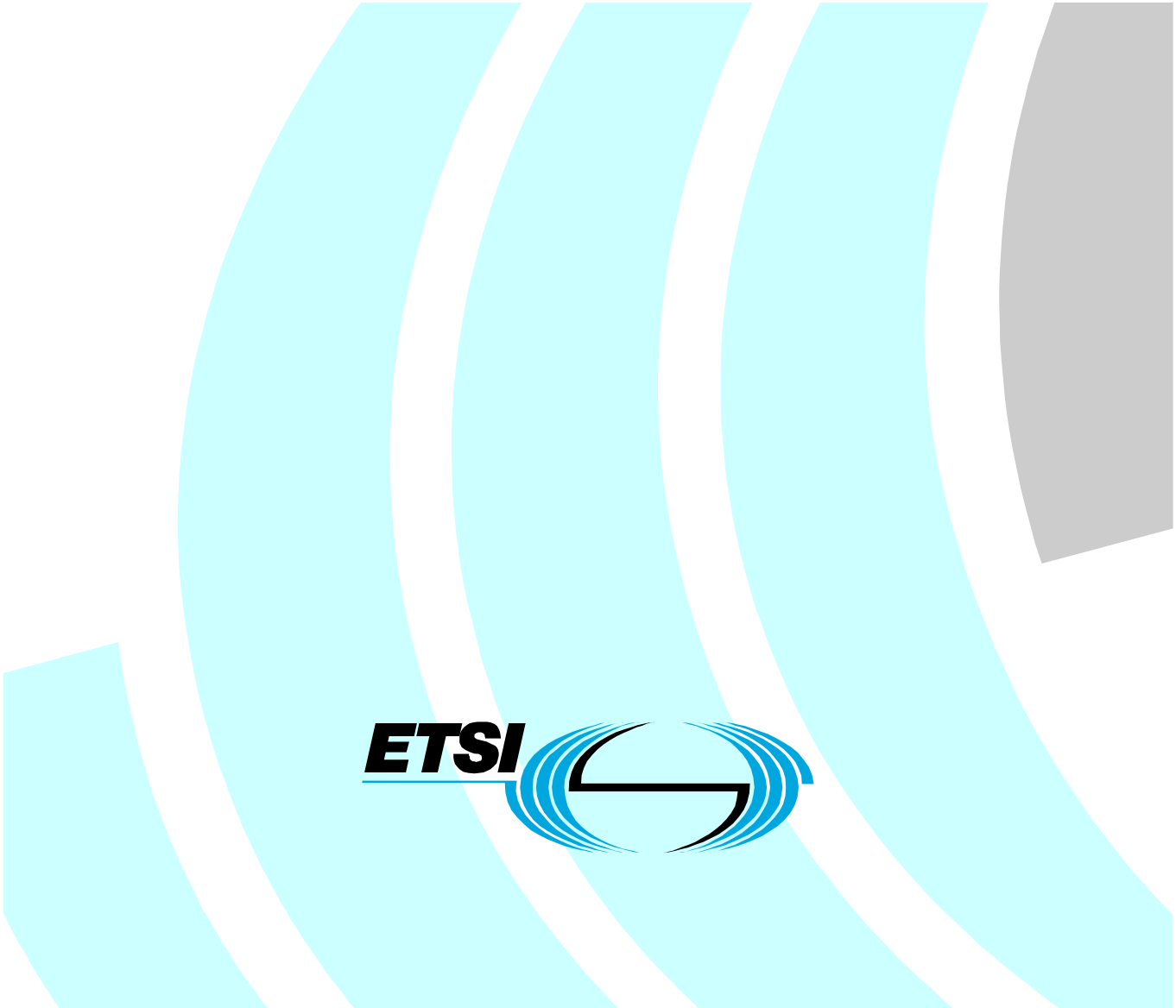
ETSI TS 102 176-2 V1.2.1 (2005-07)

Technical Specification

Electronic Signatures and Infrastructures (ESI); Algorithms and Parameters for Secure Electronic Signatures; Part 2: Secure channel protocols and algorithms for signature creation devices



Reference
RTS/ESI-000039-2

Keywords
e-commerce, electronic signature, 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

Contents

Intellectual Property Rights 4
Foreword 4
Introduction 4
1 Scope 5
2 References 5
3 Definitions and abbreviations 6
3.1 Definitions 6
3.2 Abbreviations 6
4 Maintenance activities 6
5 Secure messaging for smart cards 6
5.1 General 6
5.2 Channel keys establishment 7
5.2.1 Authentication steps 7
5.2.2 Session Key creation 8
5.2.3 Computation of channel keys 9
5.2.4 Computation of the send sequence counter SSC 10
5.3 Secure Messaging Mode 10
5.3.1 CLA byte 10
5.3.2 TLV coding of command and response message 10
5.3.3 Treatment of SM-Errors 10
5.3.4 Padding for checksum calculation 11
5.3.5 Message structure of Secure Messaging APDUs 11
5.3.5.1 Cryptograms 11
5.3.5.2 Cryptographic Checksums 13
Annex A (normative): Use of TDES and AES 15
Annex B (informative): Major changes from previous versions 17
History 18

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 Electronic Signatures and Infrastructures (ESI).

The present document is part 2 of a multi-part deliverable covering the Algorithms and Parameters for Secure Electronic Signatures, as identified below:

Part 1: "Hash functions and asymmetric algorithms";

Part 2: "Secure channel protocols and algorithms for signature creation devices".

Introduction

The present document provides for security and interoperability for the application of the underlying mathematical algorithms and related parameters for electronic signatures in accordance with the Directive 1999/93/EC [1] of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures.

The first part of the present document defines a list of cryptographic algorithms together with the requirements on their parameters, as well as the recommended combinations of algorithms in the form of "signature suites" to be used with the data structures defined in the documents developed under the EESSI (European Electronic Signature Standardization Initiative). The present document contains several informative annexes which provide useful information on a number of subjects mentioned in the text.

The present part of this technical standard (symmetric algorithms and protocols for secure channels) defines a list of symmetric algorithms and protocols to be used with protocols to construct a secure channel between an application and a signature creation device (SCDev) providing either only integrity or both integrity and confidentiality. Such a secure channel may be used during the operational phase of a signature creation device to remotely download a private key in the signature creation device, remotely extract a public key from the signature creation device when the key pair has been generated by the signature creation device or/and remotely download a public key certificate and associate it with a private key already stored in the signature creation device.

With the kind permission of CEN Management Centre, some parts of the present document reproduce text from CEN Workshop Agreement (CWA) (CWA 14890-1 [7]), a publication which is CEN copyright.

Whereas the CWA 14890-1 is restricted to the usage of Triple DES (TDES) only, the present document gives a more general approach for the application of different symmetric algorithms. It recommends the usage of AES, the successor of DES, approved by NIST.

1 Scope

The present document defines a set of symmetric algorithms and protocols to be used to construct a secure channel between an application and a signature creation device providing either only integrity or both integrity and confidentiality. Such a secure channel is required during the operational phase of a signature creation device to remotely download a private key in the signature creation device, remotely extract a public key from the signature creation device when the key pair has been generated by the signature creation device or/and remotely download a public key certificate and associate it with a private key already stored in the signature creation device.

The protocols and algorithms defined in the present document are consistent with the following document:

- CWA 14890-1 [7]: "Application Interface for Smart Cards used as Secure Signature Creation Devices - Part 1: Basic requirements".

The secure channel is always restricted to the both partners of the communication and can be defined even in a proprietary way without loss of interoperability. The present document gives one possibility to set up the secure channel, other methods may be used as well and are not ruled out hereby.

Patent related issues are out of the scope of the present document.

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.

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

[1] Directive 1999/93/EC of the European Parliament and of the Council of 13 December 1999 on a Community framework for electronic signatures.

[2] ISO/IEC 7816-4 (2005): "Identification cards - Integrated circuit cards - Part 4: Organization, security and commands for interchange".

[3] ISO/IEC 9797-1 (1999): "Information technology - Security techniques - Message Authentication Codes (MACs) - Part 1: Mechanisms using a block cipher".

[4] ISO/IEC 11568-2 (1994): "Banking - Key management (retail) - Part 2: Key management techniques for symmetric ciphers".

[5] "The order of encryption and authentication for protecting communications (or: How secure is SSL?)" by Hugo Krawczyk. In Advances in Cryptology - CRYPTO 2001, volume 2139 of Lecture Notes in Computer Science, pages 310-331, Springer-Verlag, 2001.

[6] ANSI X9.63: "Public Key Cryptography for the Financial Services Industry: Key Agreement and Key Transport using Elliptic Curve Cryptography".

[7] CWA 14890-1: "Application Interface for smart cards used as Secure Signature Creation Devices - Part 1: Basic requirements".

[8] FIPS Publication 46-3 (1999): "Data Encryption Standard (DES)", National Bureau of Standards.

[9] FIPS Publication 197 (2001): "Advanced Encryption Standard (AES)", National Institute of Standards and Technology.

3 Definitions and abbreviations

3.1 Definitions

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

host application: application able to establish a secure channel with the SCDev

interface device: device that is the physical interface by which the communication between the card and the host application is handled

NOTE: The communication may be with a contact interface, a contactless interface or both.

3.2 Abbreviations

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

AES	Advanced Encryption Standard
APDU	Application Protocol Data Unit
CLA	CLass byte of an APDU
CWA	CEN Workshop Agreement
DES	Data Encryption Standard
DO	Data Object
FCP	File Control Parameters
HA	Host Application
IFD	InterFace Device
MAC	Message Authentication Code
SAGE	Security Algorithms Group of Experts (from ETSI)
SCDev	Signature-Creation Device
SM	Secure Messaging
TDES	Triple DES

4 Maintenance activities

As a response to relevant developments in the area of cryptography and technology, activities for the maintenance of the symmetric algorithms and protocols for secure channels shall enable dynamic updating of the lists of recommended algorithms and protocols. An initial list of recommended symmetric algorithms and protocols for secure channels is given in the present document.

The present document describes the establishment of two symmetric channel keys using symmetric cryptography only, and does not consider an option for asymmetric cryptography. However, in the future, there can be evolutions towards asymmetric mechanisms for establishing secure channels keys between HA and SCDev.

The maintenance activity is carried by ETSI ESI with the cooperation of the SAGE group. In order to allow an easy follow up of the present document, a history of the changes will be maintained.

5 Secure messaging for smart cards

5.1 General

The secure channel, while being used, is based on symmetric channel keys. There are two channel keys: one for the computation of a Message Authentication Code (MAC) and another one to be used for confidentiality when needed. These channel keys may be preinstalled or dynamically negotiated.

The former case is called "Static SM" where static symmetric channel keys are reserved for secure messaging. In that case the channel keys are always available in the card. A key agreement/derivation method is therefore not required.

In the later case, symmetric channel keys must be established using symmetrical or asymmetric cryptography. The present document does not consider, for the moment, asymmetrical cryptography to establish the negotiated channel keys. However, in the future, there can be evolutions towards asymmetric mechanisms for establishing secure channel keys between HA and SCDev.

When symmetrical cryptography is used to establish the channel keys, these keys are derived after the establishment of a single Session Key K_{SK}. Once the channel keys are established, a trusted channel is then available to protect or conceal the information transmitted over the interface from either side.

5.2 Channel keys establishment

According to ISO/IEC 7816-4 [2] a cryptographic mechanism for confidentiality consists of an algorithm in a mode of operation. In the absence of explicit indication and when no mechanism is implicitly selected for confidentiality, a default mechanism shall apply.

When symmetrical cryptography is used, a single Session Key is established after a successful mutual authentication. The key used for confidentiality K_{ENC} and MAC computation K_{MAC} are derived from the Session Key. They shall be available on HA and SCDev side. The keys K_{ENC} and K_{MAC} used in authentication protocol are replaced as soon as a fresh session key is negotiated by HA and SCDev.

For the HA, the TDES algorithm SHALL be supported while the AES algorithm SHOULD be supported.

For the SCDev, either the TDES algorithm or the AES algorithm SHALL be supported.

NOTE: The AES algorithm is an alternative for future use which currently may not be supported by SCDevs. The current protocol was designed to support a single algorithm (TDES) and does not allow to negotiate the algorithm: the host has to know in advance the single algorithm supported by the SCDev or it extracts this information from elsewhere, for example from the file control parameters (FCP file descriptor extension tag "85") of the file containing the key according to ISO/IEC 7816-4 [2].

The mode of operation SHALL be CBC i.e. cipher-block-chaining.

5.2.1 Authentication steps

The authentication scheme follows the protocol described in CWA 14890-1 [7], section 8.7.1. We use in the following the notation $E[K_{ENC}](data)$ to describe the encryption of "data" using key K_{ENC}. The notation $MAC[K_{MAC}](data)$ describes the computation of a MAC over "data" using key K_{MAC}.

Step	IFD	Transmission	SCDev
1	READ BINARY of file EF.SN.SCDev or GET DATA respectively	→ ←	Read data from specified file SN.SCDev as response
2	GET CHALLENGE	→ ←	RND.SCDev
3	MUTUAL AUTHENTICATE Generate Key K_{HA} S = RND.HA || SN.HA || RND.SCDev || SN.SCDev || K_{HA} $E[K_{ENC}](S)$ || $MAC[K_{MAC}](E[K_{ENC}](S))$	→	SCDev decrypts input and compares RND.SCDev with the previous response. Verify RND.SCDev, SN.SCDev Generate Key K_{SCDev} Generate Session Key K_{SK} (see 5.2.2) Generate SSC.SCDev
3	Verify RND.HA, SN.HA Generate Session Key K_{SK} (see 5.2.2) Generate SSC.HA	←	Return: R = RND.SCDev || SN.SCDev || RND.HA || SN.HA || K_{SCDev} $E[K_{ENC}](R)$ || $MAC[K_{MAC}](E[K_{ENC}](R))$
Both sides authenticated and session key seeds available.			

K_{ENC} is for example a TDES key being used in a DES in CBC mode (see annex A "Use of TDES"). The IV for the CBC-encryption is always set to all zero bytes, e.g. "00000000 00000000" for TDES.

K_{MAC} is for example a TDES key being used according checksum calculation. The initial check block is set to all zero bytes. e.g. "00000000 00000000" and the MAC consists of the first bytes (at least four) from the final output. The length required by CWA 14890-1 [7] is 8 bytes.

No padding is required for the encryption on either side because the data block is constructed to be a multiple of the blocksize (i.e. 8 for TDES).

NOTE: The encryption mechanism requires no padding, since CWA 14890-1 [7] **always** uses the padding indicator for cryptograms set to "01" see table 9.1 and clause 5.3.2. This implies that "the padding consists of one mandatory byte set to "80" followed, if needed, by 0 to k-1 bytes set to"00" until the respective data block is filled up to k bytes (ISO/IEC 7816-4 [2], see table 30 and section 6.2.3.1)". Therefore complete blocks are encrypted whatever the blocksize is.

RND.SCDev and K_{SCDev} are random numbers which are generated by the SCDev where RND.HA and K_{HA} are random numbers which are generated by the HA. The random numbers RND.SCDev and RND.HA are 8 bytes long.

The random numbers K_{SCDev} and K_{HA} are 32 bytes long each and are used to generate the session key K_{SCDev}. SN. HA and SN.SCDev are the 8 least significant bytes of the serial numbers of the HA and the SCDev, respectively.

The structure of the serial numbers used here is out of scope of the present document, it is only important that SN.HA and SN.SCDev are derived from some fixed data available at HA and SCDev and that they occupy 8 bytes each.

5.2.2 Session Key creation

The goal of the authentication procedure is the agreement of channel keys for building cryptograms and cryptographic checksums with the block cipher algorithm. In a first step, the 32-byte values K_{HA} and K_{SCDev} are xor-ed to build the Session Key K_{SK}:

$$K_{SK} = K_{HA} \oplus K_{SCDev}$$

Then the actual channel keys are built from K_{SK} according to clause 5.2.3.

5.2.3 Computation of channel keys

The key derivation protocol for the channel keys is described here using the TDES:

- **K_i (ENC):** describes the TDES key being used to encrypt and decrypt data;
- **K_i (MAC):** describes the TDES key being used to compute and verify a cryptographic checksum;
- **i = a:** describes the first 8 bytes of the TDES key;
- **i = b:** describes the second 8 bytes of the TDES key.

Two 16-byte channel keys are required for secure messaging: one for MAC computation and one for confidentiality protection, if required.

Key derivation from the common secret K_{SK} is performed according to ANSI X9.63 [6]. Let c be a 32 bit counter. Both HA and SCDev compute:

- $HASH_1 = h_{SM}(K_{SK} \| c)$ with c=1; and
- $HASH_2 = h_{SM}(K_{SK} \| c)$ with c=2.

where the hash function h_{SM} is defined here as SHA-1.

NOTE 1: The mixing properties of SHA-1 are not affected be the recently published attacks. Therefore it may be used here without breaches of security. In the future a different hash algorithm may be recommended.

Bytes 1..8 of $HASH_1$ form the key K_a(ENC), and bytes 9..16 build the key K_b(ENC).

Bytes 1..8 of $HASH_2$ form the key K_a(MAC), and bytes 9..16 build the key K_b(MAC).

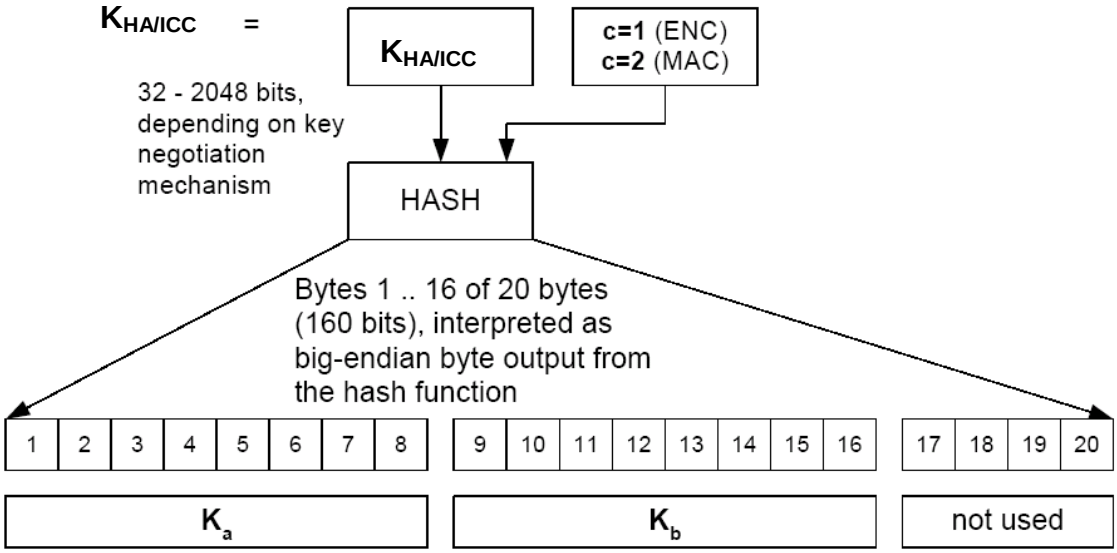


Figure 1: Building TDES-channel keys from hash output (here ICC=SCDev)

The 16-byte channel key used for confidentiality protection is computed using $HASH_1$. The concatenation $K_a \| K_b$ form the 16-byte channel key.

The 16-byte channel key used for MAC computation is computed using $HASH_2$. The concatenation $K_a \| K_b$ form the 16-byte channel key.

NOTE 2: A TDES key is 16 bytes, among which one bit in each byte) is a parity bit. So there are only 112 bits really used by the TDES computation. Parity bits are not considered.

For AES-128 channel keys the derivation proceeds in a similar way. The first 16 bytes of $HASH_1$ build the key for confidentiality and the first 32 bytes of $HASH_2$ || $HASH_3$ build the MAC computation keys K_a and K_b, where $HASH_3$ is computed accordingly as h_{SM} (K_{SK} || c) with c=3.

5.2.4 Computation of the send sequence counter SSC

After successful device authentication, the send sequence counter SSC, which is an 8-bit value, is generated as follows:

- The starting value for the SSC is:

  SSC = RND.SCDev (4 least significant bytes) || RND.HA (4 least significant bytes)

- The RND.SCDev and RND.HA are taken from the values of the device authentication protocol described in clause 5.2.1.

NOTE: The send sequence counter SSC must be increased (+1) each time before a MAC is calculated i.e. if the starting value is x, in the next command the value of SSC is x+1. The SSC value of the first response will then be x+2.

5.3 Secure Messaging Mode

The format of a plain text message is compliant with the definitions in ISO/IEC 7816-4 [2] when it is transmitted using secure messaging.

5.3.1 CLA byte

The presence of Secure Messaging is indicated in b3 and b4 of the CLA byte of the command APDU. According to ISO/IEC 7816-4 [2] clause 6.2.3.1 the bits b3 and b4 are set to 1 indicating that the command header is included in the message authentication.

5.3.2 TLV coding of command and response message

If Secure Messaging is applied the command and response message shall be TLV coded according to ISO/IEC 7816-4 [2].

Tag	Meaning
"81"	Plain value (to be protected by CC)
"87"	Padding-content indicator byte ("01" for ISO-Padding) followed by the cryptogram
"8E"	Cryptographic checksum (MAC)
"97"	Le (to be protected by CC)
"99"	Processing status (SW1-SW2, protected by MAC)

For cryptograms the padding indicator PI is always set to "01", i.e. padding according to ISO/IEC 7816-4 [2] (80 ...00).

NOTE: The plain value SM DOs are always set to Tag "81", because the structure of the data in the data field is irrelevant for the SM view.

The cryptographic checksum shall integrate any secure messaging data object having an odd tag number.

5.3.3 Treatment of SM-Errors

When the SCDev recognizes an SM error while interpreting a command, then the status bytes must be returned without SM. In ISO/IEC 7816-4 [2] the following status bytes are defined to indicate SM errors:

- "6987": Expected SM data objects missing;
- "6988": SM data objects incorrect.

NOTE: Further SM status bytes can occur in application specific contexts.

When the SCDev returns status bytes without SM DOs or with an erroneous SM DO the SCDev deletes the session keys. As a consequence the secure session is aborted.

5.3.4 Padding for checksum calculation

The padding mechanism according to ISO/IEC 7816-4 [2] (80 ...00) is applied.

5.3.5 Message structure of Secure Messaging APDUs

For secure messaging the TDES algorithm or the AES algorithm shall be used.

5.3.5.1 Cryptograms

Cryptograms are built with the symmetric algorithm (TDES or AES) in CBC-Mode with the Null vector as Initial Check Block.

A cryptogram (Tag = 87 x) is always followed by a cryptographic checksum with Tag = 8E x. Encryption must be done first on the data, followed by the computation of the cryptographic checksum on the encrypted data. This order is in accordance with ISO/IEC 7816-4 [2] and has security implications as described in [5].

The command header shall be included into the cryptographic checksum.

The actual value of Lc will be modified to Lc after application of secure messaging. If required, an appropriate data object may optionally be included into the APDU data part in order to convey the original value of Lc.

Figure 2 shows an example how an unprotected command APDU is protected using secure messaging with both integrity and confidentiality. If encryption is not required, the data object "87" is replaced with a plain text data object "81" that conveys the plain data (no padding) in its value field.

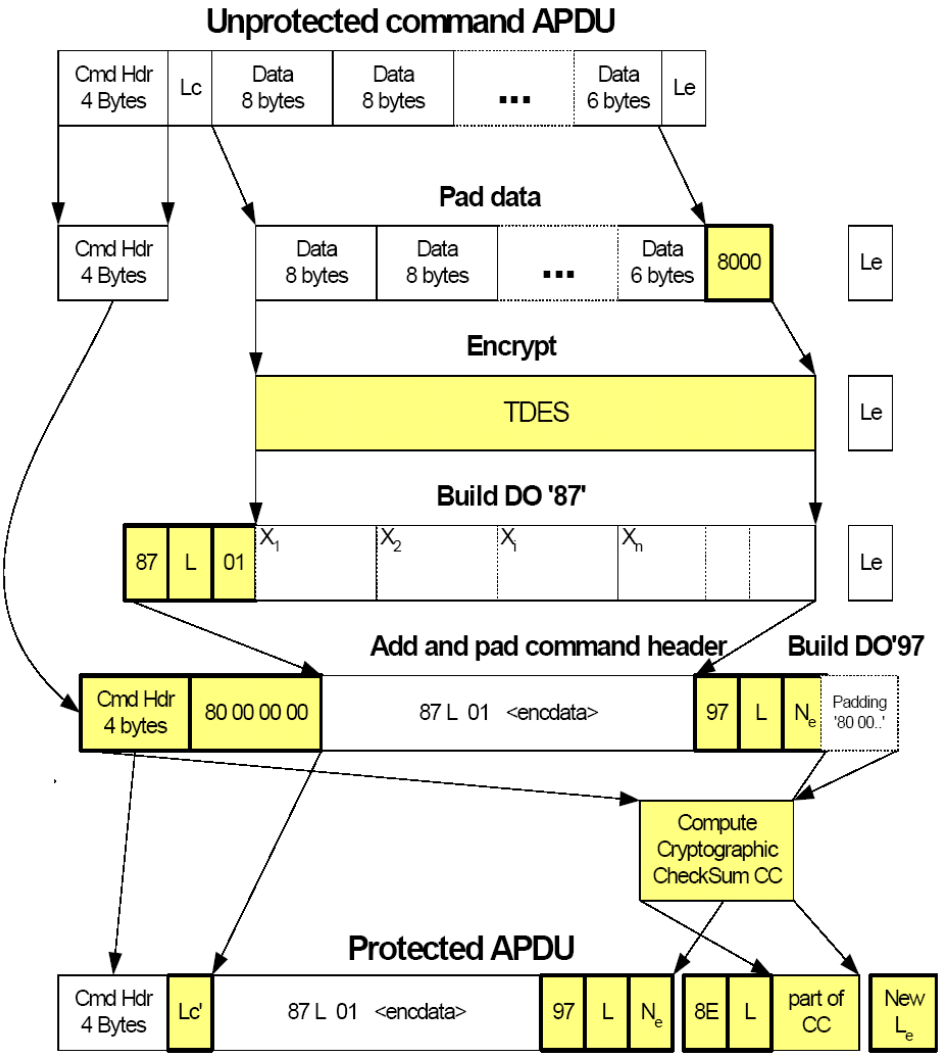


Figure 2: Example for protecting an APDU command using TDES for secure messaging with both integrity and confidentiality

Figure 3 shows an example how an unprotected response APDU is protected using secure messaging with both integrity and confidentiality.

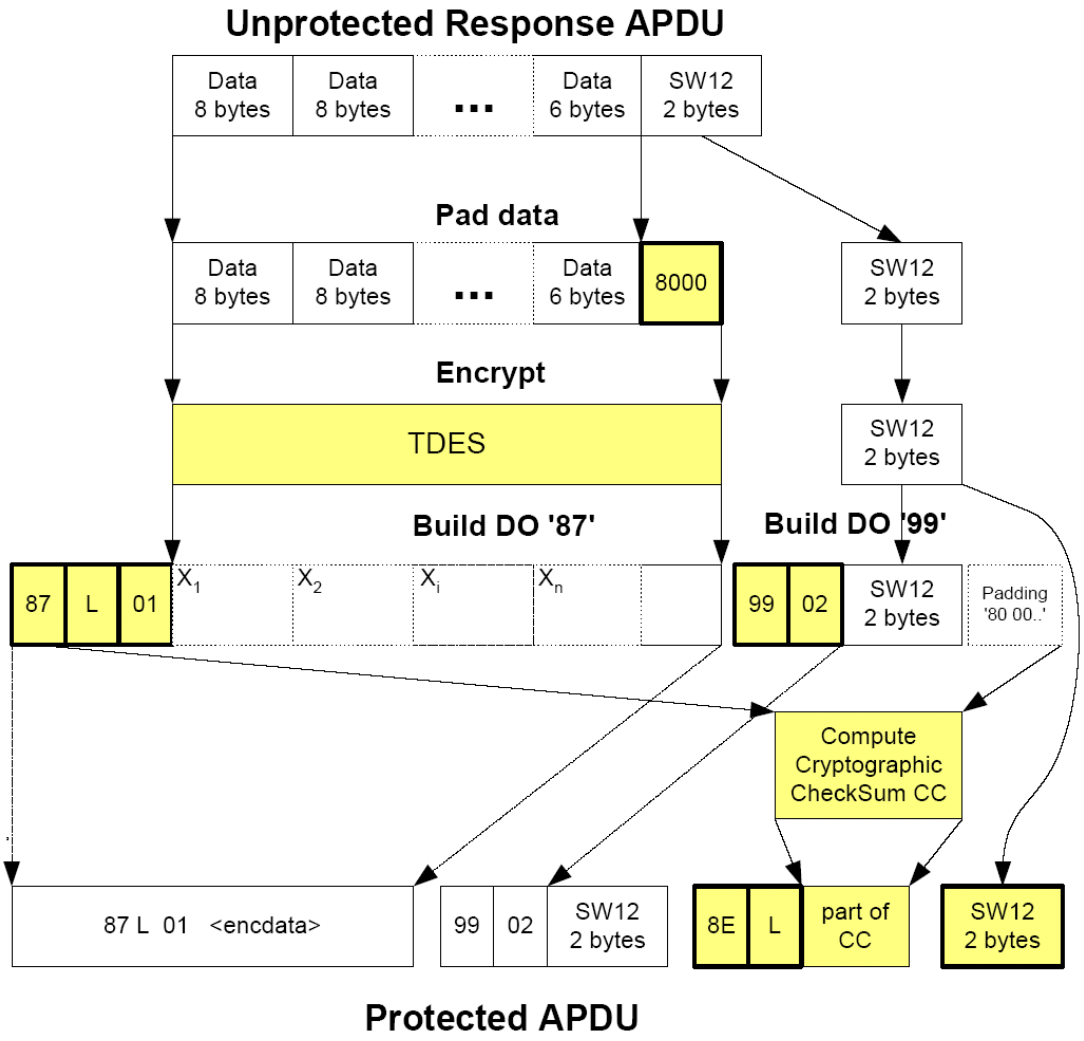


Figure 3: Example for protecting an APDU response using TDES for secure messaging with both integrity and confidentiality

If encryption is not required, the data object "87" is replaced with a plain text data object "81" that conveys the plain data (no padding) in its value field.

NOTE: Some existing applications transmit the DO "99" (secured SW12) only if no data is present in the response. If the DO "99" is not present in the response, the HA shall correctly process using the unprotected SW12 at the end of the response. An attacker, however, cannot remove DO "99" from the response because the verification of the CC (MAC) would fail.

5.3.5.2 Cryptographic Checksums

The keys K_a and K_b are derived from the common freshly generated session key. The data part is split in data blocks with 8-bytes (for TDES) or 16-bytes (for AES) length each. As an example, figures 2 and 3 indicate the 8-byte subblocks with the notation X_i.

In the TDES case, cryptographic checksums are built according to ISO/IEC 7816-4 [2] (clause 6.2.3.1) as follows (the basic mechanism is to build a MAC according ISO/IEC 9797-1 [3] with the block cipher DES, padding method 2, MAC algorithm 3, MAC length of at least four bytes):

- Initial stage: The initial check block Y_0 is E[K_a] (SSC).
- Sequential Stage: The check blocks Y_1, .. , Y_n are calculated using K_a.
- Final Stage: The cryptographic checksum is calculated from the last check block Y_n as follows: E [K_a] (D [K_b] (Y_n)).

Here E [K] ( ) means single encryption with DES and key K, respectively D [K] ( ) decryption with DES. Figure 6 in annex A illustrates this mechanism.

In the case of AES, cryptographic checksums are built according to ISO/IEC 7816-4 [2] (clause 6.2.3.1), using the EMAC construction of ISO/IEC 9797-1 [3] with the block cipher AES:

- Initial stage: The initial check block Y_0 is E [K_a] (SSC).
- Sequential Stage: The check blocks Y_1, .. , Y_n are calculated using K_a.
- Final Stage: The cryptographic checksum is calculated from the last check block Y_n as follows: E [K_b] (E [K_a] (Y_n)).

Here E [K] ( ) means (single) encryption with AES.

Annex A (normative): Use of TDES and AES

Figure A.1 shows the application of keys in TDES (see also ISO/IEC 11568-2 [4]).

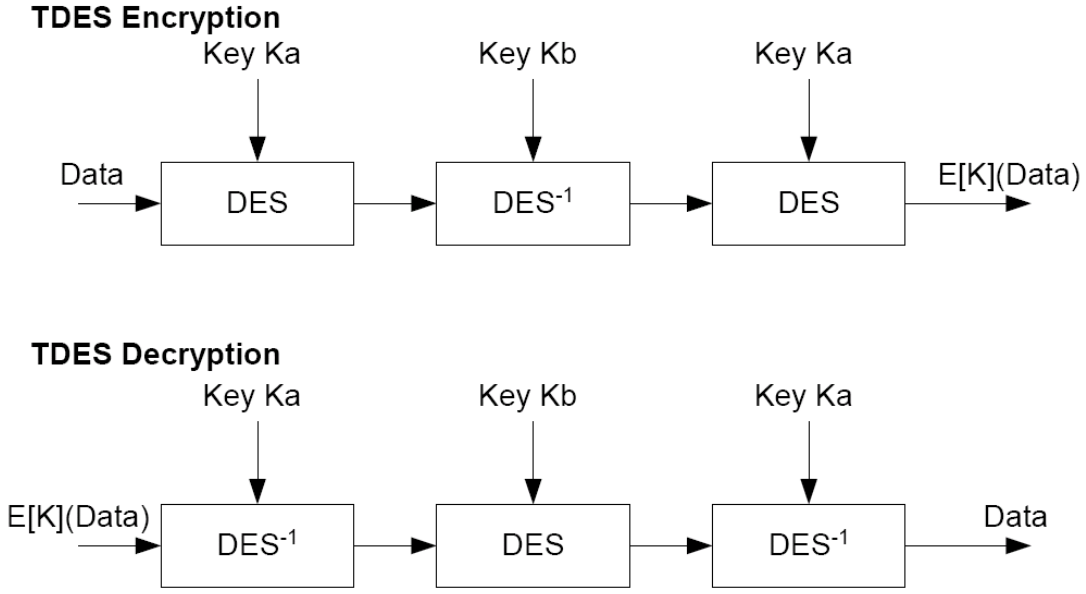


Figure A.1: TDES Encryption/Decryption

For AES the encryption/decryption consists of one block only. The CBC mode of encryption is described in figure A.2.

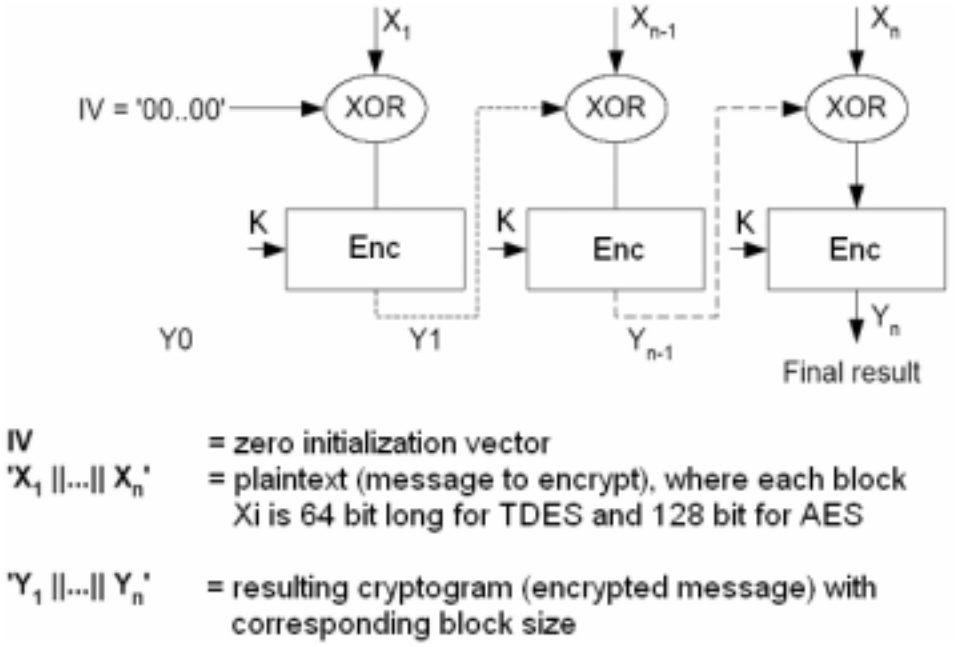


Figure A.2: CBC Encryption/Decryption

The encryption is started with the initial value which is set to a zero vector. The IV is xor-ed with the first plaintext block of the APDU. The result of this encryption is processed accordingly.

The cryptographic checksum (CC) is in the TDES case calculated as retail MAC according to figure A.3.

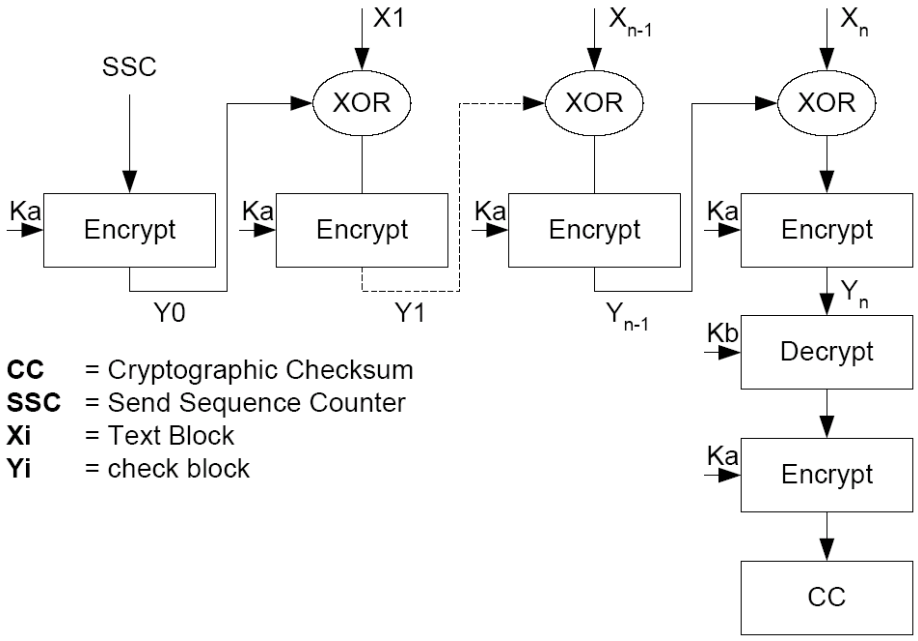


Figure A.3: Retail MAC computation with TDES

The first step performs a single encryption with they key K_a on the send sequence counter. The resulting cryptogram Y_0 is xor-ed with the first plaintext block X_1 from the actual data to be protected. Figures 2 and 3 illustrate how the text blocks X_i are built from the actual APDU data. Then the xor-result is encrypted again with the key K_a.

The second to the last step continue up to the last encryption which results in $Y_{n.}$ The final step is performed on Y_n. K_b is used with decryption, followed by an ecryption with K_a.

If the MAC computation is used with AES, then EMAC computation (MAC algorithm 2 as defined in ISO/IEC 9797-1 [3]) must be used. The last two operations, decryption with K_a and encryption with K_b are replaced by a single encryption with K_b as shown in the figure A.4.

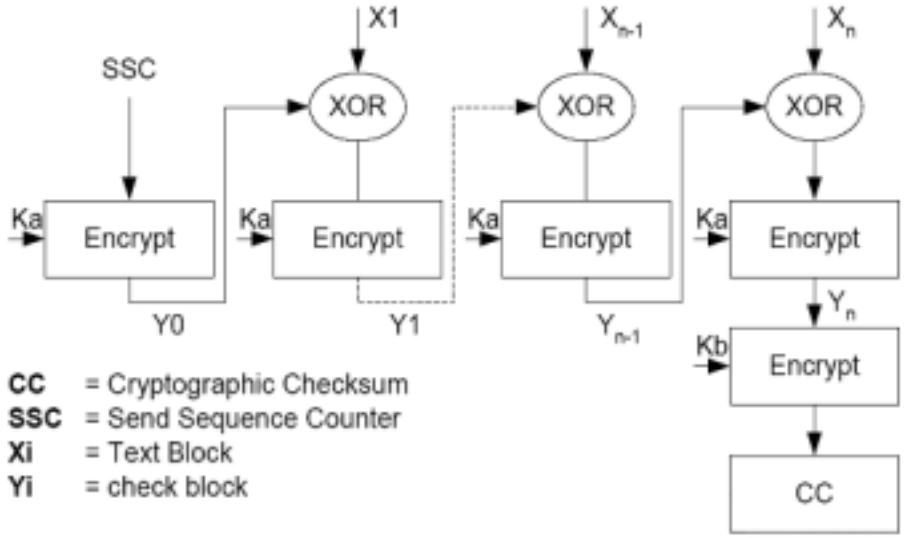


Figure A.4: EMAC computation with AES

Annex B (informative): Major changes from previous versions

This annex is currently empty in the first version of the present document. It will later on contain a description of the major changes between the several versions, so that an history can be easily be done.

History

Document history		
V1.1.1	March 2003	Publication as SR 002 176
V1.2.1	July 2005	Publication