SIM7070_SIM7080_SIM7090 Series_SSL _Application Note

LPWA Module

SIMCom Wireless Solutions Limited
Building B, SIM Technology Building, No.633, Jinzhong Road
Changning District, Shanghai P.R. China
Tel: 86-21-31575100
support@simcom.com
www.simcom.com

Document Title:	SIM7070_SIM7080_SIM7090 Series_SSL_Application Note
Version:	1.00
Date:	2020.08.11
Status:	Released

GENERAL NOTES

SIMCOM OFFERS THIS INFORMATION AS A SERVICE TO ITS CUSTOMERS, TO SUPPORT APPLICATION AND ENGINEERING EFFORTS THAT USE THE PRODUCTS DESIGNED BY SIMCOM. THE INFORMATION PROVIDED IS BASED UPON REQUIREMENTS SPECIFICALLY PROVIDED TO SIMCOM BY THE CUSTOMERS. SIMCOM HAS NOT UNDERTAKEN ANY INDEPENDENT SEARCH FOR ADDITIONAL RELEVANT INFORMATION, INCLUDING ANY INFORMATION THAT MAY BE IN THE CUSTOMER'S POSSESSION. FURTHERMORE, SYSTEM VALIDATION OF THIS PRODUCT DESIGNED BY SIMCOM WITHIN A LARGER ELECTRONIC SYSTEM REMAINS THE RESPONSIBILITY OF THE CUSTOMER OR THE CUSTOMER'S SYSTEM INTEGRATOR. ALL SPECIFICATIONS SUPPLIED HEREIN ARE SUBJECT TO CHANGE.

COPYRIGHT

SIMCom Wireless Solutions Limited
Building B, SIM Technology Building, No.633 Jinzhong Road, Changning District, Shanghai P.R. China
Tel: +86 21 31575100
Email: simcom@simcom.com

For more information, please visit:
https://www.simcom.com/download/list-863-en.html

For technical support, or to report documentation errors, please visit:
https://www.simcom.com/ask/ or email to: support@simcom.com

About Document

Version History

Version	Date	Owner	What is new
V1.00	2020.08.11	Wei.Zhang	All

Scope

This document applies to the following products

Name	Type	Size(mm)	Comments
SIM7080G	CAT-M/NB	17.6*15.7 *2.3	N/A
SIM7070G/SIM7070E	CAT-M/NB/GPRS	24*24*2.4	N/A
SIM7070G-NG	NB/GPRS	24*24*2.4	N/A
SIM7090G	CAT-M/NB	14.8*12.8*2.0	N/A

Contents

1 Introduction

1.1 Purpose of the document

Based on module AT command manual, this document will introduce SSL application process.

The following applications of SIM7070_SIM7080_SIM7090 series module support SSL:
HTTP, FTP, TCPUDP, MQTT and EMAIL.

Developers could understand and develop application quickly and efficiently based on this document.

1.2 Related documents

[1] SIM7070_SIM7080_SIM7090 Series_AT Command Manual
[2] SIM7070_SIM7080_SIM7090 Series_FS_Application Note
[3] SIM7070_SIM7080_SIM7090 Series_HTTP(S)_Application Note
[4] SIM7070_SIM7080_SIM7090 Series_FTP(S)_Application Note
[5] SIM7070_SIM7080_SIM7090 Series_TCPUDP(S)_Application Note
[6] SIM7070_SIM7080_SIM7090 Series_MQTT(S)_Application Note
[7] SIM7070_SIM7080_SIM7090 Series_Email_Application Note
[8] RFC7925
[9] RFC2246

1.3 Conventions and abbreviations

In this document, the GSM engines are referred to as following term:

- ME (Mobile Equipment);
- MS (Mobile Station);
- TA (Terminal Adapter);
- DCE (Data Communication Equipment) or facsimile DCE (FAX modem, FAX board);

In application, controlling device controls the GSM engine by sending AT Command via its serial interface. The controlling device at the other end of the serial line is referred to as following term:

- TE (Terminal Equipment);
- DTE (Data Terminal Equipment) or plainly "the application" which is running on an embedded system;

2 SSL Introduction

- SSL (Secure Sockets Layer), a security protocol. It was put forward by Netscape in the first version of Web browser. The aim is to provide security and data integrity for network communications. SSL encrypts the network connections at the transport layer.

- SSL uses public key technology to ensure the confidentiality and reliability of communication between two applications and to ensure that communication between client and server applications is not eaves dropped by attackers. It can be supported at both ends of the server and client, and has become an industrial standard for secure communication over the Internet. Current Web browsers generally combine HTTP and SSL to achieve secure communication. This Agreement and its successor are TLS (Transport Layer Security, TLS).

- TLS uses key algorithm to provide endpoint authentication and communication security on the Internet, It is based on the public key infrastructure. In typical implementations, however, only the network server is authenticated reliably, while the client is not necessarily. This is because the public key infrastructure is generally commercial, and electronic signature certificates usually need to be paid for. The protocol is designed to enable master-slave architecture application communication itself to prevent tapping, tampering, and message forgery

- DTLS (Datagram Transport Layer Security) is the data transmission layer security protocol. TLS cannot be used to ensure the security of the data transmitted on UDP, so Datagram TLS extends the existing TLS protocol architecture to support UDP, that is datagram transmission. DTLS 1.0 is based on TLS 1.1, and DTLS 1.2 is based on TLS 1.2.

- DTLS, TLS and SSL encrypt network connections at the transport layer, DTLS is above the UDP transport protocol, and TLS is above the TCP transport protocol.

2.1 SSL Versions and Cipher Suites

The following SSL versions are supported.

Version
TLS1.0
TLS1.1
TLS1.2
DTLS1.0

DTLS1.2

The following table shows SSL cipher suites supported by SIM7070_SIM7080_SIM7090 series module. For detailed description of cipher suites, please refer to *RFC 2246-The TLS Protocol Version 1.0*.

Code of Cipher Suites	Name of Cipher Suites
0x008A	PSK_WITH_RC4_128_SHA
0x008B	PSK_WITH_3DES_EDE_CBC_SHA
0x008C	PSK_WITH_AES_128_CBC_SHA
0x008D	PSK_WITH_AES_256_CBC_SHA
0x00A8	PSK_WITH_AES_128_GCM_SHA256
0x00A9	PSK_WITH_AES_256_GCM_SHA384
0x00AE	PSK_WITH_AES_128_CBC_SHA256
0x00AF	PSK_WITH_AES_256_CBC_SHA384
0xC0A8	PSK_WITH_AES_128_CCM_8
0x002F	RSA_WITH_AES_128_CBC_SHA
0x0033	DHE_RSA_WITH_AES_128_CBC_SHA
0x0035	RSA_WITH_AES_256_CBC_SHA
0x0039	DHE_RSA_WITH_AES_256_CBC_SHA
0x003C	RSA_WITH_AES_128_CBC_SHA256
0x003D	RSA_WITH_AES_256_CBC_SHA256
0x0067	DHE_RSA_WITH_AES_128_CBC_SHA256
0x006B	DHE_RSA_WITH_AES_256_CBC_SHA256
0x009C	RSA_WITH_AES_128_GCM_SHA256
0x009D	RSA_WITH_AES_256_GCM_SHA384
0x009E	DHE_RSA_WITH_AES_128_GCM_SHA256
0x009F	DHE_RSA_WITH_AES_256_GCM_SHA384
0xC004	ECDH_ECDSA_WITH_AES_128_CBC_SHA
0xC005	ECDH_ECDSA_WITH_AES_256_CBC_SHA
0xC009	ECDHE_ECDSA_WITH_AES_128_CBC_SHA
0xC00A	ECDHE_ECDSA_WITH_AES_256_CBC_SHA
0xC00E	ECDH_RSA_WITH_AES_128_CBC_SHA
0xC00F	ECDH_RSA_WITH_AES_256_CBC_SHA
0xC013	ECDHE_RSA_WITH_AES_128_CBC_SHA
0xC014	ECDHE_RSA_WITH_AES_256_CBC_SHA
0xC023	ECDHE_ECDSA_WITH_AES_128_CBC_SHA256
0xC024	ECDHE_ECDSA_WITH_AES_256_CBC_SHA384
0xC025	ECDH_ECDSA_WITH_AES_128_CBC_SHA256
0xC026	ECDH_ECDSA_WITH_AES_256_CBC_SHA384
0xC027	ECDHE_RSA_WITH_AES_128_CBC_SHA256
0xC028	ECDHE_RSA_WITH_AES_256_CBC_SHA384

0xC029	ECDH_RSA_WITH_AES_128_CBC_SHA256
0xC02A	ECDH_RSA_WITH_AES_256_CBC_SHA384
0xC02B	ECDHE_ECDSA_WITH_AES_128_GCM_SHA256
0xC02C	ECDHE_ECDSA_WITH_AES_256_GCM_SHA384
0xC02D	ECDH_ECDSA_WITH_AES_128_GCM_SHA256
0xC02E	ECDH_ECDSA_WITH_AES_256_GCM_SHA384
0xC02F	ECDHE_RSA_WITH_AES_128_GCM_SHA256
0xC030	ECDHE_RSA_WITH_AES_256_GCM_SHA384
0xC031	ECDH_RSA_WITH_AES_128_GCM_SHA256
0xC032	ECDH_RSA_WITH_AES_256_GCM_SHA384
0xC0AE	ECDHE_ECDSA_WITH_AES_128_CCM_8
0xC09C	RSA_WITH_AES_128_CCM
0xC09D	RSA_WITH_AES_256_CCM
0xC09E	DHE_RSA_WITH_AES_128_CCM
0xC09F	DHE_RSA_WITH_AES_256_CCM
0xC0A0	RSA_WITH_AES_128_CCM_8
0xC0A1	RSA_WITH_AES_256_CCM_8
0xC0A2	DHE_RSA_WITH_AES_128_CCM_8
0xC0A3	DHE_RSA_WITH_AES_256_CCM_8
0xCC13	ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256
0xCC14	ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256
0xCC15	DHE_RSA_WITH_CHACHA20_POLY1305_SHA256

2.2 Supported Certificate format

- SSL Certificates is a file that uses digital encryption technology to encrypt the details of the publisher's information.

- SSL certificates are encoded in the binary format specified in the X.509 ITU-T standard, which is a well-established telecom standard for storing structured binary data. The binary X.509 data is sometimes stored as raw binary data and sometimes encoded using Base64 encoding. Several X.509 encoding formats exists. We can handle the following formats: .PEM, .DER, and .P7B.

- Typically, a certificate's private key and public certificate are stored as two .PEM encoded files. Root certificates (CAs) are typically stored in the container format .P7B, which can store multiple CAs. The root certificates are used by SSL function when certifying (authenticating) the peer side.

3 AT Commands that support SSL

The module provides AT commands that can be used by device terminals as follows:

Command	Description
AT+CSSLCFG	Configure SSL parameters of a context identifier
AT+CASSLCFG	Set SSL Certificate and Timeout Parameters(TCPUDP)
AT+SHSSL	Select SSL Configure(HTTP)
AT+FTPSSL	Select FTP SSL Configure
AT+SMSSL	Select SSL Configure
AT+EMAILSSL	Set Email SSL function

For detail information, please refer to "SIM7070_SIM7080_SIM7090 Series_AT Command Manual".

4 Certificate Management

When SSL establishes communication, it is necessary to verify the identity of both sides of the communication, which is divided into one-way authentication and two-way authentication.
One way authentication is the client to verify the certificate of the server. The server sends the server certificate to the client. The client verifies that the root certificate that issued the server certificate is trustworthy, and if so continues the communication process.

After the two-way authentication client verifies the server certificate, the client needs to send its own certificate to the server and let the server verify its client certificate. The validation process is the same, all need to confirm whether the root certificate of the certificate can be trusted.

Module uses its own binary format for storing certificates, a format optimized for speed and size. We provide AT command that can convert standard .PEM, .DER, and .P7B formats into its internal format.

By default, SSL only performs server-side authentication, client-side authentication is optional.

The following example is to visit Baidu web as an example.

4.1 Configure SSL parameters

Module can support the existence of 6 configuration files with sequence numbers from 0-5.

The following example SSL configuration will take the second configuration file as an example.

//Example of configure SSL parameters.	
AT+CSSLCFG="SSLVERSION",1,3	//Set the protocol type of SSL version 1 means the second configuration file. 3 means TLS1.2
OK	
AT+CSSLCFG="CIPHERSUITE",1,0,0x009c	//Configure the ciphersuite. 1 means the second configuration file. 0 means cipher_index. 0x009c means TLS_RSA_WITH_AES_128_GCM_SHA256. You can check it from the Supported SSL Cipher Suites table in SSL Versions and Cipher Suites

	You can choose not to fill, and it will automatically default. Otherwise, select at least 1 cipher suite or at most 8 cipher suites. Range:0-7 This is an option.
OK	
…	
AT+CSSLCFG="CIPHERSUITE",1,7,0x002f	
OK	
AT+CSSLCFG="SNI",1,www.baidu.com	//Configure SNI 1 means the second configuration file. www.baidu.com is severname. SNI (Server Name Indication) is an extension of TLS and is used to solve the situation where a server has multiple domain names. This is an option.
OK	
AT+CSSLCFG="IGNORERTCTIME ",1,1	//Configure whether to ignore time. The first 1 means the second configuration file. The second 1 means ignore the RTC time. If it's 0 that means do not ignore the RTC time. This is an option.
OK	
AT+CSSLCFG="CTXINDEX",1	//Query all the parameters that have been set. This is an option.
+CSSLCFG: 3,0x009c,0x0035,0xcca8,0xc030,0xcca8,0xc013,0x009d,0x002f,1,""	
OK	

4.2 Import and Convert Root CA

A root CA certificate can be used by module during the initial SSL handshake. The root CA is then used as a trusted third party and module uses the root CA for certifying the peer's certificate. Certifying the peer side is optional in Module, however, if used, the peer side must provide a certificate signed by a trusted root CA as part of the SSL handshake.

The root CA certificate(s) must be stored on a file system. In many cases, only one root CA certificate is required in the device. The same root CA can be used to sign all certificates the device must verify. The below example programs show how to store and use one root CA.

```
//Example of import and convert Root CA.
AT+CFSINIT                                  //Initialize AT relate to file system functions
OK
AT+CFSWFILE=3,GlobalSignRootCA.crt,0,383,   //Import CA take Baidu web as an example.
10000                                       3 means this file will put in the “customer”
                                            directory.
                                            GlobalSignRootCA.crt is a CA file that will be
                                            written to “customer” directory.
                                            0 is starting writing point, 383 is the size of CA,
                                            10000 means that you must complete it in 10
                                            seconds.
OK
AT+CSSLCFG="CONVERT",2,CertumGlobalSe       //Convert GlobalSignRootCA.crt, and save it to file
rvicesCASHA2.crt                            system. 2 is convert root CA
OK
AT+CFSDFILE=3,GlobalSignRootCA.crt          //After convert, delete temp GlobalSignRootCA.crt
                                            which is in “customer” directory because of
                                            security considerations.
                                            3 means it belongs to the “customer” directory.
                                            GlobalSignRootCA.crt is root CA name.
OK
AT+CFSTERM                                  //Release AT relates to file system functions.
OK
```

NOTE

Perform the above steps at least once. As long as the certificate is not updated, there is no need to repeat the operation.

4.3 Import and Convert Certificate

```
//Example of import and convert certificate.
AT+CFSINIT                                  //Initialize AT relate to file system functions
OK
AT+CFSWFILE=3,Client.cer,0,1024,10000       //Import Client.cer.
                                            3 means this file will put in the “customer” directory.
                                            Client.cer is a certificate file that will be written to
                                            “customer” directory.
                                            0 is starting writing point,1024 is the size of
```

	certificate,10000 means that you must complete it in 10000 ms.
OK	
AT+CFSWFILE=3,Client_key.pem,0,512,10000	//Import Client_key.pem 3 means this file will put in the “customer” directory. Client_key.pem is private key that will be written to “customer” directory. 0 is starting writing point,512 is the size of private key,10000 means that you must complete it in 10000 ms.
OK	
AT+CSSLCFG="CONVERT",1,Client.cer,Client_key.pem,"simcom"	//Convert Client.cer and Client_key.pem , and save it to file system. 1 means convert certificate. Client.cer is certificate. Client_key.pem is private key of Client.cer. If Client_key.pem is encrypted,”simcom” is the corresponding password.
OK	
AT+CFSDFILE=3,Client.cer	//After convert, delete temp Client.cer which is in “customer” directory because of security considerations. 3 means it belongs to the “customer” directory. Client.cer is certificate name.
OK	
AT+CFSDFILE=3,Client_key.pem	// After convert, delete temp Client_key.pem which is in “customer” directory because of security considerations. 3 means it belongs to the “customer” directory. Client_key.pem is private key.
AT+CFSTERM	//Release AT relates to file system functions.
OK	

NOTE

Perform the above steps at least once. As long as the certificate is not updated, there is no need to repeat the operation.

4.4 Import and Convert PSK

PSK(pre-shared key) is a series of keys that have been determined and known by both parties before

communication, and this series of keys relies on PSK Identify (referred to as PSK ID) for indexing.

After the client sends the Client Key Exchange with the PSK Client Params, the server will index the PSK ID to find the preset key (this key is also stored locally on the client), and then use an algorithm to combine this key and Parameters such as the random numbers of both parties generate the final symmetric key.

//Example of import and convert PSK.	
AT+CFSINIT	//Initialize AT relate to file system functions
OK	
AT+CFSWFILE=3,psktable.secret,0,512,10000	//Import psktable.secret. 3 means this file will put in the “customer” directory. psktable.secret will be written to “customer” directory. 0 is starting writing point,512 is the size of psktable.secret. 10000 means that you must complete it in 10 second. The psktable (psktable.secret) format as follows: <Identity>:<psk_key> Identity and psk_key are corresponding. Each Identity correspond to a psk_key. For example: user_zhao:313233 It should be noted that the Identity is string type and psk_key is hexadecimal string(e.g: If the psk_key is string “123”, you must write that “313233”)
OK	
AT+CSSLCFG="CONVERT",3,psktable.secret	//Configure the type of psktable to be converted, and 3 is psktable.
OK	
AT+CFSDFILE=3,psktable.secret	//After convert, delete temp psktable.secret which is in “customer” directory because of security considerations. 3 means it belongs to the “customer” directory.
OK	psktable.secret is PSK file name.
AT+CFSTERM	//Release AT relates to file system functions.
OK	

NOTE

Perform the above steps at least once. As long as the certificate is not updated, there is no need to repeat the operation.