TELIUM SDK : SSL Security Guidance

ICO-OPE-00891-V2

www.ingenico.com
28/32, boulevard de Grenelle, 75015 Paris - France / (T) +33 (0)1 58 01 80 00 / (F) +33 (0)1 58 01 91 35
Ingenico – S.A. au capital de 53 086 309 € / 317 218 758 RCS PARIS

Contents

1 Document Information

1_1 Evolution follow-up

Revision	Type of modification	Author	Date
V1	Document Creation from ICO-PE-046-GU-EN	Vincent GOMES	July 2013
V2	Update document format. Update on SSL weaknesses	Antoine WHAAP	15/07/2015

1_2 Document validity

	Name	Function	Signature	Date
Verified by	Antoine WHAAP	Security Engineer		15/07/2015
Verified by	Vincent GOMES	Software Engineer		15/07/2015
Approved by	Patrice FIVEL	Security Manager		15/07/2015

2 INTRODUCTION

This manual provides security guidance for developers of SSL/TLS solutions. It recommends best practices for keys and certificates management. It describes a framework for the Public Key Infrastructure [PKI].

For the specification of the effective PKI, developers must respect the mandatory requirements described in this document.

Applications must use the SSL/TLS protocol to protect any financial information exchanged through Internet. The use of other security protocols instead of TLS/SSL is prohibited in the scope of the POS Terminal Security Program (PTS Program) and PCI PTS “Open Protocols” module.

Because of the specificity of the platform, the perimeter of these certifications is limited to the use of one of the following security protocol supplied by Ingenico: SSLv3 / TLS1.0 / TLS1.1 / TLS1.2.

If third party developers want to add a new security protocol or use another SSL/TLS library than the one certified by Ingenico, they will have to request for an additional certification. This additional approval must be taken in charge by the third-party developers.

Developers shall also read the document [ICO-OPE-00892-PackIP_SecurityGuidance_UserGuide] which describes best practices for implementing IP enabled applications, according to the requirements of the POS Terminal Security Program (PTS) and the PCI PTS “Open Protocol” module.

The following of this document presents a model of PKI which is compliant with these security requirements.

3 requirements

This section lists the basic requirements for the PKI definition.

- For the terminal, the PKI framework imposes some requirements for the SSL profile definition, mainly to be compliant with the security requirements of the POS Terminal Security (PTS Program) and of the PCI PTS “Open Protocols” module.
- Only the protocol versions SSLv3 or TLS 1.0 or TLS 1.1 or TLS 1.2 must be used to transfer financial information. **SSLv2 must not be used.**
- Only the algorithms: 3DES and AES must be used for encryption of the SSL/TLS session.
- The minimum length of the 3DES or AES keys must be at least 128 bits.
- Authentication method must use RSA or DSS algorithms.
  - The length of public keys must be at least 1024 bits.
  - The length of public keys must be at least 2048 bits for PCI PTSv4 products.
  - It is recommended to have a minimum length of 65537 for the exponent.
- The use of SHA-1 is prohibited for all digital signatures of certificates used to establish an SSL/TLS connection.
- The use of MD5 is prohibited.

> The definitions of profiles which are not compliant with these requirements are out of the scope of the POS Terminal Security Program (PTS Program) and of the PCI PTS “Open Protocols” module.

The mutual authentication is recommended but not mandatory.

The PCI PTS “Open Protocols” module recommends using the SHA-2 hash algorithm for the integrity checking of the exchanged frames. As the SHA-2 algorithm is not supported by the cipher suites of the SSLv3, TLS 1.0 and TLS 1.1 protocols, the TELIUM SDK provides the function “calculate_hash” allowing an application to calculate the SHA-2 digest of the data to be sent.
The SHA-2 algorithm is fully supported by the protocol TLS1.2.
It is recommended to use the upgraded Telium OpenSSL DLL supporting TLSv1.2 for every new application implementation.

Note that SSL protocol is inherently weak and should be removed unless required on an interim basis to facilitate interoperability as part of a migration plan.

4 SSL Configuration application

In the terminal, the SSL configuration is monitored by a dedicated application - named SSL manager [SSL_MGR]. It is in charge of creating and modifying the SSL profiles. It offers also services for the maintenance of the PKI.20

Creation of a simple authentification profile

The creation of a simple authentication profile is made by an SSL script file. The script describes the modifications to update a profile. There are two types of modifications: the creation of a new profile or the modification of an existed profile.

The script indicates first the name of the profile, then the type of modification (CREATION or UPDATE).
Then it contains a list of operations for adding or removing certificates (CA certificates or CRL certificates).

The removal operations concern only the modifications to an existing certificate. A profile creation script only contains operations to add certificates to the new profile.

The SSL script files must be signed according the PED PCI requirements, to allow their treatments. It prevents attackers from entering untrusted certificates in the terminal or from altering an existing profile. To force the signature of these files and their checking by the system, the SSL_MGR application must use the extension .PDF (Parameter Description File). The downloading operation of this type of files locate the files in the FLASH folder “/SYSTEM”.

SSL script files can be loaded locally (using the LLT tool or an USB key) or remotely using downloading servers (TMS, FTP servers…).

The SSL script file does not allow the creation of mutual authentication profiles. However it makes possible the affectation of a client certificate to a profile (cf. section: 6 Creation of mutual authentication profile).

5 SSL Configuration server

The SSL_MGR application can request services of remote servers to get SSL configuration data. These servers are named SSL configuration servers [SC servers].

The communications between SSL_MGR applications and SC servers are encrypted using the SSL protocol. This point imposes that the SSL_MGR applications have previously initialised SSL profiles, used to connect the SC servers.

These profiles can be simple authentication profiles or mutual authentication profiles. They must be compliant to the basic requirements.

The SC servers allow to manage remotely all the SSL configuration. It provides the following services:

- Registration of a terminal in the PKI,
- Creation of new profiles
- Updating of exiting profiles
- Revocation or renewal of CA certificates.

The SC server can allow creating mutual authentication profiles. It interfaces with the certification authorities to sign the certificates generated by the terminal (cf. section: 6 Creation of mutual authentication profile).

6 Creation of mutual authentication profile

A mutual authentication profile differs from a simple authentication profile by the fact that it contains a client certificate.

A client certificate identifies uniquely the terminal and transmits its public RSA key. In order to protect the private key associated to this public key, the RSA key pair of the terminal must be generated by the terminal. The private key is stored securely in the terminal and must never leave it.

The length of a public key is at least 1024 bits (at least 2048 bits for PCI PTSv4 products). To generate the key, the SSL_MGR application should call the function “rsaKeyGen” (please refer to the Telium SDK documentation [Modules>Communication>TCPIP>SSL functions]).

From the generated public key, the terminal generates a certificate. This certificate contains at least the terminal’s serial number. The other fields are not specified in the scope of this document. However the content of this field must be clearly explained the PKI specification document.

Then, from its certificate, the SSL_MGR application generates a certificate signature request (CSR) coded in the PKCS#10 format. This request is sent to the SC server. The SC server validates the request, and it submits it to its certification authority to get its signature. At this time, the SC server can immediately return the signed certificate or just send a response acknowledging the certificate signature to the SSL_MGR application. In this last case, the signed certificate will loaded later in the terminal, using a script file or during a future connection to the SC server.

To generate the CSR, the SSL_MGR application should call the function “rsaCsrGen”.

7 SSL configuration protocol

The PKI defines a dedicated protocol to communicate with the SC server. This section describes the requirements of this protocol. It does not describe its format.

The SSL_MGR connects the SC server using an existing profile, compliant with the basic requirements (described above). It can be a simple authentication profile.

The protocol defines an identification message sent by the terminal. It allows the server to identify the requesting terminal. This message contains mainly two fields: the terminal's serial number and a password.

The password field allows authorizing locally the SSL configuration request. Its goal is to prevent the non-authorized applications or terminals from accessing the SC server's services.

The password can be:

- Entered by an operator using the SSL_MGR application,
- Or a secret information only shared by the SSL_MGR application and the SC server
- Or derived from a signature algorithm

The identification message is optional when the profile used to connect the SC server is a mutual authentication profile. In this case, the SC server authenticates and authorizes the terminal from its client certificate.

The SC server sends an acknowledgment response allowing the terminal to send its requests.

There are two types of requests:

- [FIRST_INITreq] is used to request the registration of a terminal in the PKI. In the case of mutual authentication, the terminal sends its certificate signature request.

- [UPDATEreq] is sent to request the SC server to update the current configuration of the terminal. This request is used to manage the certificates (revocation, renewal…).

8 Other requirements

The TELIUM SDK provides SSL/TLS enabled applications with a very rich set of primitives to enforce the PKI requirements. It is a complete toolkit which allows applications:

- to get information about the session,
- to access information fields of X509 certificates,
- to decode PKCS#12 format
- to monitor the lifetimes of the different keys
- …

[END OF THE DOCUMENT]