ES 201 209-1 V1.1.1 (1997-07)

ETSI Standard

Identification card systems; Telecommunications IC cards and terminals; Interoperability with synchronous prepaid cards; Part 1: Requirements for off-line and on-line configurations

European Telecommunications Standards Institute

Reference
DES/PTS-00209-1 (b7090icp.PDF)

Keywords
Card, payphone, interoperability, security

ETSI Secretariat

Postal address
F-06921 Sophia Antipolis Cedex - FRANCE

Office address
650 Route des Lucioles - Sophia Antipolis
Valbonne - 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

X.400
c= fr; a=atlas; p=etsi; s=secretariat

Internet
secretariat@etsi.fr
http://www.etsi.fr

Contents

Intellectual Property Rights 6
Foreword 6
1 Scope 7
2 Normative references 7
3 Definitions, symbols and abbreviations 8
3.1 Definitions 8
3.2 Symbols 9
3.3 Abbreviations 10
4 System overview 10
4.1 Introduction 10
4.2 Roles involved 10
4.3 The interface model 12
4.4 Security architecture 13
4.5 The Security Module (SM) life cycle process model 14
4.6 The key management process model 15
4.6.1 Overview 15
4.6.2 The key management implementation model 15
4.6.3 Definitions of the keys 16
4.7 The billing and accounting process model 17
4.8 User Memory Card (UMC) characteristics 18
4.8.1 General characteristics 18
4.8.2 UMC recognition 18
4.8.3 Counter operation 19
4.8.4 UMC counter handling by the terminal 19
4.8.5 UMC counter interpretation by the SM 19
5 Procedures 20
5.1 Introduction 20
5.2 Key handling for interoperability (symmetric key scheme) 20
5.2.1 Key management procedures of the Trusted Third Party (TTP) 20
5.2.2 Encryption of keys in the card management system 22
5.2.3 Handover of keys to the Operators' Management System 23
5.2.4 Key download procedures 23
5.2.4.1 Initial key download 23
5.2.4.2 Updating of keys 25
5.2.5 Adding a new application 26
5.2.6 Removing an application 27
5.3 Normal operation of the terminal and UMC handling 27
5.3.1 Preparatory steps (e.g. identify UMC type) 28
5.3.2 Blacklist 28
5.3.3 Key diversification 29
5.3.4 Card authentication 29
5.3.5 Decrease UMC counter 30
5.3.6 Increase SM counter 32
5.3.7 Extract counter from the SM 33
5.4 Payment and settlement 33
5.5 Operational differences for network based SM 34
6 Data elements 35
6.1 Data description technique 35
6.2 System operator/Card Issuer interface (Interface 4) 35
6.3 Operator's Management System (OMS)/terminal application interface (Interface 3) 38
6.4 Terminal/UMC interface (Interface 2) 40
6.5 Data elements to be held in the UMC 41

6.6 Terminal/SM interface (Interface 1) 42
6.6.1 Preparation and selection 42
6.6.2 UMC authentication 42
6.6.3 Counter increase 43
6.7 Data elements to be held in the SM 43
6.8 SM/system interface (Interface 1 or A) 46
6.8.1 Counter extraction with a MAC 46
6.8.2 Download a key 47

7 Compliance with the terms of this document 48

Annex A (normative): Security requirements 50

A.1 General 50

A.2 Security requirements on the TTP 50

A.3 Basic security requirements 51

Annex B (normative): Key transfer procedures 52

B.1 The procedure for key transfer using a secure device 52

B.2 The procedure for key transfer using public key mechanisms 53
B.2.1 Principles 53
B.2.2 Public key initialization 54
B.2.3 Certificates initialization 54
B.2.3.1 A certificate computation 54
B.2.3.2 B certificate computation 54
B.2.4 Key initialization 55
B.2.4.1 A operations 55
B.2.4.2 B operations 55

Annex C (normative): UMC counter methodologies 57

C.1 Introduction 57

C.2 Size and location of the counter 57
C.2.1 Electrical value 58
C.2.2 Counter orientation 58
C.2.3 Special bits 59
C.2.4 Higher abacus stages 59
C.2.5 Lower abacus stage 59

Annex D (normative): Methods of operation for the counter on the UMC 61
D.1 Introduction 61
D.2 Method A 61
D.3 Method B 62
D.4 Method C 63
D.5 Method D 64
D.6 Method E 65
Annex E (informative): Recommendations on items subject to bilateral agreement between Card Issuers and terminal/system operators 66
E.1 Introduction 66
E.2 Commercial matters 66
E.3 Technical matters 66
E.4 Security matters 67
Annex F (informative): Fixed Number Dialling (FND) 68
Annex G (informative): Additional data elements 70
G.1 System operator/Card Issuer interface 70
Annex H (informative): MAC generation: an example 72
H.1 Introduction 72
H.2 Detailed description of the data 72
H.3 Detailed description of the algorithm 73
Annex J (informative): Bibliography 74
J.1 Introduction 74
J.2 Informative references 74
History 75

Intellectual Property Rights

ETSI has not been informed of the existence of any Intellectual Property Right (IPR) which could be, or could become essential to the present document. However, pursuant to the ETSI Interim IPR Policy, no investigation, including IPR searches, has been carried out. No guarantee can be given as to the existence of any IPRs which are, or may be, or may become, essential to the present document.

Foreword

This ETSI Standard (ES) has been produced by ETSI Project (EP) Pay Terminals and Systems (PTS).

The present document is part 1 of a multi-part document covering Identification card systems; Telecommunications IC cards and terminals; Interoperability with synchronous prepaid cards, as identified below:

Part 1: "Requirements for off-line and on-line configurations";

Part 2: "Security requirements".

1 Scope

The purpose of this document is to securely:

- enable system operators to accept a User Memory Card (UMC) for payment purpose;
- enable system operators to claim money for proven UMC usage from Card Issuers;
- enable system operators as well as UMC and Security Module (SM) issuers to deal with key management issues.

In order to achieve this, the document describes:

- models containing assumptions about roles of involved parties;
- security architecture;
- the minimum functionality required by an application to handle the UMC-SM interfaces;
- methods of recognition of UMCs;
- the minimum data elements required at the SM interface;
- the SM interface required to handle UMC;
- SM interfaces to handle system requirements to provide secure information exchange between system operator and Card Issuer;
- the interface to the management system that enables the secure exchange of relevant information (including accounting and billing) between the system operators and the Card Issuers.

This document is based on the assumption that the SM is an ICC and that the card architecture is based on EN 726-7 [2]. Other implementations of the SM need to fulfil the functional requirements of this document to ensure interoperability of UMCs. The Security Module (SM) may be based on the functions specified in EN 726-7 [2].

Details of these other implementations are outside the scope of this document.

2 Normative references

References may be made to:

a) specific versions of publications (identified by date of publication, edition number, version number, etc.), in which case, subsequent revisions to the referenced document do not apply; or

b) all versions up to and including the identified version (identified by "up to and including" before the version identity); or

c) all versions subsequent to and including the identified version (identified by "onwards" following the version identity); or

d) publications without mention of a specific version, in which case the latest version applies.

A non-specific reference to an ETS shall also be taken to refer to later versions published as an EN with the same number.

[1] EN 726-3 (1994): "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 3: Application independent card requirements".

[2] prEN 726-7: "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 7: Security Module".

[3] ETR 115 (1994): "Terminal Equipment (TE); General concerns for the parties involved during the telecommunication integrated circuit card life-cycle".

[4] ETR 165 (1995): "Human Factors (HF); Recommendation for a tactile identifier on machine readable cards for telecommunications terminals".

[5] ISO 4127: "Codes for the representation of currencies and funds".

[6] ISO/IEC 7816-1 (1987): "Identification cards - Integrated circuit(s) cards with contacts - Part 1: Physical characteristics".

[7] ISO/IEC 7816-2 (1988): "Identification cards - Integrated circuit(s) cards with contacts - Part 2: Contact locations and minimum size".

[8] ISO/IEC 7816-3 (1990) and Amendments 1 & 2: "Identification cards - Integrated circuit(s) cards with contacts - Part 3: Electronic signals and transmission protocols".

[9] ISO/IEC 7816-4 (1995): "Identification cards - Integrated circuit(s) cards with contacts - Part 4: Inter industry commands for interchange".

[10] ISO/IEC 7816-5 (1994): "Identification cards - Integrated circuit(s) cards with contacts - Part 5: Registration system for applications in IC cards".

[11] ISO/IEC 7816-6: "Identification cards - Integrated circuit(s) cards with contacts - Part 6: Inter-industry data elements".

[12] ISO 10202-1: "Financial transaction cards - Security architectures of financial transaction systems using Integrated Circuit Cards - Part 1: Card life cycle".

[13] ISO/IEC CD 11770-3 (1996): "Key Management - Part 3: Mechanisms using asymmetric techniques".

[14] ITU-T Recommendation X.680 | ISO/IEC 8824-1 (1994): "Information technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1): Specification of basic notation".

[15] ITU-T Recommendation X.682 | ISO/IEC 8824-3 (1994): "Information technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1): Constraint specification".

[16] ITU-T Recommendation X.690 | ISO/IEC 8825-1 (1994): "Information technology - Open Systems Interconnection - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)".

[17] ITU-T Recommendation X.691 | ISO/IEC 8825-2 (1994): "Information technology - Open Systems Interconnection - ASN.1 encoding rules: Specification of Packed Encoding Rules (PER)".

[18] ITSEC: "Information Technology Security Evaluation Criteria (ITSEC), Provisional Harmonised Criteria", Version 1.2, June 1991 (ISBN 92-826-3004-8).

[19] ITU-T Recommendation E.164 (1991): "Numbering plan for the ISDN era".

3 Definitions, symbols and abbreviations

3.1 Definitions

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

Card Issuer: The party that is responsible for the data on an UMC (including the secret key), and ultimately receives the payment for units when an UMC is purchased.

card management system: A system belonging to a Card Issuer which is used to maintain records which may include data related to UMCs issued by that party and payment claimed by the system operators. Data such as management information and secure details of revenue taken from UMCs may be exchanged with one, or more, Operators' Management Systems.

key: A sequence of symbols that controls the operations of encipherment and decipherment.

key management: The generation, storage, distribution, deletion, archiving and application of keys in accordance with a security policy.

Operator's Management System (OMS): A system belonging to a system operator which communicates with a number of terminals to distribute management information and collect data, such as secure details of revenue obtained from UMCs.

off-line: A terminal, or a terminal plus connecting unit (including Security Module functionality), can handle an application stand-alone during a transaction. From time to time however, an information exchange will take place with the system.

on-line: A connection to the system is needed during each transaction.

pre-payment: A payment method using an IC card, where the card contains a pre-payment application. The pre-paid value is stored in the card and offers access to one, or more, applications. Pre-paid value means that payment is received in advance.

Secure Device (SD): A device to store the key K_{TTP} pre-loaded by the Trusted Third Party (TTP) and to encrypt loading keys K_{load}. During operation a loading key can also be stored and further keys can be encrypted. With the secure device different key formats can be adopted.

Security Module (SM): A device containing logically and physically protected secrets - algorithm(s), related key(s), security procedures and information to protect applications in such a way that unauthorized access is not feasible. In order to achieve this the module may in addition be further physically, electrically and logically protected (as in EN 726-7 [2]).

system operator: An organization that operates a telecommunications system which accepts payment for access to its services by means of UMCs issued by Card Issuer(s).

telecommunication unit: A telecommunication unit represents a certain amount of service from a specified service provider.

NOTE: A telecommunication unit may represent monetary unit(s) but also a charge pulse of e.g. the telephone network.

terminal operator: An organization that operates terminals which accept payment for access to their services by means of UMCs issued by Card Issuer(s).

terminal: A device which provides a user with access to the telecommunications system of a system operator, accepting payment by means of UMCs issued by a Card Issuer.

Trusted Third Party (TTP): A security authority, or its agent, trusted by other entities with respect to security related activities. In particular, a TTP is trusted for the purposes of key management.

User Memory Card (UMC): The UMC is a memory card with a synchronous protocol and cryptographic security. The user pays the Card Issuer for units which may subsequently be used for access to a service from a terminal or system operator. It is hence a pre-payment card. It has at least mechanisms for:

- one way authentication of the UMCs identity (internal authentication);
- handling of a signed counter value, i.e. the counter value is included in the authentication;
- allowing only a decrease in the value of the counter.

An UMC is called a memory card because it is a type of ICC that contains read/write non-volatile memory and hard wired logic but no microprocessor. ISO 7816 [6 to 11] specifies asynchronous ICCs. Only certain aspects apply to UMCs that use a synchronous protocol.

3.2 Symbols

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

. . .	In ITU-T Recommendation X.680 [14] data descriptions ellipsis indicate that extra data elements could be added in the future at that point in the data type.

3.3 Abbreviations

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

ASN.1	Abstract Syntax Notation number One
BCD	Binary Coded Decimal (a scheme for using 4 bits to encode 1 decimal digit)
CHV	Card Holder Verification
CI	Card Issuer
DF	Dedicated File
EEPROM	Electrically Erasable and Programmable ROM
EF	Elementary File
FND	Fixed Number Dialling
IC	Integrated Circuit
ICC	IC Card
IIN	Issuer Identification Number
ITSEC	Information Technology Security Evaluation Criteria
K_{TTP}	Key TTP
K_{Mttp}	Master Key ttp
MAC	Message Authentication Code
OMS	Operator's Management System
PIN	Personal Identification Number
ROM	Read-Only Memory
SD	Secure Device
SM	Security Module
TLC	Transport Locking Code
ttp	trusted third party
TTP	Trusted Third Party
umc	user memory card
UMC	User Memory Card
Vpp	programming voltage

4 System overview

4.1 Introduction

This document describes the role of various parties and system components. Not all of the requirements in this document are relevant to all parties, or system components.

Conformance testing is outside the scope of this document.

In this clause an overview of the document is given. It describes the scenario of accepting User Memory Cards of different Card Issuers (CI) at terminals of different operators. It starts with a description of the parties involved in the process.

4.2 Roles involved

In the System the following five main roles are involved:

- system operator;
- Card Issuer;
- SM manufacturer;
- user;
- Trusted Third Party (TTP).

In certain cases a single party may undertake several of these roles.

In the following, a short description of each of these roles is given:

System Operator:

The system operator gives service, accepts UMCs as the payment mechanism at a terminal, uses and manages Security Modules and a (background) system connected to the terminals and Security Modules, for the purpose of securely claiming payment from Card Issuers.

These 3 roles of giving service, managing Security Modules, and claiming money from Card Issuers (and other necessary management functions) may be all performed by a single entity or may be split between commercial entities. An entity that performs the first two roles only and relies on some other entity for the (background) system that can securely claim payment from Card Issuers is referred to in this document as a Terminal Operator.

Card Issuer:

The Card Issuer is the party issuing the UMC. The Card Issuer generates and owns the secret keys, which are needed for the operation of the UMCs. The Card Issuer receives the funds, with which the services delivered via the terminal need to be paid. The Card Issuer receives from the system operator the proof that a certain number of units have been decreased at terminals of the system operator from UMCs issued by the Card Issuer.

SM Manufacturer:

The SM manufacturer loads certain parts of the application into the Security Module. The SM manufacturer is responsible for demonstrating that the SM fulfils specified security requirements. This may be achieved via a security evaluation (such as an ITSEC evaluation and/or a reverse engineering evaluation).

User:

The User pays for the UMC and is then the owner of the UMC. They use the UMC to pay for the services they can get at the terminal.

NOTE 1: When the user has paid for the UMC, the Card Issuer may have no rights over the UMC at all and the user may be the owner. Alternatively the Card Issuer may retain rights to the UMC itself, and the user only have the right to use the value on the card.

Trusted Third Party (TTP):

The task of the TTP is to initialize a mechanism, which allows keys to be downloaded from a Card Issuer into the Security Module of a system operator without either party getting to know the keys in plain text form. To do so, the TTP loads an initial loading key into the Security Modules. To enable the Card Issuer to encrypt the keys without knowledge of the initial loading key, the same initial loading key can be installed into some other security device. Thus the Card Issuer can encrypt his operational keys with this secure device and send the encrypted keys to the system operator.

After download into the Security Modules by the system operator, they are decrypted within the Security Modules.

NOTE 2: For economical and practical reasons, the role of the TTP might be fulfilled by the Security Module manufacturer, because the Security Module manufacturer has to guarantee somehow that correct mechanisms (software and hardware) are implemented in the Security Module, and has physical access to the Security Module in the initial phase.

The tasks of a TTP include the following:

- Optionally generate the data field "Master Key (K_{Mttp})" in a secure environment;
- Generate (or optionally diversify from a master key) a Terminal Operator specific data field "K_{TTP}";
- Install the data field "K_{TTP}" in a Secure Device, or use another secure transfer mechanism;
- Install the data field "K_{TTP}" in the Security Modules of the Terminal Operator;
- Protect the TESA-7 algorithms physically.

4.3 The interface model

The interface model that is used as the basis of this document is illustrated in figure 1.

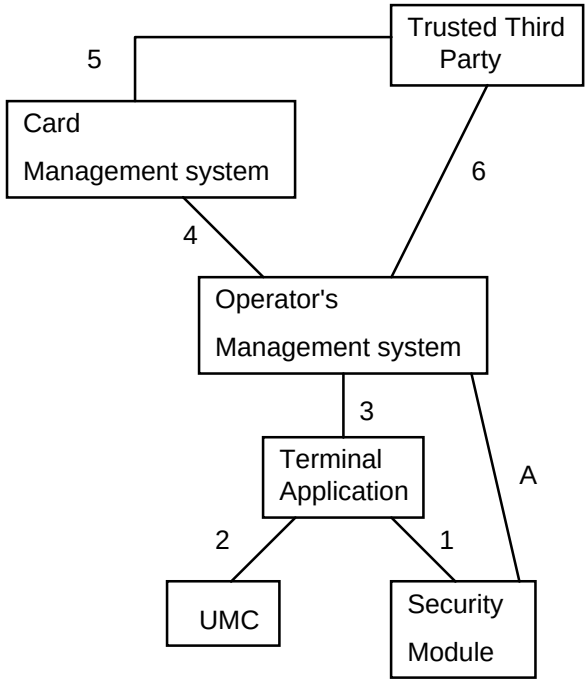


Figure 1: The interface model

This model takes the view of a terminal or system operator and identifies seven different interfaces that are amenable to standardization through this document. These are briefly described here, and are specified (directly or by reference) in subsequent clauses or are specified as being out of scope for this document.

Interface 1 is between the Security Module and the Terminal Application. The Security Module may be co-located with the terminal or may be accessible across a network. This interface is optional as the Security Module may support the interface specified in subclauses 4.8.5, 5.2.4 - 5.2.6, 5.3.6 - 5.3.7, 6.6 and 6.8 or the interface to the Security Module may be considered to be a matter local to the terminal operator. However, if the local interface is adopted it shall still be capable of supporting the information flows required at Interfaces 2 and 3.

Interface A is only available if the SM is in the network. It allows direct exchange of information between the SM and the Operator's Management System without needing to pass this information through the Terminal Application.

Interface 2 is between the UMC, when inserted in a terminal, and the Terminal Application. It is described in subclauses 4.8.3 - 4.8.4, 5.3 and 6.4.

Interface 3 is between a terminal and the management system of some system operator. It is specified in subclause 6.3.

Interface 4 is for the interchange of information between a terminal management system and a card management system. It is specified in subclause 6.2.

Interface 5 is for the interchange of key information between a Card Issuer and a TTP. It is specified in subclauses 4.6 and 5.2.

Interface 6 is for the placement of key information from a TTP into the SMs of a terminal operator. It is specified in subclauses 4.6 and 5.2.

Within the domain of an operator the system components and their interfaces are a local matter. Any architecture and configuration within the operator domain is permissible, so long as the external interfaces numbered 2, 4 and 6 are supported. Support of Interface A is optional.

4.4 Security architecture

The interfaces which are particularly relevant to the security architecture are highlighted in figure 2.

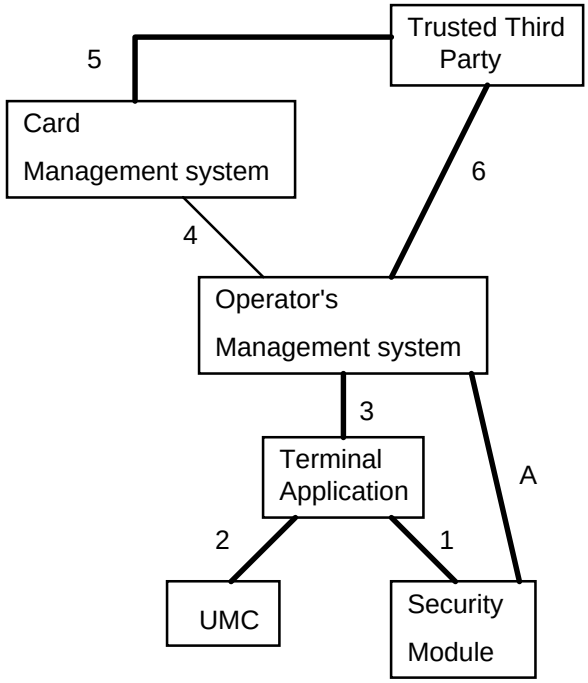


Figure 2: The security related aspects of the interface model

The security architecture takes into account the fact that the SM and the UMC can be regarded as being secure, but the terminal application is not regarded as secure. The SM checks that the UMC is authentic. However, the terminal application does act as the master. That is, it issues commands to the UMC and the SM. The UMC and SM can never initiate commands, only make responses to terminal application commands. This situation is illustrated in figure 3. The SM (and potentially also the UMC) can check the sequence of commands, as well as their content, and respond appropriately if the content or command sequence does not obey set rules. The terminal operator and/or Card Issuer may still require to be assured that the terminal application conforms to its specification and operates correctly under all circumstances. In particular, they need to be assured that UMCs from Card Issuers with which the terminal operator has agreements are correctly handled.

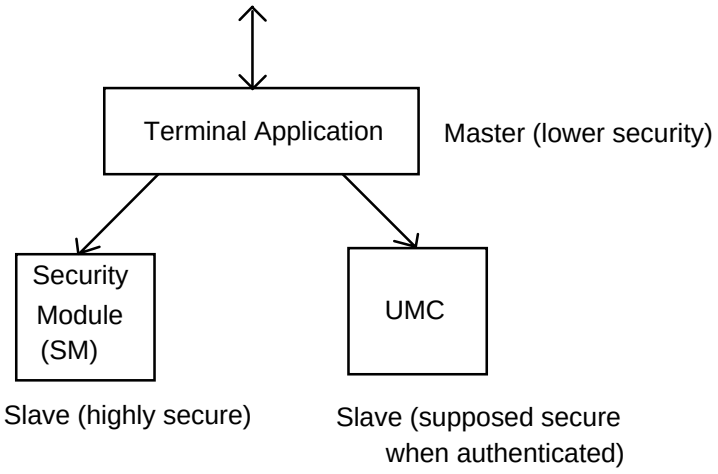


Figure 3: The security architecture model for card authentication

A functional model of the security architecture is given. There are different phases to be distinguished. In an initialization phase physical components of the system such as:

- Security Module;
- secure device (if used);
- UMC,

are exchanged between different parties. The UMC is given to the Users by the Card Issuer. The Security Module is given to the TTP by the Security Module manufacturer. The TTP passes it to the system operator after having installed the initial loading key. The security device is given to the TTP, which installs also in the secure device the initial loading key. Afterwards it is passed to the Card Issuer.

In the operational phase there are several different actions to be considered:

- key download;
- use of UMC (including authentication and decrementing);
- counter increase of SM counter;
- counter extraction of SM counter.

Key download and counter extraction are mainly important for the interface between the system operator and the Card Issuer, but they are also important for the interface between the Security Module and system of the system operator. Counter increase and the use of the UMC are related to the interface between the UMC and the Security Module, which is indirect via the terminal.

It is recommended that all data are classified into "not security sensitive", "security sensitive" and "highly security sensitive" and then to decide which data should be secured by additional means (integrity checksum, encryption) during a data transfer.

4.5 The Security Module (SM) life cycle process model

The life cycle for a SM that is used as the basis of this document is illustrated in figure 4 (see ETR 115 [3]).

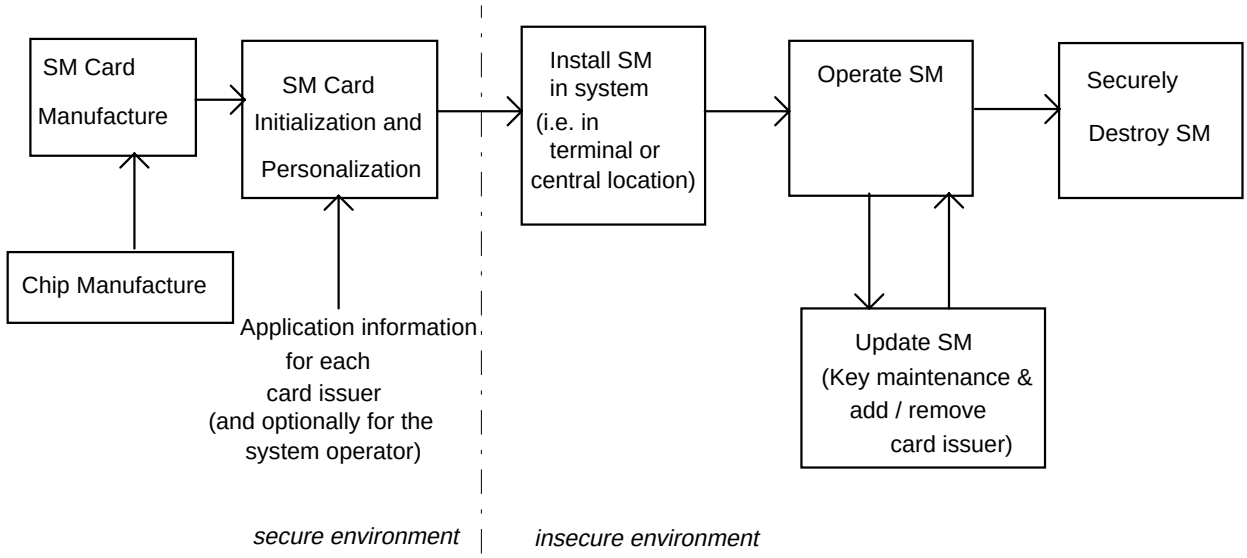


Figure 4: The SM life cycle process model

The life cycle starts with the manufacture of the chip. It is immediately vulnerable to theft and misuse and shall be protected. This is achieved by physical security, organisational security, and internal logical security (see ISO 10202-1 [12]). The chip is passed to the SM card manufacture stage. The SM card is then passed to a TTP. This role may be fulfilled by a logically separate part of the SM manufacture's organization, or by a completely different organization. The TTP loads the applications from the Card Issuers and any specific applications for the client system

operator (as specified by that operator), the required cryptographic algorithm code and a secret key. This stage is known as "personalization".

In practice, personalization may occur in two phases. The first phase involves the loading of the secure operating system and the formation of an empty file structure. The second phase then involves the loading of the initial Card Issuer specific data.

The personalized SM is handed over to the client system operator for installation in the terminal network and the SM enters the operational phase of its life. During this phase it will generally be necessary to load new keys, delete old keys, load, modify and delete data in dedicated and elementary files. Finally the SM shall be securely destroyed at the end of its working life.

4.6 The key management process model

4.6.1 Overview

UMC related keys are required for the correct operation of an UMC. These UMC related keys need to be passed to the SM in encrypted form. A TTP takes the keys from the Card Issuer and provides tools for the secure transfer to the SMs. This data is also required to allow the UMC to be programmed. This process is illustrated in figure 5.

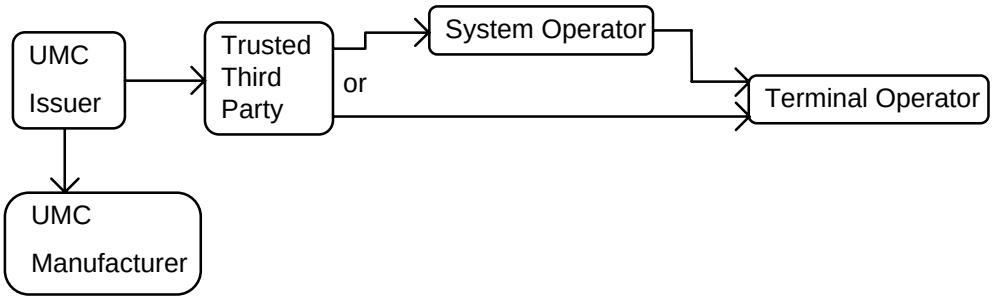
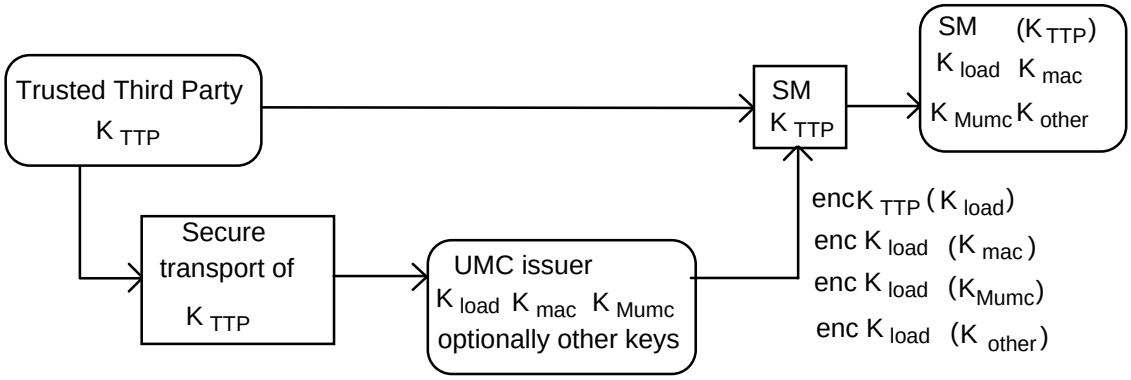


Figure 5: The basic security architecture process model

4.6.2 The key management implementation model

The key management implementation model that is used as the basis of this document is illustrated in figure 6.



NOTE: This scheme is related to the key management required for key management for the Card Issuer. It does not address the management of other keys that may be loaded onto the SM such as a key for generating a MAC on counter values for use by the system operator, or a certificate key.

Figure 6: The basic key management implementation model

The task of the TTP is to initialize a mechanism, which allows keys to be downloaded from a Card Issuer into the Security Module of a system operator without either party getting to know the keys in plain text form. To do so, the TTP generates and loads an initial loading key into the Security Modules. To enable the Card Issuer to encrypt the keys without knowledge of the initial loading key, the same initial loading key can be installed into some other security

device. Thus the Card Issuer can encrypt his operational keys with this secure device and send the encrypted keys to the system operator.

After download into the Security Modules by the system operator, they are decrypted within the Security Modules.

4.6.3 Definitions of the keys

The main keys required for the operation of this architecture are described below. Additional keys may be required for any particular implementation.

Counter MAC Key (K_{MAC})

For the secure readout of counters a key is necessary to create a signature on the value of the counter. This key K_{MAC} will be specific for each Card Issuer, and shall not be known to the System Operator. Each Card Issuer will have at any time one K_{MAC} installed in a specific Security Module. At any time this K_{MAC} can be replaced by a new version.

Initial Loading Key (K_{TTP})

To enable the download of keys into a Security Module managed by a non-trusted party, a special construct is necessary between Card Issuer, System Operator, and a TTP. This construct can be featured by a Secure Device containing the initial loading key K_{TTP} The key K_{TTP} is generated by a TTP specific for every Card Issuer. It might either be derived out of a master key in a secure way or specifically be chosen for every instance of the key. It is handed over to the Card Issuers.

The Card Issuer can encrypt the loading keys using the initial loading key K_{TTP}. These encrypted keys can be downloaded by the System Operator to the Security Module, which decrypts the key and install it in the proper location without the System Operator being able to read this key.

The K_{TTP} should never be known to any party but the TTP.

The K_{TTP} shall be specific for each combination of System Operator and TTP, i.e. this TTP will determine this K_{TTP} for each System Operator ordering Security Modules from this TTP.

If a secure device is used, the Secure Device is handed over by the TTP to the Card Issuer directly, i.e. the System Operator will not be an intermediate partner in this process. The System Operator will however order the Secure Device from a Security Module manufacturer and will have it delivered to the Card Issuer. The secure device encrypts the Card Issuer's keys. This document does not address how these encrypted keys are entered into the SM.

Master Initial Loading Key (K_{Mttp})

This is an optional key that may be used for two purposes:

a) As a master key for generating Card Issuer specific Initial Loading Keys (K_{TTP}) via a key diversification algorithm. The same algorithm needs to be in the SM.

b) As a management key for the secure creation of file sets for additional Card Issuers.

SM Key Loading Key (K_{load})

For the secure download of keys a key is necessary to encrypt the key to be installed into the Security Module. Apart from this encryption mechanism, a mechanism is necessary to insure the integrity of the keys, e.g. the use of a Message Authentication Code (MAC). This key K_{load} is again specific for each Card Issuer, and shall again not be known to the System Operator. Each Card Issuer will have at any time exactly one K_{load} installed in a single Security Module. At any time this K_{load} can be replaced by a new version.

UMC Authentication Key (K_{umc})

The UMC Authentication Key (K_{umc}) is unique to each UMC. It is generated from an UMC authentication Master Key (K_{Mumc}) and the UMC's identification data by means of a key diversification algorithm. It is placed on the UMC by the Card Issuer, and is used internally within the UMC when the UMC is authenticated by a terminal, and when it is used to pay for a transaction. It is very important that this key remains hidden inside the UMC and that it is not possible to read it out.

UMC Authentication Master Key (K_{Mumc})

The Card Issuer generates an UMC authentication Master Key (K_{Mumc}). It use this with a key diversification algorithm and card identification data to generate a separate authentication key (K_{umc}) for each UMC it issues. The Card Issuer also transfers this UMC authentication Master Key (K_{Mumc}) in encrypted form to the Security Modules that will be used for authenticating UMC transactions. This document specifies the use of TTP for this transfer as an aid to maintaining overall security. The actual secure physical transfer of this important key may be achieved using a secure device that encrypts the key and keeps it hidden from the terminal operator or by any other means of secure (usually encrypted) transfer.

The Security Modules use the UMC authentication Master Key (K_{Mumc}) together with a key diversification algorithm and card identification data to independently generate authentication key (K_{umc}) for an UMC it is requested to authenticate. It uses this key in the UMC authentication algorithm.

It is recommended that at least 3 versions of K_{Mumc} are supported.

NOTE: Some systems require the support of 4, or more, versions of K_{Mumc}.

UMC Certification Master Key (K_{Mcer}) optional

The aim of the certificate is to increase the UMC security during its life cycle. The certificate is a 16-bit value which can be written in the UMC at a specific place and at a specific time. For example it can be written just before the delivery of the card to the customer. Then in case of theft of the UMC before certificate inscription, the UMC is valueless.

The certificate can also be used to reload the UMC. In this case it proves that the UMC has been reloaded with true units.

The certificate is computed with UMC data and a secret key K_{Mcer}. The SM shall support at least 4 K_{Mcer} versions, if the certification process is supported.

The SM will check the certificate value during the authentication procedure. This operation is not mandatory.

The K_{Mcer} key is loaded using the same procedure as K_{Mumc}.

4.7 The billing and accounting process model

The billing and accounting model that is used as the basis of this document is illustrated in figure 7.

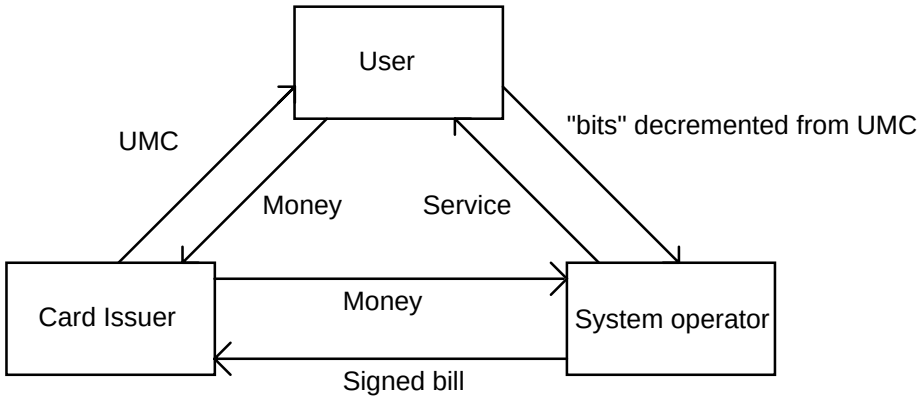


Figure 7: The billing and accounting process model

The model shows an overview of the way the payment system works. The complete cycle consists of 3 steps:

1. The Card Issuer sells UMCs to the user i.e. they give the UMCs to the user and receive money for it.

2. The user receives service at a terminal of the system operator. The terminal application has to ensure that the counter in the UMC is decreased by the correct amount over time as the service is used and that the counter in the Security Module is increased by the same amount.

3. The system operator presents the counters of the Security Module to the Card Issuer and receives money from the Card Issuer. This is represented in figure 7 as the "signed bill", which is just the counter value "signed" with a MAC.

4.8 User Memory Card (UMC) characteristics

4.8.1 General characteristics

This document considers an UMC to be a synchronous memory card with the following common features:

- Identification area;
  - this area contains data needed to identify the chip type and the Card Issuer.
- Counter;
  - the counter is an abacus counter. The counting area consists of several counter stages. Counting is done by writing bits in the relevant counter stages. A counter backup procedure exists and can be used to inform the terminal that a problem occurred in the counter set up procedure.
- Authentication;
  - to check the validity of the card a one way authentication mechanism is used. This mechanism is based on a strong authentication algorithm (cryptographic method not based on a PIN or CHV), using a secret key, a challenge and card authentication data including the counting area, the identification area and possibly other data.
- In accordance with ISO 7816-1 and ISO 7816-2 [6, 7] and 3 (section 1 to 4.2.7) [8].

The UMC has two different access configurations: Issuer configuration and User configuration. In Issuer configuration access is permitted to certain memory locations on presentation of the correct value for the "Transport Locking Code". Some ICC types do not have this "Transport Locking Code" facility and thus can be programmed without presenting a correct value. Issuer configuration is intended to be used by IC manufacturers, and by ICC manufacturers in the personalization phase. Once the User configuration has been entered it shall not be possible to revert to Issuer configuration. In User configuration access shall be strictly controlled by security logic on the IC.

The overall physical dimensions of the UMC shall be in accordance with the ID-1 type as specified in ISO 7816-1 [6]. The location, size and position of the electrical contacts shall be in accordance with ISO 7816-2 [7]. The surface profile of the electrical contacts shall conform to ISO 7816-1 [6].

The card may carry an indentation on one edge, or some other physical feature, which enables a person with impaired vision to insert the UMC into the terminal in the correct orientation (see ETR 165 [4]).

The UMC shall have electrical contacts as denoted in ISO 7816-2 [7]. The Vpp contact shall not be used.

4.8.2 UMC recognition

There are different low level protocols used by UMC chip manufacturers. Some UMC readers will only be able to use one of these protocols. Operators of such terminals will only be able to accept UMCs from Card Issuers using that protocol. If the reader can use more than one, then it has to recognize which one is implemented on any UMC presented to it. The terminal shall try each protocol in turn until one yields readable data. If all available possibilities fail to produce good data, then the terminal shall reject the UMC as unreadable. The reader shall not electrically damage the UMC nor modify the UMC counter during these operations.

It shall be possible for the terminal application to be able to recognize from the data on an UMC, which Card Issuer issued that UMC (so that the correct files in the MS can be selected for UMC authentication and counter increment).

The terminal application shall read at least the first 2 Bytes on the card to determine if the card is in the interoperability scheme outlined in this document. The terminal application shall read additional Bytes to recognize all the characteristics associated with the card and the Card Issuer.

It is recommended that the terminal reads at least the first 4 Bytes on the UMC to positively identify the Card Issuer.

According to ISO 7816-3 (section 6.2) [8] the first 2 Bytes of the memory may be used for UMC identification.

These 2 Bytes of data are allocated by PRO ELECTRON and are unique to a chip type and card manufacturer. A chip manufacturer requests a specific value for the UMC first 2 Bytes. The Card Issuer name is optional, only the chip manufacturer is mandatory. These first 2 Bytes may also be unique to a Card Issuer. In the case where the first 2 Bytes are not unique to a Card Issuer, the UMC data shall be such that the Card Issuer can be uniquely identified by the first 2 Bytes in conjunction with the 3rd and 4th Bytes. The meaning of other card data may then be determined from the data format used by that Card Issuer and supplied to the system/terminal operator as part of their bilateral agreement.

The first 2 Bytes are managed by PRO ELECTRON.

In the interoperability scheme, the value of all the first 2 Bytes used by each parties are included in the bilateral agreement, PRO ELECTRON is only able to guarantee that each code is reserved for an unique chip manufacturer, but PRO ELECTRON is not allowed to give any information on the codes used by any Card Issuer and these data are considered to be sensitive. The address of PRO ELECTRON is as follows:

PRO ELECTRON

140 avenue LOUISE (6th floor)

1050 BRUSSELS BELGIUM

4.8.3 Counter operation

The counter on the UMC consists of 4 or 5 rows of memory and operates on the abacus principle. A number of variants of the basic scheme are in current use and these are described in annex D. Card Issuers shall adopt one of these standard schemes. Two approaches to handling counter values in the terminal application and the SM are possible. The first is to store the counter method against each card type and then to handle each counter method separately. As each method is a variant of the basic abacus counting principle, it should be possible to store a set of parameters for each card type and to use a single algorithm with a set of parameters to determine absolute counter values and the difference between two counter values. Detailed information on the counter methodologies is given in annex C to assist designers.

4.8.4 UMC counter handling by the terminal

The general abacus counting mechanism described in subclause 4.8.3 may be further constrained such that its use is restricted to a particular counter bit format. Examples of counter formats currently in use are given in annex D.

Where a Card Issuer requires that their UMCs are maintained in a particular counter format, this shall be defined in the bilateral agreement between the Card Issuer and terminal/system operators. All terminals of that terminal/system operator shall only accept UMCs from that Card Issuer if the counter is in the specified format, and shall only decrement bits from the counter such that this format is maintained.

If at any point during a call, the terminal reads the counter in an illegal format, it shall abort the call and not increase the SM counter.

4.8.5 UMC counter interpretation by the SM

To ensure that the secure counter in an SM is only increased securely in relation to bits decremented from an UMC, the SM shall be able to interpret the counter bit pattern from the UMC as a bit value. Any SM shall be able to interpret counter bit patterns for all UMCs accepted by the connected terminal(s).

Where bits in the UMC counter area which can be changed are used for purposes other than the counting operation, the SM shall not attach any value to those bits.

Where a number of different counter bit formats are to be supported, the SM may optionally use a general counting method together with a set of parameters to modify it for the different abacus counter formats. Parameters may be used for the following purposes:

To indicate the value of a counting bit in each counter row.

To indicate how to identify an erased/written counter bit.

...

Other parameters may be used to further define the accepted counter formats and to control the minimum number of authentication signature bits required for secure counter increase.

If different counting methods or counter parameters are supported for one Card Issuer, the counter method or parameters used by the SM shall be securely linked to the K_{Mumc} selected.

5 Procedures

5.1 Introduction

All procedures described in the following subclauses are based on the system overview described in clause 4.

In this subclause the following procedures are described:

- Key handling.
- Normal operation of the Terminal- and UMC-handling.
- Payment and settlement.
- Operational differences for network based Security Module (SM).

Each procedure uses several data elements. The structures of the data elements are described in clause 6.

There are 3 mandatory different kinds of keys for the Card Issuer to be downloaded into the system of the operator, load key, MAC key and UMC master key. Other keys may also be used for specific purposes. The purpose of the load key is to have an additional key hierarchy between the initial loading key of the TTP and the operational keys (MAC key and UMC master key). The MAC key serves as a key to sign the counter in the Security Module, where the counter is specific for the Card Issuer. The UMC master key is the key that is diversified into individual keys for each Card Issued by the Card Issuer.

The Card Issuer initiates the key management process when issuing new types of cards requiring new keys, or when one, or more, current keys have expired. The new keys shall be loaded in all the SM which accepts these UMCs.

5.2 Key handling for interoperability (symmetric key scheme)

5.2.1 Key management procedures of the Trusted Third Party (TTP)

It is required that the Trusted Third Party maintains a security management system to fulfil the security requirements.

The first step of the Trusted Third Party is to generate for each Card Management system a dedicated Initial Loading Key (K_{TTP}). That dedicated K_{TTP} shall only be known to the TTP. That K_{TTP} therefore needs to be stored in a very secure way.

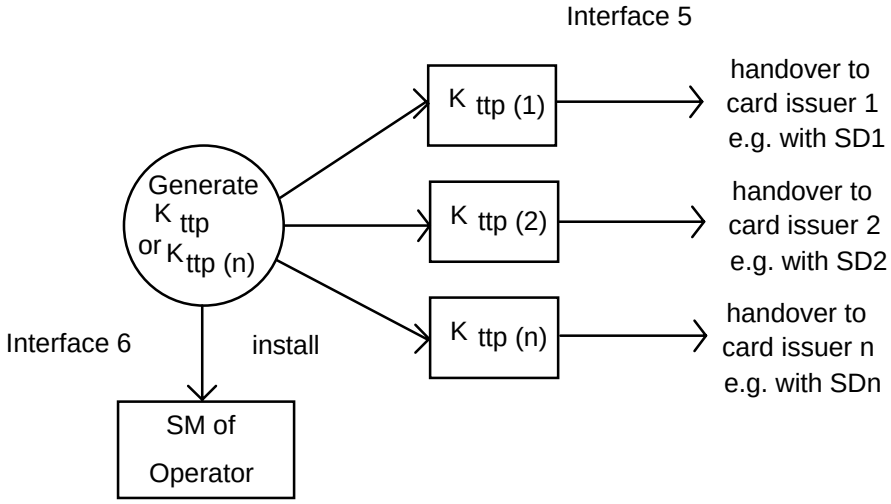


Figure 8a: TTP key generation - single key or randomly generated keys

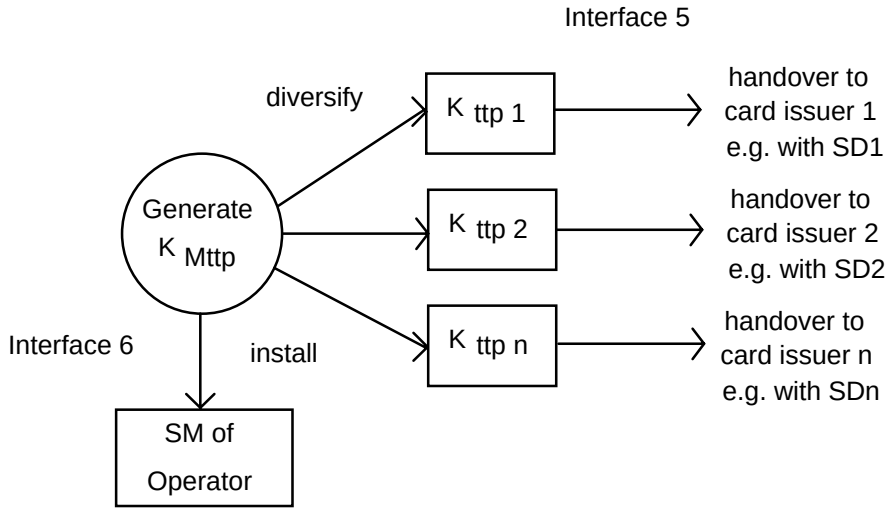


Figure 8b: TTP key generation from a master key

There are 3 schemes for generating and handling K_{TTP}. The first is to use the same value for K_{TTP} for each Card Issuer. Only this one value needs to be loaded into the SM. The main disadvantage of this scheme is that if this single value of K_{TTP} is used for the whole of the SM life and it becomes known to anyone other than the TTP, then the complete system is compromised. Therefore this scheme is not recommended. This scheme is illustrated in figure 8a.

NOTE: A viable variation on this scheme is to replace the current version of K_{TTP} with K_{load} each time keys are downloaded so that on each key download cycle $K_{\mathbf{TTP}}$ is given a new value.

The second is for the TTP to choose an arbitrary value for $K_{\mathbf{TTP}}$ for each Card Issuer. This scheme is illustrated in figure 8a. It implies that SMs have to be pre-loaded with the $K_{\mathbf{TTP}}$ values for each Card Issuer. If an unanticipated Card Issuer enters the system then SMs from operators intending to interoperate with UMCs from that Card Issuer will have a new $K_{\mathbf{TTP}}$ loaded under the control of the TTP.

The third scheme is illustrated in figure 8b. A master key (K_{Mttp}) is generated by the TTP. A Card Issuer identification parameter and a diversification algorithm are used to generate separate values for K_{TTP} for each Card Issuer. The single master key (K_{Mttp}) value is loaded into each SM. Whenever a Card Issuer downloads new keys to an SM, it has to generate the correct value of initial loading key, K_{TTP}, for that Card Issuer by using the same diversification algorithm, Card Issuer identifier and master key (K_{Mttp}) value. This method is more secure than the first method. It is not as secure as the second method, but does not require the permanent storage of many K_{TTP} values in the SM and the attendant update problems.

The generated K_{TTP} (first and second scheme) or K_{Mttp} (third scheme) will be installed in the SMs of an Operator during a personalization process of the SMs. All SMs will then be handed over to the operator.

If that operator wants now to accept the cards of a Card Issuer n it is necessary to hand over the K_{TTP} n to the Card management system to prepare the further steps (encryption of the keys of the Card management system, hand over to the Operator's Management System and installation in the SMs of the operator).

5.2.2 Encryption of keys in the card management system

During a key exchange the following data elements need to be encrypted before they are handed over from a Card Management System to an Operator's Management System:

- K_{load};
- K_{Mumc};
- K_{MAC};
- optionally K_{Mcer}.

The encryption of these keys of the Card Management system can be done by means of the secure device which is handed over from a TTP and plugged in to the system or by a piece of software in the system.

The following tasks shall be performed:

- Generate new keys (e.g. K_{load}, and K_{MAC}).
- Receive the data field K_{load} in a secure way. It is recommended that the Card Management uses a secure central computer to handle the cryptographic data. This central computer shall be located in a secure area.
- The data field K_{load} is encrypted with the data field K_{TTP} and the recommended algorithm agreed between the partners in the required format.
- The encrypted K_{load} is then handed over again to the Card Management system.
- An encryption tool encrypts the K_{Mumc}, K_{MAC}, and any other keys with the K_{load}.
- The data fields K_{Mumc} and K_{MAC} are handed over in encrypted form.

After the initialization phase carried out by the TTP, the TTP hands over the Security Modules to the Terminal Operator. After that step the Security Modules are prepared to receive cryptographic data of the Card Management System.

It is recommended that TESA-7 is used as the key diversification and the key encryption algorithm.

After the encryption of the relevant keys the encrypted data needs to be handed over to the Terminal Operator in the data elements described in clause 7.

If a secure device is used, then the procedure is given in annex B.1.

If a secure device is not used, then some other mechanism has to be agreed between the parties. Such alternative mechanisms are not specified in this document.

If a secure device is used, then the Card Management System provides an interface for the Secure Device.

5.2.3 Handover of keys to the Operators' Management System

During a key exchange the following data elements need to be handed over from a Card Management System to an Operator's Management System:

$encK_{TTP}$ (K_{load});

$encK_{load}$(K_{Mumc});

$encK_{load}$(K_{MAC});

optionally $encK_{load}$(K_{Mcer}).

In addition to these encrypted keys, sufficient information has to be known by both parties to identify the use of the keys (e.g. key identification, format, and algorithm).

It is optional for K_{load} to replace K_{TTP} and become the new "K_{TTP}" for the next download of "K_{load}". This depends on the method to be used for the subsequent update of K_{load}.

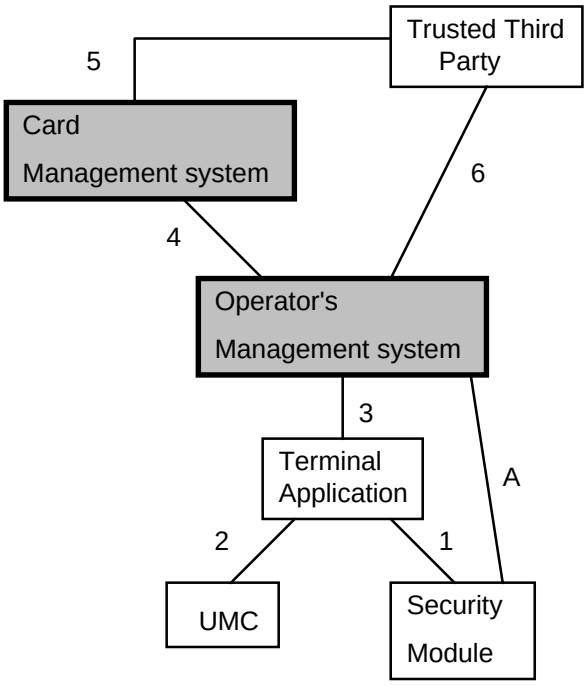


Figure 9: The key exchange model

During a key exchange between Card Issuer and terminal operator the highlighted boxes are relevant.

5.2.4 Key download procedures

5.2.4.1 Initial key download

After a secure hand over of the encrypted keys from a Card Management System to a Operator's Management System the encrypted keys need to be downloaded into the Security Modules. These encrypted keys are then decrypted and installed in the Security Module.

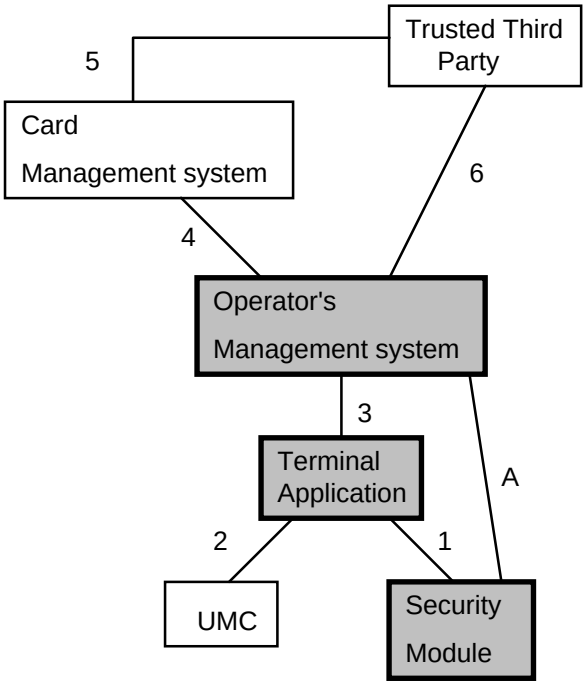


Figure 10: The key download procedures model

During a key download procedure the highlighted boxes are relevant.

For a secure download of the encrypted data of a Card Management System into the Security Module the following procedures are necessary:

Download of the encrypted data fields into the Terminal Application in a secure way: it is strongly recommended that a further cryptographic layer is used for a secure download of data fields.

If a Security Module structured according to EN 726-7 [2] is used then the procedure is as follows:

- The Terminal Application or Operator's Management System selects the DF of the SM.
- The data field "$encK_{TTP}(K_{load})$" is handed over to the SM.
- The SM decrypts the data field and installs the K_{load} in a special EF in the SM.
- The data field "$encK_{load}(K_{Mumc})$" is handed over to the SM, decrypted in the SM and installed.
- The data field "$encK_{load}(K_{MAC})$" is handed over to the SM, decrypted in the SM and installed.
- Other keys if required (e.g. for certification).

It is optional for K_{load} to replace K_{TTP} and become the new "K_{TTP}" for the next download of "K_{load}". This depends on the method to be used for the subsequent update of K_{load}.

The encryption mechanism shall also include a process to ensure the integrity of the key, e.g. the use of a Message Authentication Code (MAC). The integrity mechanism (e.g. MAC) shall be calculated from the key and optionally additional data (e.g. to securely link the data to the key).

5.2.4.2 Updating of keys

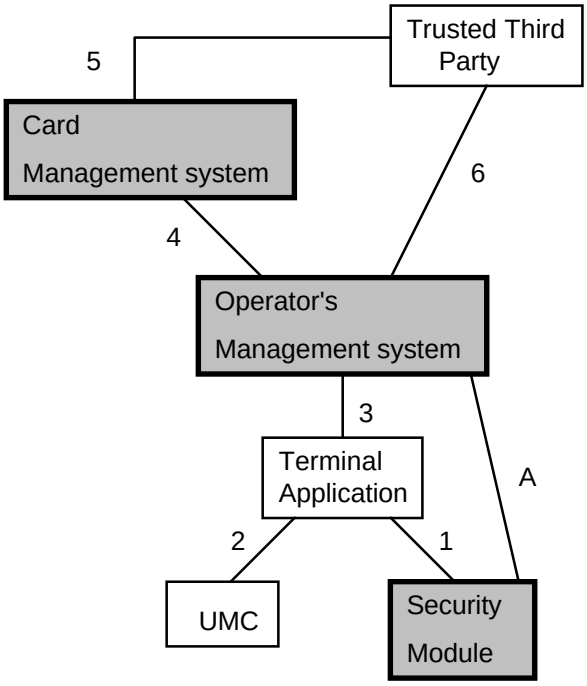


Figure 11: The key updating procedures model

During the update of the keys held in a Security Module, the highlighted boxes are relevant.

The following procedures relate to the update of any of the keys related to a specific Card Issuer in the Security Module. These procedures apply to the following keys, which originate from the Card Issuer:

- K_{load};
- K_{MAC};
- K_{Mumc};
- other keys, such as a certification key, K_{Mcer}, for checking certificates on the UMC.

In the case of any optional keys which originate from another source (e.g. an operator), the procedures below should be modified as appropriate e.g. to use a TTP (and optionally a secure device).

All keys other than K_{load} are encrypted with K_{load}. For K_{load} there are two possibilities:

a) all versions of K_{load} are encrypted with the same K_{TTP};

b) a new version of K_{load} is encrypted using the previous version of K_{load}.

These procedures assume that initial key download has already been performed (as defined in subclause 5.2.4.1). In this state the Security Module application related to a Card Issuer has been initialized with a loading key K_{load} specific for that Card Issuer.

The procedure for the updating of keys shall meet the following basic requirements.

1) The Card Issuer shall encrypt the new key using K_{load} (or K_{TTP} for K_{load}). The encryption mechanism shall also include a process to ensure the integrity of the key, e.g. the use of a Message Authentication Code (MAC). The integrity mechanism (e.g. MAC) shall be calculated from the key and optionally additional data (e.g. to securely link the data to the key).

2) The encrypted key data shall be handed over to the operator for transmission via the Operator's Management System to the Security Module. Additional data may also be necessary to the key updating process, such as identification of the key to be replaced and a key version number.

3) The Security Module shall verify the integrity of the key (e.g. by checking the MAC for that key or a sufficient redundancy in the key) and decrypt it using K_{load} (or K_{TTP} or the previous version of K_{load} for K_{load}). If the key has been successfully verified, the Security Module shall store the new key, replacing the previous key. If the key is not successfully verified, the Security Module shall not allow the new key to be used. The previous key may be retained in this case.

4) The Security Module shall indicate whether or not the key update operation was successful. An indication shall be returned to the Operator's Management System if the update operation failed. Returning information of a successful key update to the Operator's Management System is optional.

The operator shall ensure that all Security Modules under its control are maintained with the latest versions of all keys.

It is recommended that each key in a Security Module is associated with a key version number.

Depending on the individual requirements for specific keys, it may be necessary to store more than one version of the same key. When a key is updated, a key version number can be used to select the version of key to be replaced.

The mechanism used for the updating of keys shall ensure that it is not possible to install old versions of keys in place of new versions, by replaying previous key download messages to the Security Module. This can be achieved by comparing the version numbers of keys in the Security Module. In this case there shall be a secure link between the key and version number.

Other procedures for the updating of keys are not excluded by this specification provided they can be shown to meet the basic security requirements defined in these procedures.

5.2.5 Adding a new application

A new application could consist of a new DF structure, new parameters for an existing file structure, and so on. There are two methods for adding an application for a new Card Issuer to an existing Security Module. The main difference between these two methods is whether the DF for all applications are created at Security Module personalization, or whether the DF for each new application is created as the need arises. Data to be loaded may include:

- algorithm identifiers;
- counting method; and
- card types.

a) The first method is for the Security Module to be personalized by the TTP, with a number of DFs created ready for future use by Card Issuers. The memory allocated to each DF shall take into account the likely requirements of future applications. In each DF, only the initial management key(s) and, optionally, the file structure will be loaded. No other file contents will be created (other than initializing non-re-setable counters to 0). The initial management keys shall be different for the DF of each Card Issuer and will be held by the TTP. When a new Card Issuer application is to be added, the relevant TTP key(s) for that DF shall be securely given to the Card Issuer (e.g. using a secure device), so the Card Issuer can replace the initial management key(s), load their data and enable the application.

b) The second method is to allow new applications to be added by creating a new DF during the operational phase of the Security Module. This method is more flexible as the application identifier and memory can be allocated to the DF when the requirements of the application are known. To achieve this there shall be a master key at the MF level, known by a TTP, which can be used to create DFs. For this method, the Card Issuer shall request a DF of a defined size to be created. This DF will be created and loaded with initial management key(s). At this stage the procedures for initializing the application, are the same as for the first method.

It shall not be possible for an Untrusted Party to download executable code into a Security Module without the use of a Trusted Party. Additional procedures, to be included in the bilateral agreements between the terminal/system operator (see annex E.4) shall be used to ensure that the security of any other application in the Security Module is not compromised. All code shall be evaluated and certified (ROM and EEPROM), in combination with the code already in the SM, before being downloaded. The download command shall be protected by cryptographic data specific to each SM.

5.2.6 Removing an application

Invalidating an application shall be possible. In the SM, under the control of the TTP, some special commands shall allow the invalidation of all data related to a Card Issuer. The SM may be able to reuse the free memory for another Card Issuer.

5.3 Normal operation of the terminal and UMC handling

In this subclause the important normal operations of the terminal regarding the UMC handling and the Security Module as far it is involved are described. The basic operations are:

- identify type of UMC and other preparatory steps;
- check Blacklist;
- key diversification;
- UMC authentication;
- decrease of counter in the UMC;
- increase of counter in the SM;
- extraction of the counter from the SM.

Furthermore, if a certificate is used, there is an additional operation to check the certificate.

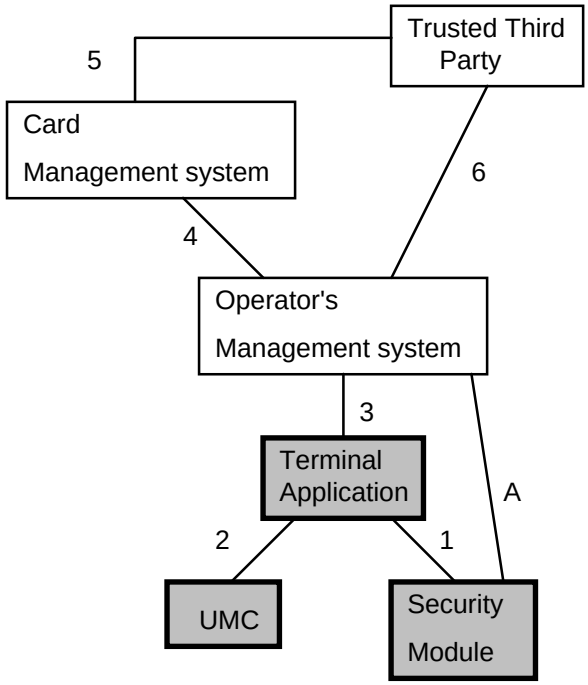


Figure 12: Security during normal operation

During a normal operation the highlighted boxes shown in figure 12 are relevant.

Thus the procedures during normal operation described here are relevant for the following interfaces:

- Interface 3 between the terminal application and the Operator's Management System.
- Interface 2 between the terminal application and the UMC.
- Interface 1 between the terminal application and the Security Module.

- Interface A between the Security Module and the Operator's Management System in the case of an SM in the network (see subclauses 5.5 and 6.8).

5.3.1 Preparatory steps (e.g. identify UMC type)

After the UMC is inserted into the terminal, the terminal determines the protocol used by that UMC and then resets the card.

The terminal reads out the data elements of the UMC.

The terminal shall determine the UMC type (see subclause 4.8.2) and use this information to enable it to correctly interpret data from the UMC and to identify the necessary parameters for UMC authentication (e.g. Key version authentication algorithm).

The terminal shall determine the Card Issuer of each UMC with the help of up to the first 4 Bytes, and shall use this information when accessing the SM.

Based on the information read from the UMC, the terminal shall decide whether to accept the UMC or not. If it is accepted then the terminal shall proceed as described below. The decision shall include information relevant to features supported by that card type.

The terminal shall check if any counter backup bits (i.e. an anti-tearing mechanism) are active. If so it shall initiate a recovery cycle. If possible, illegal use of any counter backup mechanism shall be detected. If it is the terminal shall reject the card and/or record the UMC data and raise an alarm to the management system.

The terminal may reject any UMC that has an illegal pattern of bits in the counter area or record this situation against the card identifier and continue operation (as defined in the bilateral agreement). This may happen as a result of the initial read of the card data or on any subsequent reading of data. The SM may also check the counter value bit pattern during authentications.

The terminal shall check if the card is a Fixed Number Dialling (FND) card only. This check may have already been performed in recognizing the card type. If so then the terminal shall either not accept the card or lock out keypad dialling and only offer connection to the fixed number if the card authenticates correctly. If the card is Fixed Number Dialling only, then the terminal may display a message to the user to this effect and request user confirmation. If the user confirms then the terminal shall dial the fixed number, otherwise the terminal shall reject the service request. Alternatively the terminal shall dial the fixed number directly.

The terminal shall check if a certificate ought to be present. This check may have already been performed in recognizing the card type. If so, it shall initiate the check that the certificate is valid. If it is then the terminal shall proceed to the next step. If it is not valid then the terminal shall reject the service request. Optionally it may also raise an alarm and/or capture data from the UMC for latter analysis.

If the certification key is present, then the SM is able to check the UMC certificate. This operation is done during UMC authentication. The selection key mechanism is the same as the authentication key selection. The certificate checking depends on bilateral agreement (case of interoperability between an operator using certificates and another which does not use certificates).

The key diversification scheme includes UMC data Bytes from Byte 0 to Byte 13.

The terminal knows the DF in the Security Module to start the application.

The terminal shall check the UMC counter and calculates the monetary value in the local currency.

The terminal shall check if there is sufficient value on the card for the required service.

5.3.2 Blacklist

The terminal shall have access to a Blacklist, and optionally a Whitelist of UMC numbers or UMC batch numbers. The terminal shall check whether Blacklist or Whitelist checking needs to be performed for UMCs from that Card Issuer. If not the terminal shall omit Blacklist and Whitelist checking. Otherwise the terminal checks if there is a value limit to the check. If so it shall check whether the original UMC value is below the agreed limit required for Blacklist/Whitelist checking. Otherwise the terminal shall check that each UMC presented is on the Whitelist of valid, issued UMC in

circulation, if one is provided, and is not on the Blacklist of UMCs that have been stopped, withdrawn from service, time expired or have been used up.

The correct handling of the Blacklist and/or Whitelist is not secured by the Security Module. Thus it cannot be regarded as secure.

The size of the Blacklist and/or Whitelist, and other details, should be the subject of agreement between the terminal/system operator and the Card Issuer (see also annex E), and may be limited by the amount of storage available at the terminal.

5.3.3 Key diversification

Every UMC contains a secret UMC-key K_{umc}. To be able to calculate the card individual keys in the Security Modules it is necessary to have card keys derived from a master key K_{Mumc}. These master keys are stored in the Security Module and the card individual keys are calculated from the card id (ID) and the master key (as identified by the master key identifier) with an algorithm for key diversification KEY_DIV:

$$K_{umc} = KEY\_DIV[K_{Mumc}](ID).$$

The master keys are currently 16-Byte long. The card identifier (ID) shall uniquely and unambiguously identify that card and only that card. The card identifier (ID) shall also unambiguously be associated with a value for K_{umc}.

The algorithms for key diversification have to be implemented in the Security Modules. Therefore it is important to use one, or a few, common algorithms. It is recommended that TESA-7 is used. A TTP shall protect the TESA-7 algorithms physically.

5.3.4 Card authentication

The authentication of the UMC is executed on request of the terminal. The authentication is necessary to recognize valid cards inclusive of the value stored in the card. Furthermore the authentication is necessary to handle the counters in the Security Module securely as described in subclause 5.3.6.

The authentication is based on the card individual keys K_{umc}. The authentication function AUT_UMC takes as input a random number R and 16 Bytes of data stored in the EEPROM of the UMC together with the key K_{umc}:

$$S = AUT\_UMC[K_{umc}](R, X),$$

and computes a signature S consisting of at least n bits where the value of n is agreed in the bilateral agreement (see annex E item 14). It is strongly recommended to use at least 16-bit signatures. If the UMC provides signatures shorter than n bits, then for the calculation it is necessary to use successive authentication functions using different random challenges to build up a value of n bits or more.

The 16-Byte data X includes the counter of the UMC.

To summarize the location of data, algorithms and keys:

- the UMC contains the 16-Byte data X, the UMC-key K_{umc} and the authentication algorithm AUT_UMC;
- the Security Module contains the secret master key K_{Mumc} and both algorithms, the key diversification algorithms KEY_DIV and the authentication algorithms AUT_UMC. Furthermore the Security Module needs a cryptographically secure (pseudo-)random number generator to calculate a random number R for each authentication.

NOTE: The procedure below describes the procedure and processes required for initial authentication. The actual order of events and details are indicative only.

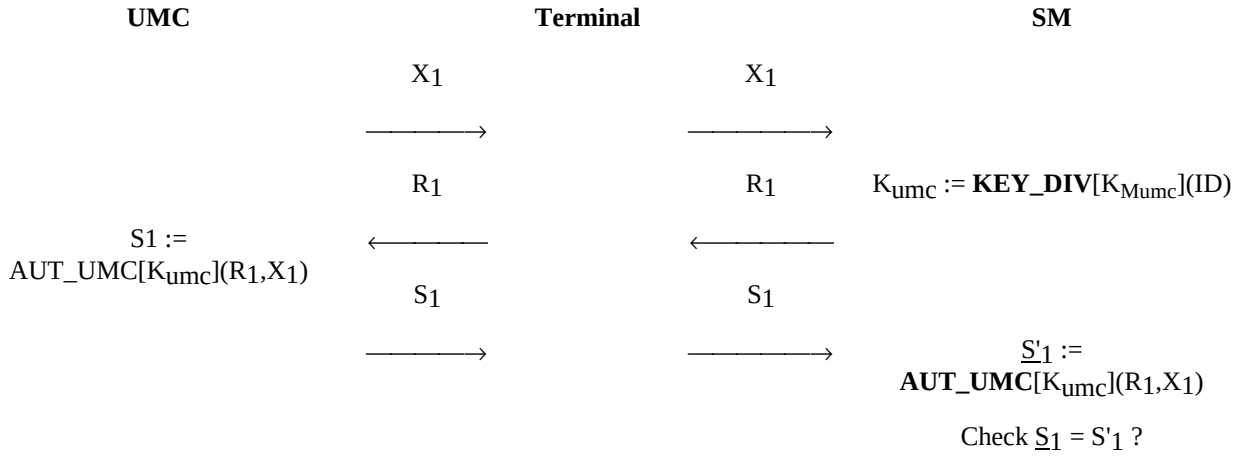


Figure 13: Initial UMC-authentication

1. The terminal reads the 16-Byte card data X_1, consisting of the Card Identity ID, the counter value in the Card C and further information, and sends it to the Security Module.

2. With the identity of the card ID, the Security Module selects the correct master key K_{Mumc}, and computes the UMC key:

$$K_{umc} = KEY\_DIV[K_{Mumc}](ID).$$

It stores K_{umc} and the data X_1 temporarily.

3. The Security Module generates a random number R_1, stores it temporarily and sends it to the terminal.

4. The terminal sends the R_1 to the UMC.

5. The UMC computes a n-bit response, with $n \geq 4$

$$S_1 := AUT\_UMC[K_{umc}](R_1,X_1).$$

It is strongly recommended that 16-bit, or longer, responses are used. If the UMC generates less bits than required then steps 3 to 5 shall be repeated, with different random numbers (or a number generated by a suitable function in the SM or terminal application) and the individual signature bits concatenated together to form an overall response of the required length. The Card Issuer defines the minimum length of the responses and that minimum length shall be enforced by the SM.

6. Te terminal reads S_1 from the UMC and sends it to the Security Module.

7. The Security Module computes

$$S'_1 := AUT\_UMC[K_{umc}](R_1,X_1)$$

and checks internally whether $S'_1 = S_1$.

If this initial authentication is successful, then the call can start.

5.3.5 Decrease UMC counter

With each billing information from the network or a possible self tariffing of the terminal application, the terminal calculates the value to be debited.

The terminal holds the exchange rates for the counter bits for every co-operating Card Issuer.

The terminal calculates further the number of bits to be debited from the card according to the exchange rate.

To handle the UMC counter correctly, the terminal has the information about the location and use of the counter method. Where different UMC authentication algorithms or methods of interpreting the UMC counter (counting mechanism or bit value) are supported, the selection of these shall be linked in the SM to the K_{Mumc} selected.

The terminal debits the required bits from the card.

The terminal displays the remaining value on the card on the display.

If the value on the card reaches zero effective value remaining during the provision of service, then the terminal may offer continuation of the service after the removal of the UMC and its replacement with one with value remaining. An initial authentication shall be performed on the replacement UMC. The terminal may also offer a switch to a different payment method (such as cash), if one is available. If the terminal is not able to offer continuation, or the user does not continue within a given time period then the terminal shall terminate the service.

The terminal requests further authentications of the card. To prove the authenticity of the number of bits taken from the card, further authentications are necessary. Figure 14 gives a detailed description of the k-th authentication procedure.

NOTE 1: The procedure below describes the procedure and processes required for authentication. The actual order of events and details are indicative only.

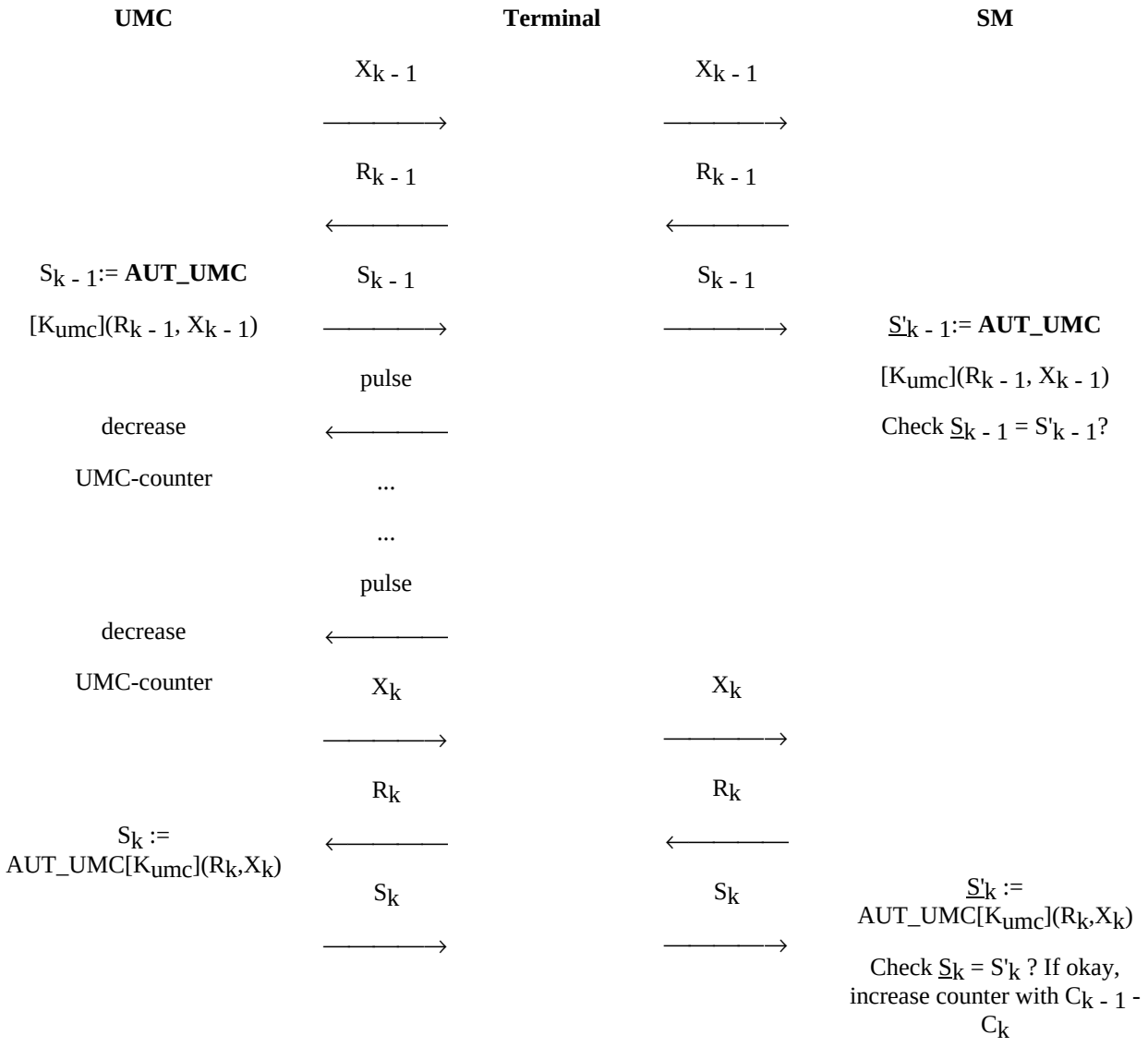


Figure 14: The k-th UMC-authentication procedure

It is assumed, that the authentication for k-1 has been successful. Then, after a certain number of pulses, a subsequent authentication is executed as follows:

1. The terminal reads up to 16-Byte data X_k, consisting of the card identity ID, the current counter C_k and further data out of the EEPROM of the UMC, or at least the data that should change, and sends it to the Security Module.

2. The Security Module stores X_k temporarily.

3. The Security Module generates a random number R_k, stores it temporarily and sends it to the terminal. It is not essential that the random number, R_k, is different from the previous value, R_{k-1}, for one UMC if some other data in the calculation has changed.

4. The terminal sends the R_k to the UMC.

5. The UMC computes a n-bit response, with $n \geq 4$.

$$S_k := AUT\_UMC[K_{umc}](R_k, X_k).$$

It is strongly recommended that 16-bit, or longer, responses are used. If the UMC generates less bits than required then steps 3 to 5 should be repeated, with different random numbers (or a number generated by a suitable function in the SM) and the individual signature bits concatenated together to form an overall response of the required length. The Card Issuer defines the minimum length of the responses and that minimum length shall be enforced by the SM (see annex E).

6. The terminal reads S_k from the UMC and sends it to the Security Module.

7. The Security Module computes

$$S'_k := AUT\_UMC[K_{umc}](R_k,X_k)$$

and checks internally whether $S'_k = S_k$.

8. If the check is correct, then the Security Module increases the counter corresponding to K_{Mumc} by the difference of the last two counter values

$C_k - C_{k-1}$,

where C_{k-1} is the UMC counter value read at the previous authentication. The terminal lets the service continue.

In the case of an unsuccessful authentication the service is aborted by the terminal and the counter in the Security Module is not updated for the last change ($C_k - C_{k-1}$). Hence, with this procedure the counter in the Security Module is updated if and only if two consecutive authentication procedures were successful.

NOTE 2: Thus if an authentication fails or can not be performed (e.g. because the user has removed the UMC) then C_k and hence $C_k - C_{k-1}$ can not be computed. The system/terminal operator can not claim any money for this final amount and is therefore at risk for this amount. Thus it is an advantage for the system/terminal operator to ensure that the terminal authenticates and decrements the UMC frequently, preferably for every metering event.

For subsequent authentications with changed counters it might be sufficient to use the same random for each authentication. In the case of enhanced security by repeated authentication to enlarge the number of response bits, it is necessary to use different random numbers.

5.3.6 Increase SM counter

For each Card Issuer the Security Module handles one, or more, counters summing up all units debited from cards of this Card Issuer. One of the important functions of the Security Module is to ensure, that the counter(s) for a certain Card Issuer are only increased by a certain value, if a signed message of an UMC is received, proving that the counter of the UMC is decreased by the same value. Therefore it is necessary that the Security Module is able to interpret the counter value for all Card Issuers for which it is handling cards.

For instance, if a Card Issuer issues different series of UMCs for which the bits on the UMCs represent different monetary values then different SM counters are necessary.

In this case, the selection of the SM counter shall be securely linked to the selected K_{Mumc} within the Security Module, to prevent an incorrect value being claimed. UMCs for which different bits represent different monetary values shall use a different K_{Mumc}.

5.3.7 Extract counter from the SM

As mentioned in the previous subclause the SM handles counters for each Card Issuer. These counters are extracted with a signature. The key used for this signature is the MAC key, chosen by the Card Issuer and not known in plain to the system operator. The billing between system operator and Card Issuer is based on these counters. The billing process is described in subclauses 4.7 and 5.4. Annex H describes an example method of MAC generation. A unique identification of the counter shall be used in the MAC calculation. This shall consist at least of the unique identity of the SM and the unique identity of the counter within the SM (e.g. containing the full path of the counter file). The use of this information shall be secured within the SM to ensure the correct identification is used.

5.4 Payment and settlement

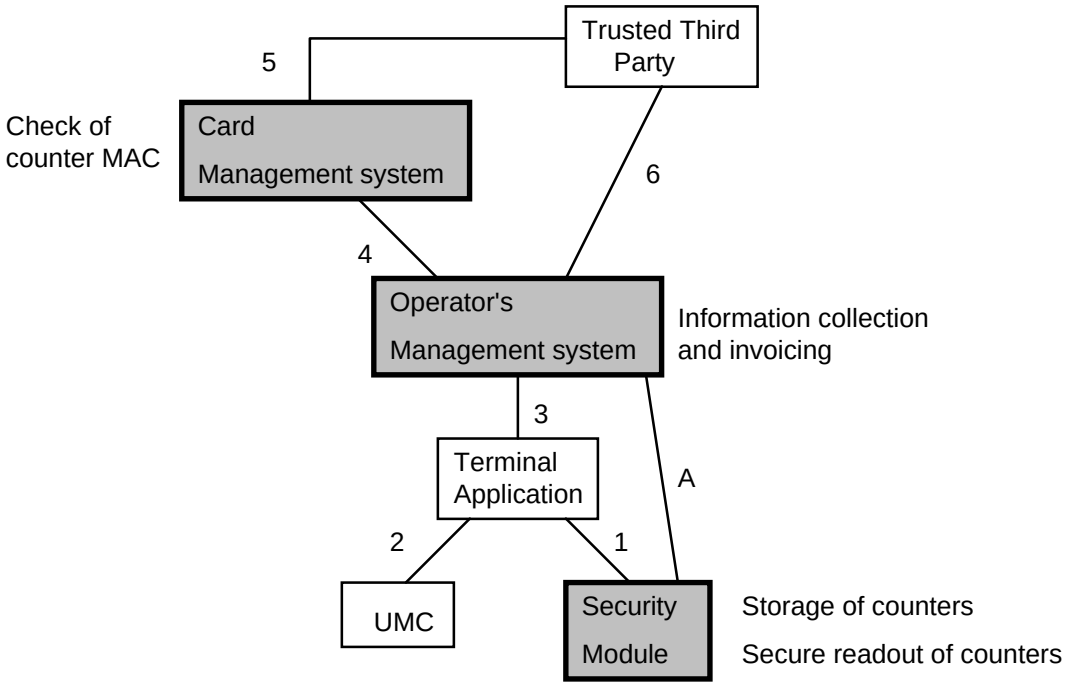


Figure 15: The billing and accounting interface model

For secure billing the boxes highlighted in figure 15 are relevant. In more detail the following steps take place for secure billing:

- The counters and MACs are periodically read out of the Security Module and transferred to a central billing system.

The Data elements "counter_value" and "MAC" are described in subclause 6.8.1.

- The billing system stores for each Security Module the data elements "counter_value" and "MAC" for the first day of a billing period and the data element "counter and MAC" for the current day.

- After the end of a billing period the two data elements "counter_value" and "MAC" for the first day of a billing period and the data elements "counter_value" and "MAC" for last day of a billing period are stored. The difference between the two counters is the consumed value of the billing period.

- The billing system adds all data elements "consumed values of a SM". That summary is the basis for a bill.

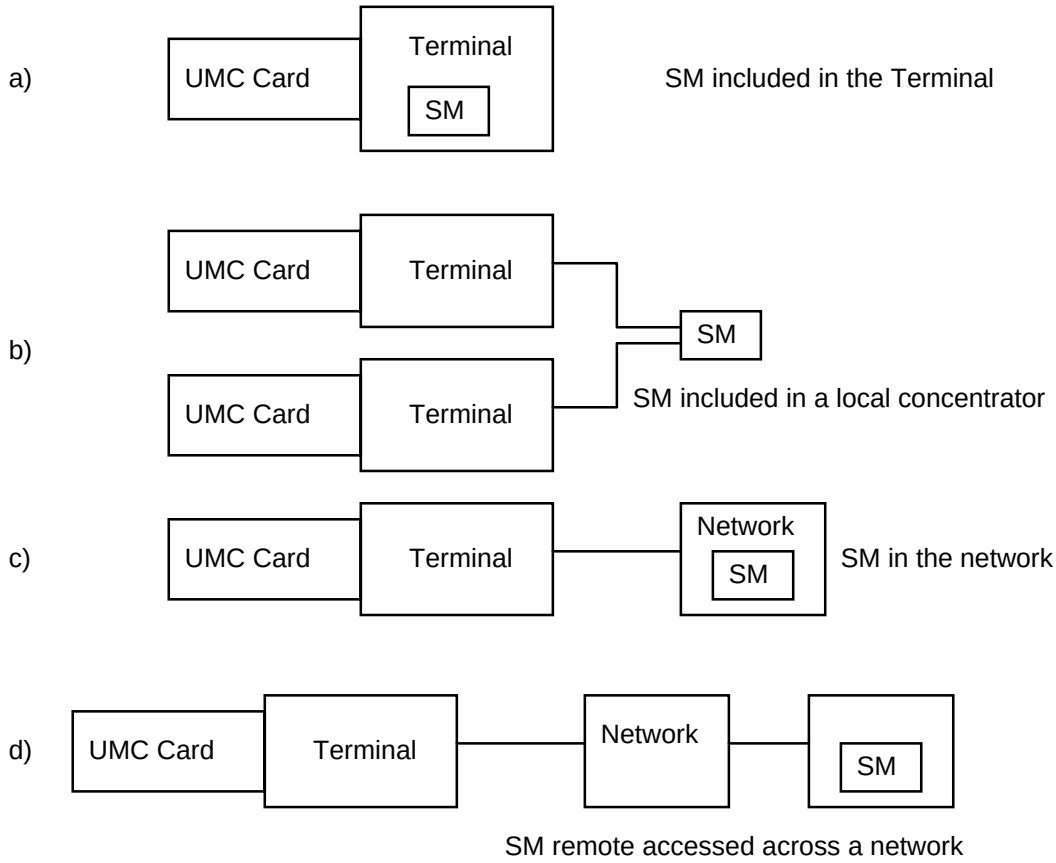
- The Card Issuer will receive all data elements "counter_value" and "MAC" for the first day of a billing period and the data elements "counter_value" and "MAC" for last day.
- The Card Issuer may check the MAC and then if correct, the summary minus an agreed amount is then paid to the operator.
- For further billing periods the start value will be the value of the last day of the foregoing period.

For the secure counter readout see subclause 5.3.7.

An alternative billing arrangement may be used by bilateral agreement between an operator and a Card Issuer.

5.5 Operational differences for network based SM

The Security Module can also be located outside of the terminal (see EN 726-7 [2]). Figure 16 (taken from EN 726-7 [2]) shows the 3 different possible configurations. This choice can be made for security reasons (e.g. to decrease the risk of SM theft) or for economic reasons (e.g. sharing an SM between different terminals). This leads to the need to consider further requirements on the SM to deal with other constraints and functions.



NOTE: These network based Security Module configurations are taken from EN 726-7 [2].

Figure 16: Network based SM configurations

Configuration a) is the one covered by other parts of this document. The main differences required if configurations b), c) or d) are adopted are covered in this subclause.

The major difference is that the SM is shared in order to manage different terminals, thus allowing a decrease in the total number of SMs required for a given number of terminals.

This leads to the need to consider "parallel" authentication processing of different cards in different terminals. This introduces the requirement for the management of a **context mechanism** allowing the creation and retrieval of the necessary environment for management of one card process amongst others.

Data elements to be stored in each context shall allow access to the relevant data, counter and key file(s).

The ability to perform parallel processing also implies that the SM has to process faster than for an SM dealing with only a single terminal. In order to reduce the processing time numerous card management specific commands (essentially **macro commands**) could be used.

Location of the SM in the network allows management of terminals of different operators by one SM (e.g. in the case of private terminal operators). In order to deal with this case and with the corresponding demands for secure settlement, it is necessary to provide the terminal operator with its own DF and necessary keys (i.e. at least the K_{load} and K_{MAC} keys).

As the data link between the UMC and the SM may not be continuously available during the provision of service period, the signature process by the UMC and SM verification could be made at different time periods. It is strongly recommended that in this case the terminals are authenticated by the SM to ensure that the correct DFs are used, and that the terminals can not be masqueraded.

6 Data elements

6.1 Data description technique

In this subclause Abstract Syntax Notation One (ASN.1) (ITU-T Recommendations X.680 and X.682 [14, 15]) is used purely as a means of describing the data elements in a succinct manner, yet precise enough for the purpose. Where ASN.1 is used it lists whether data elements are mandatory or optional, and gives guidelines for how they might be structured. The exact structure of the data elements is not mandated. The interface specifications do not contain all the detail that would be required to completely specify each interface in an interoperable manner. Complete interface protocols could be specified that included the data elements given here. Such specifications could also include other data elements to complete the protocol. ASN.1 could be used for these complete specifications and the standardized encoding rules (ITU-T Recommendations X.690 and X.691 [16, 17]) could be applied to obtain the bit patterns actually exchanged. However, the use of other data description techniques and encoding methods are not precluded by this document for these more detailed interface specifications.

6.2 System operator/Card Issuer interface (Interface 4)

The Card Issuer shall provide all the system operators, with which they have an agreement, information on UMCs (or UMC batches) which have been stopped because they have been reported stolen or are thought to be being used fraudulently (the blacklist).

```
Blacklist ::= SEQUENCE OF
SEQUENCE    {umc_issuer_id          UMC_Issuer_Id,
                                    . . .,
                                    SEQUENCE OF
             SEQUENCE    {CHOICE
                         {umc_batch_id      UMC_Batch_Id,
                          individual_Ids     SEQUENCE    {umc_id_first   UMC-id,
                                                          umc_id_last    UMC-id OPTIONAL},
                          date_and_time_of_issue          Date_and_Time       OPTIONAL,
                          initial_value                   Value               OPTIONAL,
                          initial_units                   Units               OPTIONAL,
                          date_and_time_of_stop           Date_and_Time       OPTIONAL,
                          stop_reason                     Stop_Reason         OPTIONAL,
                                                          . . .
                         },
                                    . . .
             }
```

The provision of Whitelist information (i.e. information on cards actually issued and in current circulation) may optionally be provided by the Card Issuer to system operators. The basic data elements for this information are specified in annex G.

System operators shall regularly provide each Card Issuer, with which they have an agreement, usage information that shall also have the status of an invoice against which the Card Issuer shall provide payment to the system operator at agreed time intervals. The basic data elements for this invoice information are specified below.

```
Invoice_info ::= SEQUENCE          {invoice_id              Invoice_Id          ,
                                    umc_issuer_id           UMC_Issuer_Id       OPTIONAL,
                                    operator_id             Operator_Id         OPTIONAL,
                                                            . . .,
                                    summary                 SEQUENCE
                                   {period_start_date_and_time      Date_and_Time,
                                    period_end_date_and_time        Date_and_Time,
                                    total_usage                     Units           OPTIONAL,
                                    value_claimed                   Value,
                                                                    . . .
                                   },
                                                            . . .,
                                    detail                  SEQUENCE OF
                SEQUENCE           {terminal_id                     Terminal_id         OPTIONAL,
                                    sm_id                           SM-id,
                                    counter_id                      Counter_Id,
                                    umc_type                        UMC-Type,
                                    value_claimed                   Value,
                                    date_and_time_of_first_entry    Date_and_Time       OPTIONAL,
                                    initial_value                   Units               OPTIONAL,
                                    initial_value_challenge         OctetString         OPTIONAL,
                                    initial_value_mac               MAC                 OPTIONAL,
                                    date_and_time_of_last_entry     Date_and_Time       OPTIONAL,
                                    final_value                     Units,
                                    final_value_challenge           OctetString,
                                    final_value_mac                 MAC,
                                                                    . . .
                                   }
                                                            . . .
                                    }

Date_and_Time  ::=  UTCTime (specified in ITU-T Recommendation X.680 [14])
MAC            ::=  OctetString
Units          ::=  INTEGER
Value          ::=  CHOICE        {units              INTEGER,
                                   monetary_value      SEQUENCE        {amount     INTEGER,
                                                                        currency_id Currency_Id,
                                                                       . . .},
                                   . . .}
```

```
Currency_Id      ::=    PrintableString (SIZE 3)          DEFAULT "ECU"
```

- - The value shall be a 3 letter currency code taken from ISO 4127 [5]

The Card Issuer provides cryptographic keys in a secure manner. They are entered into the SM by means of the key management and maintenance process described in subclause 4.6.

```
Key&P_Download_Data ::= SEQUENCE
    {sm_Id              CHOICE  {sm_Id          SM_Id,
                                 sm_Id_list     SET OF SM-Id,
                                 all            NULL,
                                 . . .
                                 }              OPTIONAL,
     encrypted_keys     SET OF SEQUENCE
                          {key_type            ENUMERATED    {kload, kMAC, kMumc, kMcer, . . .},
                           version_no          INTEGER       OPTIONAL,
                           encrypted_key       BIT STRING,
                           mac                 BIT STRING,
                           . . .
                          },
    . . .
    }
```

It is recommended that at least 3 versions of K_{Mumc} are supported. Optionally if certificates are supported then it is recommended that at least 4 versions of K_{Mcer} are supported.

Additional data may need to be transferred between the Card Issuer and the system operator. For example, data to link keys, algorithms and counting methods. Details of how this data is transferred are not specified here.

6.3 Operator's Management System (OMS)/terminal application interface (Interface 3)

If the operating management system is owned by a different party from that owning the terminal then the interface between them shall fulfil the requirements of this document.

If they are both owned by the same party then implementation details are a local matter. However, even in this case the interface has to be capable of meeting the information exchange requirements of this document such that it is possible to confirm to the other interfaces and the required functionality.

The physical, electrical and lower layer protocol aspects of this interface are not constrained by this document. (They are for further study).

When communication is established between the Management system and the terminal, it is recommended that there shall be a mutual authentication exchange before any other data is transferred. If this fails for either party then that party shall abort the communication. Periodic re-authentication may take place, or a continuous authentication mechanism may be used. All information exchanged should be classified. Depending on the classification level, security may be applied to ensure the integrity and confidentiality of the data as appropriate to its classification.

Information to be conveyed in the direction Operator's Management System to Terminal Application:

request for a particular counter value	
request for all counter values	optional
read tariff table	optional
load tariff table	
read currency conversion table	optional
load currency conversion table	
request call records	optional
read blacklist	optional
load blacklist	
incremental update to blacklist	optional
download new key	
download application to SM	optional
download new software to the terminal	optional
load table of accepted cards	
acknowledge alarm information	

Information to be conveyed in the direction Terminal Application to Operator's Management System:

response to request for a particular counter value	
response to request for all counter values	optional
response to read tariff table	optional
response to load tariff table	
response to read currency conversion table	optional
response to load currency conversion table	
response to request call records	optional
response to read blacklist	optional
response to load blacklist	
response to incremental update to blacklist	optional

response to download new key

response to download application to SM optional

response to download new software to the terminal optional

response to load table of accepted cards

send alarm information

Additional optional information may be exchanged with either end initiating the transfer.

6.4 Terminal/UMC interface (Interface 2)

The physical and electrical aspects of this interface shall be as specified in ISO 7816-1, -2 and -3 (section 1 to 4.2.7) [6, 7 and 8].

NOTE 1: The synchronous protocol and command set are not fully specified in ISO 7816-3 [8]. Thus this International Standard is usually augmented by proprietary specifications. There are presently no publicly available specifications or standards that fully specify the UMC to terminal interface.

The UMC can not issue commands to the terminal; it can only respond to commands from the terminal. The terminal issues commands and associated data to the UMC by asserting combinations of electrical signals on the electrical contacts on the card (see ISO 7816-2 [7] and proprietary specifications).

In User configuration the minimum command set available is:

AUTHENTICATE;

RESET;

READ;

WRITE;

ERASE.

Data-elements handed over through this interface shall support the card recognition and card authentication processes. Optionally other processes may be supported.

Following data elements shall be interchanged to support authentication operations:

from UMC to terminal:

- UMC_data:
  - identification data ID: 64 bits, these are the first 64 bits of the UMC-memory area;
  - current counter C value (see annex D);
  - additional data D, also written on the card.

NOTE 2: The counting method and the location of the counter have to be determined from the card recognition process and data provided by the Card Issuer.

Following data elements may optionally be transmitted from UMC to the terminal to support other functions. It is possible that these data elements are part of the additional data D.

- Certificate CER, written on the card, used to validate the card (passive authentication): 16 bits;
- face value (i.e., the initial value on the card);
- expiry date;

- flag for counter value less than face value;
- empty flag;
- FND number;
- currency indicator;
- counter backup (i.e. an anti-tearing mechanism);
- other additional data;

from terminal to UMC:

- random R: 32 or 48 bits;

from UMC to terminal:

- signature S, calculated by the card as a function of (Kumc, UMC_data, R): 4, 8, 12 or 16 bits.

6.5 Data elements to be held in the UMC

```
UMC_DATA ::= SET    {
    icc_identification              ICC_Identification,
    icc_type                        ICC_Type                OPTIONAL,
    face_value                      Face_Value              OPTIONAL,
    personalisation_flag            BOOLEAN,
    counter                         Counter,
    anti_tearing_flags              Anti_tearing_flags,
    authentication_key_Kumc         Authentication_key_Kumc,
    first_use_flag                  BOOLEAN                 OPTIONAL,
    empty_flag                      BOOLEAN                 OPTIONAL,
    reload_flag                     BOOLEAN                 OPTIONAL,
    currency_id                     Currency_Id             OPTIONAL,
    master_key_identifier           Master_key_identifier   OPTIONAL,
    expiry_date                     Expiry_date             OPTIONAL,
    certificate                     Certificate             OPTIONAL,
    card_invalid_flag               BOOLEAN                 OPTIONAL,

    services                    SEQUENCE OF CHOICE      {
            fnd_option          SEQUENCE    {national_call_flag     BOOLEAN,
                                             fnd_number             FND_Number,
                                                                    . . .
                                            }
                        . . .
                                        }   OPTIONAL,

                        . . .
```

```
}
```

The data element "ICC_Identification" may include information to identify the type of IC, Card Issuer and other data such as a unique serial number.

6.6 Terminal/SM interface (Interface 1)

6.6.1 Preparation and selection

The terminal needs to select the information for a particular Card Issuer when issuing commands to the SM. Two different ways of achieving this are recommended.

a) The terminal may issue a "Select" command to the SM. All subsequent commands to the SM are governed by this select command until another "Select" command is issued. This mode of operation minimizes the amount of data transferred between the terminal and the SM.

b) The select information may be included with each command to the SM. This mode of operation allows commands concerning different UMCs to be interleaved.

The application may be selected using Application identifiers (registered under the procedures given in ISO 7816-5 [10]) or using information related to an UMC issuer.

6.6.2 UMC authentication

The minimum data to be exchanged between the terminal and the SM in order to achieve the UMC authentication is the following.

The terminal initiates the sequence by requesting the SM to generate a random number and by providing the necessary UMC data to the SM. Then the SM replies with the answer "OK" or "Not OK" only.

```
Authenticate_UMC_request ::=        SET
 {umc_data              SET         {umc_identification          Identification_Area,
                                     umc_counter                 Counter,
                                     umc_ad_data                 Data_Area     OPTIONAL,
                                     . . .
                                     }                           OPTIONAL,
 master_key_id          SET         {umc_issuer_Application_id   Umc_Issuer_Aid,
                                     master_key_File_Id          Master_Key_Ef_Id,
                                     master_key_number           Key_Num,
                                     . . .
                                     },
 random_number          Bit String,
 umc_signature          Bit String,
                        . . .
 }
```

The master key has to be diversified internally by the SM with the UMC information and an appropriate algorithm given as a parameter or implicitly chosen.

NOTE: This data element can be used in different ways and in either direction of data flow. No particular sequence of information transfer is implied by the structure of this data element.

```
Authenticate_UMC_response ::=     SET
    { result                      ENUMERATED {positive, negative, . . .}
                                  . . .
    }
```

6.6.3 Counter increase

The following set of data presents the minimum information to be exchanged between the terminal and the SM in order to achieve the secure increase of a counter.

The basic condition to achieve this function is to have already successfully authenticated the UMC (see previous subclause). This scheme supposes that the SM is able to calculate the difference between the previous value of the UMC Counter area and the current one. The security of this operation relies on the UMC signature. The SM replies "OK" or "Not OK".

```
Increase_Counter_request ::=      SET
    {umc_counter                  UMC_Counter_Value,
     counter_id                   Counter_Id                OPTIONAL,
     random_number                Bit String,
     umc_signature                Bit String,
                                  . . .
    }
```

NOTE: Some of these data elements may be stored in the SM and implicitly selected.

```
Increase_Counter__response ::=    SET
    { result                      ENUMERATED {positive, negative, . . .}
                                  . . .
    }
```

6.7 Data elements to be held in the SM

An SM is a device containing logically and physically protected secrets such as algorithms and related keys. The SM itself needs to operate in such away, and be the subject of strict security procedures such that unauthorized access is rendered infeasible or prohibitively costly compared to the benefits gained.

The following description is aimed at describing the minimum information set needed to achieve the required data security.

It shall not be possible to read certain secret information, particularly key values and algorithm parameters out of the SM.

```
SM_DATA ::=          SET
{key_set        SET   {
      key_Mttp       SEQUENCE      {key_Mttp_version      INTEGER          OPTIONAL,
                                    kMttp                  KEY,
                                    algo_Id                Algo_Id          OPTIONAL,
                                    algo_params            Algo_params      OPTIONAL,
                                . . . }                    OPTIONAL,
                . . .    },
 sm_identification    SET           {sm_id                 SM_id,
                                    sm_manufacturer_id     SM_MAN_id
                                . . . },
                                    . . .,
card_issuer_data      SEQUENCE OF SET
{card_issuer_id       SET    {umc_issuer_idUMC_issuer_id,
                              umc_issuer_Application_id    UMC_ISSUER_AId
                                . . . },
{key_set        SET   {
      key_ttp        SEQUENCE      {key_ttp_version       INTEGER          OPTIONAL,
                                    K_TTP                  KEY,
                                    algo_Id                Algo_Id          OPTIONAL,
                                    algo_params            Algo_params      OPTIONAL,
                                . . . },
      key_load       SEQUENCE      {key_load_version      INTEGER,
                                    kload                  KEY,
                                    kl_algo_Id             Algo_Id          OPTIONAL,
                                    kl_algo_params         Algo_params      OPTIONAL,
                                . . . },
      key_mac        SEQUENCE      {key_mac_version       INTEGER,
                                    kmac                   KEY,
                                    km_algo_Id             Algo_Id          OPTIONAL,
                                    km_algo_params         Algo_params      OPTIONAL,
                                . . . },
                                    - - other keys could be added
                . . .    }
key_Mumc              SEQUENCE OF SEQUENCE
```

```
                    {key_Mumc_version          INTEGER,
                     kMumc          KEY,
                     kMu_algo_Id                Algo_Id          OPTIONAL,
                     kMu_algo_params            Algo_params      OPTIONAL,
                . . .  },
key_Mcer        SEQUENCE OF SET
                    {key_Mcer_version          INTEGER,
                     kMcer          KEY,
                     kMc_algo_Id                Algo_Id          OPTIONAL,
                     kMc_algo_params            Algo_params      OPTIONAL,
                . . .  }        OPTIONAL,
umc_param       SEQUENCE    {length_of_signature   INTEGER,
                     number_of_loops            INTEGER
- - Number of challenge response loops to achieve the required security strength,
                     abacus_type                ABACUS           OPTIONAL,
                     . . .      }    OPTIONAL,
umc_counter     SEQUENCE OF
                SET  {counter_id    Counter_Id,
                     type_of_counter            INTEGER          OPTIONAL,
                     counter   Bit String (SIZE counter_size),
                . . .  },
. . .       },
. . .
}
```

If the SM complies with EN 726-7 [2], the keys may be stored in the key files EF_{KEY_MAN} or EF_{KEY_OP}. In this case the key file EF_{KEY_MAN} or EF_{KEY_OP} is mandatory for the MF and each DF. It is used to manage the memory as specified in EN 726-7 [2].

The data element key_ttp is mandatory for initial key download, but may optionally be replaced by the element key_load after initial key download has taken place.

It is recommended that at least 3 versions of K_{Mumc} are supported. Optionally if certificates are supported then it is recommended that at least 4 versions of K_{Mcer} are supported.

The data field "sm_identification" shall be sufficient to uniquely identify a SM. The data field "sm_manufacturer_id" shall uniquely identify a SM manufacturer.

The values for the data elements "counter_id" and/or "type_of_counter" imply the value of each bit of the "counter" value (i.e. the currency used). A different combination shall be used for each currency used by a Card Issuer.

```
ABACUS ::= ENUMERATED {Method_A, Method_B, Method_C1, Method_C2, Method_D1, Method_D2,
Method_E, . . .}
```

See annex D for the specification of each method.

If the Security Module covers requirements of EN 726-7 [2] then the data elements could be presented as follows:

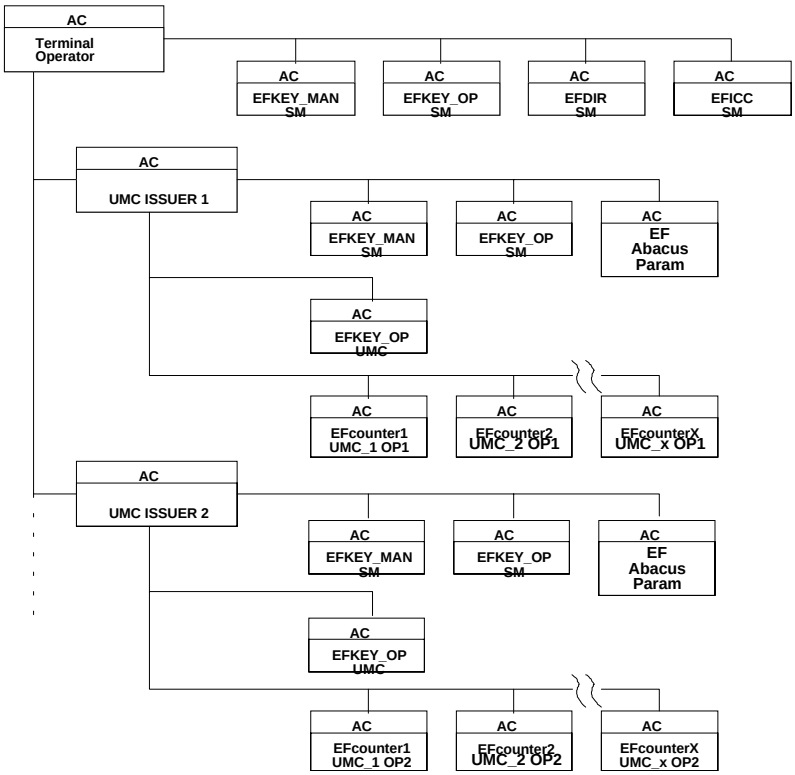


Figure 17: Example Security Module file map

6.8 SM/system interface (Interface 1 or A)

If Interface A is implemented then these data elements are passed over that interface, otherwise they flow over Interface 1 (see figure 1).

6.8.1 Counter extraction with a MAC

The following set of data presents the minimum information to be exchanged between the terminal and the SM in order to achieve extraction of a secured counter with an associated MAC.

```
MAC_Counter_request ::=          SET
    {card_issuer_id      Card_Issuer_Id                OPTIONAL,
     sm_id               SM_Id                         OPTIONAL,
     counter_id          Counter_Id                    OPTIONAL,
     mac_key_num         Key_Num                       OPTIONAL,
     initial_value       Initial_Value                 OPTIONAL,
     . . .
    }

MAC_Counter_response ::=          SET
    {card_issuer_id      Card_Issuer_Id                OPTIONAL,
     sm_id               SM_Id,
     counter_id          Counter_Id,
     mac_key_num         Key_Num                       OPTIONAL,
     initial_value       Initial_Value                 OPTIONAL,
     counter_value       Bit String (SIZE counter_size) OPTIONAL,
     mac                 Bit String                    OPTIONAL,
     result              ENUMERATED {positive, negative, . . .},
                         . . .
    }
```

The Security Module shall use the SM_Id and Counter_Id, plus any "initial value" on the request, as initial values to be fed into the MAC computation.

A globally unique identification of that counter (such as a globally unique SM identifier ("SM-Id") plus a local counter identifier) shall be included in the MAC calculation, plus optionally an "Initial_Value" that includes at least a random number or sequence number.

A unique identification of the counter shall be used in the MAC calculation. This shall consist at least of the unique identity of the SM (SM_Id) and the unique identity of the counter (Counter_Id) within the SM (e.g. containing the full path of the counter file). The use of this information shall be secured within the SM to ensure the correct identification is used.

6.8.2 Download a key

The following set of data represents the minimum information to be exchanged between the terminal and the SM to download a new key into an existing key location.

The key to be loaded is enciphered and signed to achieve both confidentiality and authenticity during the communication. This is performed by a strong enough algorithm and a key (K_{load}) present both in the Card Issuer management system and in the SM.

The SM shall not disclose any information which could be used for cryptographic analysis other than the final binary answer "OK" or "Not OK" and status messages as specified in EN 726-3 [1].

```
Download_Key_request ::=        SET
    {ciphered_key               Bit String,
     mac                        Bit String              OPTIONAL,
     key_location_num           Key_Location_Num,
     key_file_id                Key_File_Id,
     key_to_change              Key_Num                 OPTIONAL,
     key_version_number         Key_Version_Number      OPTIONAL,
                                . . .
    }
```

If the explicit MAC field is omitted, then some other mechanism shall be used to ensure the integrity of the key, such as redundancy in the key.

```
Download_Key_response ::=       SET
    { result                ENUMERATED {positive, negative, . . .}
                            . . .
    }
```

7 Compliance with the terms of this document

This document specifies a system architecture for achieving interoperability of pre-paid memory type smart cards (known as User Memory Cards or UMCs). That is various entities can issue these cards and they can be used in terminals operated by various other entities. For this system to work it is necessary for each card issuing entity to have a technical and commercial agreement with one, or more, entities operating terminals. The cards issued by an issuer can only be used in the terminals operated by entities with which the issuer has such an agreement. This needs to be made clear by Card Issuers to their customers. Thus the system architecture specified provides for limited interoperability rather than universal interoperability.

This architecture could be applied by a single commercial entity that performed all the different roles specified in this document. If the provisions of this document are faithfully applied then that entity would conform to the provisions of this document. However, the architecture defines different roles. Different commercial entities can perform the functions of the different roles and co-operate to fulfil the provisions of this document.

The main roles defined in this document are:

- system operator;
- Card Issuer;
- SM manufacturer;
- UMC user;
- Trusted Third Party (TTP).

An entity only needs to comply with the provisions of this document that apply to the role or roles that it claims to perform.

This document also specifies the system components and interfaces between these components that comprise the overall system architecture. An implementor of a system component specified in this document only has to comply with the functions and interfaces specified that apply to that system component. The main system components and interfaces between them are shown in figure 1.

This document only specifies the interfaces between system components at a relatively high level of abstraction. Other specifications may specify these interfaces and associated functions in more detail. These other specifications comply with this document if they provide for the functions and transfer of information specified in this document. They may

provide extra features not explicitly specified in this document. They shall not provide any features or information transfer explicitly or implicitly prohibited by this document.

NOTE: Other specifications may reference this document and claim compliance with it under the terms of this clause. These specifications may be international, regional, national or proprietary in nature.

Annex A (normative): Security requirements

A.1 General

The Partners want to achieve mutual acceptance of UMCs in their telecommunication systems. It is mandatory to use a secure transfer mechanism for the transfer of keys (see annex B) The creation of the keys is the task of a TTP.

Problem: **The level of trust of the TTP** shall **be evident**. The confidentiality and the quality of the Keys shall be ensured.

A.2 Security requirements on the TTP

In this clause the keys referred to are the initial loading key and any others required.

1) Secure generation, administration, and storage of keys for a Partner.

2) Secure hand over of keys to other parties e.g. secure loading of Keys into the Secure Devices (SD) - if used.

3) Secure loading of Keys into the Security Modules.

4) Additional security measures for secure data handling e.g. CHV and PIN creation and administration.

5) Physical protection of the TESA-7 algorithms.

To fulfil the role the TTP has to use software and hardware. This software and hardware, and their physical location are subject to the following requirements:

6) It shall be ensured, that only authorized persons are able to start the personalization software.

7) In general any unauthorized modification of data or keys shall be prevented. It shall be possible to recognize unauthorized actions, e.g. audit trails.

8) Access control mechanisms shall guarantee the confidentiality of keys and other protected data. The logical and/or physical storage of keys shall be secure.

9) The generation of keys shall be secure. The use of weak keys shall be prevented. It is important to verify random number generator.

10) Every personalization action shall be evident.

11) The software and hardware environments have to be considered.

To find out the **level of trust**, an **ITSEC Evaluation** of the TTP Software should be done. The minimum evaluation level should be at least **E2** and the minimum strength of mechanisms should be **medium**.

12) It shall be ensured that only trustworthy personnel is employed.

13) Protection of buildings and rooms against criminal actions shall be ensured.

14) In order to prevent unauthorized external access to the personalization software the personalization device should not be connected to an external network. If a network solution is necessary the physical and therefore also the logical connection of the personalization device should be limited to the secure personalization environment.

To find out the coverage of these minimum physical and organisational security requirements, an audit of the location of the TTP should be done.

A.3 Basic security requirements

The SM shall comply with the following security requirements.

Req 1 The SM operating system (mask) shall ensure the integrity and confidentiality of the initialization key(s) (e.g. K_{TTP}), the UMC algorithms and all the keys and data belonging to a Card Issuer. These keys, UMC algorithms, and confidential data shall remain secret in the SM and shall not be communicated outside the SM. Any program code downloaded shall comply with these requirement too.

Req 2 The mask shall ensure the integrity and confidentiality of each Card Issuer's secure and confidential storing of cryptographic keys and confidential data towards other Card Issuers. These keys and data shall remain secret in the SM. Any program code downloaded shall comply with these requirement too.

Req 3 Creation, deletion of a Card Issuer, cryptographic key loading and code downloading shall only be allowed after successful authentication.

Req 4 The UMC counter in the SM shall not be resetable, it shall only be possible to increment it. The only way to increment the UMC counter unit in the SM shall only be after two consecutive successful UMC authentications.

Req 5 The download of any additional software shall be secured under the control of the authorized TTP.

Req 6 The SM shall follow a one way life cycle.

Req 7 The SM shall provide physical and logical mechanisms to fulfil the former requirements.

It is recommended that the system operator provides means to prove to the outside world that the SM mask and all the downloaded software are authentic. The nature of this proof and the procedures to be followed are subject to bilateral agreement (see annex E item 21).

In order to meet these requirements, the mask, the additional downloaded software and the chip shall implement adapted security enforcing functions and mechanisms. Appropriate confidence in these functions will be needed.

To provide this confidence, it is mandatory that a security evaluation and certification (e.g. ITSEC [18]) shall be completed according to a bilateral agreement. The Target Of Evaluation shall include chip, mask and additional downloaded software. The security enforcing functions described in the security target shall fulfil the former requirements. Typically the evaluation level should be greater or equal to ITSEC E3 and the minimum strength of the mechanism high. Each piece of downloaded software shall be evaluated to the same requirements in order to be allowed to be downloaded.

The evaluation report (if produced) shall remain confidential because it contains some confidential information. Depending on bilateral agreement, it shall be possible that some parties exchange the present document.

Annex B (normative): Key transfer procedures

There are several possibilities for transferring keys. Two methods are described in this annex. Others may be adopted by bilateral agreement (see annex E).

B.1 The procedure for key transfer using a secure device

If a secure device is used, then the Card Management System provides an interface for the Secure Device. The tasks of the Secure Device are the following:

- In the secure device, the key K_{TTP} is pre-loaded by the TTP.
- The secure device receives the data field K_{load} in a secure way. It is highly recommended that the Card Management uses a special central system to handle the cryptographic data. This system should be located in a physically secured area.
- The secure device encrypts the data field K_{load} with the recommended algorithm and the data field K_{TTP} that has been previously installed in the SD.
- The encrypted K_{load} is then handed over to the Card Management system.
- The K_{load} is installed in the SD for the next steps.
- The data fields KM_{umc} and K_{MAC} are handed over to the SD. Optionally other keys (e.g. certification key K_{Mcer}).
- The SD encrypts the K_{Mumc} and K_{MAC} with the K_{load}.

The hand over of the keys K_{load}, KM_{umc} and K_{MAC} from the card management system might be realized in encrypted form, i.e. there is a further key handled by the Card Management System and the secure device to encrypt these keys at the interface.

Due to the fact, that these keys are handled in clear within the secure device, the secure device might transform additionally the format of the keys.

If a secure device is used, the procedure is shown in the following picture:

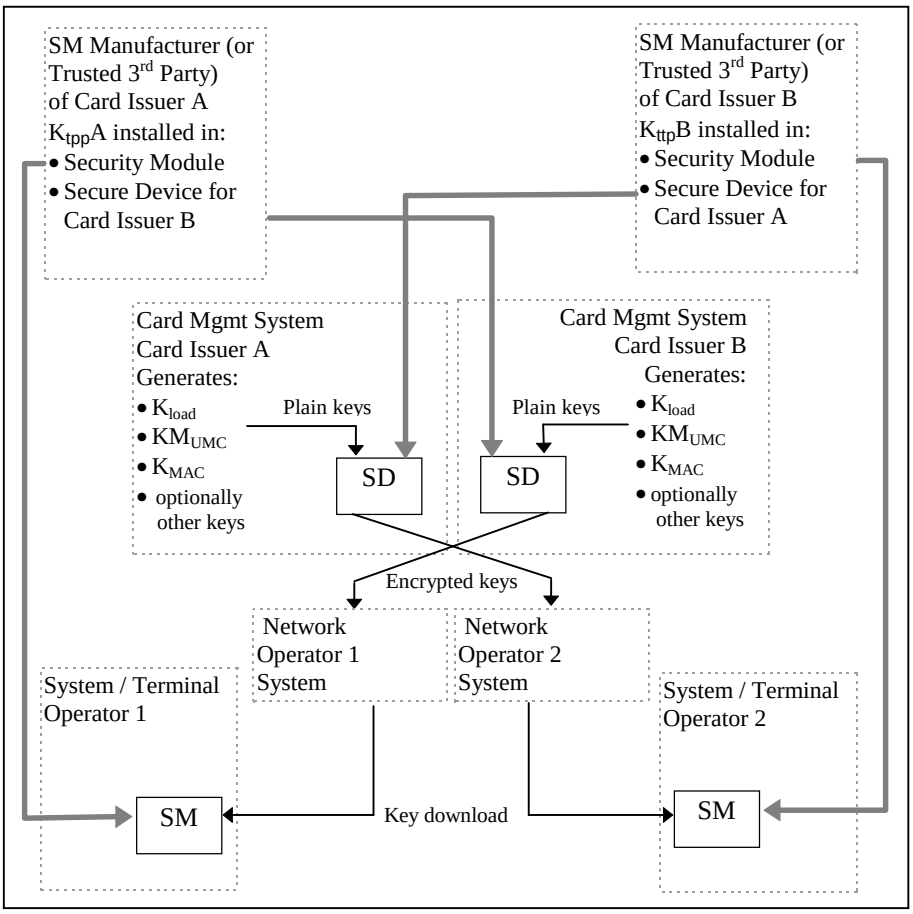


Figure 18: The key management procedures model

If a secure device is not used, then some other mechanism has to be agreed between the parties. Such alternative mechanisms are not specified in this document.

After the encryption of the relevant keys the encrypted data needs to be handed over to the Terminal Operator in the data format described in clause 7.

B.2 The procedure for key transfer using public key mechanisms

B.2.1 Principles

This subclause describes the principle of a public key scheme for K_{LOAD} initialization. See ISO/IEC CD 11770-3 [13].

The problem is: The Card Issuer (A) wants to load its K_{loadA} master key in operator's (B) Security Module (SM) while using a public key scheme.

A and B are identified by identifiers Id_A and Id_B. The application identities are supposed to be managed in a secure way: each application identity is unique.

The TTP has an asymmetric signature system (S_{TTP}, V_{TTP}). All the operators use card or special device to operate V_{TTP}. A and B know the public value V_{TTP}. This value is loaded in the SM or other device in a secure way (the integrity of V_{TTP} shall be guaranteed).

Notation: $S_A(m)$ denotes an entity A's private signature transformation applied to message m. No assumption is made on the nature of the signature transformation. In the case of a signature system with message recovery, $S_A(m)$ denotes the signature itself. In the case of a signature system with text hashing, $S_A(m)$ denotes the message m together with the signed hash value. (See ISO/IEC CD 11770-3 [13]).

SM_B has an asymmetric Encipherment/Decipherment functionalities (E_B, D_B). These functionalities are chosen by TTP and loaded by TTP in a secure way in the SM during personalization phase. These values cannot be modified.

Operator A has an asymmetric signature system (S_A, V_A). These functionalities are generated by TTP. A can compute with a chip card (strongly advised). This chip card remains in A premises.

TVP is a date and time dependent value.

To summarize:

Entity	Secret key	Public key
TTP	S_{TTP}	V_{TTP}
A	S_A	V_A
B	D_B	E_B

Data information knowledge:

Data	TTP	A	B
S_{TTP}	Yes	No	No
V_{TTP}	Yes	Yes	Yes
S_A	Yes	Yes	No
V_A	Yes	Yes	Yes
E_B	Yes	No	Yes
D_B	Yes	Yes	Yes

B.2.2 Public key initialization

1. TTP generates S_{TTP}, V_{TTP}, E_B, D_B, V_A and S_A.
2. TTP loads V_{TTP}, E_B and D_B in SM_B in a secure way.
3. TTP sends V_{TTP}, V_A and S_A to A in a secure way.

B.2.3 Certificates initialization

B.2.3.1 A certificate computation

1. A sends V_A and Id_A to TTP.
2. TTP computes a certificate on V_A and Id_A: $CER\_VA = S_{TTP}(V_A, Id_A, \text{other relevant data})$.
3. TTP sends to A CER_VA, A stores it.

B.2.3.2 B certificate computation

1. B sends E_B and Id_B to TTP.
2. TTP computes a certificate on E_B and Id_B: $CER\_EB = S_{TTP}(E_B, Id_B, \text{other relevant data})$.
3. TTP sends to B CER_EB, B stores it.

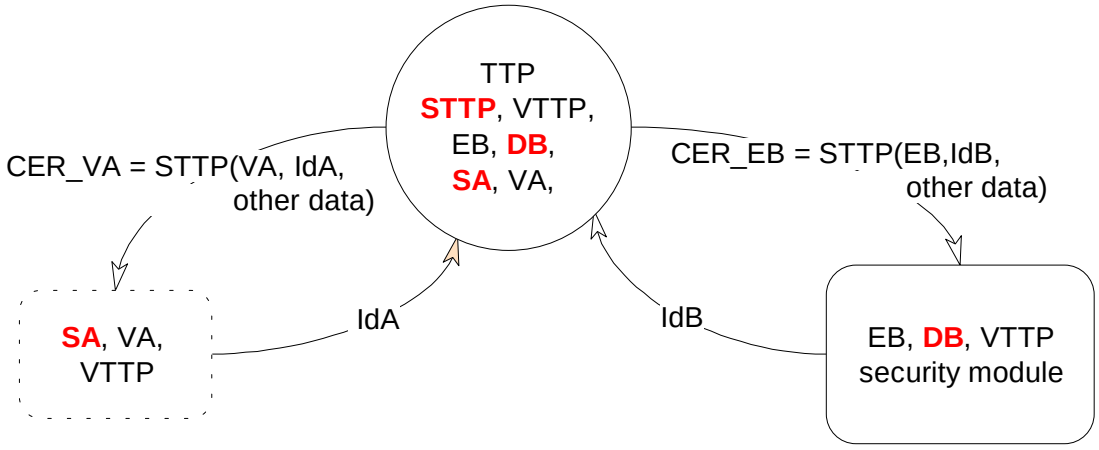


Figure 19: The public key initialization procedures model

B.2.4 Key initialization

B.2.4.1 A operations

1. A sends Id_A and CER_VA to B.
2. A receives CER_EB from B.
3. A checks CER_EB certificate using V_{TTP} and recovers Id_B and E_B.
4. A enciphers K_{loadA} and Id_A with B's public key E_B: BE = $E_B(Id_A, K_{loadA}$, other relevant data).
5. A signs this block with its secret key S_A: KT = $S_A(Id_A$, TVP, BE).

B.2.4.2 B operations

1. B sends Id_B and CER_EB to A.
2. B receives CER_VA from A.
3. B checks CER_VA certificate using V_{TTP} and recovers Id_A and V_A.
4. B checks KT certificate V_{TTP} and recovers BE.
5. B uses D_B to decipher BE, extracts K_{loadA} and accepts K_{loadA}.

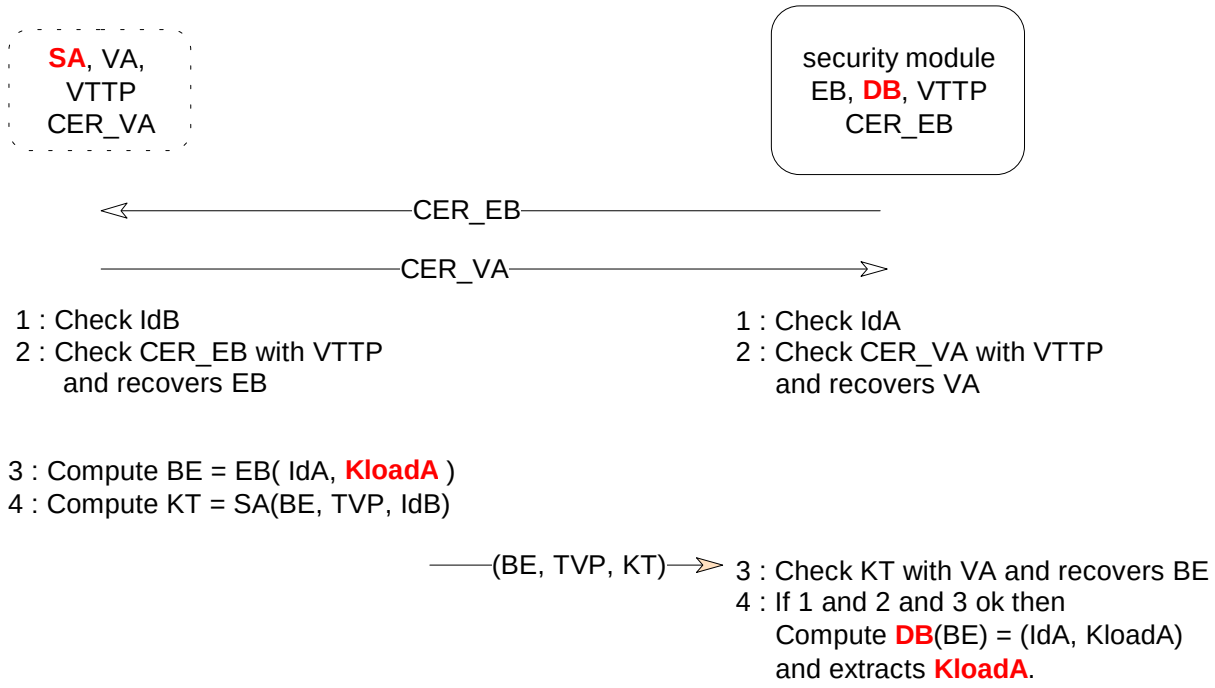


Figure 20: The public key certificate operations

With some modification, this scheme could be used to load the other keys.

Annex C (normative): UMC counter methodologies

C.1 Introduction

The UMC counter shall operate on an abacus principle, with the counter area consisting of a number of rows of 8 counter bits. Each row is assigned an octal value which is used in the counting process (dependent on the counter format).

Value may be changed by programming bits totalling the required value. To enable a flexible counting mechanism, a row of bits may be erased (reloaded) when programming a bit (reloading bit) in the next higher counter stage. The counter mechanism shall be such that it shall not be possible to increase the number of available units.

A number of different possible counter formats exist (see annex D). It is not mandatory for all Security Modules and terminals to support all counter formats, but it is recommended that they should be designed to be flexible enough to adapt to the listed formats (e.g. different electrical values).

Factors that need to be considered for UMC counter operation may include:

1) Size and location of the counter:

   Number of Bytes limited to 4 or 5

   Address of the highest Byte of the counter
   Address of the lowest Byte of the counter

2) Electrical value:

   E: value associated with an erased bit

   P: value associated with a programmed bit

3) Organization of the counting procedure: from the physical lower addresses to the higher (left hand) or from the physical higher addresses to the lower (right hand)

4) Use of each bit in the counter

   R Used for reloading (or personalization) only - not available for counting, to be set to the value P

   N Never used for counting - dedicated to application specific, test or personalization purposes

   x Available for counting, these bits are the non R and N bits

C.2 Size and location of the counter

This document supports two abacus families, containing 4 or 5 Bytes.

The 5 Bytes abacus counter is located in UMC bits 64 to 103. Each counter stage consists of 1 Byte. The octal value associated with the abacus stages are 8^0, 8^1, 8^2, 8^3 and 8^4.

The 4 Bytes abacus counter is located in UMC bits 64 to 95. The abacus is organized in 3 stages: the lowest and intermediate stages consist of 1 Byte. The third stage consists of 2 Bytes. The octal value associated with the abacus stages are 8^0, 8^1 and 8^2.

The maximum value which can be stored in a 5-Byte counter may be up to 32 768 units depending on the use of special bits.

The maximum value which can be stored in a 4-Byte counter may be up to 1 087 units depending on the use of special bits.

C.2.1 Electrical value

Depending on the technology, a programmed or erased bit can be represented as either a high electrical level (logical value 1) or a low electrical level (logic value 0). To cover both possibilities in the rest of this subclause, the following notation will be used:

P to represent a programmed bit;

E to represent an erased bit.

Depending on the convention used, either an erased or a programmed bit is considered to be available for counting.

If an erased bit is available for counting, the UMC counter counts down as units are used. The counter value relates to unused units.

If a programmed bit is available for counting, the UMC counter counts up as units are used. The counter value relates to the units that have been used on the UMC. For the rest of this document this is considered to be decreasing the remaining units on the UMC.

C.2.2 Counter orientation

The counter can be left or right oriented.

In this subclause the notation $[X]^k$ means field of k contiguous bits set to the value "X".

If the counter is left oriented, bits are programmed from the lower UMC addresses to the higher UMC addresses. The reloading R bits are located in the lower physical bit address of each abacus Byte.

In the case where no special bits are used in a left oriented counter, the only allowed bit patterns are:

$[P]^m[E]^n$ with (m+n) = 8 and m ≥ 1 (5-Byte counter and lower 2 stages of 4-Byte counter)

$[P]^m[E]^n$ with (m+n) = 16 Highest stage of 4-Byte counter)

Table C.1: Convention for left oriented 5-Byte abacus counter

	lowest address							highest address	x bit value
row 5	R	x	x	x	x	x	x	x	4 096
row 4	R	x	x	x	x	x	x	x	512
row 3	R	x	x	x	x	x	x	x	64
row 2	R	x	x	x	x	x	x	x	8
row 1	R	x	x	x	x	x	x	x	1

Table C.2: Convention for left oriented 4-Byte abacus counter

	lowest address							highest address	x bit value
row 4	x	x	x	x	x	x	x	x	64
row 3	x	x	x	x	x	x	x	x	64
row 2	R	x	x	x	x	x	x	x	8
row 1	R	x	x	x	x	x	x	x	1

If the counter is right oriented, bits are programmed from the higher UMC addresses to the lower UMC addresses. The reloading R bits are located in the higher physical bit address of each abacus Byte.

In the case where no special bits are used in a right oriented counter, the only allowed bit patterns are:

$[E]^m[P]^n$ with (m+n) = 8 and n ≥ 1 (5-Byte counter and lower 2 stages of 4-Byte counter)

$[E]^m[P]^n$ with (m+n) = 16 (Highest stage of 4-Byte counter)

Table C.3: Convention for right oriented 5-Byte abacus counter

	lowest address						highest address	x bit value
row 5	x	x	x	x	x	x	R	4 096
row 4	x	x	x	x	x	x	R	512
row 3	x	x	x	x	x	x	R	64
row 2	x	x	x	x	x	x	R	8
row 1	x	x	x	x	x	x	R	1

Table C.4: Convention for right oriented 4-Byte abacus counter

	lowest address						highest address	x bit value
row 4	x	x	x	x	x	x	x	64
row 3	x	x	x	x	x	x	x	64
row 2	x	x	x	x	x	x	R	8
row 1	x	x	x	x	x	x	R	1
NOTE: The 4-Byte abacus counter does not require reloading bits in rows 3 and 4.								

C.2.3 Special bits

The lower stage (row 1) and the upper stages (top 2 rows for the 4-Byte abacus, top row for the 5-Byte abacus) of the abacus counter may contain special bits used for purposes other than counting value or reloading (e.g. application specific, test or personalization purposes).

Use of such special bits in other stages of the counter is not permitted.

C.2.4 Higher abacus stages

Certain bits in the higher counter stages (top 2 rows for the 4-Byte abacus, top row for the 5-Byte abacus) may be always set to the programmed value P (e.g. fuse or test bits) and may not be used to represent counting value.

Other bits in the higher counter stages may be used for application specific purposes, and may be subject to modification by the terminal. These bits are not used in the counting procedure. In the case these bits are used, the terminal and Security Module shall not include these bits when calculating the UMC value.

C.2.5 Lower abacus stage

This stage can be used in one of the following two ways:

1. The reload bit R is required to be programmed to P by the terminal whenever the row is erased.

2. The bit which by default is the reload bit R in this stage is available for counting, and is not programmed to P by the terminal whenever the row is erased. In this case the bit at the other end of the row may be used for application specific purposes and may be subject to modification by the terminal. This bit is not used in the counting procedure. The terminal and Security Module shall not include this bit when calculating the UMC value.

Table C.5: Convention for lower abacus stage in left oriented abacus counter

	lowest address						highest address	x bit value
row 1	R or x	x	x	x	x	x	x	1
							or N	

Table C.6: Convention for lower abacus stage in right oriented abacus counter

	lowest address						highest address	x bit value
row 1	x or N	x	x	x	x	x	R or x	1

Annex D (normative): Methods of operation for the counter on the UMC

D.1 Introduction

This annex describes in detail the counting methods in current use. Card Issuers are encouraged to follow one of these conventions when decrementing the counter as determined from the card data. Additionally, every reload cycle shall be followed immediately by decrementing the left-hand bit of the reloaded row except in row 1 for methods A, D and E, and by decrementing the right-hand bit of the reloaded row except in row 1 for methods B and C.

D.2 Method A

The counter operates on the "abacus" principle. The counter is set to its required value by writing "0s" directly to the required addresses in the counter area. The maximum value that can be stored is 16 383 units. The convention shown in table D.1 shall be followed, i.e. the "left-hand" bit in each row shall always be written to "0" and shall not be used for the purpose of calculating the value in the counter.

Table D.1: Method A convention for decrementing value from the counter

	lowest address							highest address	Bit value
row 5	P	R	x	x	x	T	T	T	4 096
row 4	R	x	x	x	x	x	x	x	512
row 3	R	x	x	x	x	x	x	x	64
row 2	R	x	x	x	x	x	x	x	8
row 1	N/U	x	x	x	x	x	x	x	1

X = available for counting, i.e. erased by process of clearing the TLC.

P = personalization bit. Set to "0" in user configuration.

R = used for reloading only. Always set to "0". Not available for counting.

T = Reserved for test purposes. Never used for storage of value.

N/U = never used for storage of value. Always set to "0".

"1" = value stored (erased bit).

"0" = value destroyed (written bit).

The value storage bits in the counter area shall always be decremented from left to right, i.e. starting at the lowest address in a row and ending at the highest address in a row. Bits in row 5 have the highest value. They are equivalent to 8 bits in row 4, and so on. Bits in row 1 have one unit of value each.

The terminal shall follow these conventions when decrementing the counter. Additionally, every reload cycle shall be followed immediately by decrementing the left-hand bit of the reloaded row, so as to preserve the convention of not using the left-hand bits for counting.

D.3 Method B

The counter operates on the "abacus" principle. The counter is set to its required value by writing "1s" directly to the required addresses in the counter area. The maximum value that can be stored is 1 087 units. The convention shown in table D.2 shall be followed, i.e. the "right-hand" bit in row 1 and row 2 shall always be written to "1" and shall not be used for the purpose of calculating the value in the counter.

Table D.2: Method B convention for decrementing value from the counter

	lowest address							highest address	Bit value
row 4	x	x	x	x	x	x	x	x	64
row 3	x	x	x	x	x	x	x	x	64
row 2	x	x	x	x	x	x	x	R	8
row 1	x	x	x	x	x	x	x	R	1

X = available for counting, i.e. erased by process of clearing the TLC.

R = used for reloading only. Always set to "1". Not available for counting.

"0" = Erased bit.

"1" = Written bit.

The value storage bits in the counter area shall always be decremented from right to left, i.e. starting at the highest address in a row and ending at the lowest address in a row. Bits in row 4 and row 3 have the highest value. They are equivalent to 8 bits in row 2. Bits in row 2 are equivalent to 8 bits in row 1. Bits in row 1 have one unit of value each.

The terminal shall follow these conventions when decrementing the counter. Additionally, every reload cycle shall be followed immediately by decrementing the right-hand bit of the reloaded row 1 and row 2, so as to preserve the convention of not using these right-hand bits for counting.

D.4 Method C

The counter operates on the "abacus" principle. The counter is set to its required value by writing directly to the required addresses in the counter area (value depending on the chip type). The maximum value that can be stored is 12 287 units. The convention shown in table D.3 shall be followed, i.e. the "right-hand" bit in row shall always be written (value depending on the chip type) and shall not be used for the purpose of calculating the value in the counter.

Table D.3: Method C convention for decrementing value from the counter

	lowest address						highest address	Bit value
row 5	N/U	N/U	N/U	N/U	N/U	x	x	R	4 096
row 4	x	x	x	x	x	x	x	R	512
row 3	x	x	x	x	x	x	x	R	64
row 2	x	x	x	x	x	x	x	R	8
row 1	x	x	x	x	x	x	x	R	1

X = available for counting, i.e. erased by process of clearing the TLC.

R = used for reloading only. Bit always written. Not available for counting.

N/U = never used for storage of value. Bit always written.

Written bit convention

"0" = erased bit then "1" = written bit Method C1

or

"1" = erased bit then "0" = written bit Method C2

The value storage bits in the counter area shall always be decremented from right to left, i.e. starting at the highest address in a row and ending at the lowest address in a row. Bits in row 5 have the highest value. They are equivalent to 8 bits in row 4, and so on. Bits in row 1 have one unit of value each.

The terminal shall follow these conventions when decrementing the counter. Additionally, every reload cycle shall be followed immediately by decrementing the left-hand bit of the reloaded row, so as to preserve the convention of not using the right-hand bits for counting.

D.5 Method D

The counter operates on the "abacus" principle. The counter is set to its required value by writing "0s" directly to the required addresses in the counter area. The maximum value that can be stored is 32 768 units. The convention shown in table D.4 shall be followed, i.e. the "left-hand" bit in each row shall always be written to "0" and shall not be used for the purpose of calculating the value in the counter.

Table D.4: Method D convention for decrementing value from the counter

	lowest address							highest address	Bit value
row 5	P	R/x	x	x	x	x/T	x/T	x/T	4 096
row 4	R	x	x	x	x	x	x	x	512
row 3	R	x	x	x	x	x	x	x	64
row 2	R	x	x	x	x	x	x	x	8
row 1	N/U/x	x	x	x	x	x	x	x	1

X = available for counting, i.e. erased by process of clearing the TLC.

P = personalization bit. Set to "0" in user configuration.

R = used for reloading only. Always set to "0". Not available for counting.

x/T = May be used for test purposes, or for storage of value depending on the maximum value required.

N/U/x = Preferably not used for storage of value.

"1" = value stored (erased bit)

"0" = value destroyed (written bit) Method D1

or

"0" = value stored (erased bit)

"1" = value destroyed (written bit) Method D2

The value storage bits in the counter area shall always be decremented from left to right, i.e. starting at the lowest address in a row and ending at the highest address in a row. Bits in row 5 have the highest value. They are equivalent to 8 bits in row 4, and so on. Bits in row 1 have one unit of value each.

The terminal shall follow these conventions when decrementing the counter. Additionally, every reload cycle shall be followed immediately by decrementing the left-hand bit of the reloaded row, so as to preserve the convention of not using the left-hand bits for counting. An exception to this convention can be made for row 1 allowing acceptance of cards having this lowest bit with value stored.

NOTE: This method is a variant of Method A

D.6 Method E

The counter operates on the "abacus" principle. The counter is set to its required value by writing "0"s directly to the required addresses in the counter area. The maximum value that can be stored is 21 064 units. The convention shown in table D.5 shall be followed, i.e. the "left-hand" bit in row 5 shall always be written to "0".

Table D.5: Method E convention for decrementing value from the counter

	lowest address							highest address	
row 5	P	X	X	X	X	T	T	T	4 096
row 4	R	X	X	X	X	X	X	X	512
row 3	R	X	X	X	X	X	X	X	64
row 2	R	X	X	X	X	X	X	X	8
row 1	X	X	X	X	X	X	X	X	1

X = Available for counting, i.e. erased by process of clearing the TLC.

P = Personalization bit. Set to "0" in user configuration.

R = Used for reloading. Available for counting.

T = Reserved for test purposes.

"1" = Value stored (erased bit).

"0" = Value destroyed (written bit).

The value storage bits in the counter area shall always be decremented from left to right, i.e. starting at the lowest address in a row and ending at the highest address in a row.

Only in row 5 each erased bit is evaluated no matter whether the decrementing is done from left to right or vice versa.

Bits in row 5 have the highest value. They are equivalent to 8 bits in row 4, and so on. Bits in row 1 have one unit of value each.

The terminal shall follow these conventions when decrementing the counter. Additionally, every reload cycle shall be followed immediately by decrementing the left-hand bit of the reloaded row except in row 1.

Annex E (informative): Recommendations on items subject to bilateral agreement between Card Issuers and terminal/system operators

E.1 Introduction

This document specifies minimum functionality to achieve interoperability of UMCs in different terminal operators terminals. It is necessary to take into account the existing situation which has prevented the adoption of a single solution to some key aspects such as how the terminal can recognize the UMCs from different Card Issuers. Even if a single solution could be adopted it would still be necessary for each system operator to enter into a bilateral agreement with the Card Issuers that system operator decides to deal with. It is recommended that such bilateral agreements cover, at least, the following information:

E.2 Commercial matters

1) Official contact personnel, their roles and their contact information.

2) The means used to transfer information, and network addresses and protocols to be used if electronic communication is used.

3) The procedures to be followed if MACs on counter values are asserted not to be correct.

4) Who keeps the counter values, and for how long.

5) The method of splitting value between the system/terminal operator and the Card Issuer.

6) The method by which the Card Issuer will pay the system/terminal operator.

7) Whether blacklists and/or whitelists will be used.

8) Whether call records need to be supplied back to the Card Issuer.

E.3 Technical matters

9) The means of distinguishing UMCs from that Card Issuer from all other UMCs.

10) The monetary value of counter units, and how to distinguish if more than one is used by that Card Issuer.

11) The location of the counter on the card and its method of operation.

12) What data is included in the calculation of the card signature and the exact order of the data. Does the terminal application have to manipulate the data sent to the SM, or does the terminal application have to pass the card data to the SM unchanged.

13) The UMC authentication algorithm used (or how this UMC authentication algorithm is going to be loaded onto all the system operator's SMs or new SMs provided).

14) The number of UMC signature bits required to perform an increase of the SM counter.

15) The size and structure of the DF - EF suite required (or available) in the SMs.

16) Details of the number and function of the electrical contacts and the synchronous protocol used on the UMC to Terminal interface.

17) Use of any additional features such as Fixed Number Dialling or certificates.

E.4 Security matters

18) Which method will be used for secure transfer of Card Issuer's keys into the terminal/system operator's Security Modules.

19) Which security requirement specifications will apply to each of the secure components used in the system that are relevant for interoperability, such as the SM and the secure device (if used). The parties also need to consider their requirements for evaluating or proving that these security relevant components meet their specifications. The software (including firmware), hardware and processes all need to be considered.

The target for security needs to be specified (see annex A).

20) How the role of the TTP is to be fulfilled. The process for the management of keys needs to be specified and evaluated. This includes the Card Issuer, the TTP, and the system and terminal operators.

21) The nature of the proof and the procedures to be followed by the system operator to prove that the SM mask and all the downloaded software are authentic.

Annex F (informative): Fixed Number Dialling (FND)

Pre-paid phone cards can have an optional Fixed Number Dialling (FND) facility. When this option is activated on a card, the card can only be used to pay for calls to the fixed number. This fixed number is stored on the card. The terminal has to recognize that the card is for a fixed number only, to lock out the keypad and automatically dial the number.

This annex gives an example of how to implement Fixed Number Dialling.

At least 15 digit numbers need to be supported (see ITU-T Recommendation E.164 [19]). Country/geographic area codes are allocated by the ITU and are 1, 2 or 3 digits long. The norm is 3 digits for a country code though many European countries have 2 digit codes at present (e.g. the UK is 44).

The terminal shall determine whether the card is Fixed Number Dialling. If it is then the terminal locks out the key pad and executes the following procedure. The terminal first reads the national call flag.

If it is set it dials the number as on the card. This option is used for special national numbers (such as 0800 numbers). A warning needs to be printed on the card that the card will only work in the stated country.

If the national call flag is not set, then the full international number is stored on the card (up to 15 digits). The terminal determines whether the country code part is for the country it is in (as far as the network is concerned) or whether an international call is required. It adds the appropriate prefix and dials either just the national number or the international number according to its determination.

Binary Coded Decimal encoding of a 15 digit number takes 7 Bytes plus 4 bits (60 bits). It is recommended that Binary Coded Decimal encoding is used as straight binary encoding of the 15 digit decimal number requires 50 bits and thus does not save much memory on the card and is much harder for the terminal application to handle. It is recommended that the standard practice is adopted of setting unused digit positions to "F" in hex (i.e. "1111" in binary). Alternatively variable length encoding may be used with "F" in hex as the terminator. A full 15 digit number has no final value of "F".

For example assuming Binary Coded Decimal encoding on the card, Swiss telephone number (+41) 22 730 5994 would be stored on the card as: 41227305994FFFF or 41227305994F.

A terminal in the UK would dial:	00	41227305994
A terminal in Switzerland would dial:	0	227305994
A terminal in France would dial:	19	41227305994

It is recommended that the data is laid out on the UMC as in table F.1.

Table F.1: Recommended layout for Fixed Number Dialling data

	highest							lowest
row 8	N1	N1	N1	N1	R	R	B	F
row 7	N3	N3	N3	N3	N2	N2	N2	N2
row 6	N5	N5	N5	N5	N4	N4	N4	N4
row 5	N7	N7	N7	N7	N6	N6	N6	N6
row 4	N9	N9	N9	N9	N8	N8	N8	N8
row 3	N11	N11	N11	N11	N10	N10	N10	N10
row 2	N13	N13	N13	N13	N12	N12	N12	N12
row 1	N15	N15	N15	N15	N14	N14	N14	N14

F = Fixed Number Dialling flag.

B = National Call flag.

R = Reserved for future use.

Nn = Used to store the BCD encoded value of the n th digit of the number.

"1" = flag not set (bit not programmed).

"0" = flag set (bit programmed).

Using this example the N1 nibble would have the value "4", N2 the value "1", and so on.

Annex G (informative): Additional data elements

G.1 System operator/Card Issuer interface

The Card Issuer may provide system operators with information about valid cards in circulation (known as Whitelist information.

```
Whitelist ::= SEQUENCE OF
SEQUENCE    {umc_issuer_id          UMC_Issuer_Id          OPTIONAL,
                                    . . .,
                                    SEQUENCE OF
             SEQUENCE      {CHOICE
                            {umc_batch_id       UMC_Batch_Id,
                             individual_Ids     SEQUENCE    {umc_id_first   UMC-id,
                                                             umc_id_last    UMC-id OPTIONAL},
                             date_and_time_of_issue          Date_and_Time       OPTIONAL,
                             initial_value                   Value               OPTIONAL,
                             initial_units                   Units               OPTIONAL,
                                                             . . .
                            }
                                    . . .
             }
```

The Card Issuer may provide system operators with a response to invoice information sent by the system operator to the Card Issuer.

```
Invoice_info_response ::=  SEQUENCE
                    {invoice_id             Invoice_Id,
                     umc_issuer_id          UMC_Issuer_Id                   OPTIONAL,
                     operator_id            Operator_Id                     OPTIONAL,
                                            . . .,
 summary    SEQUENCE     {period_start_date_and_time     Date_and_Time,
                          period_end_date_and_time       Date_and_Time,
                          agreed_usage                   Units               OPTIONAL,
                          agreed_value                   Value               OPTIONAL,
                    . . . },
                     agreed_is_claimed      BOOLEAN,
                     diagnostic_code        Diagnostic_invoice_info_response    OPTIONAL,
                                            . . .,
```

```
        disputed_detail          SEQUENCE OF
    SEQUENCE    {sm_id                          SM-id,
                 umc_type                       UMC-Type,
                 date_and_time_of_first_entry   Date_and_Time    OPTIONAL,
                 date_and_time_of_last_entry    Date_and_Time    OPTIONAL,
                 usage                          Units            OPTIONAL,
                 value_claimed                  Value,
                 mac                            MAC,
            . . . }                             OPTIONAL
    . . . }
```

System operators may also regularly provide each Card Issuer, with which they have an agreement, call record information for either each use of an UMC, or first and last use of an UMC.

```
Call_record_info ::= SEQUENCE    {umc_issuer_id      UMC_Issuer_Id          OPTIONAL,
                                  operator_id        Operator_Id            OPTIONAL,
                                                     . . .,
                                  summary            SEQUENCE
                                 {period_start_date_and_time    Date_and_Time,
                                  period_end_date_and_time      Date_and_Time,
                                  sm_id                         SM-id,
                                  site_tel_no                   Telephone_number,
                                                                . . .
                                 },
                                  detail             SEQUENCE OF  Call_record,
                                                     . . .
                                 }

Call_record ::= SEQUENCE    {umc_type                   UMC-Type,
                             umc_id                     UMC-Id,
                             date_and_time_of_start     Date_and_Time    OPTIONAL,
                             date_and_time_of_finish    Date_and_Time    OPTIONAL,
                             initial_value              Units            OPTIONAL,
                             final_value                Units            OPTIONAL,
                             first_use                  BOOLEAN          OPTIONAL,
                             no_value_left              BOOLEAN          OPTIONAL,
                             expiry_date                Date             OPTIONAL,
                                                        . . .
                            }
```

Annex H (informative): MAC generation: an example

H.1 Introduction

This example for MAC generation uses the DES algorithm. However, the generation of MACs is not limited to this algorithm. Other algorithms such as TESA-7 could be used.

For each Card Issuer, there should be a key in the SM for MAC generation:

$$K_{MAC}$$

With this key a MAC is calculated for every counter together with some additional data. This data, secured with the MAC, is sent regularly to the background system, where it can be stored, or sent further to the co-operating partner. The co-operating partner can check the integrity of the counters by inspection of the MACs.

An example using this data might consist of:

(Random, Path of the SID (Security Module ID), SID, Path of the counter, Counter),
where the path information is a constant value.

The MAC is calculated by **DES** enciphering in **CBC-Mode**, where the last 64-bit enciphering block is used as MAC. Thus for the MAC calculation a 64-bit DES-key is applied.

H.2 Detailed description of the data

Before the MAC generation is started, an 8-Byte random is chosen, which is used as initialization value (IV). The data to be MACed consists of 5 blocks with 8-Byte each:

Block	status (viewed from Card Issuer)	content
1.	constant	path of EF_{ID} 4 Byte, INS, P1, P2, Le of READ Binary (1 Byte each)
2.	dynamic	first 8-Byte of the Security Module ID (SID)
3.	dynamic	9th and 10th Byte of the SID padded with 6 Byte zeros
4.	constant	path of $EF_{counter}$ 4 Byte, INS, P1, P2, Le of READ Record (1 Byte each)
5.	dynamic	counter (4 Byte) padded with 4 Byte zeros

H.3 Detailed description of the algorithm

Let b_1 to b_5 be the 5 data blocks defined above and IV the 8-Byte random value. Further define y := DES(K, x) to be the 8-Byte cipher text, received by a DES encryption of the 8-Byte plain text x with the DES-Key K. Let $\oplus$ define the XOR operation. Then the MAC is calculated as follows:

$$c_1 := DES[K_{MAC}](IV \oplus b_1)$$

$$c_i := DES[K_{MAC}](c_{(i-1)} \oplus b_i) \quad \text{for } i = 2, \ldots, 5$$

where the MAC is chosen to be:

$$MAC(IV, b_1, \ldots, b_5) := c_5.$$

Applying such a scheme, the first 8-Byte block of the input data could be manipulated in principle. But the first data block is constant, so there is nothing to be reached in doing so, and the manipulation can be detected, when the constant value is known.

Annex J (informative): Bibliography

J.1 Introduction

This annex contains references to relevant background material. Whilst they are not referred to directly in the text they have been retained in this annex because readers of this document may find them useful background reading on the operational use of smart or memory cards in terminal systems.

J.2 Informative references

- EN 726-1 (1994): "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 1: System Overview".
- EN 726-2 (1995): "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 2: Security framework".
- EN 726-4 (1994): "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 4: Application independent card related terminal requirements".
- EN 726-5: "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 5: Payment methods".
- EN 726-6 (1995): "Identification Card Systems - Telecommunication Integrated Circuit Cards and Terminals - Part 6: Telecommunication features".
- EN 1038: "Identification card systems - Telecommunications applications - Integrated circuit(s) card payphone".
- ISO/IEC 7812-1: "Identification cards - Numbering system and registration procedure for issuer identifiers - Part 1: ".
- ISO/IEC 7812-2: "Identification cards - Numbering system and registration procedure for issuer identifiers - Part 2: ".
- ISO 10202-4: "Financial transaction cards - Security architectures of financial transaction systems using Integrated Circuit Cards - Part 4: Secure Application Module".
- ISO 10202-5: "Financial transaction cards - Security architectures of financial transaction systems using Integrated Circuit Cards - Part 5: Use of algorithms".
- ISO/IEC JTC1/SC27 N 1082: "Guidelines for the use and management of Trusted Third Parties; Part 1: General overview".
- ETSI TCR-TR 028 (July 1995): "Network Aspects (NA); Security Techniques Advisory Group (STAG); Glossary of security terminology".

History

Document history		
V1.1.1	May 1997	Membership Approval Procedure MAP 9729: 1997-05-20 to 1997-07-18
V1.1.1	July 1997	Publication

ISBN 2-7437-1638-X
Dépôt légal : Juillet 1997