

BG95&BG77 HTTP(S)

Application Note

LPWA Module Series

Rev. BG95&BG77_HTTP(S)_Application_Note_V1.0

Date: 2019-08-12

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://www.quectel.com/support/sales.htm>

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

<http://www.quectel.com/support/technical.htm>

Or email to: support@quectel.com

GENERAL NOTES

QUECTEL OFFERS THE INFORMATION AS A SERVICE TO ITS CUSTOMERS. THE INFORMATION PROVIDED IS BASED UPON CUSTOMERS' REQUIREMENTS. QUECTEL MAKES EVERY EFFORT TO ENSURE THE QUALITY OF THE INFORMATION IT MAKES AVAILABLE. QUECTEL DOES NOT MAKE ANY WARRANTY AS TO THE INFORMATION CONTAINED HEREIN, AND DOES NOT ACCEPT ANY LIABILITY FOR ANY INJURY, LOSS OR DAMAGE OF ANY KIND INCURRED BY USE OF OR RELIANCE UPON THE INFORMATION. ALL INFORMATION SUPPLIED HEREIN IS SUBJECT TO CHANGE WITHOUT PRIOR NOTICE.

COPYRIGHT

THE INFORMATION CONTAINED HERE IS PROPRIETARY TECHNICAL INFORMATION OF QUECTEL WIRELESS SOLUTIONS CO., LTD. TRANSMITTING, REPRODUCTION, DISSEMINATION AND EDITING OF THIS DOCUMENT AS WELL AS UTILIZATION OF THE CONTENT ARE FORBIDDEN WITHOUT PERMISSION. OFFENDERS WILL BE HELD LIABLE FOR PAYMENT OF DAMAGES. ALL RIGHTS ARE RESERVED IN THE EVENT OF A PATENT GRANT OR REGISTRATION OF A UTILITY MODEL OR DESIGN.

Copyright © Quectel Wireless Solutions Co., Ltd. 2019. All rights reserved.

About the Document

History

Revision	Date	Author	Description
1.0	2019-08-12	Terrence YANG/ Sherlock ZHAO	Initial

Contents

About the Document	2
Contents	3
Table Index	4
1 Introduction	5
1.1. The Process of Using HTTP(S) AT Commands.....	5
1.2. Description of HTTP(S) Header.....	6
1.2.1. Customize HTTP(S) Request Header	6
1.2.2. Output HTTP(S) Response Header.....	6
1.3. Description of Data Mode	6
2 Description of HTTP(S) AT Commands	8
2.1. AT+QHTTPCFG Configure Parameters for HTTP(S) Server	8
2.2. AT+QHTTPURL Set URL of HTTP(S) Server	10
2.3. AT+QHTTPGET Send GET Request to HTTP(S) Server	11
2.4. AT+QHTTPPOST Send POST Request to HTTP(S) Server via UART/USB	13
2.5. AT+QHTTPPOSTFILE Send POST Request to HTTP(S) Server via File.....	15
2.6. AT+QHTTPREAD Read Response from HTTP(S) Server via UART/USB	16
2.7. AT+QHTTPREADFILE Store the Response from HTTP(S) Server to a File.....	17
3 Examples	19
3.1. Access HTTP Server.....	19
3.1.1. Send HTTP GET Request and Read the Response	19
3.1.2. Send HTTP POST Request and Read the Response.....	21
3.1.2.1. POST Body Obtained from UART/USB	21
3.1.2.2. POST Body Obtained from File System.....	22
3.2. Access HTTPS Server	23
3.2.1. Send HTTPS GET Request and Read the Response.....	23
3.2.2. Send HTTPS POST Request and Read the Response	25
3.2.2.1. POST Body Obtained from UART/USB	25
3.2.2.2. POST Body Obtained from File System.....	27
4 Error Handling	30
4.1. Executing HTTP(S) AT Commands Fails.....	30
4.2. PDP Activation Fails.....	30
4.3. DNS Parse Fails.....	30
4.4. Entering Data Mode Fails	31
4.5. Sending GET/POST Requests Fails.....	31
4.6. Reading Response Fails.....	32
5 Summary of Error Codes	33
6 Summary of HTTP(S) Response Codes	35
7 Appendix A References	36

Table Index

TABLE 1: SUMMARY OF ERROR CODES	33
TABLE 2: SUMMARY OF HTTP(S) RESPONSE CODES	35
TABLE 3: RELATED DOCUMENTS	36
TABLE 4: TERMS AND ABBREVIATIONS	36

1 Introduction

BG95/BG77 supports HTTP(S) applications through accessing HTTP(S) servers. This document is a reference guide to all the AT commands defined for HTTP(S).

1.1. The Process of Using HTTP(S) AT Commands

Through TCP/IP AT commands, customers can configure a PDP context, activate/deactivate the PDP context and query the PDP context status. And through HTTP(S) AT commands, customers can send HTTP(S) GET/POST requests to HTTP(S) server, and read HTTP(S) response from HTTP(S) server. The general process is as follows:

- Step 1:** Configure <APN>, <username>, <password> and other parameters of a PDP context by AT+QICSGP.
- Step 2:** Activate the PDP context by AT+QIACT, then the assigned IP address can be queried by AT+QIACT?.
- Step 3:** Configure the PDP context ID and SSL context ID by AT+QHTTPCFG.
- Step 4:** Configure SSL context parameters by AT+QSSLCFG.
- Step 5:** Set HTTP(S) URL by AT+QHTTPURL.
- Step 6:** Send HTTP(S) request. AT+QHTTPGET can be used for sending HTTP(S) GET request, and AT+QHTTPPOST/AT+QHTTPPOSTFILE can be used for sending HTTP(S) POST request.
- Step 7:** Read HTTP(S) response information by AT+QHTTPREAD/AT+QHTTPREADFILE.
- Step 8:** Deactivate the PDP context by AT+QIDEACT.

NOTES

1. Please refer to *Quectel_BG95&BG77_TCP(IP)_AT_Commands_Manual* for more details of AT+QICSGP, AT+QIACT and AT+QIDEACT commands.
2. Please refer to *Quectel_BG95&BG77_SSL_AT_Commands_Manual* for more details of AT+QSSLCFG command.

1.2. Description of HTTP(S) Header

1.2.1. Customize HTTP(S) Request Header

By default, HTTP(S) request header is filled by the module automatically. It can also be customized by configuring <request_header> as 1 via AT+QHTTPCFG command, and then inputting HTTP(S) request header according to the following requirements:

1. Follow HTTP(S) header syntax.
2. The value of URI in HTTP(S) request line and the "Host:" the header must be in line with the URL configured by AT+QHTTPURL command.
3. The HTTP(S) request header must end with <CR><LF>.

The following example shows a valid HTTP(S) POST request header:

```
POST /processorder.php HTTP/1.1<CR><LF>
Host: 220.180.239.212:8011<CR><LF>
Accept: */*<CR><LF>
User-Agent: QUECTEL_MODULE<CR><LF>
Connection: Keep-Alive<CR><LF>
Content-Type: application/x-www-form-urlencoded<CR><LF>
Content-Length: 48<CR><LF>
<CR><LF>
Message=1111&Appleqty=2222&Orangeqty=3333&find=1
```

1.2.2. Output HTTP(S) Response Header

By default, HTTP(S) response header will not be output. Output of HTTP(S) response header can be enabled by configuring <response_header> to 1 via AT+QHTTPCFG command, and then the HTTP(S) response header will be output together with HTTP(S) response body after executing AT+QHTTPREAD or AT+QHTTPREADFILE command.

1.3. Description of Data Mode

The COM port of BG95/BG77 module has two working modes: AT command mode and data mode. In AT command mode, the inputted data via COM port will be treated as AT commands. While in data mode, it will be treated as data.

Inputting "+++" or pulling up DTR (AT&D1 should be set first) can make the COM port exit from data mode. 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.

When AT+QHTTPURL, AT+QHTTPPOST and AT+QHTTPREAD are executed, the COM port will enter data mode. If customers are using “+++” or DTR to make the port exit from data mode, the executing procedure of these commands will be interrupted before the response is returned. In such a case, the COM port cannot reenter data mode by executing ATO command.

2 Description of HTTP(S) AT Commands

2.1. AT+QHTTPCFG Configure Parameters for HTTP(S) Server

The command is used to configure the parameters for HTTP(S) server, including configuring a PDP context ID, customizing HTTP(S) request header, outputting HTTP(S) response header and querying SSL settings. If the Write Command only executes one parameter, it will query the current settings.

AT+QHTTPCFG Configure Parameters for HTTP(S) Server

Test Command AT+QHTTPCFG=?	Response +QHTTPCFG: "contextid",(1-16) +QHTTPCFG: "requestheader",(0,1) +QHTTPCFG: "responseheader",(0,1) +QHTTPCFG: "sslctxid",(0-5) +QHTTPCFG: "contenttype",(0-3) OK
Read Command AT+QHTTPCFG?	Response +QHTTPCFG: "contextid",<contextID> +QHTTPCFG: "requestheader",<request_header> +QHTTPCFG: "responseheader",<response_header> +QHTTPCFG: "sslctxid",<sslctxID> +QHTTPCFG: "contenttype",<content_type> OK
Write Command AT+QHTTPCFG="contextid" [<contextID>]	Response If <contextID> is present, configure the PDP context ID: OK Or +CME ERROR: <err> If <contextID> is omitted, query the current settings: +QHTTPCFG: "contextid",<contextID> OK
Write Command	Response

<p>AT+QHTTPCFG="requestheader"[,<request_header>]</p>	<p>If <request_header> is present, configure whether to enable customization of HTTP(S) request header: OK Or +CME ERROR: <err></p> <p>If <request_header> is omitted, query the current settings: +QHTTPCFG: "requestheader",<request_header></p> <p>OK</p>
<p>Write Command AT+QHTTPCFG="responseheader"[,<response_header>]</p>	<p>Response</p> <p>If <response_header> is present, configure whether to enable output of HTTP(S) response header: OK Or +CME ERROR: <err></p> <p>If <response_header> is omitted, query the current settings: +QHTTPCFG: "responseheader",<response_header></p> <p>OK</p>
<p>Write Command AT+QHTTPCFG="sslctxid"[,<sslctxID>]</p>	<p>Response</p> <p>If <sslctxID> is present, configure the SSL context ID used for HTTP(S): OK Or +CME ERROR: <err></p> <p>If <sslctxID> is omitted, query the current settings: +QHTTPCFG: "sslctxid",<sslctxID></p> <p>OK</p>
<p>Write Command AT+QHTTPCFG="contenttype"[,<content_type>]</p>	<p>Response</p> <p>If <content_type> is present, configure the data type of HTTP(S) body: OK Or +CME ERROR: <err></p> <p>If <content_type> is omitted, query the current settings: +QHTTPCFG: "contenttype",<content_type></p> <p>OK</p>

Parameter

<contextID>	Integer type. PDP context ID. The range is 1-16, and the default value is 1.
<request_header>	Integer type. Disable or enable customization of HTTP(S) request header. 0 Disable 1 Enable
<response_header>	Integer type. Disable or enable to output HTTP(S) response header. 0 Disable 1 Enable
<sslctxID>	Integer type. SSL context ID used for HTTP(S). The range is 0-5, and the default value is 1. The SSL parameters can be configured by AT+QSSLCFG, and for details of this command, please refer to <i>Quectel_BG95&BG77_SSL_AT_Commands_Manual</i> .
<content_type>	Integer type. Data type of HTTP(S) body. 0 application/x-www-form-urlencoded 1 text/plain 2 application/octet-stream 3 multipart/form-data
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.

2.2. AT+QHTTPURL Set URL of HTTP(S) Server

The URL must begin with "http://" or "https://", which indicates an HTTP or HTTPS server will be accessed.

AT+QHTTPURL Set URL of HTTP(S) Server

Test Command AT+QHTTPURL=?	Response +QHTTPURL: (1-700),(1-65535) OK
Read Command AT+QHTTPURL?	Response [+QHTTPURL: <URL>] OK
Write Command AT+QHTTPURL=<URL_length>[,<timeout>]	Response a) If the parameter format is correct, and it is not sending HTTP(S) GET/POST requests at present: CONNECT TA switches to transparent access mode, and then the URL can be inputted. When the total size of the inputted data

reaches **<URL_length>**, TA will return to command mode and report the following code:

OK

If **<timeout>** has reached, but the received length of URL is less than **<URL_length>**, TA will return to command mode and report the following code:

+CME ERROR: <err>

b) If the parameter format is incorrect or other errors occur:

+CME ERROR: <err>

Parameter

<URL>	String type. The URL of HTTP(S) server.
<URL_length>	Integer type. The length of URL. The range is 1-700. Unit: byte.
<timeout>	Integer type. The maximum time for inputting URL. The range is 1-65535, and the default value is 60. Unit: second.
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.

2.3. AT+QHTTPGET Send GET Request to HTTP(S) Server

According to the configured **<request_header>** parameter in **AT+QHTTPCFG="requestheader"[, <request_header>]** command, **AT+HTTPGET** Write Command has two different formats. If **<request_header>** is set to 1, after **AT+QHTTPGET** command has been sent, "CONNECT" may be output in 125s to indicate that the connection is successful. If it is not output during the time, then "+CME ERROR: <err>" will be output.

After **AT+HTTPGET** Write Command has been sent, it is recommended to wait for a specific period of time (**<rsptime>**) for "+QHTTPGET: <err>[,<httprcode>[,<content_length>]]" to be output after "OK" is reported.

In "+QHTTPGET: <err>[,<httprcode>[,<content_length>]]", the **<httprcode>** parameter can only be reported when **<err>** is 0. If HTTP(S) response header contains "CONTENT-LENGTH", then **<content_length>** will be reported.

AT+QHTTPGET Send GET Request to HTTP(S) Server

Test Command	Response
AT+QHTTPGET=?	+QHTTPGET: (1-65535),(1-2048),(1-65535)
	OK

<p>Write Command If <request_header> equals 0 (disable to customize HTTP(S) request header) AT+QHTTPGET[=<rsptime>]</p>	<p>Response a) If the parameter format is correct and no other errors occur: OK</p> <p>When the module has received response from HTTP(S) server, it will report the following URC: +QHTTPGET: <err>[,<httprspcode>[,<content_length>]]</p> <p>b) If the parameter format is incorrect or other errors occur: +CME ERROR: <err></p>
<p>Write Command If <request_header> equals 1 (enable to customize HTTP(S) request header) AT+QHTTPGET=<rsptime>,<data_length>[,<input_time>]</p>	<p>Response a) If HTTP(S) server is connected successfully: CONNECT</p> <p>TA switches to transparent access mode, and then the HTTP(S) GET request header can be inputted. When the total size of the inputted data reaches <data_length>, TA will return to command mode and report the following code: OK</p> <p>When the module has received response from HTTP(S) server, it will report the following URC: +QHTTPGET: <err>[,<httprspcode>[,<content_length>]]</p> <p>If the <input_time> has reached, but the length of received data is less than <data_length>, TA will return to command mode and report the following code: +CME ERROR: <err></p> <p>b) If the parameter format is incorrect or other errors occur: +CME ERROR: <err></p>
<p>Maximum Response Time</p>	<p>Determined by <rsptime></p>

Parameter

<p><rsptime></p>	<p>Integer type. The range is 1-65535, and the default value is 60. Unit: second. It is used to configure the timeout for the HTTP(S) GET response “+QHTTPGET: <err>[,<httprspcode>[,<content_length>]]” to be output after “OK” is returned.</p>
<p><data_length></p>	<p>Integer type. The length of HTTP(S) request information, including HTTP(S) request header and HTTP(S) request body. The range is 1-2048. Unit: byte.</p>
<p><input_time></p>	<p>Integer type. The maximum time for inputting HTTP(S) request information. The range is 1-65535, and the default value is 60. Unit: second.</p>
<p><err></p>	<p>Integer type. The error code of operation. Please refer to Chapter 5 for details.</p>

<httpsrcode>	Please refer to Chapter 6 .
<request_header>	Integer type. Disable or enable to customize HTTP(S) request header. 0 Disable 1 Enable
<content_length>	Integer type. The length of HTTP(S) response body. Unit: byte.

2.4. AT+QHTTPPOST Send POST Request to HTTP(S) Server via UART/USB

The command is used to send HTTP(S) POST request. According to the configured <request_header> parameter in AT+QHTTPCFG="requestheader"[,<request_header>] command, the AT+QHTTPPOST Write Command has two different formats:

- If <request_header> is set to 0, then only the POST body should be inputted via UART/USB port.
- If <request_header> is set to 1, then both POST header and POST body should be inputted via UART/USB port.

After AT+QHTTPPOST command has been sent, "CONNECT" may be output in 125s to indicate the connection is successful. If it is not received during the time, "+CME ERROR: <err>" will be output. It is recommended to wait for a specific period of time (refer to the maximum response time below) for "+QHTTPPOST: <err>[,<httpsrcode>[,<content_length>]]" to be output after "OK" is reported.

AT+QHTTPPOST Send POST Request to HTTP(S) Server via UART/USB	
Test Command AT+QHTTPPOST=?	Response +QHTTPPOST: (1-1024000),(1-65535),(1-65535) OK
Write Command If <request_header> equals 0 (disable to customize HTTP(S) request header) AT+QHTTPPOST=<data_length>[,<input_time>,<rsptime>]	Response a) If the parameter format is correct, the HTTP(S) server has been connected successfully, and the HTTP(S) request header has been sent completely: CONNECT TA switches to transparent access mode, and then the HTTP(S) POST body can be inputted. When the total size of the inputted data reaches <data_length>, TA will return to command mode and report the following code: OK When the module has received response from HTTP(S) server, it will report the following URC:

	<p>+QHTTPPOST: <err>[,<httprspcode>[,<content_length>]]</p> <p>If the <input_time> has reached, but the received length of data is less than <data_length>, TA will return to command mode and report the following code:</p> <p>+CME ERROR: <err></p> <p>b) If the parameter format is incorrect or other errors occur:</p> <p>+CME ERROR: <err></p>
<p>Write Command</p> <p>If <request_header> equals 1 (enable to customize HTTP(S) request header)</p> <p>AT+QHTTPPOST=<data_length>[,<input_time>,<rsptime>]</p>	<p>Response</p> <p>a) If the parameter format is correct and the HTTP(S) server has been connected successfully:</p> <p>CONNECT</p> <p>TA switches to the transparent access mode, and then the HTTP(S) POST header and body can be inputted. When the total size of the inputted data reaches <data_length>, TA will return to command mode and report the following code:</p> <p>OK</p> <p>When the module has received response from HTTP(S) server, it will report the following URC:</p> <p>+QHTTPPOST: <err>[,<httprspcode>[,<content_length>]]</p> <p>If the <input_time> has reached, but the length of received data is less than <data_length>, TA will return to command mode and report the following code:</p> <p>+CME ERROR: <err></p> <p>b) If the parameter format is incorrect or other errors occur:</p> <p>+CME ERROR: <err></p>
<p>Maximum Response Time</p>	<p>Determined by network and <rsptime></p>

Parameter

<p><data_length></p>	<p>Integer type. If <request_header> is 0, it indicates the length of POST body, and if <request_header> is 1, it indicates the length of HTTP(S) request information, including HTTP(S) request header and HTTP(S) request body. The range is 1-1024000. Unit: byte.</p>
<p><input_time></p>	<p>Integer type. The maximum time for inputting POST body or HTTP(S) request information. The range is 1-65535, and the default value is 60. Unit: second.</p>
<p><rsptime></p>	<p>Integer type. The range is 1-65535, and the default value is 60. Unit: second. It is used to configure the timeout for the HTTP(S) POST response</p>

	“+QHTTPPOST: <err>[,<httprcode>[,<content_length>]]” to be output after “OK” is returned.
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.
<httprcode>	Please refer to Chapter 6 .
<request_header>	Integer type. Disable or enable to customize HTTP(S) request header. 0 Disable 1 Enable
<content_length>	Integer type. The length of HTTP(S) response body. Unit: byte.

2.5. AT+QHTTPPOSTFILE Send POST Request to HTTP(S) Server via File

The command can be used to send HTTP(S) POST request via a file. According to the <request_header> configuration in AT+QHTTPCFG="requestheader"[,<request_header>] command, the file operated by AT+HTTPPOSTFILE command has two different formats:

- If <request_header> is set to 0, the file in file system will be POST body only.
- If <request_header> is set to 1, the file in file system will be POST header and body.

The module will report “+QHTTPPOSTFILE: <err>[,<httprcode>[,<content_length>]]” to indicate the executing result of AT+QHTTPPOSTFILE command. The <httprcode> parameter can only be reported when <err> equals 0. It is recommended to wait for a specific period of time (refer to the maximum response time below) for “+QHTTPPOSTFILE: <err>[,<httprcode>[,<content_length>]]” to be output after “OK” is reported.

AT+QHTTPPOSTFILE Send POST Request to HTTP(S) Server via File

Test Command AT+QHTTPPOSTFILE=?	Response +QHTTPPOSTFILE: <file_name>,(1-65535) OK
Write Command AT+QHTTPPOSTFILE=<file_name>[,<rsptime>] If <request_header> equals 1, the specified file must contain HTTP(S) request header information.	Response a) If the parameter format is correct and the HTTP(S) server has been connected successfully: OK When the module has received response from HTTP(S) server, it will report the following URC: +QHTTPPOSTFILE: <err>[,<httprcode>,<content_length>]

	b) If the parameter format is incorrect or other errors occur: +CME ERROR: <err>
Maximum Response Time	Determined by <rsptime>

Parameter

<file_name>	String type. File name. The max length of the file name is 80 bytes.
<rsptime>	Integer type. The range is 1-65535, and the default value is 60. Unit: second. It is used to configure the timeout for the HTTP(S) POST response “+QHTTPPOSTFILE: <err>[,<httprcode>,<content_length>]” to be output after “OK” is returned.
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.
<httprcode>	Please refer to Chapter 6 .
<request_header>	Integer type. Disable or enable to customize HTTP(S) request header. 0 Disable 1 Enable
<content_length>	Integer type. The length of HTTP(S) response body.

2.6. AT+QHTTPREAD Read Response from HTTP(S) Server via UART/USB

This command is used to retrieve HTTP(S) response information from HTTP(S) server via UART/USB port, after HTTP(S) GET/POST requests are sent. It must be executed after “+QHTTPGET: <err>[,<httprcode>[,<content_length>]]”, “+QHTTPPOST: <err>[,<httprcode>[,<content_length>]]” or “+QHTTPPOSTFILE: <err>[,<httprcode>,<content_length>]” are received.

AT+QHTTPREAD Read Response from HTTP(S) Server via UART/USB	
Test Command AT+QHTTPREAD=?	Response +QHTTPREAD: (1-65535) OK
Write Command AT+QHTTPREAD[=<wait_time>]	Response a) If the parameter format is correct and the HTTP(S) response is read successfully: CONNECT <Output HTTP(S) response information> OK +QHTTPREAD: <err>

If <wait_time> reaches or other errors occur, but the HTTP(S) response has not been output completely, it will report the following code:

+CME ERROR: <err>

b) If the parameter format is incorrect or other errors occur:

+CME ERROR: <err>

Parameter

<wait_time>	Integer type. The maximum interval time between receiving two packets of data. The range is 1-65535, and the default value is 60. Unit: second.
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.

2.7. AT+QHTTPREADFILE Store the Response from HTTP(S) Server to a File

This command is used to store the HTTP(S) response information from HTTP(S) server to a specified file, after HTTP(S) GET/POST requests are sent, thus allowing customers to retrieve the response information from the file. It must be executed after “+QHTTPGET: <err>[,<httprspcode>[,<content_length>]]”, “+QHTTPPOST: <err>[,<httprspcode>[,<content_length>]]” or “+QHTTPPOSTFILE: <err>[,<httprspcode>,<content_length>]” is reported.

AT+QHTTPREADFILE Store the Response from HTTP(S) Server to a File

Test Command AT+QHTTPREADFILE=?	Response +QHTTPREADFILE: <file_name>,(1-65535) OK
Write Command AT+QHTTPREADFILE=<file_name>[,<wait_time>]	Response a) If the parameter format is correct: OK When the response from HTTP(S) server is read or <wait_time> reaches, it will report: +QHTTPREADFILE: <err> b) If the parameter format is incorrect or other errors occur: +CME ERROR: <err>

Parameter

<wait_time>	Integer type. The maximum interval time between receiving two packets of data. The range is 1-65535, and the default value is 60. Unit: second.
<file_name>	String type. File name. The max length of file name is 80 bytes.
<err>	Integer type. The error code of operation. Please refer to Chapter 5 for details.

3 Examples

3.1. Access HTTP Server

3.1.1. Send HTTP GET Request and Read the Response

The following examples show how to send HTTP GET request and enable output of HTTP response header, as well as how to read HTTP GET response.

```
//Example of how to send HTTP GET response.
AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.
OK

AT+QHTTPCFG="responseheader",1 //Allow to output HTTP response header.
OK

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
OK //No context activated currently.

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

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

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
+QIACT: 1,1,1,"10.7.157.1"

OK

AT+QHTTPURL=23,80 //Set the URL of the HTTP server which will be accessed.
CONNECT
http://www.sina.com.cn/ //Input the URL whose length should be 23 bytes. (This URL is
only an example. Please input the correct URL in practical test.)
OK
```

```
AT+QHTTPGET=80 //Send HTTP GET request and the maximum response time is
                80s.

OK

+QHTTPGET: 0,200,547256 //If HTTP response header contains "CONTENT-LENGTH"
                        information, then the <content_length> (547256) will be
                        reported.

//Example of how to read HTTP response.
//Solution 1: Read HTTP response information and output it via UART port.
AT+QHTTPREAD=80 //Read HTTP response information and output it via UART. The
                maximum time to wait for HTTP session to be closed is 80s.

CONNECT
HTTP/1.1 200 OK <CR><LF> //HTTP response header and body.
Content-Type: text/html<CR><LF>
Vary: Accept-Encoding<CR><LF>
X-Powered-By: shci_v1.03<CR><LF>
Server: nginx<CR><LF>
Date: Fri, 27 Dec 2013 02:21:43 GMT<CR><LF>
Last-Modified: Fri, 27 Dec 2013 02:20:01 GMT<CR><LF>
Expires: Fri, 27 Dec 2013 02:22:43 GMT<CR><LF>
Cache-Control: max-age=60<CR><LF>
Age: 1<CR><LF>
Content-Length: 547256<CR><LF>
X-Cache: HIT from xd33-85.sina.com.cn<CR><LF>
<CR><LF>
<body>
OK

+QHTTPREAD: 0 //Read HTTP response header and body successfully.

//Solution 2: Read HTTP response information through storing it to UFS file.
AT+QHTTPREADFILE="1.txt",80 //Read HTTP response header and body through storing
                             them to "1.txt", and the maximum time to wait for HTTP
                             session to close is 80s.

OK

+QHTTPREADFILE: 0 //HTTP response header and body are stored successfully.
```

3.1.2. Send HTTP POST Request and Read the Response

3.1.2.1. POST Body Obtained from UART/USB

The following examples show how to send HTTP POST request and retrieve POST body via UART port, as well as how to read HTTP POST response.

```

AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.
OK

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
OK //No context activated currently.

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

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

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
+QIACT: 1,1,1,"172.22.86.226"

OK

AT+QHTTTPURL=59,80 //Set the URL of the HTTP server which will be accessed.
CONNECT
http://api.efxnow.com/DEMOWebServices2.8/Service.asmx/Echo? //Input the URL whose length is
59 bytes. (This URL is only an
example. Please input the
correct URL in practical test.)
OK

AT+QHTTTPPOST=20,80,80 //Send HTTP POST request. The maximum input body time is
80s and the maximum response time is 80s.
CONNECT
Message>HelloQuectel //Input the POST body whose length should be 20 bytes. (The
POST body is only an example. Please input the correct POST
body in practical test.)
OK

+QHTTTPPOST: 0,200,177 //If the HTTP response header contains "CONTENT-LENGTH"
information, then the <content_length> (177) will be reported.

```

```

AT+QHTTPREAD=80 //Read HTTP response body and output it via UART. The
                    //maximum time to wait for HTTP session to be closed is 80s.

CONNECT
<?xml version="1.0" encoding="utf-8"?>
<string xmlns="httpHTTPs://api.efxnow.com/webservices2.3">Message='HelloQuectel' ASCII:72
101 108 108 111 81 117 101 99 116 101 108 </string> //Output the HTTP response body.
OK

+QHTTPREAD: 0 //HTTP response body is output successfully.

```

3.1.2.2. POST Body Obtained from File System

The following examples show how to send HTTP POST request and retrieve POST body via file system, as well as how to store HTTP POST response to file system.

```

AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.
OK

AT+QIACT? //Query the list of the currently activated contexts and their IP
            //addresses.
OK //No context activated currently.

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

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

AT+QIACT? //Query the list of the currently activated contexts and their IP
            //addresses.
+QIACT: 1,1,1,"172.22.86.226"

OK

AT+QHTTPURL=59,80 //Set the URL of the HTTP server which will be accessed.
CONNECT
http://api.efxnow.com/DEMOWebServices2.8/Service.asmx/Echo? //Input URL whose length is 59
                                                            //bytes. (This URL is only an
                                                            //example. Please input the
                                                            //correct URL in practical test.)
OK

//POST request information from UFS file, and read the HTTP response information through storing it to a

```

UFS file.

AT+QHTTPPOSTFILE="2.txt",80 //Send HTTP POST request. The POST body is obtained from "2.txt", and the maximum response time is 80s.

OK

+QHTTPPOSTFILE: 0,200,177 //HTTP POST request is sent successfully. And then the HTTP response body can be read via either AT+QHTTPREAD or AT+QHTTPREADFILE command.

AT+QHTTPREADFILE="3.txt",80 //Read HTTP response body through storing it to "3.txt", and the maximum time to wait for HTTP session to close is 80s.

OK

+QHTTPREADFILE: 0 //HTTP response body is stored successfully.

3.2. Access HTTPS Server

3.2.1. Send HTTPS GET Request and Read the Response

The following examples show how to send HTTPS GET request and enable output of HTTPS response header, as well as how to read HTTPS GET response.

//Example of how to send HTTPS GET request.

AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.

OK

AT+QHTTPCFG="responseheader",1 //Allow to output HTTPS response header.

OK

AT+QIACT? //Query the list of the currently activated contexts and their IP addresses.

OK

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

OK

AT+QIACT=1 //Activate context 1.

OK //Activated successfully.

AT+QIACT? //Query the list of the currently activated contexts and their IP addresses.

+QIACT: 1,1,1,"10.7.157.1"

OK

AT+QHTTPCFG="sslctxid",1 //Set SSL context ID as 1.

OK

AT+QSSLCFG="sslversion",1,1 //Set SSL version as 1 which means TLSv1.0.

OK

AT+QSSLCFG="ciphersuite",1,0x0005 //Set SSL cipher suite as 0x0005 which means RC4-SHA.

OK

AT+QSSLCFG="seclevel",1,0 //Set SSL verify level as 0 which means CA certificate is not needed.

OK

AT+QHTTPURL=22,80 //Set the URL of HTTPS server which will be accessed.

CONNECT

https://www.alipay.com //Input URL whose length is 22 bytes. (This URL is only an example. Please input the correct URL in practical test.)

OK

AT+QHTTPGET=80 //Send HTTPS GET request and the maximum response time is 80s.

OK

+QHTTPGET: 0,200,21472 //If the HTTPS response header contains "CONTENT-LENGTH" information, then the <content_length> (21472) will be reported.

//Example of how to read HTTPS response.

//Solution 1: Read HTTPS response information and output it via UART.

AT+QHTTPREAD=80 //Read HTTPS response information and output it via UART. The maximum time to wait for HTTPS session to be closed is 80s.

CONNECT //HTTPS response header and body.

HTTP/1.1 200 OK<CR><LF>

Server: nginx/1.2.7<CR><LF>

Date: Fri, 27 Dec 2013 02:38:27 GMT<CR><LF>

Content-Type: text/html; charset=GB18030<CR><LF>

Content-Length: 10750<CR><LF>

Connection: keep-alive<CR><LF>

<CR><LF>

```

<body>
OK

+QHTTPREAD: 0 //Read HTTPS response header and body successfully.

//Solution 2: Read HTTPS response information through storing it to UFS file.
AT+QHTTPREADFILE="4.txt",80 //Read HTTPS response header and body through storing it to
                              "4.txt", and the maximum time to wait for HTTPS session to be
                              closed is 80s.

OK

+QHTTPREADFILE: 0 //HTTPS response header and body are stored successfully.

```

3.2.2. Send HTTPS POST Request and Read the Response

3.2.2.1. POST Body Obtained from UART/USB

The following examples show how to send HTTPS POST request and retrieve POST body via UART port, as well as how to read HTTPS POST response.

```

AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.
OK

AT+QIACT? //Query the list of the currently activated contexts and their IP
           addresses.
OK //No context activated currently.

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

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

AT+QIACT? //Query the list of the currently activated contexts and their IP
           addresses.
+QIACT: 1,1,1,"172.22.86.226"

OK

AT+QHTTPCFG="sslctxid",1 //Set SSL context ID as 1.
OK

AT+QSSLCFG="sslversion",1,1 //Set SSL version as 1 which means TLSv1.0.

```

OK

AT+QSSLCFG="ciphersuite",1,0x0005 //Set SSL cipher suite as 0x0005 which means RC4-SHA.

OK

AT+QSSLCFG="secllevel",1,2 //Set SSL verify level as 2 which means CA certificate, client certificate and client private key should all be uploaded by AT+QFUPL command.

OK

AT+QFUPL="cacert.pem" //Upload CA certificate to UFS.

CONNECT

<Input file bin data>

+QFUPL:1216,7648

OK

AT+QFUPL="clientcert.pem" //Upload client certificate to UFS.

CONNECT

<Input file bin data>

+QFUPL:1216,5558

OK

AT+QFUPL="clientkey.pem" //Upload client private key to UFS.

CONNECT

<Input file bin data>

+QFUPL:1706,538

OK

AT+QSSLCFG="cacert",1,"cacert.pem" //Configure the path of CA certificate for SSL context 1.

OK

AT+QSSLCFG="clientcert",1,"clientcert.pem" //Configure the path of client certificate for SSL context 1.

OK

AT+QSSLCFG="clientkey",1,"clientkey.pem" //Configure the path of client private key for SSL context 1.

OK

AT+QHTTPURL=45,80 //Set the URL of the HTTPS server which will be accessed.

```

CONNECT
HTTPs://220.180.239.212:8011/processorder.php //Input the URL whose length is 45 bytes. (This
URL is only an example. Please input the correct
URL in practical test.)

OK

AT+QHTTPPOST=48,80,80 //Send HTTPS POST request. The maximum input body time is
80s and the maximum response time is 80s.

CONNECT
Message=1111&Appleqty=2222&Orangeqty=3333&find=1 //Input the POST body whose length
should be 48 bytes. (This POST body is
only an example. Please input the
correct one in practical test.)

OK

+QHTTPPOST: 0,200,285 //If the HTTPS response header contains
"CONTENT-LENGTH" information, then the <content_length>
(285) will be reported.

AT+QHTTPREAD=80 //Read HTTPS response body and output it via UART. The
maximum time to wait for HTTPS session to be closed is 80s.

CONNECT //Read HTTPS response body successfully.
<html>
<head>
<title>Quectel's Auto Parts - Order Results</title>
</head>
<body>
<h1>Quectel's Auto Parts</h1>
<h2>Order Results</h2>

<p>Order processed at 02:49, 27th December</p><p>Your order is as follows: </p>1111
message<br />2222 apple<br />3333 orange<br /></body>
</html>

OK

+QHTTPREAD: 0 //HTTPS response body is output successfully.

```

3.2.2.2. POST Body Obtained from File System

The following examples show how to send HTTPS POST request and retrieve POST body from file system, as well as how to store HTTPS POST response to file system.

```

AT+QHTTPCFG="contextid",1 //Configure the PDP context ID as 1.
OK

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
OK //No context activated currently.

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

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

AT+QIACT? //Query the list of the currently activated contexts and their IP
addresses.
+QIACT: 1,1,1,"172.22.86.226"
OK

AT+QHTTPCFG="sslctxid",1 //Set SSL context ID as 1.
OK

AT+QSSLCFG="sslversion",1,1 //Set SSL version as 1 which means TLSv1.0.
OK

AT+QSSLCFG="ciphersuite",1,0x0005 //Set SSL cipher suite as 0x0005 which means RC4-SHA.
OK

AT+QSSLCFG="secllevel",1,2 //Set SSL verify level as 2 which means CA certificate, client
certificate and client private key should all be uploaded by
AT+QFUPL command.
OK

AT+QFUPL="cacert.pem" //Upload CA certificate to UFS.
CONNECT
<Input file bin data>
+QFUPL:1216,7648
OK

AT+QFUPL="clientcert.pem" //Upload client certificate to UFS.
CONNECT
<Input file bin data>
+QFUPL:1216,5558

```

OK

AT+QFUPL="clientkey.pem" //Upload client private key to UFS.

CONNECT

<Input file bin data>

+QFUPL:1706,538

OK

AT+QSSLCFG="cacert",1,"cacert.pem" //Configure the path of CA certificate for SSL context 1.

OK

AT+QSSLCFG="clientcert",1,"clientcert.pem" //Configure the path of client certificate for SSL context 1.

OK

AT+QSSLCFG="clientkey",1,"clientkey.pem" //Configure the path of client private key for SSL context 1.

OK

AT+QHTTPURL=45,80 //Set the URL of HTTPS server which will be accessed.

CONNECT

https://220.180.239.212:8011/processorder.php //Input the URL whose length should be 45 bytes. (This URL is only an example. Please input the correct URL in practical test.)

OK

//POST request information from UFS file, and read the HTTPS response information through storing it to a UFS file.

AT+QHTTPPOSTFILE="5.txt",80 //Send HTTPS POST request. The POST body is obtained from "5.txt", and the maximum response time is 80s.

OK

+QHTTPPOSTFILE: 0,200,285 //HTTPS POST request is sent successfully. And then the HTTPS response body can be read via either AT+QHTTPREAD or AT+QHTTPREADFILE command.

AT+QHTTPREADFILE="6.txt",80 //Read HTTPS response body through storing it to "6.txt". The maximum time to wait for HTTPS session to be closed is 80s.

OK

+QHTTPREADFILE: 0 //HTTPS response body is stored successfully.

4 Error Handling

4.1. Executing HTTP(S) AT Commands Fails

When executing HTTP(S) AT commands, if “ERROR” response is received from the module, please check whether the (U)SIM card is inserted and whether it is “+CPIN: READY” returned when executing AT+CPIN?.

4.2. PDP Activation Fails

If it is failed to activate a PDP context by AT+QIACT command, please check the following configurations:

1. Query the PS domain status by AT+CEREG? (for LTE Cat M and Cat NB2 networks) or AT+CGREG? (for EGPRS network) and make sure the PS domain has been registered.
2. Query the PDP context parameters by AT+QICSGP=<contextID> command and make sure the APN of the specified PDP context has been set.
3. Make sure the specified PDP context ID is neither used by PPP nor activated by AT+CGACT command.
4. The module supports maximum three PDP contexts activated simultaneously under LTE Cat M/EGPRS and maximum two under LTE Cat NB2.

If all above configurations are correct, but activating the PDP context by AT+QIACT command still fails, please reboot the module to resolve this issue. After rebooting the module, please check the configurations mentioned above at least three times and each time at an interval of 10 minutes to avoid frequently rebooting the module.

4.3. DNS Parse Fails

When executing AT+QHTTPGET, AT+QHTTPPOST and QHTTPPOSTFILE commands, if “+CME ERROR: 714” (714: HTTP(S) DNS error) is returned, please check the following aspects:

1. Make sure the domain name of HTTP(S) server is valid.
2. Query the status of the PDP context by AT+QIACT? command to make sure the specified PDP context has been activated successfully.
3. Query the address of DNS server by AT+QIDNSCFG command to make sure the address of DNS server is not null or "0.0.0.0".

If the DNS server address is null or "0.0.0.0", there are three solutions:

1. Reassign a valid DNS server address by AT+QIDNSCFG command.
2. Deactivate the PDP context by AT+QIDEACT command, and then re-activate the PDP context via AT+QIACT command.
3. If the module has registered to NB-IoT network, execute AT+QCFG="nccconf",101 command to enable ePCO, and then re-register to network via AT+CFUN=0/1 command or reboot the module.

4.4. Entering Data Mode Fails

When executing AT+QHTTPURL, AT+QHTTPGET, AT+QHTTPPOST and AT+QHTTPREAD commands, if "+CME ERROR: 704" (704: HTTP(S) UART busy) is returned, please check whether there are other ports in data mode, since the module only supports one port in data mode at a time. If any, please re-execute these commands after other ports have exited from data mode.

4.5. Sending GET/POST Requests Fails

When executing AT+QHTTPGET, AT+QHTTPPOST and QHTTPPOSTFILE commands, if a failed result is received, please check the following configurations:

1. Make sure the URL inputted via AT+HTTPURL command is valid and can be accessed.
2. Make sure the specified server supports GET/POST commands.
3. Make sure the PDP context has been activated successfully.

If all above configurations are correct, but sending GET/POST requests by AT+QHTTPGET, AT+QHTTPPOST and AT+QHTTPPOSTFILE commands still fails, please deactivate the PDP context by AT+QIDEACT and re-activate the PDP context by AT+QIACT to resolve this issue. If activating the PDP context fails, please refer to **Chapter 4.2** to resolve it.

4.6. Reading Response Fails

Before reading response by AT+QHTTPREAD and AT+QHTTPREADFILE commands, customers should execute AT+QHTTPGET, AT+QHTTPPOST and AT+QHTTPPOSTFILE commands and wait until the following URC information are reported:

```
+QHTTPGET: <err>[,<httprcode>,<content_length>]"/  
+QHTTPPOST: <err>[,<httprcode>,<content_length>]"/  
+QHTTPPOSTFILE: <err>[,<httprcode>,<content_length>]"/
```

During executing AT+QHTTPREAD and AT+QHTTPREADFILE commands, if customers encounter some errors, such as: "+CME ERROR: 717" (717: HTTP(S) socket read error), please resend HTTP(S) GET/POST requests to HTTP(S) server by AT+QHTTPGET, AT+QHTTPPOST and AT+QHTTPPOSTFILE commands. If sending GET/POST requests to HTTP(S) server fails, please refer to **Chapter 4.5** to resolve it.

5 Summary of Error Codes

The error code <err> indicates an error related to mobile equipment or network. The details about <err> are described in the following table.

Table 1: Summary of Error Codes

<err>	Meaning
0	Operation successful
701	HTTP(S) unknown error
702	HTTP(S) timeout
703	HTTP(S) busy
704	HTTP(S) UART busy
705	HTTP(S) no GET/POST requests
706	HTTP(S) network busy
707	HTTP(S) network open failed
708	HTTP(S) network no configuration
709	HTTP(S) network deactivated
710	HTTP(S) network error
711	HTTP(S) URL error
712	HTTP(S) empty URL
713	HTTP(S) IP address error
714	HTTP(S) DNS error
715	HTTP(S) socket create error
716	HTTP(S) socket connect error
717	HTTP(S) socket read error

718	HTTP(S) socket write error
719	HTTP(S) socket closed
720	HTTP(S) data encode error
721	HTTP(S) data decode error
722	HTTP(S) read timeout
723	HTTP(S) response failed
724	Incoming call busy
725	Voice call busy
726	Input timeout
727	Wait data timeout
728	Wait HTTP(S) response timeout
729	Memory allocation failed
730	Invalid parameter

6 Summary of HTTP(S) Response Codes

<httprcode> indicates the response codes from HTTP(S) server. The details about <httprcode> are described in the following table.

Table 2: Summary of HTTP(S) Response Codes

<httprcode>	Meaning
200	OK
403	Forbidden
404	Not found
409	Conflict
411	Length required
500	Internal server error

7 Appendix A References

Table 3: Related Documents

SN	Document Name	Remark
[1]	RFC2616	Hyper Text Transport Protocol
[2]	Quectel_BG95&BG77_TCP(IP)_AT_Commands_Manual	BG95&BG77 TCP/IP AT Commands Manual
[3]	Quectel_BG95&BG77_FILE_AT_Commands_Manual	BG95&BG77 FILE AT Commands Manual
[4]	Quectel_BG95&BG77_AT_Commands_Manual	BG95&BG77 AT Commands Manual
[5]	Quectel_BG95&BG77_SSL_AT_Commands_Manual	BG95&BG77 SSL AT Commands Manual

Table 4: Terms and Abbreviations

Abbreviation	Description
DNS	Domain Name Server
DTR	Data Terminal Ready
ePCO	Extended Protocol Configuration Options
HTTP	Hyper Text Transport Protocol
HTTPS	Hyper Text Transfer Protocol Secure
PPP	Point-to-Point Protocol
SSL	Security Socket Layer
URI	Uniform Resource Identifier
URL	Uniform Resource Locator