SSL/TLS User Guide

1vv0300989 Rev. 14 – 2017-11-28

TELIT
TECHNICAL
DOCUMENTATION

[04.2016]

Mod. 0809 2016-08 Rev.7

SPECIFICATIONS ARE SUBJECT TO CHANGE WITHOUT NOTICE

NOTICES LIST

While reasonable efforts have been made to assure the accuracy of this document, Telit assumes no liability resulting from any inaccuracies or omissions in this document, or from use of the information obtained herein. The information in this document has been carefully checked and is believed to be reliable. However, no responsibility is assumed for inaccuracies or omissions. Telit reserves the right to make changes to any products described herein and reserves the right to revise this document and to make changes from time to time in content hereof with no obligation to notify any person of revisions or changes. Telit does not assume any liability arising out of the application or use of any product, software, or circuit described herein; neither does it convey license under its patent rights or the rights of others.

It is possible that this publication may contain references to, or information about Telit products (machines and programs), programming, or services that are not announced in your country. Such references or information must not be construed to mean that Telit intends to announce such Telit products, programming, or services in your country.

COPYRIGHTS

This instruction manual and the Telit products described in this instruction manual may be, include or describe copyrighted Telit material, such as computer programs stored in semiconductor memories or other media. Laws in the Italy and other countries preserve for Telit and its licensors certain exclusive rights for copyrighted material, including the exclusive right to copy, reproduce in any form, distribute and make derivative works of the copyrighted material. Accordingly, any copyrighted material of Telit and its licensors contained herein or in the Telit products described in this instruction manual may not be copied, reproduced, distributed, merged or modified in any manner without the express written permission of Telit. Furthermore, the purchase of Telit products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Telit, as arises by operation of law in the sale of a product.

COMPUTER SOFTWARE COPYRIGHTS

The Telit and 3rd Party supplied Software (SW) products described in this instruction manual may include copyrighted Telit and other 3rd Party supplied computer programs stored in semiconductor memories or other media. Laws in the Italy and other countries preserve for Telit and other 3rd Party supplied SW certain exclusive rights for copyrighted computer programs, including the exclusive right to copy or reproduce in any form the copyrighted computer program. Accordingly, any copyrighted Telit or other 3rd Party supplied SW computer programs contained in the Telit products described in this instruction manual may not be copied (reverse engineered) or reproduced in any manner without the express written permission of Telit or the 3rd Party SW supplier. Furthermore, the purchase of Telit products shall not be deemed to grant either directly or by implication, estoppel, or otherwise, any license under the copyrights, patents or patent applications of Telit or other 3rd Party supplied SW, except for the normal non-exclusive, royalty free license to use that arises by operation of law in the sale of a product.

USAGE AND DISCLOSURE RESTRICTIONS

I. License Agreements

The software described in this document is the property of Telit and its licensors. It is furnished by express license agreement only and may be used only in accordance with the terms of such an agreement.

II. Copyrighted Materials

Software and documentation are copyrighted materials. Making unauthorized copies is prohibited by law. No part of the software or documentation may be reproduced, transmitted, transcribed, stored in a retrieval system, or translated into any language or computer language, in any form or by any means, without prior written permission of Telit

III. High Risk Materials

Components, units, or third-party products used in the product described herein are NOT fault-tolerant and are NOT designed, manufactured, or intended for use as on-line control equipment in the following hazardous environments requiring fail-safe controls: the operation of Nuclear Facilities, Aircraft Navigation or Aircraft Communication Systems, Air Traffic Control, Life Support, or Weapons Systems (High Risk Activities"). Telit and its supplier(s) specifically disclaim any expressed or implied warranty of fitness for such High Risk Activities.

IV. Trademarks

TELIT and the Stylized T Logo are registered in Trademark Office. All other product or service names are the property of their respective owners.

V. Third Party Rights

The software may include Third Party Right software. In this case you agree to comply with all terms and conditions imposed on you in respect of such separate software. In addition to Third Party Terms, the disclaimer of warranty and limitation of liability provisions in this License shall apply to the Third Party Right software.

TELIT HEREBY DISCLAIMS ANY AND ALL WARRANTIES EXPRESS OR IMPLIED FROM ANY THIRD PARTIES REGARDING ANY SEPARATE FILES, ANY THIRD PARTY MATERIALS INCLUDED IN THE SOFTWARE, ANY THIRD PARTY MATERIALS FROM WHICH THE SOFTWARE IS DERIVED (COLLECTIVELY “OTHER CODE”), AND THE USE OF ANY OR ALL THE OTHER CODE IN CONNECTION WITH THE SOFTWARE, INCLUDING (WITHOUT LIMITATION) ANY WARRANTIES OF SATISFACTORY QUALITY OR FITNESS FOR A PARTICULAR PURPOSE.

NO THIRD PARTY LICENSORS OF OTHER CODE SHALL HAVE ANY LIABILITY FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING WITHOUT LIMITATION LOST PROFITS), HOWEVER CAUSED AND WHETHER MADE UNDER CONTRACT, TORT OR OTHER LEGAL THEORY, ARISING IN ANY WAY OUT OF THE USE OR DISTRIBUTION OF THE OTHER CODE OR THE EXERCISE OF ANY RIGHTS GRANTED UNDER EITHER OR BOTH THIS LICENSE AND THE LEGAL TERMS APPLICABLE TO ANY SEPARATE FILES, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

APPLICABILITY TABLE

PRODUCTS

	Platform Version ID[1]	Technology
GL865 SERIES	10	2G
GE865-QUAD		
GE864 V2 SERIES		
GL868-DUAL		
GE910 SERIES	13	
GE910-QUAD V3	16	
GE866-QUAD		
GL865 V3 SERIES		
GL868-DUAL V3		
HE910 SERIES	12	3G
UE910 SERIES		
UL865 SERIES		
UE866 SERIES		
LE910 Cat1 SERIES	20	4G
LE910 V2 SERIES		
LE866 SERIES	23	
ME866A1 SERIES		

[1] Platform Version ID is a reference used in the document. It identifies the different SW versions, e.g. 10 for SW version 10.xx.xxx, 13 for SW version 13.xx.xxx, etc.

CONTENTS

TABLES LIST

AT Command List

The following list, organized in alphabetical order, shows the AT commands covered by this User Guide. The number close to each command indicates the page of the first AT command occurrence.

1. INTRODUCTION

1.1. Scope

This document describes the set of the Telit AT commands regarding the SSL/TLS protocols use.

1.2. Audience

The guide is intended for users that need to develop applications based on secure connection channels. The reader is expected to have knowledge in wireless technology as well as in SSL/TLS security protocols.

1.3. Contact Information, Support

For general contact, technical support services, technical questions and report documentation errors contact Telit Technical Support at:

- TS-EMEA@telit.com
- TS-AMERICAS@telit.com
- TS-APAC@telit.com

Alternatively, use:

http://www.telit.com/support

For detailed information about where you can buy the Telit modules or for recommendations on accessories and components visit:

http://www.telit.com

Our aim is to make this guide as helpful as possible. Keep us informed of your comments and suggestions for improvements.
Telit appreciates feedback from the users of our information.

1.4. Text Conventions

Danger – This information MUST be followed or catastrophic equipment failure or bodily injury may occur.

Caution or Warning – Alerts the user to important points about integrating the module, if these points are not followed, the module and end user equipment may fail or malfunction.

Tip or Information – Provides advice and suggestions that may be useful when integrating the module.

All dates are in ISO 8601 format, i.e. YYYY-MM-DD.

1.5. Related Documents

[1] AT Command Reference Guide, 80000ST10025a

[2] Telit Modules Software User Guide, 1vv0300784

[3] IP Easy User Guide, 80000ST10028A

[4] Virtual Serial Device, Application Note, 80000NT10045A

[5] Telit 3G Modules Ports Arrangements, User Guide, 1vv0300971

[6] Telit 3G Modules AT Commands Reference Guide, 80378ST10091A

[7] LE910 V2 Series AT Commands Reference Guide, 80446ST10707A

[8] RFC 4217 Standard

[9] RFC 2228 Standard

[10] LE910 V2, LE910 Cat1 Ports Arrangements User Guide, 1vv0301252

[11] LE866 Series AT Commands Reference Guide, 80471ST10691A

[12] LE866, ME866A1 Ports Arrangements User Guide, 1vv0301469

2. PRELIMINARY INFORMATION

The present guide introduces the AT commands used to manage SSL sockets, and provides examples describing their use. To have detailed syntax information on the AT commands refer to the AT Commands Reference Guide ([1], [6], [7], and [11]) according to the Platform Version ID of the module you are using.

To work with sockets, you must configure a PDP context using the +CGDCONT command, and activate it as shown briefly below. To get more information refer to document [3].

AT+CGDCONT=<cid>,<PDP_type>,<APN>,…

Where:

<**cid**> PDP Context Identifier. Use the test command to know the <cid> range of the used module.

<**PDP_type**> a string which specifies the type of Packet Data Protocol.

<**APN**> Access Point Name, a string containing the logical name used to select GGSN or external packet data network. The ISP provides this parameter.

… other parameters.

Use the #SGACT command to activate the PDP.

AT#SGACT= <cid>,<stat>[,<userId>,<pwd>]

Where:

<**cid**> PDP Context Identifier. Use the test command to know the <cid> range of the used module.

<**stat**> context status: 0 = deactivate the context, 1 = activate the context.

… optional parameters.

Example

Define PDP context.
AT+CGDCONT=1,"IP","Access_Point_Name",…
OK

Before activating a PDP context, it must be bound to a socket through the #SCFG command.
AT#SGACT=1,1 ← activate the PDP context
#SGACT:212.195.45.65 ← returns the IP address provided by the network
OK

3. PROTOCOL SELECTION, #SSLSECCFG2 COMMAND

TLS and its predecessor SSL are cryptographic protocols used over the Internet to provide secure data communication in several applications. A classic example is the HTTPS connection between Web browsers and Web servers, see chapter 6.

For TLS protocol, see standards:

- RFC 2246 - TLS Protocol Version 1.0
- RFC 4346 - TLS Protocol Version 1.1
- RFC 5246 - TLS Protocol Version 1.2

Use the following command to select the protocol.

AT#SSLSECCFG2=<SSId>,<version>
[,<unused_A>[,<unused_B>[,<unused_C>[,<unused_D>]]]]

Where:

<**SSId**> Secure Socket ID. Use the AT#SSLSECCFG2=? test command to know the <SSId> range of the used module.

<**version**> It selects the SSL/TLS protocol version, see table below.

	<**version**> parameter			
	0	1	2	3
Platform Version ID[2]	SSL v3	TLS v1.0	TLS v1.1	TLS v1.2
10, 13, 16 (2G) See the warning note below	no	yes	yes	default
12 (3G)	yes	default	yes	yes
20 (4G)	yes	default	yes	yes
23 (4G)	yes	default	yes	yes

Tab. 1: SSL/TLS Protocol Versions

2G modules do not support #SSLSECCFG2 command. In this case, the module proposes its default protocol to the server. If the server does not support the protocol, automatically proposes TLS v1.1, or TLS1.0 protocol to the modem.

[2] See Applicability Table.

4. SSL CONFIGURATION

Before opening a SSL socket and exchange data with it, you must perform the following steps.

- Enable SSL channel
- Set authentication mode and timeouts
- Store Security data in the module if the authentication is required

4.1. Enabling a SSL Channel, #SSLEN Command

To provide communication security over a channel, enable a SSL socket using the #SSLEN command. If <Enable> parameter is not set to 1, any attempt to set SSL parameters fails.

AT#SSLEN= <SSId>,<Enable>

Where:

<**SSId**> Secure Socket ID. Use the AT#SSLEN=? test command to know the <SSId> range of the used module.

<**Enable**> status: 0 = deactivate secure socket (default), 1 = activate secure socket.

Example

AT#SSLEN=1,1 ← enable the SSL socket identified by <SSId>=1
OK

The #SSLEN command behavior depends on the number of the SSL sockets that the module supports, and on the AT instance you are using to enter the command. See chapters 4.2, and 4.3.

4.2. SSL Security Configuration, #SSLSECCFG Command

The cipher suite is the set of algorithms used to negotiate the security settings for a network connection using the SSL/TLS network protocol. The cipher suite includes:

- Key exchange algorithm used for the authentication during the handshake
- Encryption algorithm used to encrypt the message
- Hash function for data integrity

If the remote server does not support one of the cipher suites provided by the module, the handshake fails.

Here are some examples of algorithms used by the cipher suites supported by the modules (clients), see also chapter 8.1.

TLS_RSA_WITH_RC4_128_MD5 uses:

- RSA Key exchange algorithm for the authentication during handshake
- RC4_128 Encryption algorithm used to encrypt the message
- MD5 Hash function for data integrity

TLS_RSA_WITH_RC4_128_SHA uses:

- RSA Key exchange algorithm for the authentication during handshake
- RC4_128 Encryption algorithm used to encrypt the message
- SHA Hash function for data integrity

TLS_RSA_WITH_AES_256_CBC_SHA uses:

- RSA Key exchange algorithm for the authentication during handshake
- AES_256 Encryption algorithm used to encrypt the message
- SHA Hash function for data integrity

The following cipher suites are not supported by modules belonging to Platform Versions ID 10, 13, 16 (2G).

TLS _RSA_WITH_AES_128_CBC_SHA uses:

- RSA Key exchange algorithm for the authentication during handshake
- AES_128 Encryption algorithm used to encrypt the message
- SHA Hash function for data integrity

TLS_RSA_WITH_NULL_SHA uses:

- RSA Key exchange algorithm for the authentication during handshake
- NULL Encryption algorithm used to encrypt the message
- SHA Hash function for data integrity

The #SSLSECCFG command manages the cipher suites and the authentication modes as shown in the following two chapters.

4.2.1. 2G Modules

Here is the #SSLSECCFG command for 2G modules.

AT#SSLSECCFG= <SSId>,<CipherSuite>,<auth_mode>

Where:

<**SSId**> must be set to 1. Only one secure socket is available.

<**CipherSuite**> setting the value to 0, all the available cipher suites are proposed to the remote server, see chapter 8.1.1. It is responsibility of the remote server to select one of them.

Setting value different from zero, the module proposes to the remote server the couple cipher suite/protocol according to the table shown in chapter 8.1.1:

1 = TLS_RSA_WITH_RC4_128_MD5
2 = TLS_RSA_WITH_RC4_128_SHA
3 = TLS_RSA_WITH_AES_256_CBC_SHA
4 = TLS_RSA_WITH_128_CBC_SHA256
5 = TLS_RSA_WITH_AES_256_CBC_SHA256
6 = TLS_RSA_WITH_AES_128_GCM_SHA256

<**auth_mode**>: authentication mode:

0 = SSL verify none: no authentication, no security data is needed.

1 = Server authentication mode: CA Certificate storage is needed, the most common case.

2 = Server/Client authentication mode: CA Certificate (server), Certificate (client) and Private Key (client) are needed.

The authentication mode depends on the user's application and the desired protection against intruders. If the security data is required, they must be stored in PEM format via #SSLSECDATA command, refer to chapter 4.4.

To have more information on the command and cipher suites use, refer to document [1].

If you enable the unique SSL socket, identified by <SSId>=1, on an AT instance through the #SSLEN command, other AT instances cannot use the <SSId>=1 socket. To use the <SSId>=1 socket on another AT instance, you must disable the <SSId>=1 socket (enter #SSLEN=1,0 on the AT instance used to enable <SSId>=1), and activate it on the new AT instance. See chapter 4.3.1. To have information on AT instances refer to documents [4].

4.2.2. 3G/4G (Platform ID 20, 23) Modules

Here is the #SSLSECCFG command for 3G and 4G modules.

AT#SSLSECCFG= <SSId>,<CipherSuite>,<auth_mode>[,<cert_format>]

Where:

<**SSId**> Secure Socket ID. Use the AT#SSLSECCFG=? test command to know the <SSId> range of the used module.

<**CipherSuite**> setting the value to 0, all the available cipher suites are proposed to the remote server, see chapter 8.1.2, or 8.1.3 according to the used module. It is responsibility of the remote server to select one of them.

TLS_RSA_WITH_NULL_SHA cipher suite is not included when the <CipherSuite> parameter is set to 0. To select this cipher suite, it is required to set <CipherSuite> = 4.

Setting value different from zero, only one cipher suite is proposed as follows:

1 = TLS_RSA_WITH_RC4_128_MD5
2 = TLS_RSA_WITH_RC4_128_SHA
3 = TLS_RSA_WITH_AES_128_CBC_SHA
4 = TLS_RSA_WITH_NULL_SHA
5 = TLS_RSA_WITH_AES_256_CBC_SHA

<**auth_mode**> authentication mode:

0 = SSL verify none: no authentication, no security data is needed.

1 = Server authentication mode: CA Certificate storage is needed, the most common case.

2 = Server/Client authentication mode: CA Certificate (server), Certificate (client) and Private Key (client) are needed.

The authentication mode depends on the user's application and the desired protection against intruders. If the security data is required, they can be stored in one of the two formats: DER or PEM.

<**cert_format**> optional parameter. It selects the format of the certificate to be stored via #SSLSECDATA command, refer to chapter 4.4.

0 = DER format
1 = PEM format, default

Assume that the module is powered on right now, and the #SSLSECCFG command is entered without <cert_format> parameter. In this case, the default format is PEM.

If you enter the #SSLSECCFG? read command, it does not return the setting of the format to meet retro compatibility with other series.

Now, enter again #SSLSECCFG command with the <cert_format> parameter for the first time. If the read command is entered, it reports

the parameter value just used. If subsequently the <cert_format> is omitted, the #SSLSECCFG? read command reports the parameter value entered the last time.

Assume to use a module providing a set of SSL sockets. If you enable a SSL socket, identified by <SSId>=x, on an AT instance through the #SSLEN command, other AT instances cannot use the same <SSId>=x socket. To use the <SSId>=x socket on another AT instance, you must disable the <SSId>=x socket (enter #SSLEN=x,0 on the AT instance used to enable <SSId>=x), and activate it on the new AT instance. Different SSL sockets can be enabled on different AT instances. See chapter 4.3.2. To have information on AT instances refer to documents [5], [10], and [12] according to the used module.

4.3. Examples

4.3.1. #SSLEN in Modules Providing one SSL Socket

This example shows the behavior of the #SSLEN command in a module providing only one SSL socket (identified by <SSId>=1). Two terminal emulators are connected to the module. In this example, the first one is connected to USIF0/COM1, the second one is connected to USB0/COM4.

Use COM1, instance 1 (parser AT0)

AT+CGMM
HE910
OK

AT#PORTCFG?
#PORTCFG: 0,0
OK

The module provides only one SSL socket.
AT#SSLCFG=?
#SSLCFG: (1),(1),(0-1500),(0-65535),(10-5000),(0-255),(0-2),(0-2),(0),(0)
OK

AT+CMEE=2
OK

AT#SSLEN?
#SSLEN: 1,0
OK

AT#SSLEN=1,1
OK

AT#SSLEN?
#SSLEN: 1,1
OK

Connect USB cable, and use COM4, instance 2 (Parser AT1)

AT+CMEE?

+CMEE: 0
OK

AT+CMEE=2
OK

AT#SSLEN?
#SSLEN: 1,1
OK

AT#SSLEN=1,0
+CME ERROR: Resource used by another instance

Use COM1

AT#SSLEN=1,0
OK

AT#SSLEN?
#SSLEN: 1,0
OK

Use COM4

AT#SSLEN?
#SSLEN: 1,0
OK

AT#SSLEN=1,1
OK

Use COM1

AT#SSLEN?
#SSLEN: 1,1
OK

AT#SSLEN=1,0
+CME ERROR: Resource used by another instance.

4.3.2. #SSLEN in Modules Providing several SSL Sockets

This example shows the behavior of the #SSLEN command in a module providing a set of SSL sockets (example: <SSId>=1-6). Two terminal emulators are connected to the module. In this example, the first one is connected to USIF0/COM1 port, the second one is connected to USB0/COM25 port.

Use COM1, instance 1 (parser AT0)

AT+CGMM
LE866-SV1
OK

Check the current #PORTCFG configuration.
AT#PORTCFG?
#PORTCFG: 1,1
OK

The module provides a set of SSL sockets.
AT#SSLCFG=?
#SSLCFG: (1-6),(1-5),(0-1500),(0-65535),(10-5000),(0-255),(0-2),(0-2),(0),(0)
OK

AT+CMEE=2
OK

Assume to start from this SSL sockets configuration.
AT#SSLEN?
#SSLEN: 1,0
#SSLEN: 2,0
#SSLEN: 3,0
#SSLEN: 4,0
#SSLEN: 5,0
#SSLEN: 6,0
OK

AT#SSLEN=1,1
OK

AT#SSLEN?
#SSLEN: 1,1
#SSLEN: 2,0
#SSLEN: 3,0
#SSLEN: 4,0
#SSLEN: 5,0
#SSLEN: 6,0
OK

Connect USB cable, and use COM25, instance 2 (Parser AT1)

AT+CMEE?
+CMEE: 0
OK

AT+CMEE=2

OK

AT#SSLEN?
#SSLEN: 1,1
#SSLEN: 2,0
#SSLEN: 3,0
#SSLEN: 4,0
#SSLEN: 5,0
#SSLEN: 6,0
OK

AT#SSLEN=1,0
+CME ERROR: Resource used by another instance

AT#SSLEN=2,1
OK

Use COM1

AT#SSLEN=1,0
OK

AT#SSLEN?
#SSLEN: 1,0
#SSLEN: 2,1
#SSLEN: 3,0
#SSLEN: 4,0
#SSLEN: 5,0
#SSLEN: 6,0
OK

AT#SSLEN=2,0
+CME ERROR: Resource used by another instance

AT#SSLEN=3,1
OK

AT#SSLEN=4,1
OK

AT#SSLEN?
#SSLEN: 1,0
#SSLEN: 2,1
#SSLEN: 3,1
#SSLEN: 4,1
#SSLEN: 5,0
#SSLEN: 6,0
OK

Use COM25

AT#SSLEN?
#SSLEN: 1,0
#SSLEN: 2,1
#SSLEN: 3,1
#SSLEN: 4,1

#SSLEN: 5,0
#SSLEN: 6,0
OK

AT#SSLEN=1,1
OK

AT#SSLEN?
#SSLEN: 1,1
#SSLEN: 2,1
#SSLEN: 3,1
#SSLEN: 4,1
#SSLEN: 5,0
#SSLEN: 6,0
OK

AT#SSLEN=4,0
+CME ERROR: Resource used by another instance

4.4. Storing Security Data, #SSLSECDATA Command

The following types of security data can be stored in the modules:

- Certificates
- CA Certificates
- Private Key

The maximum size of security data depends on the used module. If a remote server has a certificate larger than the maximum size supported by the module, the authentication fails.

Chapter 4.5 describes a procedure to get the root CA certificate to use in a connection to an HTTPS server. See standards RFC 2459, and X509v3.

Server or Server/Client authentication is fulfilled only if you store the proper security data (certificate(s) and/or private key) in the module's NVM.

Use the following command to store, read, and delate security data.

AT#SSLSECDATA=<storeId>,<Action>,<DataType>[,<Size>]

Where:

<**storeId**> store identifier. Use the AT#SSLSECDATA=? test command to know the <storeId> range provided by the used module.

<**Action**> action identifier. Use the AT#SSLSECDATA=? test command to know the <Actions> range supported by the used module.

0 = delete security data from NVM
1 = store security data in NVM
2 = read security data from NVM
3 = store security data in RAM

<**DataType**> identifies the certificate/key to be stored, read or delete.

0 = Certificate of the client (module). It is needed when the Server/Client authentication mode has been configured.

1 = CA Certificate of the remote server, it is used to authenticate the remote server. It is needed when <auth_mode> parameter of the #SSLSECCFG command is set to 1 or 2.

2 = RSA private key of the client (module). It is needed if the Server/Client authentication mode has been configured.

<**Size**> size of the stored security data. Use the AT#SSLSECDATA=? test command to know the <Size> range provided by the used module.

Assume to store a security data. After entering the #SSLSECDATA command, the '>' prompt appears. There are two security data downloading modes according to the used certificate format set through the AT#SSLSECCFG command, see

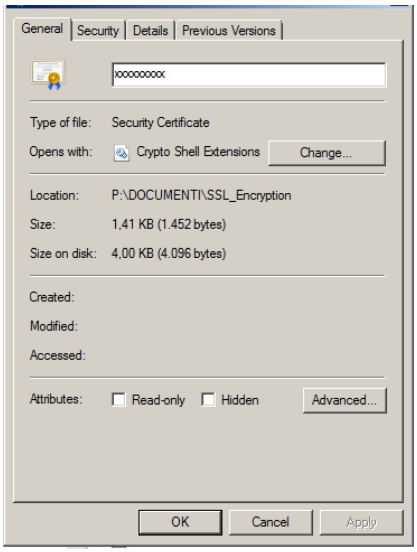
Tab. 2.

2G Modules Series	3G and 4G Modules Series
Certificates can be set only in PEM format	Certificates can be in PEM or DER format

Tab. 2: PEM and DER formats

Here are the downloading modes.

PEM format is supported by 2G/3G/4G.



Before downloading the certificate, you must know the size of the certificate expressed in bytes. Use the Property dialog box, shown on left side, to get this information.

After entering the #SSLSECDATA command, the ">" prompt appears. Now, you can enter the security data to be stored in NVM or RAM. Each certificate line must be terminated only with <LF> character (no <CR>), and no EOF character must be added at the end of the certificate file. Enter <ctrl>Z to close the certificate downloading.

Remember that the reserved chars "backspace" and "escape" are interpreted as control characters and the corresponding action is immediately executed.

DER format is supported by 3G/4G.

Before downloading the certificate, you must know the size of the certificate expressed in bytes. Use the Property dialog box, shown above. When <size> bytes are downloaded, the security data is stored and an OK message is displayed. DER format uses the binary format; therefore, the reserved chars "backspace" and "escape" are not interpreted as control characters, and the binary file includes them inside it. The data security downloading can be done with the Telit AT Controller tool.

4.5. Get the Root CA Certificate

Assume that it is required a connection to an HTTPS server via a module, and the authentication of the remote server is needed. First, you must know the root CA Certificate of the server, and then store it in the NVM of the module. Here is an example to get the root CA certificate.

To get the root CA certificate you can use a browser, running on a PC, connected to the desired HTTPS server.

During the handshake, the server sends a certificate chain, which is a list of certificates. The chain begins with the certificate of the server, and each certificate in the chain is signed by the entity identified by the next certificate in the chain. The server chain could terminate with a root CA certificate, if root CA is not sent by server it must be present locally on the client to solve the chain. The root CA certificate is always signed by the CA itself. The signatures of all certificates in the chain must be verified until the root CA certificate is reached.

Here is an example of solved certificate chain.

ServerCert → AuthorityCert1 → AuthorityCert2 … → AuthorityCertN → RootCACert

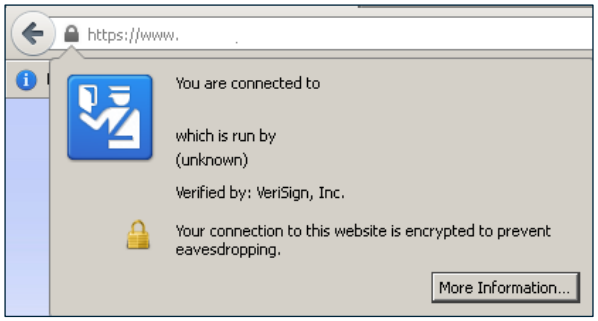
Where:

ServerCert	is the server certificate at which the client wants to be connected
AuhorityCert1…N	are certificates of intermediate authorities
RootCACert	is the certificate of a global recognized Certificate Authority

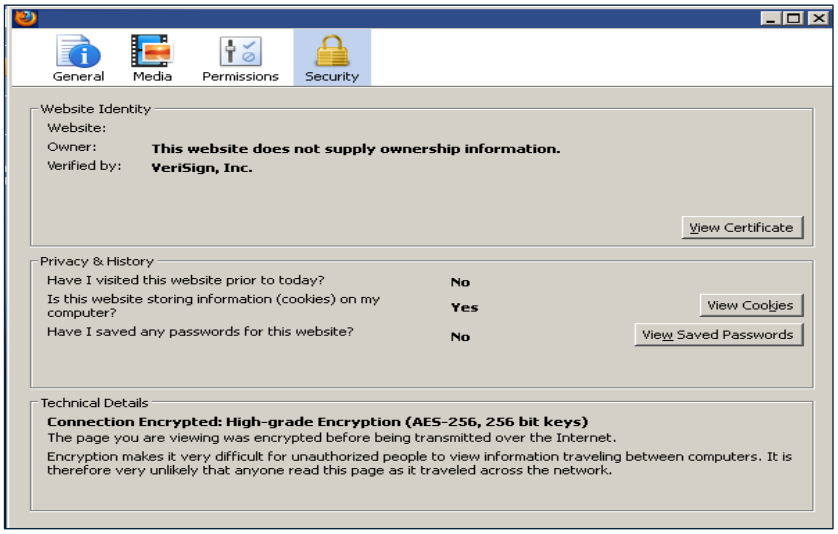
In this example is used the browser Mozilla Firefox.

After being connected to the HTTPS server, click on the lock icon on the left side of the page browser and the following dialog box appears.

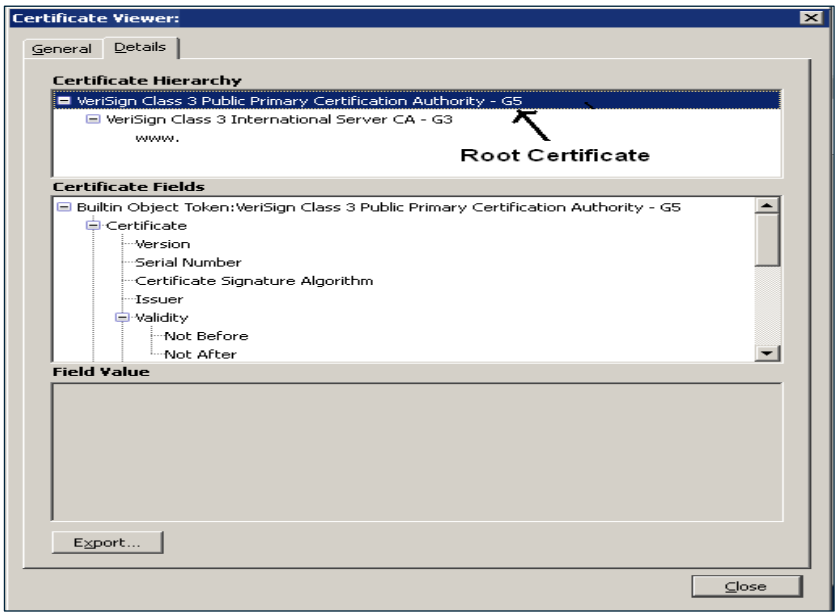
Then click on “More Information” button, the next dialog box appears.



Select the “Security” tab, and click on “View Certificate” button. The following dialog box appears.



Now, select “Details” Tab. The dialog box shows the “Certificate Hierarchy” section that contains the certificate chain for the selected website. The root CA certificate is the first one, select it and click on the “Export” button.



The root CA certificate is saved in a file in PEM format, now open the file via a text editor, the following structure is displayed:

-----BEGIN CERTIFICATE-----

……..

……..

……..

-----END CERTIFICATE-----

The root CA certificate obtained from the procedure may be different from the one sent by the server during the handshake. In this case, contact the server administrator to obtain the root CA certificate to use.

If the root CA certificate is expired, the module (client) detects the certificate expiration when it tries to perform the connection, and an error message is returned.

See example in chapter 6.1.

4.6. SSL Communication Configuration, #SSLCFG Command

Use the following command to configure the SSL socket, before opening it.

AT#SSLCFG=<SSId>,<cid>,<pktSz>,<maxTo>,<defTo>,<txTo>[,<sslSRingMode>[,<noCarrierMode>]]

Where:

<**SSId**> Secure Socket ID. Use the AT#SSLCFG=? test command to know the <SSId> range of the used module.

<**cid**> PDP Context Identifier. Use the AT#SSLCFG=? test command to know the <cid> range of the used module.

<**pktSz**> size of the packet used by the SSL/TCP/IP stack for data sending in ONLINE mode. The packet size can be changed according to the user’s application standard message size. Small <pktSz> values introduce a higher communication overhead.

<**maxTo**> socket inactivity timeout. In ONLINE mode, if there is no data exchange within this timeout period the connection is closed. Increment it if a longer idle time interval is required.

<**defTo**> timeout value used as default value by other SSL commands whenever their timeout parameters are not set.

<**txTo**> time interval after which data is sent even if <pktSz> is not reached (only in ONLINE mode). The parameter value must be tuned with user's application requirements. Small <txTo> values introduce a higher communication overhead.

<**sslSRingMode**> presentation mode of the SSLSRING: unsolicited indication, which informs the user about new incoming data that can be read in COMMAND mode. It can be disabled using value 0.

<**noCarrierMode**> permits to choose between the standard NO CARRIER indication (when the socket is closed) and two verbose modes in which additional information is added to the NO CARRIER indication.

4.7. Examples

The next section describes examples concerning the AT commands introduced in the previous chapters.

4.7.1. #SSLEN Command and SSL Parameters

4.7.1.1. 3G Modules

Before working with SSL parameters, the Secure Socket must be activated through #SSLEN command.

AT+CGMM
HE910
OK

AT+CMEE=2
OK

Use the AT#SSLEN=? test command to know the <SSId> range of the used HE910 module. It provides only one Secure Socket.
AT#SSLEN=?
#SSLEN: (1),(0,1)
OK

Check the status of Secure Socket. It is not activated.
AT#SSLEN?
#SSLEN: 1,0
OK

Check the current SSL/TLS Protocol. Modules belonging to the Platform Versions ID 10, 13, 16 provide only TLS v1.0 protocol, and do not support #SSLSECCFG2 command.
If Secure Socket is not activated, any attempt to work with SSL parameters fails.
AT#SSLSECCFG2?
+CME ERROR: SSL not activated

Check the current SSL Security Configuration.
If Secure Socket is not activated, any attempt to work with SSL parameters fails.
AT#SSLSECCFG?
+CME ERROR: SSL not activated

Check the current SSL Communication Configuration.
If Secure Socket is not activated, any attempt to work with SSL parameters fails.
AT#SSLCFG?
+CME ERROR: SSL not activated

Activate the Secure Socket <SSId>=1.
AT#SSLEN=1,1
OK

Check the current SSL/TLS Protocol.
AT#SSLSECCFG2?
#SSLSECCFG2: 1,1,0,0,0,0
OK

Check the current SSL Security Configuration
AT#SSLSECCFG?
#SSLSECCFG: 1,0,0
OK

Check the current SSL Communication Configuration
AT#SSLCFG?
#SSLCFG: 1,1,300,90,100,50,0,0,0,0
OK

4.7.2. SSL Verify None Mode

4.7.2.1. 2G Modules

Using the following #SSLSECCFG command configuration, the remote server chooses the cipher suite, and the authentication mode is SSL Verify None.

AT#SSLSECCFG=1,0,0
OK

In this case, no security data is required to be stored in NVM, the module is ready for SSL socket dial.

4.7.3. Server Authentication Mode

4.7.3.1. 2G Modules

The following #SSLSECCFG command configuration uses the TLS_RSA_WITH_RC4_128_MD5 cipher suite, and the server authentication mode.

AT#SSLSECCFG=1,1,1
OK

Store the CA certificate of the remote server in PEM format.
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

Now, the module is ready for SSL socket dial.

4.7.3.2. 3G/4G Modules

4.7.3.2.1. DER format.

Using the following #SSLSECCFG command configuration, the remote server chooses the cipher suite, and the Server authentication mode is set. DER format is selected, <cert_format> = 0.

AT#SSLSECCFG=1,0,1,0
OK

Store the CA certificate of the remote server in DER format. When <size> bytes are entered, and the CA Certificate is stored successfully, the OK message is displayed.

AT#SSLSECDATA=1,1,1,<size>
> ……
OK

Now, the module is ready for SSL socket dial.

4.7.3.2.2. PEM format

Using the following #SSLSECCFG command configuration, the remote server chooses the cipher suite, and the Server authentication mode is set. PEM format is selected, <cert_format> = 1.
AT#SSLSECCFG=1,0,1,1
OK

Store the CA certificate of the remote server in PEM format.
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

Now, the module is ready for SSL socket dial.

4.7.4. Server/Client Authentication Mode

4.7.4.1. 2G Modules

Using the following #SSLSECCFG command configuration, the remote server chooses the cipher suite, and the Server/Client authentication mode is set.
AT#SSLSECCFG=1,0,2
OK

Store the certificate of the client (module) in PEM format.
AT#SSLSECDATA=1,1,0,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

Store the CA certificate of the remote server in PEM format.
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

Store the RSA private key of the client (module).
AT#SSLSECDATA=1,1,2,<size>
[… private key …]
<ctrl>Z
OK

5. WORKING WITH SSL SOCKET

This section describes how to open a SSL socket, and exchange data using one of the following modes.

- ONLINE mode
- COMMAND mode

Use the following command to open a SSL socket.

AT#SSLD=<SSId>,<rPort>,<IPAddress>,<ClosureType>[,<connMode>[,<Timeout>]]

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

<**rPort**> remote port of the SSL server (usually 443).

<**IPAddress**> string containing an IP or hostname of the SSL server.

<**ClosureType**> enable/disable the capability to restore later the session, using the #SSLFASTD command, without repeating the handshake phase. See table below, and chapter 5.3.

	<**ClosureType**> parameter	
Platform Version ID	0	1
10, 13, 16 (2G)	SSL session id and keys are released, therefore #SSLFASTD command cannot be used to recover the last SSL session (default).	SSL session id and keys are saved and a new connection can be established without a complete handshake using #SSLFASTD command.
12 (3G)	Zero is the only allowed value, #SSLFASTD command is not supported.	N/A
20 (4G)	Zero is the only allowed value, #SSLFASTD command is not supported.	N/A
23 (4G)	Zero is the only allowed value, #SSLFASTD command is not supported.	N/A

Tab. 3: #SSLFASTD Command Availability

<**connMode**> data exchange mode:

0 = ONLINE mode. On success, the CONNECT message is returned, and from now all bytes sent to the serial port are forwarded to the remote server.

1 = COMMAND mode. On success, the OK message is returned. After that, AT parser is still alive, and data can be exchanged by means of #SSLSEND and #SSLRECV commands.

If for any reason the handshake fails (network or remote server overload, wrong certificate, timeout expiration, etc.) an ERROR response message appears.

<**Timeout**> maximum allowed TCP inter-packet delay. Modules belonging to the Platform Version ID 10 and 16 (2G technology) to manage large

certificates and avoid timeout expiration, must improve the CPU clock by means of the #CPUMODE=2 or 4 command.

5.1. Exchange Data with Secure Socket

5.1.1. ONLINE Mode

Open the SSL socket and wait CONNECT message. After receiving the CONNECT message, you can send data to the module. Data are encrypted and sent to the server through the secure socket as soon as the packet size has been reached or the txTo timeout expires; see chapter 4.6 to configure these parameters.

In ONLINE mode, you cannot enter AT commands on the used serial port or virtual port, refer to documents [4] or [5] to have information about the serial/virtual ports. Anyway, it is possible suspend the connection, without closing it, by sending the escape sequence (+++). After that, the module returns the OK response and can parse again AT commands.

ONLINE mode can be restored at any time by sending the following command.

AT#SSLO=<SSId>

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

After entering the #SSLO restore command, the CONNECT message appears, and SSL communication can continue.

If the idle inactivity timeout expires (<maxTo>, see chapter 4.6) or the remote server closes the connection, the NO CARRIER message is displayed.

5.1.2. COMMAND Mode

In COMMAND mode, data can be exchanged through a SSL socket by means of the #SSLSEND, #SSLSENDEXT and #SSLRECV commands. The data exchange is performed in blocking mode.

If SSLSRING unsolicited message has been enabled by means of the #SSLCFG command (<sslSRingMode> set to 1 or 2), any new incoming data will be notified.

At any moment, the user can switch to ONLINE mode by entering the #SSLO command described in the previous chapter.

5.1.2.1. Send Data, #SSLSEND, #SSLSENDEXT Commands

Use one of the following commands to send data:

AT#SSLSEND=<SSId>[,<Timeout>]

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

<**Timeout**> Timeout expressed in 100 msec unit. If it is omitted, the default timeout set via AT#SSLCFG will be used (<defTo>, refer to chapter 4.6).

When the command is closed with a <CR>, the '>' prompt appears. Now, you can enter the data to be sent. To close the data block, enter <ctrl>Z, then the data are forwarded to the remote server through the secure socket. Response: OK on success, ERROR on failure.

AT#SSLSENDEXT=<SSId>, <bytestosend>[,<Timeout>]

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

<**bytestosend**> Number of bytes to be sent. Use the test command to know the <SSId> range of the used module.

<**Timeout**> Timeout expressed in 100 msec unit. If it is omitted, the default timeout set via AT#SSLCFG will be used (<defTo>, refer to chapter 4.6).

When the command is closed with <CR>, the '>' prompt appears. Now, you can enter the data to be sent. When <bytestosend> bytes have been sent, operation is automatically completed. Response: OK on success, ERROR on failure.

5.1.2.2. Receive Data

Data can be received in two different ways:

- using the #SSLRECV command (the “standard” way),
- reading data from the SSLSRING: unsolicited message.

5.1.2.2.1. #SSLRECV Command

Use the following command to receive data.

AT#SSLRECV=<SSId>,<MaxNumByte>[,<Timeout>]

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

<**MaxNumByte**> Maximum number of bytes that will be read from socket. The user can set it according to the expected amount of data.

<**Timeout**> Timeout expressed in 100 msec unit. If it is omitted, the default timeout set via #SSLCFG will be used (<defTo>, refer to chapter 4.6).

On success, the data are displayed in the following format:

#SSLRECV: <numBytesRead>
… received data ….
OK

Where:

<numBytesRead> number of bytes read (equal or less than <MaxNumBytes>).

If the timeout expires, the module displays the following response

#SSLRECV: 0
TIMEOUT
OK

The ERROR message appears on failure.

5.1.2.2.2. SSLSRING: Unsolicited Message

The SSLSRING: unsolicited message, if enabled, notifies the user about any new incoming data. Configuring <sslSRingMode>=2 by means of the #SSLCFG command (see chapter 4.6) data is displayed in the URC in this format:

SSLSRING:<SSId>,<dataLen>,<data>

Where:

<**SSId**> Secure Socket ID. Use the test command to know the <SSId> range of the used module.

<**dataLen**> Number of bytes presented in the current URC. Its maximum value within a single unsolicited message is:

256 for 2G modules
1300 for 3G/4G modules

<**data**> bytes of data in ASCII format. The number of bytes is <dataLen>.

5.2. Close a Secure Socket, #SSLH Command

The following command closes the SSL socket.

AT#SSLH=<SSId>,<ClosureType>

Where:

<**SSId**>	Secure Socket ID. Use the test command to know the <SSId> range of the used module.
<**ClosureType**>	enable/disable the capability to restore later the session, using the #SSLFASTD command, without repeating the handshake phase. See chapters 5, and 5.3.

If the secure socket has been opened in ONLINE mode, the user needs to send the escape sequence (+++) before closing it with #SSLH command, unless the communication is remotely closed or the idle inactivity timeout expires (NO CARRIER message).

If the secure socket has been opened in COMMAND mode, when communication is remotely closed and all data have been retrieved (#SSLRECV), you can also close on client side and NO CARRIER message is displayed. At any moment, it is also possible to close the secure socket on client side by means of #SSLH.

5.3. Fast Dial, #SSLFASTD Command

#SSLFASTD command restores a previous suspended session avoiding full handshake and performs a fast dial, which saves time and reduces the TCP payload. It can be used if #SSLD or #SSLH command has been entered with <ClosureType> parameter set to 1, in this case the previous data security are not deleted on socket closure. Refer to chapter 5 and 5.2 respectively.

5.4. Examples

The next section describes examples concerning the AT commands introduced in the previous chapters.

5.4.1. ONLINE Mode

Suppose that the PDP context definition/activation, SSL socket enabling, and SSL socket security configuration are performed.

In this example, the secure socket is opened, connected to a SSL server having IP 123.124.125.126, and listening on port 443. After data exchange, the connection is suspended (+++). The #SSLS command is entered to check the SSL status, and then the ONLINE mode is restored using #SSLO command, and so on. At the end, the SSL socket is closed.

AT#SSLD=1,443,"123.124.125.126",0,0 ← open the SSL socket in ONLINE mode
CONNECT
...
[Bidirectional data exchange]
...
+++ ← suspend the connection
OK

AT#SSLS=1 ← query the status of the Secure Socket Id = 1
#SSLS: 1,2,<cipher_suite> ← the connection is open
OK

AT#SSLO=1 ← restore the connection
CONNECT
...
[Bidirectional data exchange]
...
+++ ← suspend again the connection
OK

AT#SSLH=1 ← close SSL socket
OK

AT#SSLS=1 ← query the status of the Secure Socket Id = 1
#SSLS: 1,1 ← the connection is closed
OK

5.4.2. COMMAND Mode

Suppose that the PDP context definition/activation, SSL socket enabling, and SSL socket security configuration are performed.

In this example, the socket is opened, connected to a SSL server having IP 123.124.125.126, and listening on port 443. The data exchange is performed using #SSLSEND, #SSLSENDEXT, and #SSLRECV commands. At the end, the SSL socket is closed.

AT#SSLD=1,443,"123.124.125.126",0,1 ← open the SSL socket in COMMAND mode
OK

AT#SSLS=1 ← query the status of the Secure Socket Id = 1
#SSLS: 1,2,<cipher_suite> ← the connection is open
OK

AT#SSLSEND=1 ← sending data
> Send this string to the SSL server!<ctrl>Z
OK

AT#SSLRECV=1,15 ← receiving data
#SSLRECV: 0
TIMEOUT ← the server has not sent a response within the timeout.

OK

AT#SSLRECV=1,15
#SSLRECV: 15
Response of the ← received data
OK

AT#SSLRECV=1,15
#SSLRECV: 6
Server ← received data
OK

"Response of the server" is the string sent by the server

AT#SSLH=1 ← close SSL socket
OK

If remote server closes data communication at the end of its data sending and no more data are available to be retrieved, communication is also closed on client side. NO CARRIER message is displayed, no #SSLH is needed.

5.4.3. Sending/Receiving Data in COMMAND Mode

Suppose that the PDP context definition/activation, SSL socket enabling, and SSL socket security configuration are performed.

In this example, the socket is opened, connected to a SSL server having IP 123.124.125.126, and listening on port 443. After data exchange in ONLINE mode, the connection is suspended and is entered the COMMAND mode. In this mode, the AT interface is active and by means of the #SSLSEND, #SSLSENDEXT and #SSLRECV commands it is possible to continue to receive and send data using the SSL socket that is still connected. At the end, the SSL socket is closed.

AT#SSLD=1,443,"123.124.125.126",0,0 ← open the SSL socket in ONLINE mode
CONNECT
...
[Bidirectional data exchange]
...
+++ ← suspend the connection and enter COMMAND mode
OK

AT#SSLS=1 ← query the status of the Secure Socket Id = 1
#SSLS: 1,2,<cipher_suite> ← the connection is open
OK

AT#SSLSEND=1 ← AT interface is still active. Send data in COMMAND mode
> Send data in command mode<ctrl>Z
OK

AT#SSLRECV=1,100 ← AT interface is still active. Receive data in COMMAND mode
#SSLRECV: 24
Response in command mode
OK

AT#SSLH=1 ← close SSL socket
OK

If remote server closes data communication at the end of its data sending and no more data are available to be retrieved, communication is also closed on client side. NO CARRIER message is displayed, and then no #SSLH is needed.

5.4.4. COMMAND Mode and SSLSRING: Unsolicited Message

These examples show how to take advantage of the unsolicited SSLSRING: feature. Mode 1 and 2 notify any incoming new record. Mode 2 shows also data, therefore #SSLRECV command is not needed.

5.4.4.1. SSLSRING: Mode = 1

Configure SSLSRING mode 1
AT#SSLCFG=1,1,300,90,100,50,1
OK

AT#SSLD=1,443,"123.124.125.126",0,1 ← open the SSL socket in COMMAND mode
OK

AT#SSLSEND=1 ← send data in COMMAND mode
> Make a request to the server<ctrl>Z
OK

SSLSRING: 1,400 ← 400 bytes are ready to be read

AT#SSLRECV=1,300 ← read only a part of received data
#SSLRECV: 300
<300 bytes>
OK

SSLSRING: 1,100 ← new SSLSRING with remaining data

AT#SSLRECV=1,100 ← read remaining data
#SSLRECV: 100
<100 bytes>
OK

NO CARRIER ← in this example the server closes the connection

5.4.4.2. SSLSRING: Mode = 2

Configure SSLSRING mode 2 plus data
AT#SSLCFG=1,1,300,90,100,50,2
OK

AT#SSLD=1,443,"123.124.125.126",0,1 ← open the SSL socket in COMMAND mode
OK

AT#SSLSEND=1 ← send data in COMMAND mode
> Make the same request of the example 1<ctrl>Z
OK

SSLSRING: 1,256,<256 bytes> ← first chunk of bytes.

SSLSRING: 1,144,<144 bytes> ← second chunk of bytes.
The module has received 400 bytes (256+144)

NO CARRIER ← in this example the server closes the connection

5.4.5. Open/Restore a SSL Socket

Suppose that the PDP context definition/activation, SSL socket enabling, and SSL socket security configuration are performed.

In this example, the socket is opened, connected to a SSL server having IP 123.124.125.126, and listening on port 443; in addition, suppose that the <ClosureType> parameter is set to 1, see chapter 5. Data exchange is performed in ONLINE mode, and then the connection is suspended and restored using #SSLFASTD command. After a new data exchange, the socket is closed definitively.

AT#SSLD=1,443,"123.124.125.126",1,0 ← open the SSL socket in ONLINE mode
CONNECT
...
[Bidirectional data exchange]
...
+++ ← suspend the connection and enter COMMAND mode
OK

AT#SSLH=1 ← close SSL socket

OK

AT#SSLFASTD=1,0 ← restore the session in ONLINE mode
...
[Bidirectional data exchange]
...
+++ ← suspend the connection
OK

AT#SSLH=1,0 ← force definitive closure
OK

6. HTTPS CONNECTION

6.1. #SSLD Command Example

Assume that you have the root CA Certificate, refer to chapter 4.5, and the PDP context definition/activation are performed.

This example shows the configuration of the SSL socket in server authentication mode, the storing of the root CA certificate, the opening of the socket, and the starting of the data exchange. After that, the HTTPS server responds to the module and closes the socket.

If <Enable> parameter is not set to 1, any attempt to set SSL security configuration fails.

Security configuration is valid for SSLD, HTTP, FTP services.

Enable the SSL socket <SSId>=1.
AT#SSLEN=1,1
OK

Set SSL Security Configuration: Secure Socket, CipherSuite, Authentication Mode, Certificate Format.
AT#SSLSECCFG=1,0,1
OK

Store the CA Certificate
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----
……..
Write the certificate got by using the procedure described in chapter 4.5
…….
-----END CERTIFICATE-----
<ctrl>Z
OK

Open the SSL socket identified by <SSId>=1. The connection is opened in ONLINE mode, <connMode>=0. In this example, HTTPS use the <rPort>=443.
AT#SSLD=1,443,"www.---",0,0
CONNECT
……
The module receives a response from the HTTPS server
……
NO CARRIER ← Server remote closure: some servers are configured in order to close the socket after a single request.

6.2. HTTP Get Command Example

In this example, it is assumed to use a 3G module.
To have information on HTTP GET command request refer to RFC 2616 standard.

Define PDP context.
AT+CGDCONT=1,"IP", "Access_Point_Name"
OK

Check the current Multi-sockets/PDP contexts configuration (default).
AT#SCFG?
#SCFG: 1,1,300,90,600,50
#SCFG: 2,1,300,90,600,50
#SCFG: 3,1,300,90,600,50
#SCFG: 4,2,300,90,600,50
#SCFG: 5,2,300,90,600,50
#SCFG: 6,2,300,90,600,50
OK

Before activating a PDP context, it must be bound to a socket. Activate PDP Context <cid>=1. The command returns the IP address assigned by the network.
AT#SGACT=1,1
#SGACT: 10.7.125.7
OK

If <Enable> parameter is not set to 1, any attempt to set SSL security configuration fails.

Security configuration is valid for SSLD, HTTP, FTP services.

Enable the SSL socket <SSId>=1. The SSL
AT#SSLEN=1,1
OK

Set SSL security configuration: Secure Socket, CipherSuite, Authentication Mode, Certificate Format.
AT#SSLSECCFG=1,0,1,1
OK

Store the CA certificate of the remote server in PEM format.
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

SSL encryption can be used only by one service at a time. Therefore, to use the SSL encryption with HTTP protocol, you must disable it for SSLD and FTP services. To do this, set to 0 the following parameters: <Enable> of the #SSLEN command, and <FTPSEn> of the #FTPCFG command.

Disable the SSL encryption for SSLD service: <Enable>=0
AT#SSLEN=1,0
OK

Check the current value of the <FTPSEn> parameter.
AT#FTPCFG?
#FTPCFG: 100,0,0
OK

Enable the SSL encryption, <ssl_enabled>=1, for the HTTP service, and configure the parameters of the HTTPS server.
AT#HTTPCFG=0,"server_address",443,0,,,1,120,1
OK

Send GET command to the HTTP server.
AT#HTTPQRY=0,0,"/"
OK ← GET command succeeds.

When the HTTP server answer is received, an URC is displayed on the terminal emulator.

#HTTPRING: 0,200,"text/html", ⋯

Type in the #HTTPRCV command to read data from HTTP server.
AT#HTTPRCV=0
<!doctype html>
<html>
……..
</html>
OK

7. FTP WITH TLS

FTPS is used when an application needs to connect securely using FTP. FTPS supports:

- authentication
- message integrity
- confidentiality

during a connection over a SSL/TLS secure socket, see standard [8].

The modules support the explicit mode described in standard [8]. In this mode, the FTPS client must explicitly request security from a FTPS server (implicit mode is a deprecated). When FTPS connection is opened towards an FTPS server, FTP command AUTH (refer to standards [8], [9]) is sent to the server to explicitly request a secure FTP connection.

To enable an FTPS connection, use:

- #FTPCFG command to enable FTPS security.
- #SSLSECCFG and #SSLSECDATA commands to configure the SSL socket, see chapters 4.2, and 4.4 respectively.

Use the FTP commands to open control connection and data connection, see document [3]. When #FTPOPEN is used, FTPS connection is opened toward the FTPS server. Any subsequent data port opening (#FTPLIST, #FTPGET, #FTPPUT …) will be in protected mode.

No TLS session reuse is performed when data connection is opened: two TLS sessions are performed within an FTP session, one for control and one for data port. Server shall be configured so that TLS reuse is not required.

The same certificates saved through #SSLSECDATA command are used for both TLS sessions, as strongly recommended by standard [8].

7.1. #FTOPEN, #FTPGET Commands Example

In this example, it is assumed to use a 3G module.

Define PDP context.
AT+CGDCONT=1,"IP", "Access_Point_Name"
OK

Check the current Multi-sockets/PDP contexts configuration (default).
AT#SCFG?
#SCFG: 1,1,300,90,600,50
#SCFG: 2,1,300,90,600,50
#SCFG: 3,1,300,90,600,50
#SCFG: 4,2,300,90,600,50
#SCFG: 5,2,300,90,600,50
#SCFG: 6,2,300,90,600,50
OK

Before activating a PDP context, it must be bound to a socket. Activate PDP Context <cid>=1. The command returns the IP address assigned by the network.
AT#SGACT=1,1
#SGACT: 10.7.125.7
OK

If <Enable> parameter is not set to 1, any attempt to set SSL security configuration fails.

Security configuration is valid for SSLD, HTTP, FTP services.

Enable the SSL socket <SSId>=1.
AT#SSLEN=1,1
OK

Set SSL security configuration: Secure Socket, CipherSuite, Authentication Mode, Certificate Format.
AT#SSLSECCFG=1,0,1,1
OK

Store the CA certificate of the remote server in PEM format.
AT#SSLSECDATA=1,1,1,<size>
> -----BEGIN CERTIFICATE-----<LF>
[…]
-----END CERTIFICATE-----<LF>
<ctrl>Z
OK

SSL encryption can be used only by one service at a time. Therefore, to use the SSL encryption with FTP protocol, you must disable it for SSLD and HTTP services. To do this, set to 0 the following parameters: <Enable> of the #SSLEN command, and <ssl_enabled> of the #HTTPCFG command.

Disable the SSL encryption for SSLD service: <Enable>=0
AT#SSLEN=1,0
OK

Disable the SSL encryption for HTTP service: <ssl_enabled>=0
AT#HTTPCFG=0,"server_address",443,0,,,0,120,1
OK

Enable the SSL encryption for FTP service: <FTPSEn>=1.
AT#FTPCFG=<tout>,<IPPignoring>,1
OK

Enter #FTPOPEN command to send toward the FTPS server the AUTH TLS command to use the explicit TLS mode. When TLS handshake is performed and secure connection is established, the <username> and <password> are sent.

AT#FTPOPEN=<server:port>,<username>,<password>[,<mode>]
OK

Now, FTP control connection is secured through TLS protocol.

Use the #FTPGET command to open a data connection and get the "file.txt" from the FTPS server.
AT#FTPGET=”file.txt”
CONNECT

Now, the data port is connected and the TLS handshake is performed, FTP data connection is secured through TLS protocol and the "file.txt" downloading is started.
…….
…….
…….
NO CARRIER

AT#FTPCLOSE ← close the FTPS connection
OK

8. APPENDIX

8.1. Preinstalled Cipher Suites

Here are the algorithms used by the cipher suites supported by the modules (clients).

8.1.1. 2G Module (Platforms ID 10, 13, 16)

The table below shows the cipher suites supported by the protocols provided by the 2G (Platform ID 10, 13, 16) modules.

	Protocols			
Cipher Suites	**SSL v3**	**TLS v1.0**	**TLS v1.1**	**TLS 1.2**
TLS_RSA_WITH_RC4_128_MD5		●		
TLS_RSA_WITH_RC4_128_SHA		●	●	●
TLS_RSA_WITH_AES_256_CBC_SHA		●	●	●
TLS_RSA_WITH_AES_128_CBC_SHA256				●
TLS_RSA_WITH_AES_256_CBC_SHA256				●
TLS_RSA_WITH_AES_128_GCM_SHA256				●

8.1.2. 3G and 4G (Platform ID 20) Modules

The table below shows the cipher suites supported by the protocols provided by the 3G and 4G (Platform ID 20) modules.

	Protocols			
Cipher Suites	**SSL v3**	**TLS v1.0**	**TLS v1.1**	**TLS 1.2**
TLS_RSA_WITH_RC4_128_MD5	●	●	●	●
TLS_RSA_WITH_RC4_128_SHA	●	●	●	●
TLS_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_RSA_WITH_AES_128_CBC_SHA256				●
TLS_RSA_WITH_AES_256_CBC_SHA256				●
TLS_RSA_WITH_AES_128_GCM_SHA256				●
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256				●
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256				●
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256				●
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256				●
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256				●

8.1.3. 4G (Platform ID 23) Modules

The table below shows the cipher suites supported by the protocols provided by the 4G (Platform ID 23) modules.

	Protocols			
Cipher Suite	**SSL 3.0**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**
TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384				●
TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384				●
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384				●
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384				●
TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_SRP_SHA_DSS_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_SRP_SHA_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_AES_256_GCM_SHA384				●
TLS_DHE_RSA_WITH_AES_256_GCM_SHA384				●
TLS_DHE_RSA_WITH_AES_256_CBC_SHA256				●
TLS_DHE_DSS_WITH_AES_256_CBC_SHA256				●
TLS_DHE_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_DHE_RSA_WITH_CAMELLIA_256_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_CAMELLIA_256_CBC_SHA	●	●	●	●
TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384				●
TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384				●
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384				●
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384				●
TLS_ECDH_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_RSA_WITH_AES_256_GCM_SHA384				●
TLS_RSA_WITH_AES_256_CBC_SHA256				●
TLS_RSA_WITH_AES_256_CBC_SHA	●	●	●	●
TLS_RSA_WITH_CAMELLIA_256_CBC_SHA	●	●	●	●
TLS_ECDHE_RSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_ECDHE_ECDSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_SRP_SHA_DSS_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_SRP_SHA_RSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_DHE_RSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_ECDH_RSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_ECDH_ECDSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_RSA_WITH_3DES_EDE_CBC_SHA	●	●	●	●
TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256				●
TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256				●
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256				●
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256				●
TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_SRP_SHA_DSS_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_SRP_SHA_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_AES_128_GCM_SHA256				●
TLS_DHE_RSA_WITH_AES_128_GCM_SHA256				●
TLS_DHE_RSA_WITH_AES_128_CBC_SHA256				●
TLS_DHE_DSS_WITH_AES_128_CBC_SHA256				●

	Protocols			
Cipher Suite	**SSL 3.0**	**TLS 1.0**	**TLS 1.1**	**TLS 1.2**
TLS_DHE_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_DHE_RSA_WITH_SEED_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_SEED_CBC_SHA	●	●	●	●
TLS_DHE_RSA_WITH_CAMELLIA_128_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_CAMELLIA_128_CBC_SHA	●	●	●	●
TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256				●
TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256				●
TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256				●
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256				●
TLS_ECDH_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_RSA_WITH_AES_128_GCM_SHA256				●
TLS_RSA_WITH_AES_128_CBC_SHA256				●
TLS_RSA_WITH_AES_128_CBC_SHA	●	●	●	●
TLS_RSA_WITH_SEED_CBC_SHA	●	●	●	●
TLS_RSA_WITH_CAMELLIA_128_CBC_SHA	●	●	●	●
TLS_ECDHE_RSA_WITH_RC4_128_SHA	●	●	●	●
TLS_ECDHE_ECDSA_WITH_RC4_128_SHA	●	●	●	●
TLS_ECDH_RSA_WITH_RC4_128_SHA	●	●	●	●
TLS_ECDH_ECDSA_WITH_RC4_128_SHA	●	●	●	●
TLS_RSA_WITH_RC4_128_SHA	●	●	●	●
TLS_RSA_WITH_RC4_128_MD5	●	●	●	●
TLS_DHE_RSA_WITH_DES_CBC_SHA	●	●	●	●
TLS_DHE_DSS_WITH_DES_CBC_SHA	●	●	●	●
TLS_RSA_WITH_DES_CBC_SHA	●	●	●	●
TLS_DHE_RSA_EXPORT_WITH_DES40_CBC_SHA	●	●	●	●
TLS_DHE_DSS_EXPORT_WITH_DES40_CBC_SHA	●	●	●	●
TLS_RSA_EXPORT_WITH_DES40_CBC_SHA	●	●	●	●
TLS_RSA_EXPORT_WITH_RC2_CBC_40_MD5	●	●	●	●
TLS_RSA_EXPORT_WITH_RC4_40_MD5	●	●	●	●
TLS_EMPTY_RENEGOTIATION_INFO_SCSV	●	●	●	●

8.2. SSL Error Codes

Telit's modules provide the AT+CMEE command to enable/disable and select the error report format. The error report can assume two formats: numerical and verbose. The table below summarizes the error reports generated by the SSL AT commands in accordance with the selected format.

Numerical format: AT+CMEE=1	Verbose format: AT+CMEE=2
830	SSL generic error
831	SSL cannot activate
832	SSL socket error
833	SSL not connected
834	SSL already connected
835	SSL already activated
836	SSL not activated
837	SSL certs and keys wrong or not stored
838	SSL error enc/dec data
839	SSL error during handshake
840	SSL disconnected

Tab. 4: SSL Error Code

9. GLOSSARY AND ACRONYMS

	Description
CA	Certification Authority
DER	Distinguished Encoding Rules
FTPS	File Transfer Protocol Secure
GGSN	Gateway GPRS Support Node
GPRS	General Packet Radio Service
HTTPS	Hyper Text Transfer Protocol over Secure Socket Layer
ISP	Internet Service Provider
NVM	Non-Volatile Memory
PDP	Packet Data Protocol
PEM	Privacy Enhanced Mail
PKCS	Public-Key Cryptography Standards
RSA	Stands for the first letter of the names of the algorithm designers
SSL	Secure Socket Layer
SUPL	Secure User Plane Location
TLS	Transport Layer Security
URC	Unsolicited Result Code

10. DOCUMENT HISTORY

Revision	Date	Changes
0	2011-10-11	First issue
1	/	/
2	2012-11-07	Added GE910 module and HE910 family modules. The document has been updated according to the added modules.
3	2012-12-14	Added notes in chapters 3.2, and 3.3.1
4	2013-03-15	Modified figures in chapter 3.3.1. Added note in chapter 3.3. Added explanation for HE910: new values 1 to 4 available of #SSLSECCFG param <cipher_suite>, new value 0 available of #SSLSECCFG param <auth_mode>, Updated Applicability Table: added GL865-DUAL V3, GL868-DUAL V3 and updated software versions.
5	2013-05-02	Update for HE910: client authentication support, FTP over TLS support. Enhancements regarding FTP over TLS for all families.
6	2013-09-13	In the Applicability Table have been added the following products: GE910-GNSS/13.00.xx4, GL865-QUAD V3/16.00xx3, GE910-QUAD V3/16.00.xx3, UE910/12.00.004
7	2013-10-10	Added products in the applicability table: UE910 V2 19.10.x21, HE910 V2 14.20.xx1, HE910 V2 14.10.xx1
8	2014-05-30	Added reference to SSLSRING feature. New sslSRingMode and noCarrierMode config parameters. Removed chunk size limitation.
9	2014-07-11	Update for HE910 regarding additional cmd mode features introduced with CR700: SSLSRING mode 2, noCarrierMode and extended range for minimum timeout of #SSLSEND/RECV

10	2015-04-07	Update for TLS_RSA_WITH_AES_256_CBC_SHA supported by HE910 Update for #SSLSECCFG2 to set TLS version Updated Applicability Table
11	2015-10-28	Updated Applicability Table: CE910-DUAL 18.22.003, CL865-DUAL 18.42.013
12	2017-06-15	The document is fully revised, and chapters are reorganized. A new template is used. Product series removed: GC864, GE864, GT86x, HE920, UE910 V2, DE910, CE910, CL865 Product series added: UL865, LE910 V2, and LE866. In the Applicability Table, has been added the Platform Version Identifier (ID). It is used as reference in the document.
13	2017-10-24	The chapters structure has been reorganized. Added: Product series: LE910 Cat1, and ME866A1; AT Command List; Chapters 6.2 HTTP Get Command Example, and 8.1 Preinstalled Cipher Suite. Updated: Chapter 1.5 Related Documents.
14	2017-11-28	Updated chapter 4.2.1, and the table in chapter 8.1.1.

SUPPORT INQUIRIES

Link to **www.telit.com** and contact our technical support team for any questions related to technical issues.

www.telit.com

Telit Communications S.p.A.
Via Stazione di Prosecco, 5/B
I-34010 Sgonico (Trieste), Italy

Telit IoT Platforms LLC
5300 Broken Sound Blvd, Suite 150
Boca Raton, FL 33487, USA

Telit Wireless Solutions Inc.
3131 RDU Center Drive, Suite 135
Morrisville, NC 27560, USA

Telit Wireless Solutions Co., Ltd.
8th Fl., Shinyoung Securities Bld.
6, Gukjegeumyung-ro8-gil, Yeongdeungpo-gu
Seoul, 150-884, Korea

Telit Wireless Solutions Ltd.
10 Habarzel St.
Tel Aviv 69710, Israel

Telit Wireless Solutions
Technologia e Servicos Ltda
Avenida Paulista, 1776, Room 10.C
01310-921 São Paulo, Brazil

[04.2016]

Mod. 0809 2016-08 Rev.7