For other AT commands, please refer to the corresponding documentation of the corresponding module.

1.3. SSL Function Coexists with Normal TCPIP Session

SSL connection can coexist with normal TCP connection. That is, you can set up one or several SSL connections and one or several normal TCP connections at the same time.

In the same foreground context, you should set up the SSL connection and the normal TCP connection with different socket index. For example, you can set up a normal TCP connection with socket index one, and set up a SSL connection with socket index three. But you could not use the same socket index to set up normal TCP connection and the SSL connection.

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

Step 1: Execute command “AT+QICSGP” to configure the APN, Username, Password of the context and so on. The command “AT+QIREGAPP” is used to register to the TCP/IP stack.

Step 2: Execute command “AT+QIACT” to activate GPRS PDP context. After the PDP context is activated, query the local IP address by command “AT+QILOCIP”.

Step 3: Execute command “AT+QIMUX=1” to enable multiple TCP/IP session.

Step 4: Execute the command “AT+QIOPEN” to establish a normal TCP connection, specify the <index> as one. After the normal TCP connection is established successfully, you can send data via the command “AT+QISEND” and receive data via the command “AT+QIRD”, and if you want to close the connection, you can execute the command “AT+QICLOSE”. For detailed syntax information about QIOPEN, QISEND, QIRD, QICLOSE, please refer to *Mxx_AT_Commands_Manual*.

Step 5: Execute the command “AT+QSSLOPEN” to establish a SSL connection, specify the <ssid> as three. After the connection is established successfully, the command “AT+QSSLSEND” is used to send data. And when the module receives data from the peer, the URC “+QSSLURC: “recv”,<cid>,<ssid>” will notify the host to read data. The host can execute the command “AT+QSSLRECV” to read data. When data transmission is accomplished, close the SSL connection by command “AT+QSSLCLOSE”.

Step 6: Deactivate GPRS PDP context by command “AT+QIDEACT”.

1.4. Error Handling

1.4.1. PDP Activation Fails

If you failed to activate PDP context by AT+QIACT command, please check the following aspects:

1. Query whether the PS domain is attached by AT+CGATT? command, if not, execute AT+CGATT=1 to attach PS domain.
2. Query the CGREG status by AT+CGREG? and make sure the PS domain is registered to.
3. Query the PDP context parameters by AT+QIREGAPP command, make sure the APN of specified

PDP context is set.

4. Make sure the specified PDP context ID is neither used by PPP nor activated by AT+CGACT command.

If the result of checking is OK, but the result of executing AT+QIACT command always fails, please reboot the module to resolve this issue. After booting the module, please check the terms mentioned above at least three times and each time at an interval of 10 minutes to avoid frequently rebooting the module.

2 Description of AT Command

2.1. AT Command Syntax

Test Command	**AT+<*x*>=?**	This command returns the list of parameters and value ranges set by the corresponding Write Command or internal processes.
Read Command	**AT+<*x*>?**	This command returns the currently set value of the parameter or parameters.
Write Command	**AT+<x>=<…>**	This command sets the user-definable parameter values.
Execution Command	**AT+<x>**	This command reads non-variable parameters affected by internal processes in the GSM engine

2.2. Description of AT Command

2.2.1. AT+QSSLCFG SSL Configuration

This AT command is used to configure the SSL version, CipherSuite, secure level, CA certificate, client certificate, client key, ignore RTC time, HTTP/HTTPS, and SMTP/SMTPS. These parameters will be used in the handshake procedure.

CTX is the abbreviation of the SSL (Secure Socket Layer) context. <ctxindex> is the index of the SSL context. standard module supports 6 SSL contexts at most. On the basis of a SSL context, several SSL connections can be established. The settings such as the SSL version and the CipherSuite are stored in the SSL context, and the settings will be applied to the new SSL connection which is associated with the SSL context.

AT+QSSLCFG SSL Configuration	
Test Command **AT+QSSLCFG=?**	Response **+QSSLCFG: "type",(0-5),"value"** **OK**
Query the setting of the context **AT+QSSLCFG="ctxindex",<ctxindex>**	Response **+QSSLCFG: <ctxindex>,<sslversion>,<seclevel>,** **<ciphersuite>,<cacert>,<clientcertname>,<clientkeyname**

	> **OK** Otherwise response **ERROR**
Configure the SSL version **AT+QSSLCFG="sslversion",<ctxindex>[,<sslversion>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "sslversion" value. **+QSSLCFG: "sslversion",<sslversion>** **OK**
Configure the CipherSuite **AT+QSSLCFG="ciphersuite",<ctxindex>[,<list of supported<ciphersuite>s>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "ciphersuite" value. **+QSSLCFG: "ciphersuite",<ciphersuite>** **OK**
Configure the authentication mode **AT+QSSLCFG="seclevel",<ctxindex>[,<seclevel>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "seclevel" value. **+QSSLCFG: "seclevel",<seclevel>** **OK**
Configure the path of root certificate **AT+QSSLCFG="cacert",<ctxindex>[,<cacertname>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "cacertname" value. **+QSSLCFG: "cacert",<cacertname>** **OK**
Configure the path of client certificate **AT+QSSLCFG="clientcert",<ctxindex>[,<clientcertname>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "clientcertname"

	value. **+QSSLCFG: "clientcert",<clientcertname>** **OK**
Configure the path of client key **AT+QSSLCFG="clientkey",<ctxindex>[,<clientkeyname>]**	Response **OK** Otherwise response **ERROR** If the third parameter is omitted, query the "clientkeyname" value. **+QSSLCFG: "clientkey",<clientkeyname>** **OK**
Configure whether to ignore the RTC time **AT+QSSLCFG="ignorertctime"[,<ignorertctime>]**	Response **OK** Otherwise response **ERROR** If the second parameter is omitted, query the "ignorertctime" value. **+QSSLCFG: "ignorertctime",<ignorertctime>** **OK**
Enable/Disable the HTTPS function **AT+QSSLCFG="https"[,<httpsenable>]**	Response **OK** Otherwise response **ERROR** If the second parameter is omitted, query the "httpsenable" value. **+QSSLCFG: "https",<httpsenable>** **OK**
Configure the SSL context index for HTTPS **AT+QSSLCFG="httpsctxi"[,<httpsctxindex>]**	Response **OK** Otherwise response **ERROR** If the second parameter is omitted, query the "httpsctxindex" value. **+QSSLCFG: "httpsctxi",<httpsctxindex>** **OK**
Configure the type of SMTP/SMTPS **AT+QSSLCFG="smtpstyle"[,<smtpstyle>]**	Response **OK** Otherwise response **ERROR**

	If the second parameter is omitted, query the “smtpstyle” value. **+QSSLCFG: “smtpstyle”,<smtpstyle>** **OK**
Configure the SSL context index for SMTPS **AT+QSSLCFG=“smtpsctxi”[,< smtpsctxindex>]**	Response **OK** Otherwise response **ERROR** If the second parameter is omitted, query the “smtpsctxindex” value. **+QSSLCFG: “smtpctxi”,<smtpsctxindex>** **OK**
Reference	

Parameter

<ctxindex>	SSL context index 0-5
<sslversion>	Configure the SSL version 0 SSL3.0 1 TLS1.0 2 TLS1.1 3 TLS1.2 4 ALL SUPPORT
<ciphersuite>	Configure the CipherSuite 0X0035 TLS_RSA_WITH_AES_256_CBC_SHA 0X002F TLS_RSA_WITH_AES_128_CBC_SHA 0X0005 TLS_RSA_WITH_RC4_128_SHA 0X0004 TLS_RSA_WITH_RC4_128_MD5 0X000A TLS_RSA_WITH_3DES_EDE_CBC_SHA 0X003D TLS_RSA_WITH_AES_256_CBC_SHA256 0XFFFF All support
<seclevel>	Configure the authentication mode 0 No authentication 1 Manage server authentication 2 Manage server and client authentication if requested by the remote server.
<cacertname>	String format, configure the server CA certificate
<clientcertname>	String format, configure the client certificate
<clientkeyname>	String format, configure the client key
<ignorertctime>	Configure whether to ignore the RTC time

	0	Do not ignore the RTC time
	1	Ignore the RTC time
<httpsenable>	Enable/disable the HTTPS function	
	0	Disable HTTPS
	1	Enable HTTPS
<httpsctxindex>	Configure the SSL context for HTTPS Httpsctxindex is the index of SSL context. If the host does not configure the httpsctxindex, the value of httpsctxindex is -1. 0-5	
<smtpstyle>	Configure the type of SMTP/SMTPS	
	0	Without SSL
	1	SSL
	2	STARTTLS
<smtpsctxindex>	Configure the SSL context for SMTPS smtpsctxindex is the index of SSL context. If the host does not configure the smtpsctxindex, the value of smtpsctxindex is -1. 0-5	

NOTES

1. The format of <cacertname>,<clientcertname> and <clientkeyname> can be as follows:
   - "RAM:filename" File is uploaded to RAM
   - "NVRAM:filename" File is uploaded to NVRAM. Support two CA certificates, one client certificate and one client private key. The filename of CA certificate must be CA0 or CA1, the filename of client certificate must be CC0, and the filename of client private key must be CK0.
     - CA[0,1] Identify a CA certificate
     - CC0 Identify a client certificate
     - CK0 Identify a client key
2. If no authentication is set, no security data is needed. If server authentication has been set, you need to configure Server CA certificate. If server and client authentication has been set, you need to configure Client certificate, Server CA certificate and Client private key.

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

AT+QSSLOPEN is used to set up a SSL connection. During the negotiation between the module and the peer, parameters configured by QSSLCFG will be used in the handshake procedure. After shaking hands with the peer 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.

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

AT+QSSLOPEN Open a SSL Socket to Connect a Remote Server	
Test Command **AT+QSSLOPEN=?**	Response **+QSSLOPEN: <ssid>,<ctxindex>,<ipaddr/domainname>,<port>,<connectmode>[,<timeout>]** **OK**
Read Command **AT+QSSLOPEN?**	Response **OK**
Write Command **AT+QSSLOPEN=<ssid>,<ctxindex>,<ipaddr/domainname>,<port>,<connectmode>[,<timeout>]**	Response If format is right, response **OK** Otherwise response **ERROR** The following URC will notify the result of connection. If connection is successful, the value of <connectcode> is 0, and is connection fails, the value of <connectcode> will be other value. **+QSSLOPEN: <ssid>,<connectcode>** Otherwise response **ERROR**
Reference	

Parameter

<ssid>	Secure socket identifier 0-5
<ctxindex>	SSL context index 0-5
<ipaddr/domainname>	String type, IP address of SSL server, or URL
<port>	The port of remote server
<connectmode>	Transferring mode 0 Non transparent mode 1 Transparent mode
<timeout>	10-300 second. The default value is 90 seconds.
<connectcode>	The result of connection 0 Success -1 Error -2 Socket is occupied

2.2.3. AT+QSSLCLOSE Close a SSL Connection

Close a SSL connection. If all of the SSL connection based on one SSL context have been closed, the module will release the SSL context.

AT+QSSLCLOSE Close a SSL Connection	
Test Command **AT+QSSLCLOSE=?**	Response **+QSSLCLOSE: (0-5)[,(0,1)]** **OK**
Read Command **AT+QSSLCLOSE?**	Response **OK**
Write Command **AT+QSSLCLOSE=<ssid>[,<closetype>]**	Response **CLOSE OK** Otherwise response **ERROR**
Reference	

Parameter

<ssid>	Secure socket identifier 0-5
<closetype>	Reserved

2.2.4. AT+QSSLSEND Send Data through SSL Connection

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

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

AT+QSSLSEND Send Data through SSL Connection	
Test Command **AT+QSSLSEND=?**	Response **+QSSLSEND: (0-5)[,(1-1460)]** **OK**
Read Command **AT+QSSLSEND?**	Response **OK**
Write Command	Response

AT+QSSLSEND=<ssid>[,<length>]	If connection is not established or disconnected, or some other errors occur: **ERROR** Response **>** Then input data to be sent. If you want to send changeable length data, tap “CTRL+Z” to send. “ESC” is used to cancel sending data. If sending is successful: **SEND OK** If the process of sending data is blocked: **SEND FAIL**
Reference	

Parameter

<ssid>	Secure socket identifier 0-5
<length>	A numeric parameter, indicates the length of sending data, it must be less than 1460 1-1460

2.2.5. AT+QSSLRECV Retrieve the Received SSL Data

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

NOTE

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

AT+QSSLRECV Retrieve the Received SSL Data	
Test Command **AT+ QSSLRECV=?**	Response **+QSSLRECV: (0,1),(0-5),(1-1500)** **OK**
Write Command	Response

AT+QSSLRECV=<cid>,<ssid>,<length>	**+QSSLRECV: <ipaddr>:<port>,TCP,<actual length><CR><LF><data>** **OK** If the buffer is empty, directly response: **OK** Otherwise response: **ERROR**
Reference	

Parameter

<cid>	Context number 0-1
<ssid>	Secure socket identifier 0-5
<length>	The maximum length of data to be retrieved. The range is 1-1500
<ipaddr>	IP address
<port>	The port of remote server
<actual length>	The actual data length obtained by QSSLRECV.

2.2.6. AT+QSSLSTATE Query Socket Connection Status

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

AT+QSSLSTATE Query Socket Connection Status	
Test Command **AT+QSSLSTATE=?**	Response **OK**
Read Command **AT+QSSLSTATE?**	Response **OK**
Write Command **AT+QSSLSTATE**	Response **+QSSLSTATE: <state>** **+QSSLSTATE: <socketindex>,<connectiontype>,<ipadd>,<port>,<socketstatus>,<sslconnectionflag>** **…** **OK**

	Otherwise response **ERROR**
Reference	

Parameter

<state>	A string parameter to indicate the status of the connection	
	“IP INITIAL”	The TCPIP stack is in idle state.
	“IP START”	The TCPIP stack has been registered to.
	“IP CONFIG”	It has been started-up to activate GPRS/CSD context.
	“IP IND”	It is activating GPRS/CSD context.
	“IP GPRSACT”	GPRS/CSD context has been activated successfully.
	“IP STATUS”	The local IP address has been gotten by the command AT+QILOCIP.
	“IP PROCESSING”	Establish connection.
	“PDP DEACT”	GPRS/CSD context was deactivated because of unknown reason.
<socketindex>	Socket index 0-5	
<connectiontype>	Connection type “TCP” “UDP”	
<ipadd>	Show IP address	
<port>	Show port number	
<socketstatus>	Socket state “INITIAL” “CONNECTING” “CONNECTED” “REMOTE CLOSING” “CLOSING” “CLOSED”	
<sslconnectionflag>	Judge whether the connection is normal TCP/UDP or TCP SSL 0 Normal TCP/UDP connection 1 TCP SSL connection	

2.2.7. AT+QSECWRITE Add a Certificate or Key

This command is used to add user certificate, user key and CA certificate to RAM or NVRAM. And the certificate and key will be stored in these storages in an encrypted way. After the certificate and key is stored in these storages, the host cannot read the data from these storages, instead, the host can only query the checksum of them. Please note that before adding a certificate or key to RAM or NVRAM, it

should not exist in the corresponding storage, if it exists already, the host should delete it first, and then add it to the corresponding storage.

AT+QSECWRITE Add a Certificate or Key	
Test Command **AT+QSECWRITE=?**	Response **+QSECWRITE: <filename >,<filesize>[,(3,200)]** **OK**
Read Command **AT+QSECWRITE?**	Response **OK**
Write Command **AT+QSECWRITE=<filename>,<filesize> [,<timeout>]**	Response If format is right, response **CONNECT** After module switches to data mode, and the certificate or key data can be inputted. When the size of the inputted data reaches <filesize> (unit: byte) or module receives "+++" sequence from UART, module will return to command mode and reply the following codes. **+QSECWRITE: <uploadsize>,<checksum>** **OK** If some errors occur, response **+CME ERROR: <err>**
Reference	

Parameter

<filename>	The name of the file to be stored. The format can be as follows: "RAM:filename" File is uploaded to RAM "NVRAM:filename" File is uploaded to NVRAM. Support two CA certificates, one client certificate and one client private key. The filename of CA certificate must be CA0 or CA1, the filename of client certificate must be CC0, and the filename of client private key must be CK0. CA[0,1] Identify a CA certificate CC0 Identify a client certificate CK0 Identify a client key
<filesize>	The size of the file to be uploaded. Unit: byte. If the file is uploaded to the RAM, the maximum size is 32768. If the file is uploaded to NVRAM, the maximum size is 2025. The minimum size is 1.

<timeout>	The time in seconds to wait for inputted data from UART. Unit: byte. 3-200. The default value is 100.
<uploadsize>	The size of the actually uploaded data. Unit: byte
<checksum>	The checksum of the uploaded data.

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

This command is used to query the checksum of a certificate or key, if the checksum is not same as the original one which owned by the user, some mistake will occur.

AT+QSECREAD Query the Checksum of a Certificate or Key	
Test Command **AT+QSECREAD=?**	Response **+QSECREAD: <filename>** **OK**
Read Command **AT+QSECREAD?**	Response **OK**
Write Command **AT+QSECREAD=<filename>**	Response **+QSECREAD: <good>,<checksum>** OK If some errors occur, response **+CME ERROR: <err>**
Reference	

Parameter

<filename>	The name of the file to be stored. The format can be as follows: "RAM:filename" Query the checksum of file which is stored in RAM "NVRAM:filename" Query the checksum of file which is stored in NVRAM. Support two CA certificates, one client certificate and one client private key. The filename of CA certificate must be CA0 or CA1, the filename of client certificate must be CC0, and the filename of client private key must be CK0. CA[0,1] Identify a CA certificate CC0 Identify a client certificate CK0 Identify a client key
<good>	Indicate the certificate or key is correct or not. When uploading the certificate or key by QSECWRITE, the checksum of certificate or key will be stored at the same time. After executing QSECREAD, QSECREAD will calculate checksum of the certificate or key

	again, and then compare this checksum with the checksum stored by QSECWRITE, if they are the same, the certificate or key is correct, otherwise the certificate or key is wrong. 0 The certificate or key is wrong 1 The certificate or key is correct
<checksum>	The checksum of the file

2.2.9. AT+QSECDEL Delete a Certificate or Key

This command is used to delete a certificate or key.

AT+QSECDEL Delete a Certificate or Key	
Test Command **AT+QSECDEL=?**	Response **+QSECDEL: <filename>** **OK**
Read Command **AT+QSECDEL?**	Response **OK**
Write Command **AT+QSECDEL=<filename>**	Response OK If some errors occur, response **+CME ERROR: <err>**
Reference	

Parameter

<filename>	The name of the file to be stored. The format can be as follows: "RAM:filename" Delete a certificate or key which is stored in RAM "NVRAM:filename" Delete a certificate or key which is stored in NVRAM. Support two CA certificates, one client certificate and one client private key. The filename of CA certificate must be CA0 or CA1, the filename of client certificate must be CC0, and the filename of client private key must be CK0. CA[0,1] Identify a CA certificate CC0 Identify a client certificate CK0 Identify a client key

2.2.10. URC

The format of SSL URC is “+QSSLURC:”. It mainly used to notify the host to read received data and disconnect the connections.

2.2.10.1. Notify to Read Data

Notify host to read data from peer.

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

Parameter

<cid>	Context number 0-2
<ssid>	Secure socket identifier 0-5

NOTES

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

2.2.10.2. Notify Disconnection

Notify host the connection has been disconnected. Lots of reasons can cause this phenomenon, such as the peer closes the connection or the state of GPRS PDP becomes deactivated. If this URC is reported, the module will close <ssid> SSL connection automatically, the host does not need to execute AT+QSSLCLOSE to close the <ssid> SSL connection.

Notify Disconnection

+QSSLURC: "closed",<ssid>	
Reference	This is a URC to notify host the connection is disconnected.

Parameter

<ssid>	Secure socket identifier 0-5

3 Example

3.1. SSL Function with Certificate and Key in RAM

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

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

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

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

OK
//Step 2: Configure and activate the PDP context.
AT+ QIFGCNT=0                                  //Set context 0 as foreground context.
OK
AT+ QICSGP=1,"CMNET"                           //Set bearer type as GPRS and the APN is
                                                "CMNET", no username and password for the
                                                APN.
```

```
OK
AT+QIREGAPP                                    //Register to TCP/IP stack.
OK
AT+QIACT                                       //Activate GPRS PDP context.
OK
AT+QILOCIP                                     //Query the local IP address.
10.1.83.188
AT+QIMUX=1                                     //Enable multiple TCPIP session.
OK
```

//Step 3: Configure SSL version, ciphersuite, server authentication and client authentication. Certificate and private key are in RAM.

```
AT+QSSLCFG="sslversion",0,2                    //Configure SSL version.
OK
AT+QSSLCFG="ciphersuite",0,"0XFFFF"            //Configure ciphersuite.
OK
AT+QSSLCFG="seclevel",0,2                      //Configure Server authentication and client
                                                 authentication.
OK
AT+QSECREAD="RAM:ca_cert.pem"                  //Check CA certificate is correct or not.
+QSECREAD: 1,4039

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

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

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

//Step 4: Setup SSL connection, send data and receive data.

```
AT+ QSSLOPEN =1, 0,"116.247.104.27",465,0      //Establish SSL connection and the socket index
                                                 is 1, and it is based on context 0,
                                                 non-transparent mode.
OK
+QSSLOPEN: 1,0                                 //Establish SSL connection successfully.
```

```
AT+QSSLSEND=1,12                          //Send 12 bytes data in the way of fixed length.
> <Input 12 bytes data>
SEND OK
AT+QSSLSEND=1                             //Send data in any byte less than 1460.
> <input some bytes data> ,<Ctrl+Z>       //After completing input data, tap "CTRL+Z" to send.
SEND OK

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

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

3.2. SSL Function with Certificate and Key in NVRAM

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

```
//Step 1: Upload the certificate and key to NVRAM.
AT+QSECWRITE="NVRAM:CA0",1614,100         //Upload the CA certificate to NVRAM.
CONNECT

<Input the CA0 data, the size is 1614 bytes>

+QSECWRITE: 1614,4039
```

```
OK
AT+QSECWRITE=”NVRAM:CC0”,1419,100          //Upload the client certificate to NVRAM.
CONNECT

<Input the CC0 data, the size is 1419 bytes>

+QSECWRITE: 1419,618

OK
AT+QSECWRITE=”NVRAM:CK0”,1679,100          //Upload the client private key to NVRAM.
CONNECT

<Input the CK0 data, the size is 1679 bytes>

+QSECWRITE: 1679,83a7

OK

//Step 2: Configure and activate the PDP context.

AT+ QIFGCNT=0                              //Set context 0 as foreground context.
OK
AT+ QICSGP=1,”CMNET”                       //Set bearer type as GPRS and the APN is “CMNET”,
                                             no username and password for the APN.
OK
AT+QIREGAPP                                //Register to TCP/IP stack.
OK
AT+QIACT                                   //Activate GPRS PDP context.
OK
AT+QILOCIP                                 //Query the local IP address.
10.1.83.188
AT+QIMUX=1                                 //Enable multiple TCPIP session.
OK

//Step 3: Configure SSL version, ciphersuite, server authentication and client authentication. Certificate
and private key are in NVRAM.

AT+QSSLCFG=”sslversion”,0,2                //Configure SSL version.
OK
AT+QSSLCFG=”ciphersuite”,0,”0XFFFF”        //Configure ciphersuite.
OK
AT+QSSLCFG=”seclevel”,0,2                  //Configure Server authentication and client
                                             authentication.
OK
AT+QSECREAD=”NVRAM:CA0”                    //Check CA certificate is correct or not.
+QSECREAD: 1,4039

OK
```

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

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

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

//Step 4: Setup SSL connection, send data, receive data

```
AT+ QSSLOPEN =1, 0,”116.247.104.27”,465,0  //Establish SSL connection and the socket index is 1,
OK                                          and it is based on context 0, non-transparent mode.

+QSSLOPEN: 1,0                             //Establish SSL connection successfully.
AT+QSSLSEND=1,12                           //Send 12 bytes data in the way of fixed length.

> <Input 12 bytes data>

SEND OK
AT+QSSLSEND=1                              //Send data in any byte less than 1460.

> <Input some bytes data> ,<Ctrl+Z>        //After completing to input data, tap “CTRL+Z” to send
                                            data.

SEND OK

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

OK
```

//Step 5: Close SSL connection

```
AT+QSSLCLOSE=1                             //Close socket index 1.
1, CLOSE OK
AT+QIDEACT                                 //Deactivate GPRS PDP context.
DEACT OK
```

3.3. Example about SSL Function Coexists with Normal TCPIP Function

```
//Step 1: Configure and activate the PDP context.
AT+ QIFGCNT=0                          //Set context 0 as foreground context.
OK
AT+ QICSGP=1,”CMNET”                   //Set bearer type as GPRS and the APN is “CMNET”,
OK                                       no username and password for the APN.
AT+QIREGAPP                            //Register to TCP/IP stack.
OK
AT+QIACT                               //Activate GPRS PDP context.
OK
AT+QILOCIP                             //Query the local IP address.
10.1.83.188

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

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

+QIRDI: 0,1,1                          //Module receives the data based on context 0, and
                                         module acts as the client, and the socket index is 1.
AT+QIRD=0,1,1,1024                     //Read the data from the module's socket buffer.
+QIRD: 116.247.104.27:6021,TCP,5       //The maximum length to retrieve is 1024. If the data
Abcde                                    length in the buffer is less than 1024, retrieve all the
OK                                       data from the buffer.

//Step 3: Configure SSL version, ciphersuite, no authentication. Setup SSL connection, send data, receive
data.
AT+QSSLCFG=”sslversion”,0,2            //Configure SSL version.
OK
AT+QSSLCFG=”seclevel”,0,0              //Configure Server authentication and client
                                         authentication.
```

```
OK
AT+QSSLCFG=”ciphersuite”,0,“0XFFFF”          //Configure ciphersuite.
OK
AT+ QSSLOPEN =3,0,”124.74.41.170”,5115,0     //Establish SSL connection and the socket index is 3,
OK                                             and it is based on context 0.

+QSSLOPEN: 3,0                               //Establish SSL connection successfully.
AT+QSSLSEND=3,12                             //Send 12 bytes data in the way of fixed length.

> <Input 12 bytes data>

SEND OK
AT+QSSLSEND=3

> <Input some bytes data> ,<Ctrl+Z>          //After completing to input data, tap “CTRL+Z” to send
                                               data.

SEND OK

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

OK

//Step 4: Close normal TCP connection and SSL connection.

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

4 Appendix A Reference

Table 3: 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]	GSM_TCPIP_Application_Note	TCPIP application note

Table 4: Terms and Abbreviations

Abbreviation	Description
ME	Mobile Equipment
TA	Terminal Adapter
MS	Mobile Station
CTX	SSL Context