ETSI TS 102 569 V7.0.0 (2007-07)

Technical Specification

Smart Cards;
UICC Security Service Module (USSM);
Stage 2
(Release 7)

Reference

DTS/SCP-T0371

Keywords

smart card, security

ETSI

650 Route des Lucioles
F-06921 Sophia Antipolis Cedex - FRANCE

Tel.: +33 4 92 94 42 00 Fax: +33 4 93 65 47 16

Siret N° 348 623 562 00017 - NAF 742 C
Association à but non lucratif enregistrée à la
Sous-Préfecture de Grasse (06) N° 7803/88

Important notice

Individual copies of the present document can be downloaded from:
http://www.etsi.org

The present document may be made available in more than one electronic version or in print. In any case of existing or perceived difference in contents between such versions, the reference version is the Portable Document Format (PDF). In case of dispute, the reference shall be the printing on ETSI printers of the PDF version kept on a specific network drive within ETSI Secretariat.

Users of the present document should be aware that the document may be subject to revision or change of status. Information on the current status of this and other ETSI documents is available at
http://portal.etsi.org/tb/status/status.asp

If you find errors in the present document, please send your comment to one of the following services:
http://portal.etsi.org/chaircor/ETSI_support.asp

Copyright Notification

Contents

Intellectual Property Rights

IPRs essential or potentially essential to the present document may have been declared to ETSI. The information pertaining to these essential IPRs, if any, is publicly available for **ETSI members and non-members**, and can be found in ETSI SR 000 314: *"Intellectual Property Rights (IPRs); Essential, or potentially Essential, IPRs notified to ETSI in respect of ETSI standards"*, which is available from the ETSI Secretariat. Latest updates are available on the ETSI Web server (http://webapp.etsi.org/IPR/home.asp).

Pursuant to the ETSI IPR Policy, no investigation, including IPR searches, has been carried out by ETSI. No guarantee can be given as to the existence of other IPRs not referenced in ETSI SR 000 314 (or the updates on the ETSI Web server) which are, or may be, or may become, essential to the present document.

Foreword

This Technical Specification (TS) has been produced by ETSI Technical Committee Smart Card Platform (SCP).

The present document defines the stage 2 description for the USSM.

The contents of the present document are subject to continuing work within EP SCP and may change following formal EP SCP approval. If EP SCP modifies the contents of the present document, it will then be republished by ETSI with an identifying change of release date and an increase in version number as follows:

Version x.y.z

where:

x the first digit:

0 early working draft;

1 presented to EP SCP for information;

2 presented to EP SCP for approval;

3 or greater indicates EP SCP approved document under change control.

y the second digit is incremented for all changes of substance, i.e. technical enhancements, corrections, updates, etc.

z the third digit is incremented when editorial only changes have been incorporated in the document.

1 Scope

The present document describes the stage two specification of the USSM. The USSM is a generic Security Service Module on a UICC, which can be used by applications on the UICC.

This document defines the architectural framework for using the USSM, the functional services for applications and how to manage the USSM on an UICC. The architecture is based on the concepts of Global Platform Card Specification [1] and the requirements as defined in TS 102 226 [13].

The concept of the USSM is flexible enough to allow additional security objects and operations to be added easily in later versions of the USSM.

2 References

The following documents contain provisions which, through reference in this text, constitute provisions of the present document.

- References are either specific (identified by date of publication and/or edition number or version number) or non-specific.
- For a specific reference, subsequent revisions do not apply.
- For a non-specific reference, the latest version applies. In the case of a reference to a EP SCP document, a non-specific reference implicitly refers to the latest version of that document *in the same Release as the present document*.

Referenced documents which are not found to be publicly available in the expected location might be found at http://docbox.etsi.org/Reference.

[1] Global Platform Card Specification, V.2.2.

[2] NIST, FIPS PUB 197: "Advanced Encryption Standard".

[3] RFC 2104: "Keyed Hashing for Message Authentication".

[4] ISO/IEC 9797-1:1999(E): "Information technology - Security techniques - Message Authentication Codes (MACs)".

[5] "PKCS #1 v2.1: RSA Cryptography Standard", RSA Laboratories.

[6] Void.

[7] FIPS PUB 46-3: Data Encryption Standard (DES).

[8] Sun Microsystems Java Card™ Specification: "Java Card™ 2.2.2 Application Programming Interface".

[9] Sun Microsystems Java Card™ Specification: "Java Card™ 2.2.2 Runtime Environment (JCRE) Specification".

[10] Sun Microsystems Java Card™ Specification: "Java Card™ 2.2.2 Virtual Machine Specification".

NOTE: SUN Java Card Specifications can be downloaded at http://java.sun.com/products/javacard/specs.html.

[11] ETSI TS 101 220: "Smart Cards; ETSI numbering system for telecommunication application providers".

[12] ETSI TS 102 225: "Smart Cards; Secured packet structure for UICC based applications".

[13] ETSI TS 102 226: "Smart Cards; Remote APDU structure for UICC based applications".

[14] ETSI TS 102 266: "Smart Cards; USSM: UICC Security Service Module; Stage 1".

[15] IETF RFC 3174 (2001): "US Secure Hash Algorithm 1 (SHA1)".

[16] IETF RFC 2104 (1997): "HMAC: Keyed-Hashing for Message Authentication".

3 Definitions and abbreviations

3.1 Definitions

For the purposes of the present document, the following terms and definitions apply:

AES: Advanced Encryption Standard, standard cryptographic algorithm [2]

Card issuer: entity that owns the card

DES: Data Encryption Standard, standard cryptographic algorithm [7]

HMAC: Keyed-Hash Message Authentication Code

HMAC-SHA1: Keyed-Hashing for Message Authentication using SHA-1 as Hash function [16]

OPEN: The central on-card administrator that owns the GlobalPlatform Registry [1]

RSA: Algorithm for public key encryption [5]

Sensitive Object: data object containing sensitive and/or protected information like keys, pins or certificates. Most objects on the USSM are sensitive and have to be protected against unauthorized access. The term might also include objects which are not sensitive (e.g. some user certificates might be not sensitive), but are handled by the USSM in the same manner.

SHA-1: Secure Hash standard as defined in [15]

Signature: an encrypted message digest of a document (for encryption asymmetric or symmetric keys can be used)

Triple-DES-Key: 16 Byte or 24 Byte long key (to be used with triple DES algorithm [7])

USSM owner: entity that controls the USSM and has the right to administer its objects. It can be the card issuer, but also an application provider

3.2 Abbreviations

For the purpose of the present document, the following abbreviations apply:

3DES	Triple DES algorithm with 16 or 24 Byte long key
CBC	Cipher Block Chaining
DAP	Data Authorisation Pattern
ECB	Electronic Code Book
OPEN	On-Card administrator that own the registry on a Global Platform Card
OTA	Over the Air
USSM	UICC Security Service Module

4 Introduction

The USSM is a general security module on the UICC, which offers security services to applications on the UICC through an API with standardized functions. The main features of an USSM are:

- storage of sensitive data (e.g. keys), which can be used for security operations.
- access to services that make use of the sensitive data
- access control per service and per group of sensitive data

- functions to manage the USSM.

The concept allows that several applications (provided that their access rights are sufficient) use the same sensitive data of the USSM, e.g. a key can be shared among several applications.

The concept of the USSM allows easy addition of new services.

The USSM stores sensitive data in keysets. The following figure shows an example of a UICC with two USSMs and three applications, which use keys of keysets in the USSMs, e.g. to communicate with various servers.

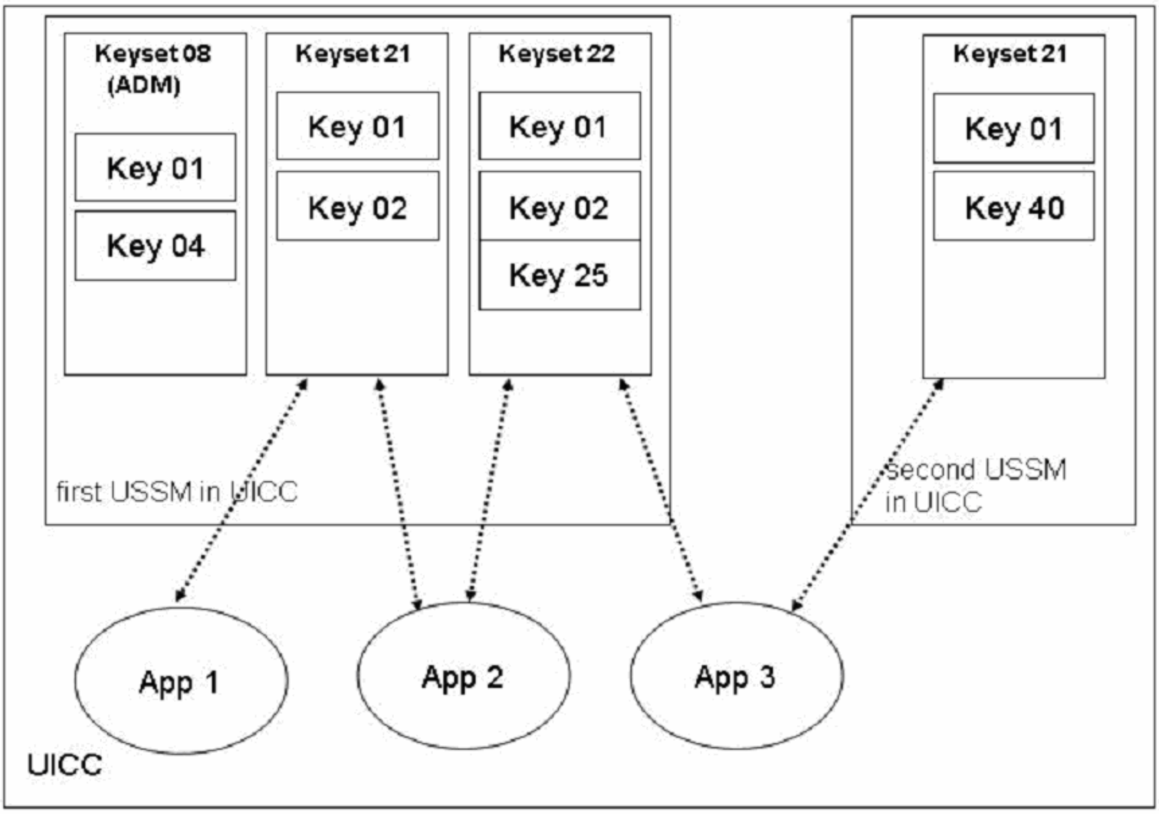

Figure 1: Example of a UICC with two USSMs and some sample Sensitive Objects

The USSM is an application which offers services to other applications on the UICC. A requested service is usually a pair of

- an operation and
- a sensitive data object

An example could be: "decrypt some data with a key"

Not all operations are allowed for a sensitive data object. Restricted access to services is realized by the use of authentication patterns, which are provided by application which requests restricted services. Furthermore the use of a sensitive data object is limited by its KEY_USAGE attribute.

Administration of the USSM is done by the external entities through APDU commands. The definition of the APDUs is out of scope of the present document.

5 Architecture

An UICC embedding a USSM shall support the Global Services Applications mechanism as defined in the Global Platform Card Specification [1].

The USSM is a Service Application in the sense of Global Platform which offers services to other applications on the UICC.

The USSM shall be registered to the OPEN during installation with the Global Platform Privilege: "Global Service" and the service parameter 'A000'.

5.1 Requesting a USSM-service

Applications requesting a service of a USSM shall request this service through the OPEN. The OPEN will return a Global Platform Service Interface of the USSM. The requesting application can then invoke the Service Interface of the USSM and use it.

The requesting application shall provide the following information before a service can be used:

- a USSM_Service_ID;
- a Keyset-ID;
- a DAP (if applicable).

At least the USSM_Service_ID and the Keyset_ID shall be provided when the Service Interface is requested, the authorization by the DAP may be performed later, before the Service Interface is used.

5.2 Storing Sensitive Data

The USSM supports the following Sensitive Objects:

- DES-keys;
- Triple-DES-keys;
- AES-keys;
- RSA-keys.

Sensitive Objects of the USSM are stored in keysets in the sense of Global Platform Card Specification [1]. A place to store a Sensitive Object is defined by a Keyset-ID and a Key-ID. Sensitive Objects are owned and managed by the USSM.

The following ranges of Keyset-IDs are defined:

Table 1: Range of Keyset-IDs

Keyset-ID Range	Purpose
'01' – '1F'	USSM administration
'20' – '5F'	Application usage
'60' – '77'	RFU
'78' – '7F'	Proprietary usage

Keys contained in a Keyset reserved for USSM administration cannot be requested for application usage and keys contained in a Keyset reserved for application usage can not be requested for USSM administration.

5.3 Access Control

Access Control occurs at two places:

- **Before accessing a Sensitive Object:**
  If the requested service for the given Keyset needs authorization, the requesting application has to provide an

authentication pattern (DAP), which is checked by the USSM. The DAP gives authorization for the triple (application, service, keyset).

- **When accessing a Sensitive Object:**
  When the requesting application invokes an operation of the Service interface, no further DAP is provided. All Sensitive Objects of the requested Keyset (except keys with Key-ID '01') can be accessed through the methods offered by the SIO.
  However if a sensitive object in the keyset has a KEY_USAGE and/or KEY_TYPE attribute attached, only operations limited by the KEY_USAGE and KEY_TYPE are possible. For example, a signing key can not be used for a general encryption. If KEY_USAGE or KEY_TYPE do not allow the operation, an exception shall be thrown.
  Furthermore, if a PIN-object is attached to the Sensitive Object, the USSM shall check that the PIN is verified by the requesting application.

5.4 Authorization

Authorization for a service and a given Keyset is needed if a key with Key-ID '01' exists in the Keyset. If there is no key with Key-ID '01' in a Keyset, all services using keys in this Keyset do not require authorization. The key with Key-ID '01' of a Keyset is reserved for verifying the DAP and cannot be used by operations/services offered to the requesting application.

If authorization is required, two methods are offered:

- The DAP is precomputed by the USSM owner and specific for an application, a Keyset, and a Service. It shall be provided by the application to the USSM when requesting a Service.

- The application has knowledge of the secret to build the DAP and can compute a response to a challenge. This can be used for hosts outside the UICC through a proxy application.

If authorization for a Service is required and authorization of the requesting application was not successful, the USSM throws an exception.

5.4.1 Precomputed DAP

The authorization pattern (DAP) allows the USSM to verify that an application is allowed to access the requested service and the requested Keyset.

It is structured as BER-TLV that consists of a list of the requested USSM_Service-IDs followed by a digital signature:

Table 2: Structure of authorization pattern (DAP)

Presence	Length	Name	Tag-Value
Mandatory	1	Tag: USSM Authentication DAP	'60'
	1	Length of USSM Authentication DAP (length of following data)	
Mandatory	1	Tag: List of requested USSM Services	'61'
	1	Length of the list of requested USSM Services	
	2	First USSM_Service_ID	
	2	Second USSM_Service_ID	
...		...	
Mandatory	1	Tag: Signature	'40'
	1	Length of Signature	
	N	Signature	

Calculating the signature:

The digital signature is calculated by first padding and then encrypting the following data structure:

Table 3: Structure used to calculate the signature

Length	Name	Value
1	Tag: AID	'41'
1	Length of AID	
N	AID of requesting application (source)	
1	Tag: AID	'42'
1	Length of AID	
N	AID of USSM (destination)	
1	Tag: List of requested USSM Services	"61"
1	Length of List of requested USSM Services	
N	List of requested USSM Services (see Table 2)	
1	Tag: Keyset-ID	'43'
1	Length	'01'
1	Keyset-ID	

This structure is first padded by '80' and then filled by as few as possible '00' until the total length is a multiple of 8 Bytes. Then the data is encrypted with the first key in the requested Keyset (key with Key-ID '01'). This key shall be of the key type DES, 3DES, or AES. The type of encryption to calculate the signature is dependent on the KEY_TYPE of this key (DES-encryption, 3DES-encryption, or AES-encryption). The encryption-method is CBC mode. The last b bytes of the result is used as the signature, where b is the blocklength of the algorithm.

5.4.2 DAP as a response to a challenge

This method may be used by an application acting as a proxy on behalf of an off-card entity.

The application will be authorized when it proves knowledge of the first key of the requested Keyset. The following procedure is used:

- The application requests a challenge using the Get Challenge Service (and may forward it to the offcard entity),
- the DAP is calculated as described for a Precomputed DAP,
- the DAP is presented using the Authorize Service.

The USSM shall grant access to all Operations / Services with security objects in the requested Keyset after successful verification of the DAP.

6 Using the USSM

6.1 Requesting a Services-Object from the USSM

Applications requesting a Service of an USSM shall provide at least the requested USSM_Service_ID through the Service interface obtained from the OPEN. The following USSM_DATA structure is passed to the USSM:

Table 4: Structure of USSM_DATA

Presence	Length	Name	Value
Mandatory	1	Tag: USSM_Service_ID	'44'
	1	Length of USSM_Service_ID	
	2	USSM_Service_ID	
Optional	1	Tag: DAP	'45'
	1	Length of DAP	
	N	DAP (Authorization Pattern)	

Presence	Length	Name	Value
Conditional	1	Tag: Keyset-ID	'43'
	1	Length of Keyset-ID	
	N	Keyset-ID	

The USSM parses the USSM_DATA and checks if the requested service requires authorization. The USSM can verify the requesting application by retrieving the AID of the requesting application through an operation provided by Global Platform Card Specification [1].

The presence of the Keyset-ID depends on the requested USSM Service.

If no authorization is required or authorization was successful, the USSM returns an object which implements the requested Service.

6.2 Services provided by the USSM

The USSM provides general and specific services. Which services are supported is indicated by the type of the USSM.

The USSM_Service_IDs are divided into the following ranges:

Table 5: Range of Service-IDs

Service-ID range	
'0000' – '5FFF'	Reserved for this specification
'6000' – '9FFF'	RFU for proprietary services
'A000' – 'FFFE'	RFU for standardization bodies outside ETSI
'FFFF'	Reserved

The following Service-IDs are defined in the present document, for details see clause 7.

Table 6: Service-IDs defined in the present document

Service-ID	Service	USSM Service Class
'0001'	Information Service	A
'0002'	Get challenge for authorization	A
'0003'	Authorise	A
'0004'	Verify PIN	A
'0010'	Generic encryption (symmetric key)	A
'0011'	Generic decryption (symmetric key)	A
'0012'	Generic signature (symmetric key)	A
'0013'	Generic verification (symmetric key)	A
'0020'	Generic encryption (RSA key)	B
'0021'	Generic decryption (RSA key)	B
'0022'	Generic signature (RSA key)	B
'0023'	Generic verification (RSA key)	B
'0029'	Read public asymmetric key	B
'0080'	Generate MAC	A
'0081'	Verify MAC	A
'0082'	Encrypt in CBC mode, no padding	A
'0083'	Decrypt in CBC mode, no padding	A
'0084'	Encrypt in CBC mode, with padding	A
'0085'	Decrypt in CBC mode, with padding	A
'0090'	Generate RSA Signature	B
'0091'	Verify RSA Signature	B
'0092'	RSA Encryption	B
'0093'	RSA Decryption	B
'00A0'	Generate HMAC-SHA1 signature	A
'00A1'	Verify HMAC-SHA1 signature	A

7 USSM-services

An object returned by the USSM implements the offered service and provides operations to "use" the sensitive data objects in the requested Keyset.

The basic functionality of Service Class A shall be supported by each USSM. Implementation of other Service Classes is optional.

A USSM may support any service. However, if support for a services class is indicated in the USSM_Type, all Services/Operations of that class shall be supported.

7.1 Generic USSM-services

7.1.1 Information Service

USSM_Service_ID = '0001'

The service returns the USSM_Type and USSM_Version, an indication of the capabilities of the USSM. This service is included in service class A.

The USSM_Type is a value where a bit is set for each supported service class. For example, a USSM which supports Class A and Class B will have the USSM_Type '03'.

Table 7: Coding of USSM Service Class

b8	b7	b6	b5	b4	b3	b2	b1	Value	Supported USSM Service Class
0	0	0	0	0	0	-	1	'01'	USSM Service Class A
0	0	0	0	0	0	1	-	'02'	USSM Service Class B
NOTE: Any other value is RFU.									

Table 8: Coding of USSM Version

USSM_Version	Coding	
1.0	'0100'	

This service does not need authorization.

No Keyset needs to be provided. If a Keyset-ID is provided in USSM_DATA, it is ignored.

Output of the operation:

- USSM_Type;
- USSM_Version.

7.1.2 Get challenge for authorization

USSM_Service_ID = '0002'

The service returns a challenge for authorization. This service is included in service class A.

Output of the operation:

- challenge

7.1.3 Authorize

USSM_Service_ID = '0003'

The service is used to present a DAP which is computed from a challenge. This service is included in service class A.

Input to the operation is:

- DAP

7.1.4 Verify a PIN

USSM_Service_ID = '0004'

The service verifies a PIN that is attached to a Sensitive Object. This service is included in service class A.

Input to the operation is:

- keyid: Key-ID of key in requested Keyset;
- PIN value.

Output of the operation:

- return code;
- retry counter.

7.1.5 Generic Encryption Service with a symmetric key

USSM_Service_ID = '0010'

The service is used to encrypt data in various modes, various padding methods and various algorithms. This service is included in service class A.

- Input to the operation is:
- data to be encrypted;
- mode: ECB or CBC;
- padding method: no padding or padding method 2 according to ISO/IEC 9797-1 [4]
- algorithm: DES or 3DES or AES;
- keyid: Key-ID of key in requested Keyset used for encryption.

Output of the operation:

- return code;
- encrypted data.

7.1.6 Generic Decryption Service with a symmetric key

USSM_Service_ID = '0011'

The service is used to decrypt data in various modes, various padding methods and various algorithms. This service is included in service class A.

Input to the operation is:

- data to be decrypted;
- mode: ECB or CBC;
- padding method: no padding or padding method 2 according to ISO/IEC 9797-1 [4];
- algorithm: DES or 3DES or AES;

- keyid: Key-ID of key in requested Keyset used for decryption.

Output of the operation:

- return code;
- decrypted data.

7.1.7 Generic Signature Service with a symmetric key

USSM_Service_ID = '0012'

The service is used to sign data in various padding methods and various algorithms. This service is included in service class A.

Input to the operation is:

- data to be signed;
- padding method: padding method 2 according to ISO/IEC 9797-1 [4];
- algorithm: DES, 3DES, AES or HMAC-SHA1;
- keyid: Key-ID of key in requested Keyset used for signing.

Output of the operation:

- - return code;
- - signature (for DES, 3DES or AES: last b Bytes of encryption in CBC mode, where b is the blocklength of the algorithm).

7.1.8 Generic Verify Service with a symmetric key

USSM_Service_ID = '0013'

The service is used to verify a signature in various modes, various padding methods and various algorithms. This service is included in service class A.

Input to the operation is:

- data to be verified;
- signature to be verified;
- For DES, 3DES or AES the signature is the last b Bytes of encryption in CBC mode, where b is the blocklength of the algorithm;
- padding method: padding method 2 according to ISO/IEC 9797-1 [4];
- algorithm: DES, 3DES, AES or HMAC-SHA1;
- keyid: Key-ID of key in requested Keyset used for verification.

Output of the operation:

- - return code.

7.1.9 Generic RSA-Encryption Service

USSM_Service_ID = '0020'

The service is used to encrypt data (e.g. a session key) with an RSA key using various padding methods. This service is included in service class B.

Input to the operation is:

- data to be encrypted;
- -adding method;
- keyid: Key-ID of RSA-key in requested Keyset used for encryption.

Output of the operation:

- - return code;
- - encrypted data.

The following padding methods are supported:

- No Padding;
- PKCS1v1.5 Padding;
- RSA_PKCS1_OAEP_Padding.

7.1.10 Generic RSA-Decryption Service

USSM_Service_ID = '0021'

The service is used to decrypt data (e.g. an encrypted session key) with an RSA key using various padding methods. This service is included in service class B.

Input to the operation is:

- data to be decrypted;
- padding method;
- keyid: Key-ID of RSA-key in requested Keyset used for decryption.

Output of the operation:

- return code
- plain data

The following padding methods are supported:

- No Padding;
- PKCS1v1.5 Padding;
- RSA_PKCS1_OAEP_Padding.

7.1.11 Generic RSA-Signature Service

USSM_Service_ID = '0022'

The service is used to sign data (e.g. a message digest) with a RSA key using various signature types. This service is included in service class B.

Input to the operation is:

- data to be signed;
- hash algorithm;

- padding schema;
- keyid: Key-ID of RSA-key in requested Keyset used for signature.

Output of the operation:

- return code;
- signature.

The following padding methods and hash algorithms are supported:

Hash Algorithms:

- SHA-1

Padding Methods:

- PKCS1v1.5 Padding;
- RSA_PKCS1_PSS_Padding.

7.1.12 Generic RSA-Verification Service

USSM_Service_ID = '0023'

The service is used to verify that a signature matches a given message digest, using a RSA key and various signature types. This service is included in service class B.

Input to the operation is:

- message digest;
- signature to verify;
- hash algorithm;
- padding schema;
- keyid: Key-ID of RSA-key in requested Keyset used for signature.

Output of the operation:

- return code.

The following padding methods and hash algorithms are supported:

Hash Algorithms

- SHA-1.

Padding Methods

- PKCS1v1.5 Padding;
- RSA_PKCS1_PSS_Padding.

7.1.13 Service to read a public asymmetric key

USSM_Service_ID = '0029'

This service is used to read a public RSA-key. This service is included in service class B.

Input to the operation is:

- keyid: Key-ID of RSA-key in requested Keyset

Output of the operation:

- return code;
- RSA-public-key.

7.2 Specific USSM-services

7.2.1 MAC-Generation

USSM_Service_ID = '0080'

This service is used to generate a MAC according to ISO/IEC 9797-1 [4] Algorithm 1, padding method 1 and 2, using 3DES or DES or AES. This service is included in service class A.

Input to the operation is:

- data to generate the MAC from;
- keyid: Key-ID of key in requested Keyset used to generate the MAC;
- padding method (1 or 2).

Output of the operation:

- MAC

7.2.2 MAC-Verification

USSM_Service_ID = '0081'

This service is used to verify a MAC according to ISO/IEC 9797-1 [4] Algorithm 1, padding method 1 and 2, using 3DES or DES or AES. This service is included in service class A.

Input to the operation is:

- data to generate the MAC from;
- MAC to verify;
- keyid: Key-ID of key in requested Keyset used to verify the MAC;
- padding method (1 or 2).

Output of the operation:

- return code.

7.2.3 Encryption in CBC mode with no padding

USSM_Service_ID = '0082'

This service is used to encrypt data with the selected key, using 3DES or DES or AES in CBC mode and with no padding. This requires that the input data has a length which is a multiple of the blocklength. This service is included in service class A.

Input to the operation is:

- data to encrypt;
- keyid: Key-ID of key in requested Keyset used for encryption.

Output of the operation:

- return code;
- encrypted data.

7.2.4 Decryption in CBC mode with no padding

USSM_Service_ID = '0083'

This service is used to decrypt data with the selected key, using 3DES or DES or AES in CBC mode and with no padding. This requires that the input data has a length which is a multiple of the blocklength. This service is included in service class A.

Input to the operation is:

- data to decrypt;
- keyid: Key-ID of key in requested Keyset used for decryption.

Output of the operation:

- return code;
- decrypted data.

7.2.5 Encryption in CBC mode with padding

USSM_Service_ID = '0084'

This service is used to encrypt data with the selected key, using 3DES or DES or AES in CBC mode and with padding method 2 according to ISO/IEC 9797-1 [4]. This service is included in service class A.

Input to the operation is:

- data to encrypt;
- keyid: Key-ID of key in requested Keyset used for encryption.

Output of the operation:

- return code;
- encrypted data.

7.2.6 Decryption in CBC mode with padding

USSM_Service_ID = '0085'

This service is used to decrypt data with the selected key, using 3DES or DES or AES in CBC mode and with padding method 2 according to ISO/IEC 9797-1 [4]. This service is included in service class A.

Input to the operation is:

- data to decrypt;
- keyid: Key-ID of key in requested Keyset used for decryption.

Output of the operation:

- return code;
- decrypted data.

7.2.7 Generation of RSA-Signature

USSM_Service_ID = '0090'

This service is used to generate a RSA signature according to RSASSA-PSS-SIGN with SHA-1 hash [5]. This service is included in service class B.

Input to the operation is:

- data to generate the signature from;
- keyid: Key-ID of RSA-key in requested Keyset used to generate the signature.

Output of the operation:

- return code;
- signature.

7.2.8 Verification of RSA-Signature

USSM_Service_ID = '0091'

This service is used to verify a RSA signature according to RSASSA-PSS-VERIFY with SHA-1 hash [5]. This service is included in service class B

Input to the operation is:

- data used to build the signature;
- signature to verify;
- keyid: Key-ID of RSA-key in requested Keyset use to verify the signature.

Output of the operation:

- return code.

7.2.9 RSA-Encryption

USSM_Service_ID = '0092'

This service is used to encrypt some data with a RSA-key according to RSAES-OEAP-ENCRYPT with SHA-1 hash [5]. This service is included in service class B.

Input to the operation is:

- data to encrypt;
- keyid: Key-ID of RSA-key in requested Keyset use for encryption.

Output of the operation:

- return code;
- encrypted data.

7.2.10 RSA-Decryption

USSM_Service_ID = '0093'

This service is used to decrypt some data with a RSA-key according to RSAES-OEAP_5-DECRYPT with SHA-1 hash [5]. This service is included in service class B.

Input to the operation is:

- data to decrypt;
- keyid: Key-ID of RSA-key in requested Keyset used for decryption.

Output of the operation:

- return code.
- decrypted data.

7.2.11 Generation of a HMAC-SHA1 signature

USSM_Service_ID = '00A0'

This service is used to generate a HMAC-SHA1 signature. This service is included in service class A.

Input to the operation is:

- data to be signed;
- keyid: Key-ID of key in requested Keyset used for signing.

Output of the operation:

- return code;
- signature.

7.2.12 Verification of a HMAC-SHA1 signature

USSM_Service_ID = '00A1'

This service is used to verify a HMAC-SHA1 signature. This service is included in service class A.

Input to the operation is:

- data to be verified;
- signature to check;
- keyid: Key-ID of key in requested Keyset used to create the signature.

Output of the operation:

- return code.

7.3 Results

The following results (success or error) are defined and may be returned by generic or specific services:

- success, ok (meaning e.g. verification of signature successful);
- PIN not verified;
- PIN verification failed;
- Key not found (Key-ID wrong);
- padding schema unknown or not allowed;
- algorithm unknown or not allowed;

- wrong data length.

8 Administration of the USSM by the USSM-owner

Administration of the USSM consists of the following tasks:

- create sensitive data objects;
- update sensitive data objects;
- delete sensitive data objects.

8.1 Administration of Sensitive Objects

Sensitive Objects of USSM are stored in Keysets. Each Sensitive Object can be identified by a Keyset-ID and a Key-ID.

The USSM shall support the PUT KEY command of Global Platform Card Specification [1] to create or update a Sensitive Object in a Keyset.

The USSM shall support the DELETE (key) command of Global Platform Card Specification [1] to delete a Sensitive Object in a Keyset.

The following values for KEY_TYPE as defined in Global Platform Card Specification [1] shall be supported for a DES-Key:

- '80': DES key of any length, where the mode (ECB / CBC) is implicitly known;
- '83': DES key of any length in ECB mode;
- '84': DES key of any length in CBC mode.

Details of the support of RSA Keys is out of scope of the current version of the present document.

Administrating the USSM using OTA security:

A UICC is able to process commands received through OTA and secured according to ETSI TS 102 225 [12].

The USSM shall be a Remote Application as defined in ETSI TS 102 226 [13], i.e. it shall be able to execute command scripts and send back a response. The USSM shall support the PUT KEY and DELETE (key) command within the command scripts. Keys from keysets ['01' - '0F'] shall be used to administer the USSM when using ETSI TS 102 225 [12] security.

Administrating the USSM using other protocols:

Other protocols than OTA security (e.g. for local administration) which use authentication and secure data transfer mechanisms may be used. Additional keysets from '10' to '1F' may be used for these protocols.

8.2 Attaching a local PIN object to a Sensitive Object

Some Sensitive Objects need to be protected by a PIN object. An example can be an authentication key, where a PIN shall be verified before a signature is calculated.

PINs attached to a Sensitive Object in a USSM are local to the USSM.

The PUT KEY command of Global Platform Card Specification [1] is used to attach an existing PIN object (identified by a PIN-ID) to a Sensitive Object. The "Key Access" field in Format 2 of the PUT KEY command shall be used to attach a PIN to the Sensitive Object.

Coding of the "Key Access" field:

Table 9: Key Access field of the PUT KEY command

Byte	Value	Meaning
1	'20'	Proprietary indicating that the following is proprietary
2	'80'	PIN-ID Tag
3	'L1'	Length
4	'XX'	PIN-ID

It is out of scope of the present document how PIN objects are created or managed. The present document specifies only how an existing PIN object, referenced by an ID, is attached to an Sensitive Object.

If a PIN object is attached to a Sensitive Object, the USSM shall verify the PIN before the Sensitive Object can be used through a service. The PIN shall be verified for each application separately.

Some PIN objects may require that the PIN shall be verified for each operation (e.g. PINs attached to a non-repudiation key), other PINs shall be verified only once per requesting application. After power-on of the USSM all PINs are in the state "not verified".

Annex A (normative): USSM API

For a card compliant to Java CardTM [88] [9] and [10], a Java Card^{TM} USSM API is pr.ovided for applications in order to use the USSM services.

The source files for the Java Card USSM API (102569_Annex_A_Java.zip and 102569_Annex_A_HTML.zip) are contained in ts_102569v020000p0.zip, which accompanies the present document.

The export files for the uicc.ussm package (102569_Annex_A_Export_Files.zip) are contained in ts_102569v020000p0.zip, which accompanies the present document.

The following table describes the relationship between each TS 102 569 specification version and its UICC API packages AID and Major, Minor versions defined in the export files.

uicc.ussm package	
AID	AID Major, Minor
A0 00 00 00 09 00 05 FF FF FF FF 89 16 00 00 00	1.0

The package AID coding is defined in TS 101 220 [11]. The USSM API packages' AID are not modified by changes to Major or Minor Version.

The Major Version shall be incremented if a change to the specification introduces byte code incompatibility with the previous version.

The Minor Version shall be incremented if a change to the specification does not introduce byte code incompatibility with the previous version.

Annex B (normative): Table of Tags

The following tags are defined in the present document:

Tag	Value
BER-TLV tag USSM Authentication DAP	'60'
List of requested USSM Services	'61'
Signature	'40'
AID (source)	'41'
AID (USSM)	'42'
Keyset-ID	'43'
USSM_Service_ID	'44'
DAP	'45'

Annex C (informative): Change history

drafting phase		
V.0.0.1	September 2005	First draft, SCPz050403
V.0.0.2	September 2005	revised during Ad Hoc #58, SCPz050405
V.0.0.3	November 2005	Handling of PIN objects added; return codes added
V.0.0.4	January 2006	Cleaning of Java specific terminology
V.1.0.0	January 2006	created at SCP-TEC#7, same as V.0.0.4
V.1.0.1	May 2006	services precised, service classes and informative Annex A added
V.1.0.2	December 2006	Annex A replaced, editorial changes, clarifications
V1.1.0	January 2007	Update to Java Card™ 2.2.2, various editorial changes
V1.2.0	March 2007	Presented to SCP-TEC#12
V2.0.0	April 2007	Presented for approval to SCP Plenary #30
V2.0.1	April 2007	editorial change, presented for approval to SCP Plenary #30; approved
V7.0.0	July 2007	ETSI Publication

History

Document history		
V7.0.0	July 2007	Publication