CEN

WORKSHOP

AGREEMENT

CWA 14890-2

May 2004

ICS 35.240.15

English version

Application Interface for smart cards used as Secure Signature Creation Devices - Part 2: Additional Services

This CEN Workshop Agreement has been drafted and approved by a Workshop of representatives of interested parties, the constitution of which is indicated in the foreword of this Workshop Agreement.

The formal process followed by the Workshop in the development of this Workshop Agreement has been endorsed by the National Members of CEN but neither the National Members of CEN nor the CEN Management Centre can be held accountable for the technical content of this CEN Workshop Agreement or possible conflicts with standards or legislation.

This CEN Workshop Agreement can in no way be held as being an official standard developed by CEN and its Members.

This CEN Workshop Agreement is publicly available as a reference document from the CEN Members National Standard Bodies.

CEN members are the national standards bodies of Austria, Belgium, Cyprus, Czech Republic, Denmark, Estonia, Finland, France, Germany, Greece, Hungary, Iceland, Ireland, Italy, Latvia, Lithuania, Luxembourg, Malta, Netherlands, Norway, Poland, Portugal, Slovakia, Slovenia, Spain, Sweden, Switzerland and United Kingdom.

EUROPEAN COMMITTEE FOR STANDARDIZATION
COMITÉ EUROPÉEN DE NORMALISATION
EUROPÄISCHES KOMITEE FÜR NORMUNG

Management Centre: rue de Stassart, 36 B-1050 Brussels

Ref. No.:CWA 14890-2:2004 D/E/F

Table of Contents

Foreword

This document is part of a series of standards for secure signature creation devices (SSCDs), developed to support the EU-directive on electronic signatures. It is dedicated to smart cards as an important representation of SSCDs (e.g. SIMs, USB tokens). The key issue of this document is to enable interoperability, so that smart cards from different manufacturers can interact with different kind of signature creation applications. The specification is applicable to smart cards supporting file system oriented applications as well as for smart cards supporting object oriented applications (e.g. Java applets).

This standard consists of two parts

- Part1 - “Basic Requirements” describes the mandatory specifications for an SSCD to be used in compliance with the ESIGN-G1 and F specifications.
- Part 2 - “Optional Features” describes additional features relevant for use with SSCDs.

The error handling of commands is out of scope of this document.

This is the second volume of the ESIGN specification for the Application Interface for smart cards used as Secure Signature Creation Device. It contains the specification of additional services, which are not required for the generation of a digital signature. They are, however, frequently used and typical in the context of digital signature applications.

1 Scope

This document specifies the application interface to SmartCards during the usage phase, used as Secure Signature Creation Devices (SSCD) to enable interoperability and usage of those cards on a national or European level.

The specification is based on the EU directive on electronic signatures [3] and takes into account E-SIGN documents and standards mentioned in the scope to be respected.

The functionalities described in this document map the general requirements of the EU directive to asymmetric techniques as required by the corresponding protection profile [6] and cover additional services, useful in signature environments.

The specification is separated into two parts.

Part 1 describes the mandatory services for the usage of Smart Cards as SSCDs. This covers the signing function, storage of certificates, the related user verification, establishment and use of trusted path and channel, key generation and the allocation and format of resources required for the execution of those functions and related cryptographic token information [17].

Part 2 describes optional services based on the same technology as available in signature devices. This covers key decipherment and client(card holder) server authentication, signature verification and related cryptographic token information [17].

2 References

The following standards contain provisions which, through reference in this text, constitute provisions of this part of ESIGN specification. At the time of publication, the editions indicated were valid.

[1] CWA 14890-1:2004, Application Interface for smart cards used as Secure Signature Creation Devices - Part 1 - Basic requirements

[2] CWA 14890-2:2004; Application Interface for smart cards used as Secure Signature Creation Devices - Part 2 - Optional Features

[3] EU Directive 1999/93/EC of the European Parliament and the council of 13 December 1999 on a Community framework for electronic signatures

[4] CWA 14170:2004, *Security Requirements for Signature Creation Applications*

[5] CWA 14171:2004, *General Guidelines for electronic signature verification*

[6] CWA 14172-1 to 8: 2004, *EESSI Conformity Assessment Guidance*

[7] ETSI SR 002 176: Electronic Signatures and Infrastructures (ESI); Algorithms and Parameters for Secure Electronic Signatures

[8] ISO 3166 : 1993, Codes for the representation of names of countries.

[9] ISO/IEC 7812-1 : 1993, Identification cards - Issuer identification - Part 1 : Numbering system.

[10] ISO/IEC 7816-3 CD Draft: 2003, Identification cards - Integrated circuit(s) cards with contacts - Part 3 : Electronic signals and transmission protocols.

[11] ISO/IEC FCD 7816-4: 2003 (Draft) Identification cards - Integrated circuit(s) cards with contacts, Part 4: Interindustry commands for interchange, Working draft dated 2003-09-16, ISO SC17 Document WG4N1750

[12] ISO/IEC FDIS 7816-5 : 2003, Identification cards - Integrated circuit(s) cards with contacts - Part 5 : Registration for application providers, FDIS dated 2003-09-15, ISO SC17 Document WG4N1751

[13] ISO/IEC FDIS 7816-6 : 2003, Identification cards - Integrated circuit(s) cards with contacts - Part 6 : Interindustry data elements for interchange - FDIS dated 2003-09-16, ISO SC17 Document WG4N1752.

[14] ISO/IEC FDIS 7816-8: 2003 , Integrated circuit(s) cards with contacts, Part 8: Commands for security operations. FDIS dated 2003-09-16, ISO SC17 Document WG4N1753.

[15] ISO/IEC FDIS 7816-9: 2003 (Draft), Integrated circuit(s) cards with contacts, Part 9: Interindustry commands for card and file management. FDIS dated 2003-09-16, ISO SC17 Document WG4N1754.

[16] ISO/IEC FDIS 7816-11: 2003(E), Integrated circuit(s) cards with contacts, Part 11: Personal verification through biometric methods. FDIS dated 2003-07-31, ISO SC17 Document WG4N1749

[17] ISO/IEC FDIS 7816-15: 2003 (Draft), Integrated circuit(s) cards with contacts, Part 15: Cryptographic Information Application, FDIS dated 2003-02-12

[18] ISO/IEC 8825-1: 1998, Information technology - ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER).

[19] ISO 9564-1:2002, Banking - Personal Identification Number (PIN) management and security - Part 1: Basic principles and requirements for online PIN handling in ATM and POS systems.

[20] ISO/IEC 9796-2:2002, Information technology - Security techniques -- Digital signature schemes giving message recovery - Part 2: Integer factorization based mechanisms (revision)

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

[22] ISO/IEC 9798 : 1998, Information technology - Security techniques - Entity authentication - Part 3: Mechanisms using digital signature techniques

[23] ISO/IEC 9979 : 1998, Information technology - Security techniques - Procedures for the registration of cryptographic algorithms.

[24] ISO/IEC 10116 : 1991, Information technology - Information technology - Security techniques - Modes of operation of an n-bit block cipher algorithm.

[25] ISO/IEC 10118-1 : 2000, Information technology - Security techniques - Hash functions - Part 1 : General.

[26] ISO/IEC 10118-2 : 2001, Information technology - Security techniques - Hash functions - Part 2 : Hash functions using an n-bit block cipher algorithm.

[27] ISO/IEC 11770-3 : 1999, Information technology — Security techniques — Key management — Part 3, Mechanisms using asymmetric techniques

[28] ISO/IEC 15408-1, Edition:1999-12, Information technology - Security techniques - Evaluation criteria for IT security - Part 1: Introduction and general model

[29] ISO/IEC 15408-2, Edition:1999-12, Information technology - Security techniques - Evaluation criteria for IT security - Part 2: Security functional requirements

[30] ISO/IEC 15408-3, Edition:1999-12, Information technology - Security techniques - Evaluation criteria for IT security - Part 3: Security assurance requirements

[31] ISO/IEC 19794-2: Biometrics – Biometric Data Interchange Formats – Part 2: Finger Minutiae (Working Draft)

[32] Wireless Identity Module Part: Security Version 12-July-2001, Wireless Application Protocol WAP-260-WIM-20010712-a, www.wapforum.org

[33] PKCS #1 v2.1: RSA Cryptography Standard, RSA Laboratories, June, 14, 2002, / www.rsasecurity.com/rsalabs/pkcs/pkcs-1/

[34] “New Directions in Cryptography” by Diffie, W. and M. E. Hellman, IEEE Transactions on Information Theory, Volume IT-22 Number 6, November 1976, pages 644-654.

[35] ISO/IEC 14888-1 : JTC1.28.08.01 “Digital signatures with appendix”

[36] ISO/IEC 14888-3 : JTC1.28.08.03 “Certificate-based mechanisms”

[37] EMV2000, Integrated Circuit Card Specification for Payment Systems Book 3, "Application Specification Version 4.0" December, 2000

[38] EuroCrypt 2002, paper by J.H. An, Y. Dodis and T. Rabin, "On the security of joint signature and encryption"

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

[40] DIN V66291-1, Chipcards with digital signature application/function according to SigG and SigV - Part 1: Application Interface, 15 December 1998, DIN Deutsches Institut für Normung e.V.: Berlin.

[41] DIN V66291-4, Chipcards with digital signature application/function according to SigG and SigV - Part 4: Basic Security Services, 17 October 2000, DIN Deutsches Institut für Normung e.V.: Berlin.

[42] “The Internet Key Exchange (IKE)” by Harkins, D. and D. Carrel, RFC2409, November 1998. URL: ftp://ftp.rfc-editor.org/in-notes/rfc2409.txt

[43] “Security Analysis of IKE's Signature-Based Key-Exchange Protocol. in Advances in Cryptology” by Canetti, R. and H. Krawczyk, Crypto 2002. 18-22 August 2002, Santa Barbara, CA:Lecture Notes in Computer Science Vol. 2045. Springer-Verlag. p. 143-161.

[44] "The Elliptic Curve Digital Signature Algorithm (ECDSA)" by D. Johnson, A. Menezes and S. Vanstone, International Journal of Information Security, 1 (1), pp. 36-63, 2001.

[45] IEEE P1363, "IEEE Standard Specifications for Public-Key Cryptography", 2000.

[46] FIPS PUB 198, “The Keyed-Hash Message Authentication Code (HMAC), 6 March 2002, *csrc.nist.gov/publications/fips/fips198/fips-198a.pdf*

[47] “Chosen ciphertext attacks against protocols based on the RSA encryption standard PKCS#1” by D. Bleichenbacher. Advances in cryptography - Crypto 1998

[48] ANSI X9.63, “Public Key Cryptography For The Financial Services Industy: Key Agreement and Key Transport Using Elliptic Curve Cryptography”,

[49] Internet-Draft : draft-ietf-cat-kerberos-pk-init-11.txt, March 2000

[50] RFC 1510: The Keberos Network Authentication Service (V5), September 1993

[51] RFC 2246: The TLS Protocol, version 1.0, January 1999

[52] RFC 3369: Cryptographic Message Syntax (CMS)

[53] RFC 2631: Diffie-Hellman Key Agreement Method, June 1999

[54] RFC 2785: Methods for Avoiding the "Small-Subgroup" Attacks on the Diffie-Hellman Key Agreement Method for S/MIME, March 2000

[55] NIST: FIPS Publication 180-1: Secure Hash Standard (SHS-1), May 1995.

[56] RFC 1321: The MD5 Message-Digest Algorithm by R.Rivest. MIT Laboratory for Computer Science and RSA Data Security, Inc., April 1992

[57] X.509: ITU-T (formerly CCITT). Recommandation X.509 (2000): Information technology - Open systems interconnection - The Directory: Public-key and attribute certificate frameworks

3 Abbreviations and notation

The abbreviations and notation is in accordance with those given in [6].

3.1 Abbreviations

APDU Application protocol data unit
ARR Access Rule Record
AT CRT for Authentication
C/S Client-Server
CA Certificate authority
CC Cryptographic checksum
CCT Cryptographic checksum template
CIA Cryptographic information appplication
CMS Card Management System
CRT Control Reference Template
CT Confidentality Template
D[key](msg) Decipherment of <msg> with <key>
DES-3 Data Encryption Standard (Triple-DES)
DF Dedicated File
DH Diffie-Hellman
DO Data Object
DS Digital Signature
DSI Digital Signature Input
DST CRT for Digital Signature
E[key](msg) Encipherment of <msg> with <key>
EF Elementary File
FCI File Control Information
FCP File Control Parameters
H_<id> Hash function using <id> algorithm
ICC Integrated Circuit(s) Card
ID Identifier.
IFD Interface Device
INS Instruction byte
KE Key Encipherment
KEI Key Encipherment Input Format
KID Key Identifier
MD5 Message Digest 5 (hash algorithm)
MF Master File
P1-P2 Parameter bytes
PK Public Key
PKI Public Key Infrastructure
PKCS Public Key Cryptography Standards
PSO PERFORM SEC. OPERATION
RCA RootCA
RFU Reserved for Future Use
RND Random number
SE Security Environment
SFI Short File Identifier
SHA Secure Hash Algorithm
SK Secret Key
SN Serial Number
SM Secure Messaging
SSCD Secure Signature Creation Device
SSD Secure Service Descriptor
UQB Usage Qualifier

4 Definitions

For the purpose of this part of the ESIGN specification, the following definitions apply. These definitions are in compliance with those given in the revision of ISO/IEC 7816-4.

4.1 C/S external authentication : The authentication of the server by the client. The client is regarded as the combination of the PC and the ICC. This external authentication is out of the scope of this specification.

4.2 C/S internal authentication : The authentication of the client by the server. The client is regarded as the combination of the PC and the ICC.

4.3 IFD: device or entity that belongs to the external world (outside the ICC).

4.4 secured channel : A communication link between the ICC and a security module (possibly also an ICC) that provides authenticity and/or integrity and/or secrecy.

5 Additional Service Selection

Additional services are typically used in the context of applications that use digital signatures.

A well known additional service is the **client/server authentication**. In this case, the ICC is used as a crypto toolbox, e.g. in order to encrypt a challenge with a private key, being stored in the ICC. This is particularly helpful in applications, where a tamper resistant device is required for client/server authentication. An ICC does have the tamper resistant quality and may therefore be used efficiently to support the application in this context.

A **document decryption** is another known service which may be performed by the IFD. A terminal application receives a document, typically encrypted with a symmetric key. The symmetric key is also encrypted with another public key. The ICC contains the appropriate private key, deciphers the symmetric key and returns it to the terminal application.

While the typical usage of a signature card is the generation of a digital signature, an application might want to verify a signature with a public key, being stored in the ICC. In this case an additional service is invoked for the **signature verification**.

Additional services provided in the ICC mandate the existance of an appropriate security environment. Associated security environments are described in the Annex B. “(informative) Security environments” on page B-42.

In addition to the descriptional information found in DF.CIA(see 16 “Cryptographic Information Application” in Part 1 page 16-122) information might be required that can be presented in Security Service Descriptors. The concept of security service descriptors is described in the Annex A. “Security Service Descriptor Templates (normative)” on page A-36.

A user verification may be required prior to the usage of additional services. The password for this user verification shall be different from the password used for the signature generation. This is to maintain the purpose of the signature generation password for the sole purpose of a ‘declaration of will’ in the case of a signature generation.

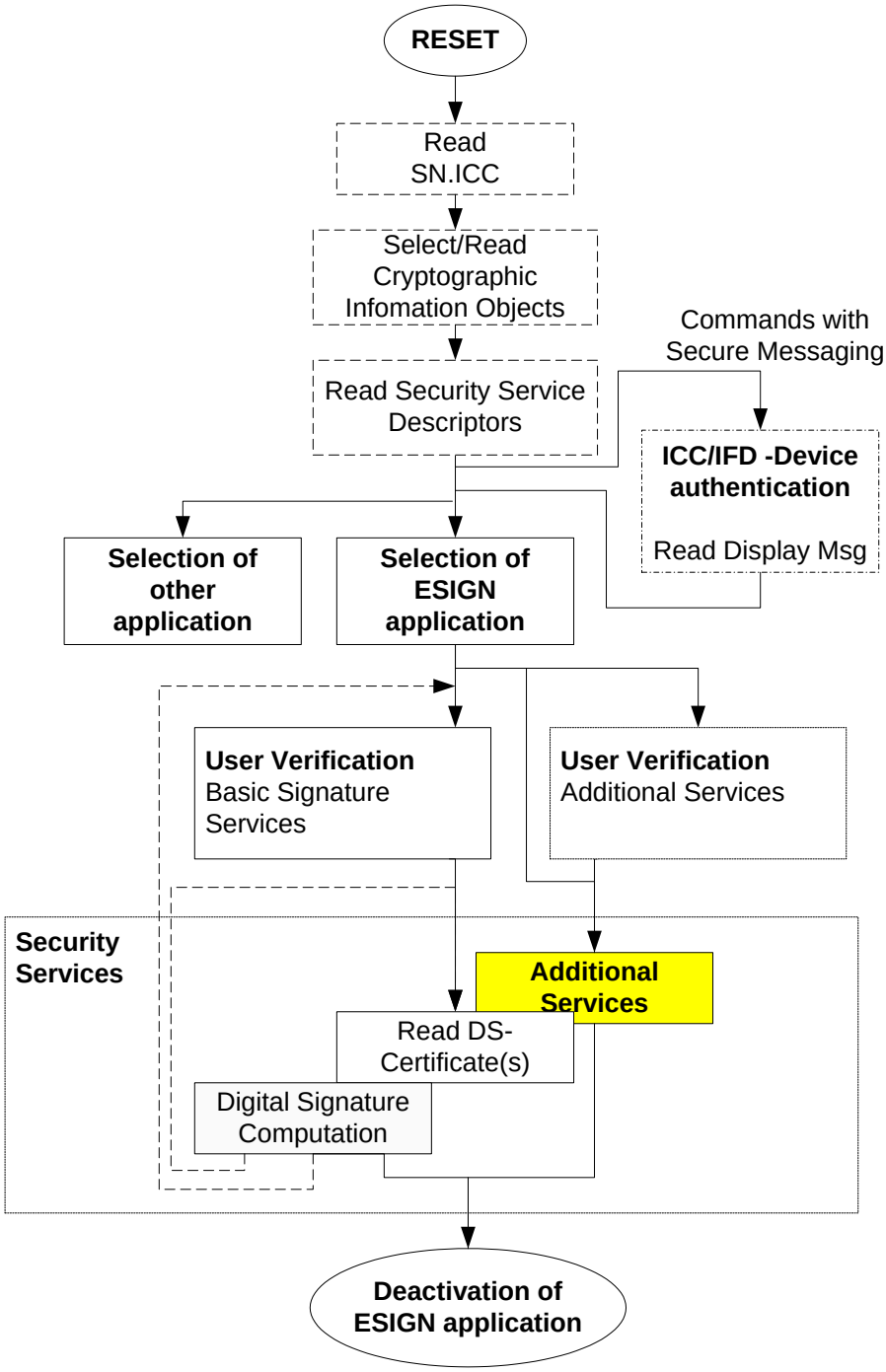


Figure 5-1. Example of additional service selection

Figure 5-1 shows the selection of additional services in the context of the ESIGN application. User verification might be required for some of the additional services. The detailed access conditions are described in the appropriate security environments.

6 Client/Server Authentication

For proving access rights to components such as servers, a PK based authentication procedure has to be performed. Such client/server Authentication (see “C/S internal authentication :” on page 4-11) is a process, being independent from the requirement of a device authentication.

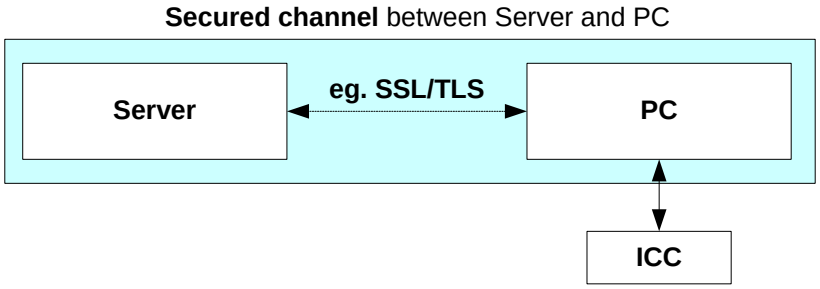


Figure 6-1. Example of client/server authentication

In the above example client/server authentication establishes a secured channel between a remote server and a PC. The ICC will be used as a cryptographic toolbox in order to provide the cryptographic functionality to the PC.

This specification does not support the authentication of the server (“C/S external authentication :” on page 4-11). The server’s certificate as well as the server protocol is application specific and therefore out of the scope of this document.

6.1 Client/Server protocols

This specification covers only the case, where the card performs a digital signature computation applying the private key for authentication in a COMPUTE DIGITAL SIGNATURE (INTERNAL AUTHENTICATE) command to the authentication input contained in the data field of the command after formatting the input.

The key pair used for client/server authentication shall be different from the device authentication keys and signature generation keys respectively. The public part of this key pair, stored with the distinguished name of the cardholder is certified by a certificate (typically X.509). Such a certificate is not interpreted by the ICC.

Relevant authentication procedures are e.g.

- the PK Kerberos protocol (for logon authentication)
- the SSL/TLS protocol
- the WTLS protocol.

All the above protocols base on the same cryptographic algorithms. In particular they are all using PKCS.1 padding format in the case of RSA. Therefore this specification describes only the PKCS #1 padding.

6.2 Steps preceeding the client/server authentication

The steps preceeding a client/server authentication are application specific. Hence this specification does not mandate the existence of those steps.

The access condititions proposed in Annex B. “(informative) Security environments” on page B-42 specify a user verification as a mandatory step prior to client/server authentication.

The reference to the password to be used for user verification in the context of client/server authentication is described in the information of DF.CIA.

6.3 Padding format

In case of RSA, the authentication input T is formatted according to PKCS #1, Version 2.1, Chapter 9.2.1 “EMSA-PKCS1-v1-5”.

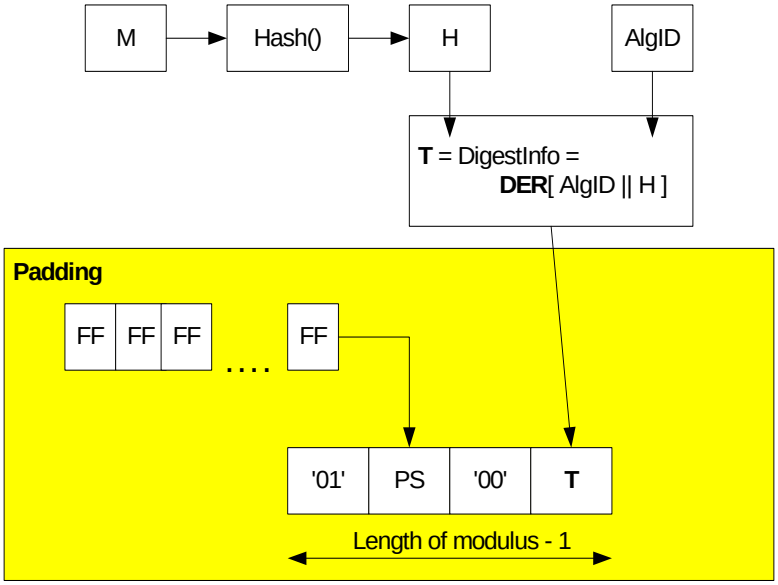


Figure 6-2. PKCS encoding of message M

PS is an octet string consisting of octets with value 'FF' (length ≥ 8). Due to security reasons the digest info T shall be smaller or equal 40 % of the length of the modulus. The formatted octet string shall consist of k-1 octets where k is the length in octets of the modulus of the private key for authentication.

6.4 Execution flow

Table 6-1 shows the execution flow of the client/server authentication. This specification covers only the internal authentication.

Stage	Step	IFD	Trans-mission	ICC
1	-	... preceeding steps(e.g. user verification) that are application specific for client/server authentication	→ ←	application specific protocol flow
INTERNAL AUTHENTICATION - server(IFD) authenticates client (ICC)				
2	1	READ BINARY of certificate (typically X.509)	→ ←	certificate
	2	MSE:SET SK.CH.AUT	→ ←	
	3	COMPUTE DIGITAL SIGNATURE or INTERNAL AUTHENTICATE	→ ←	
Client (ICC) is authenticated to server (IFD)				
3	-	... subsequent steps that are application specific to present client/ server authentication	→ ←	not specified in this document

Table 6-1. Client/Server authentication flow

6.4.1 Step 1 - Read certificate

If the IFD does not already have the clients (ICC) authentication certificate, it may read it from a global data file EF.C.CH.AUT. The client's certificate C.CH.AUT may contain public algorithm quantities which depend on the client/server authentication algorithm.

Execution Flow

```
READ BINARY( INS = 'B0'        // READ BINARY
             P1 = '85',        // Example: SFI = 05
             P2 = '00'         // offset = 0
             Le = '00')        // all data

Response = plain data as stored in EF.C.CH.AUT
```

The command and response APDUs are described in ISO/IEC 7816-4, chapter 7.2.3.

Note 1: Unless the READ BINARY selects the file using a Short File Identifier (SFI), an appropriate SELECT(EF) command must be executed prior to reading the file.

6.4.2 Step 2 - Set signing key for client/server internal authentication

The MSE command sets the key ID of the ICC's client/server private authentication key to be used for the following computation of the digital signature

Execution Flow A:

```
MSE:SET( INS = ‘22’          // MANAGE SEC ENV
         P1 = ‘41’           // computation : SET
         P2 = ‘B6’           // digital signature DST
         data=CRT data field with KeyRef(SK.CH.AUT)
```

The command and response APDUs are described in ISO/IEC 7816-4, chapter 7.5.11.

The structure and coding of the key reference data is specified in section 10.2.1 “CRT DST for selection of ICC's private client/server auth. key” on page 10-34.

Alternatively the INTERNAL AUTHENTICATE command can be used

Execution Flow B:

```
MSE:SET( INS = ‘22’          // MANAGE SEC ENV
         P1 = ‘41’           // computation : SET
         P2 = ‘A4’           // AT
         data=CRT data field with KeyRef(SK.CH.AUT)
```

The command and response APDUs are described in ISO/IEC 7816-4, chapter 7.5.11.

The structure and coding of the key reference data is specified in section 10.2.2 “CRT AT for selection of ICC's private client/server auth. key” on page 10-34

6.4.3 Step 3 - Internal authentication

The IFD sends a challenge in the data field of the INTERNAL AUTHENTICATE command. The ICC computes a signature over the challenge by using its private key, that was selected in the preceeding step.

Execution Flow:

```
PSO:COMPUTE DIGITAL SIGNATURE(  INS = ‘2A’,  // PSO COMP.DIG.SIG
                                P1 = ‘9E’,   // expected resp. DO
                                P2 = ‘9A’,   // dig. signature
                                data = see below
```

For the structure and content of the APDU refer to ISO/IEC 7816-8, chapter 5.2.

The digital signature input format is described in 6.4.4.1 “RSA” on page 6-19.

Alternatively, the INTERNAL AUTHENTICATE command can be used.

```
INTERNAL AUTHENTICATE(   INS = ‘88’,         // INT.AUT
                         P1 = ‘00’,          // No alg. info given
                         P2 = ‘00’,          // no key ref. given
                         data = see below
```

While the IFD sends only the digest T, the ICC performs the padding as described in Figure 6-2. In the following text it is not necessary to distinguish between these alternatives.

The digital signature input format is specified in 6.4.4.1 “RSA” on page 6-19

The response data field contains the plain signature without further TLV coding.

T	L	V	
-	-	SIG	= unformatted signature
-	-	‘9000’	= SW1 SW2 (or specific status bytes)

Table 6-2. Response data field for COMPUTE DIGITAL SIGNATURE/INTERNAL AUTHENTICATE

6.4.4 Client/Server authentication execution flow

The ICC generates a signature in the size of the modulus over the input data, which shall be presented in padded form such, that it is the size of the modulus.

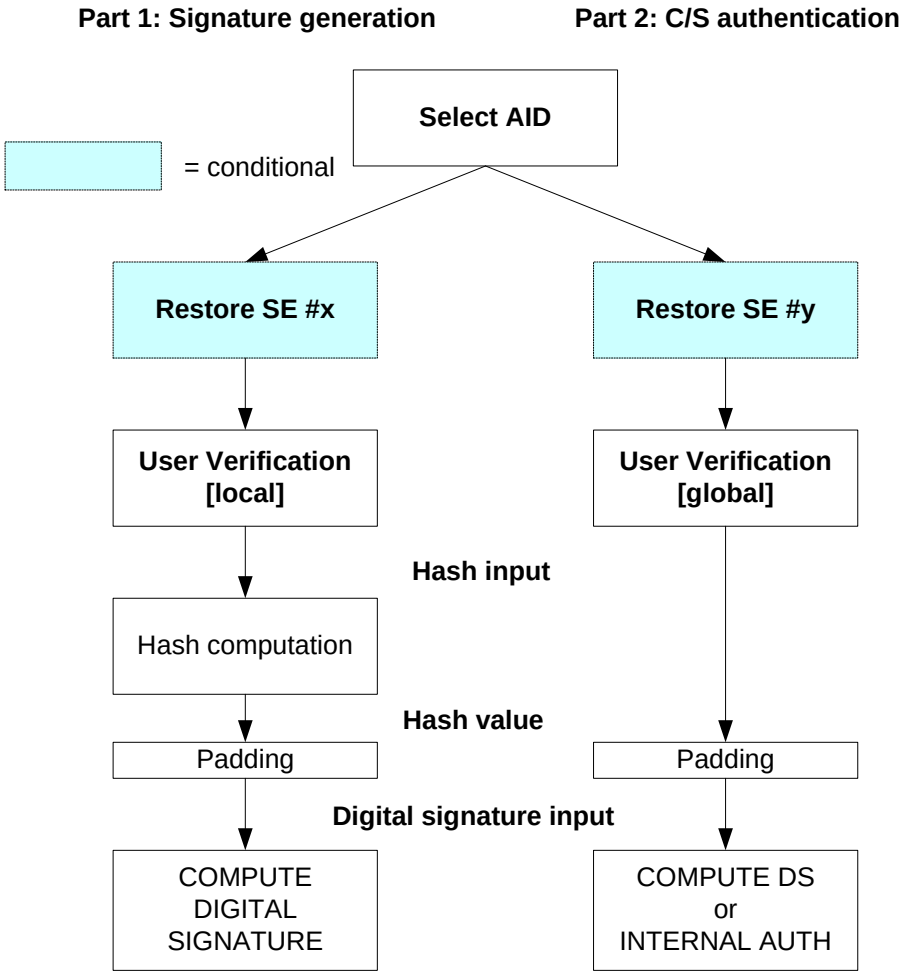


Figure 6-3. Comparison of operations in the ICC for signature creation versus C/S auth.

- The selection of a security environment is mandatory if the COMPUTE DS command is used for signature generation and client/server authentication. Using dif-

ferent commands avoids ambiguity for the usage of COMPUTE DS, hence a selection of a security environment is not necessary.

- For C/S authentication the final hash value will always be computed by the PC and sent to the ICC. The ICC provides the padding in PKCS #1 format and computes the digital signature. At least part of the padding needs to be done in the ICC in order to avoid existing attacks on the keys.
- The digest info is included in the data field of the COMPUTE DIGITAL SIGNATURE command. This mechanism is different from the computation of a digital signature as described in Part 1.
- User verification is performed with password references different for the signature generation [local password] and for C/S authentication [global password].

6.4.4.1 RSA

T	L	V	Bytes
-	-	T = DigestInfo	

Table 6-3. Command data field for KERBEROS protocol [50]

T	L	V	Bytes
-	-	T = H_MD5 || H_SHA1 see [56] and [55]	

Table 6-4. Command data field for SSL/TLS

T	L	V	Bytes
-	-	T = H_SHA1 see [56]	

Table 6-5. Command data field for WTLS

The ICC will have to perform the padding according to Figure 6-2. The value T is taken as described by the three tables above.

6.4.4.2 Other algorithms

All other algorithms use the padded (PKCS #1) hash value in the data field.

T	L	V	Bytes
-	-	Hash value	

Table 6-6. Input to other algorithms

7 Encryption Key Decipherment

Encryption key decipherment is needed when an application receives a (symmetric) message key enciphered with a public key (PK.CH.KE) that corresponds to a private key in the ICC. The IFD sends the encrypted symmetric key to the ICC. The ICC deciphers it using the private key and returns the plain symmetrical key. The IFD may then use the decrypted symmetrical key to decipher the attached message.

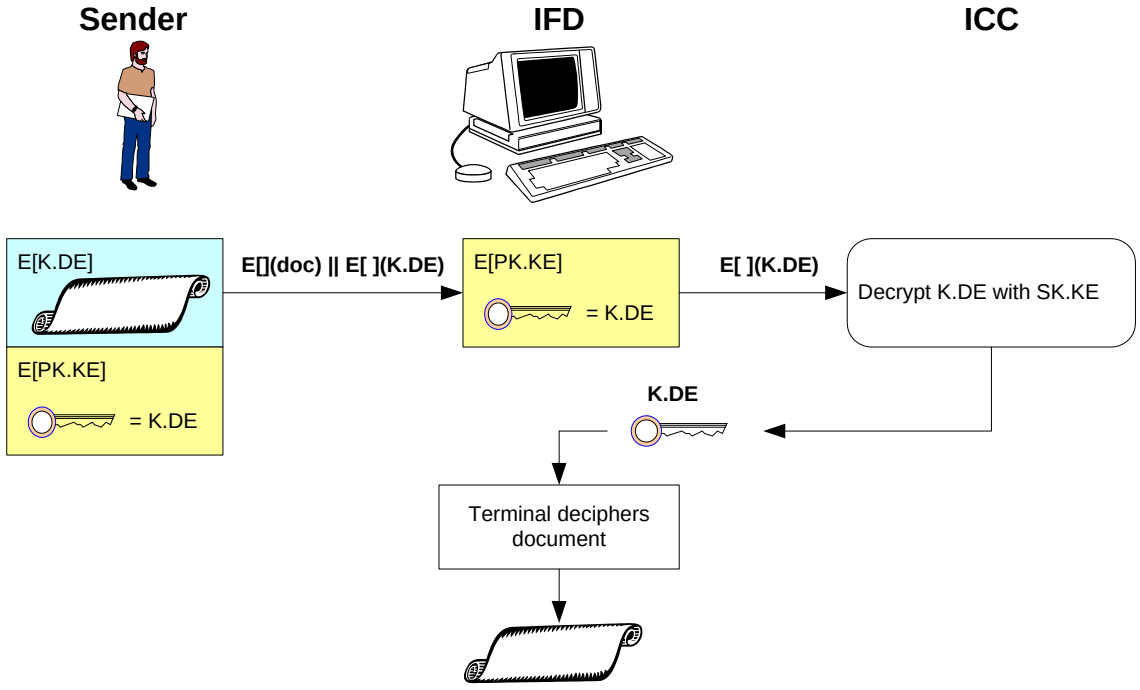


Figure 7-1. Key decipherment and Document decipherment

The following roles are involved in the document decipherment process

Sender: wants to send a document to the IFD (receiver). The sender knows the document encryption key K.DE and uses it to encrypt the document. He also encrypts that key with the ICC's public key (PK.KE).

Receiver: (IFD) wants to decipher the document. The IFD neither has the encryption key K.DE nor can it decrypt the key. Therefore the IFD strips the encrypted E[PK.KE](K.DE) from the package and sends it to the ICC.

After it has received the encryption key K.DE from the ICC, it uses K.DE in order to decipher the document.

ICC owns the SK.KE and uses it in order to decrypt the K.DE sent by the IFD.

The key decipherment service is used for the following security protocols:

- Application level encryption key decipherment
- Application level key agreement
- Key agreement or key exchange in PK Kerberos' preauthentication phase

7.1 Steps preceeding the key decryption

The steps preceeding a key decryption are application specific. Hence this specification does not mandate the existence of those steps.

7.2 Key Management with RSA

For confidential document exchange, the following scheme is applied:

- key transport is organized by **enciphering the symmetrical content encryption key** with the corresponding ICC's PK
- document enciphering with a symmetrical algorithm (e.g. DES-3).

If an enciphered document is sent, the ICC is not involved: the sender's software computes the content encryption key, enciphers the document and finally enciphers the content encryption key by applying the ICC's public key taken from the ICC's KE certificate.

The ICC's part in confidential document exchange is only the decryption of the symmetrical content encryption keys using the PSO:DECIPHER command.

The encryption key K shall be formatted according to PKCS #1, Version 2.1, Chapter 9.1.2 "EME-PKCS1-v1_5".

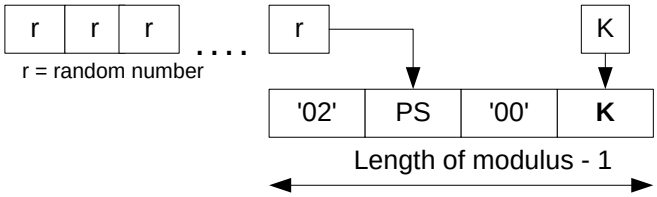


Figure 7-2. PKCS encoding of symmetric encryption key K

Formatting according to PKCS #1:

'02' || PS || '00' || K

where PS is an octet string consisting of pseudorandomly generated nonzero octets. The formatted octet string shall consist of k-1 octets where k is the length in octets of the modulus of the private key for decryption.

7.2.1 Execution flow

Figure 7-2 shows the execution flow of the key decipherment protocol.

Table 7-1. Key decipherment flow

Step	IFD	Trans-mission	ICC
1	MSE:SET SK.CH.KE	→	
		←	OK
2	PSO:DECIPHER	→	
		←	decipher remove padding return key

7.2.1.1 Step 1 - Set deciphering key

Before key decipherment can be performed, the secret key has to be selected with the MSE command.

Execution Flow:

```
MSE:SET( INS = ‘22’          // MANAGE SEC ENV
         P1 = ‘41’           // SET
         P2 = ‘B8’           // CT (confidentiality template)
         data=CRT data field with KeyRef(SK.CH.KE)

Response = empty
```

The command and response APDUs are described in ISO/IEC 7816-4, chapter 7.5.11.

The structure and coding of the key reference data is specified in section 10.2.3 “CRT CT for selection of ICC’s private key” on page 10-34.

7.2.1.2 Step 2 - Decipher key

After the key is selected, the decipher operation can be executed.

```
PSO:DECIPHER ( INS = ‘2A’,  // PSO:DECIPHER
               P1 = ‘80’,   // return plain value
               P2 = ‘86’,   // Enciphered data present
                               in the data field
               data = see below
```

The command data field contains the following data

Table 7-2. Command data field for PSO DECIPHER

‘xx’	PI (see Table 7-8)
‘xx’..’xx’	cryptogram padded according to PI

The response data field contains the content-encryption key

Table 7-3. Response data field for PSO DECIPHER

'xx'..'xx'	content-encryption key (padding already removed)
'9000'	SW1 SW2 (or specific status bytes)

7.3 Diffie-Hellman key exchange

The requirements for the ICC are based on the ephemeral mode of the Diffie-Hellman key agreement method, see RFC 2631, as referenced in CMS (Cryptographic Message Syntax).

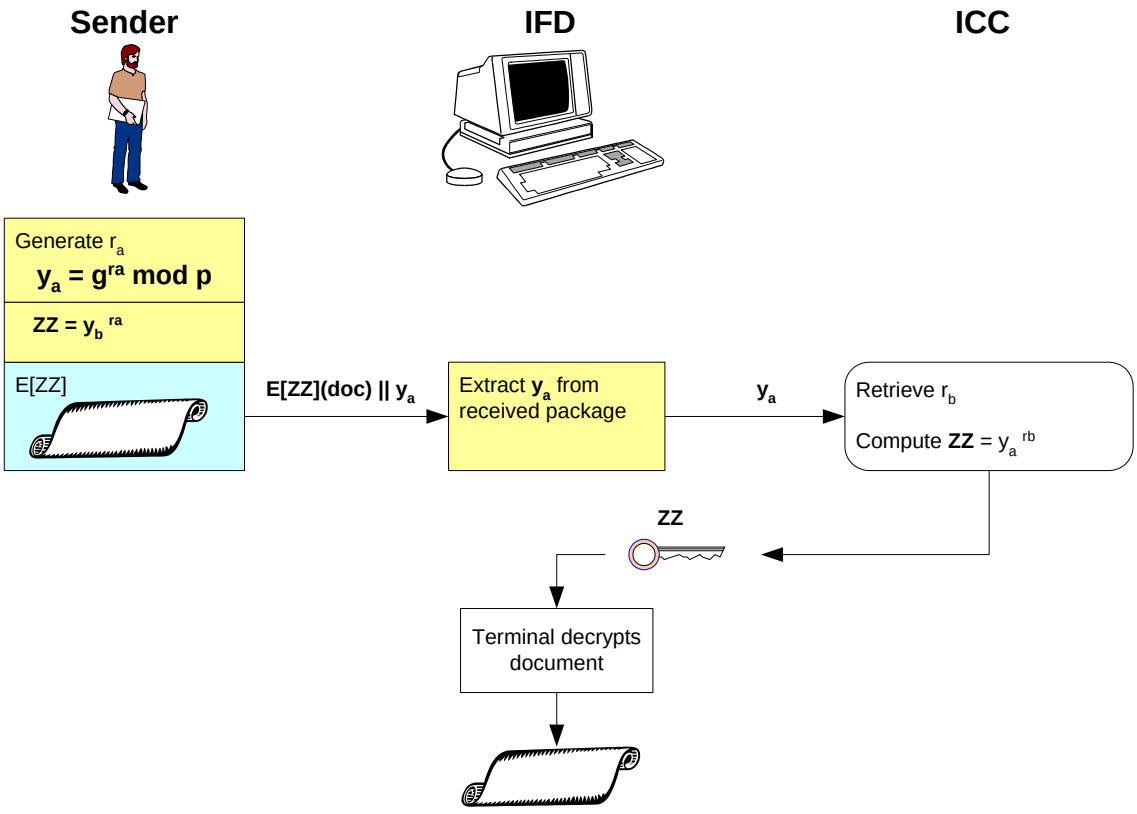


Figure 7-3. DH key freshness generation and Document decipherment

The following roles are involved in the document decipherment process

Sender: wants to send a document to the IFD (receiver). The sender knows the ICC's certified public key y_b and uses it to derive ZZ to be used for the encryption of the document. Together with the encrypted document he sends his public key y_a which was newly generated through r_a for freshness reasons.

Receiver: (IFD) wants to decipher the document. Since the ICC does not have the IFD's public key y_a it cannot generate ZZ. Therefore the IFD sends the sender's public key y_a to the ICC.

After it has received the common secret ZZ from the ICC, it uses ZZ in order to decipher the document.

ICC owns the number r_b and therefore can calculate the common secret ZZ in order to decipher the document. It returns the common secret ZZ to the IFD.

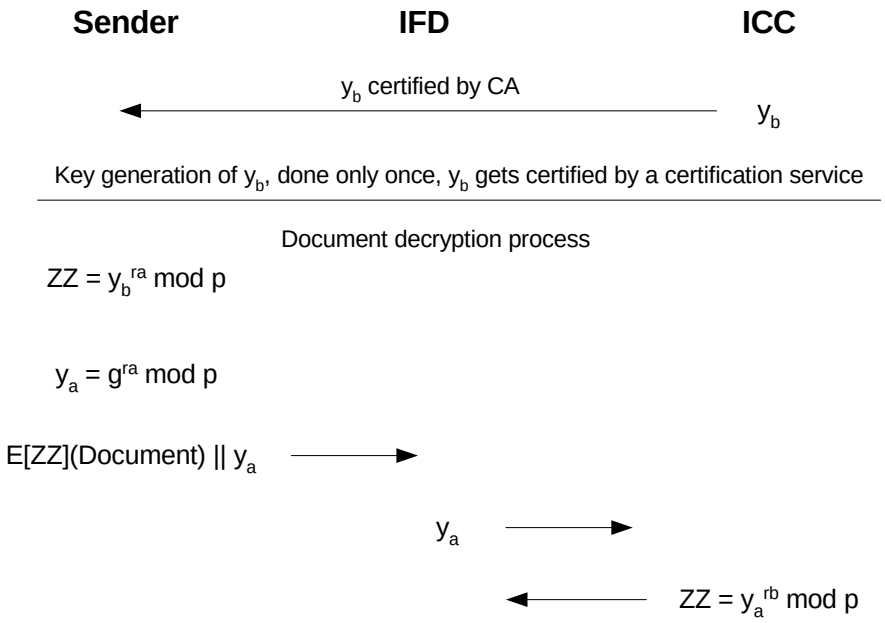


Figure 7-4. DH key exchange acc. to RFC 2631

Figure 7-4 shows the life cycle of the key tokens y_a ,y_b. The ICC's key token y_b is generated once. y_b is exported and certified and is made available to any sender that wants to send something to and IFD that uses the ICC.

For the document deciphering process freshness is provided by the sender's r_a on each document encryption.

In Figure 7-4 the cryptological perspective is repeated for the process where

y_a is the sender's public key made of $y_a = g^{ra} \bmod p$.

y_b is the ICC's certified public key made of $y_b = g^{rb} \bmod p$. It does not necessarily need to exist in the ICC after creation. The ICC keeps r_b to allow the creation of ZZ.

r_a is the IFD's secret exponent (randomly generated for freshness)

r_b is the ICC's secret exponent (generated once).

It can be summarized as follows:

The document is encrypted by a content encryption key, which is generated at random.

The recipient's public key and the sender's private key are used to generate a symmetric key, then the content-encryption key is encrypted in this symmetric key. The result ZZ is sent to the IFD for further derivation of keys.

Note: Care has to be taken in order to protect the recipient's secret key against the so called "small subgroup" attacks, see RFC 2785.

7.3.1 Execution flow

Figure 7-2 shows the execution flow of the key decipherment protocol.

Table 7-4. Key decipherment flow

Step	IFD	Trans-mission	ICC
1	MSE:SET SK.CH.KE	→	
		←	OK
2	PSO:DECIPHER	→	establish common secret ZZ
		←	OK

7.3.1.1 Step 1 : Select DH encryption key

First, the secret key SK.CH.KE has to be selected with the MSE command:

Execution Flow:

```
MSE:SET( INS = ‘22’          // MANAGE SEC ENV
         P1 = ‘41’           // SET
         P2 = ‘B8’           // CT (confidentiality template)
         data=CRT data field with KeyRef(SK.CH.KE)
```

For the structure and content of the APDU refer to ISO/IEC 7816-4, chapter 7.5.11.

The structure and coding of the key reference data is specified in section 10.2.4 “CRT CT for selection of ICC’s DH encryption key” on page 10-35.

The response data field contains the secret ZZ.

7.3.1.2 Step 2: Derivation of the shared secret.

After the key is set, the derivation of the shared secret can be done. The computed value ZZ is sent in the response.

Execution Flow:

```
PSO: DECIPHER (   INS = ‘2A’,        // PSO: DECIPHER
                  P1 = ‘80’,         // return plain value
                  P2 = ‘86’          // cryptogram (ISO/IEC 7816-4)
                  data = see below );
```

The response data field is empty

The command data field contains the following data

Table 7-5. Command data field for PSO DECIPHER

'xx'	PI (see Table 7-8) = 00 = no padding information given
'xx'..'xx'	cryptogram = y_a (public key) - no padding

The response data field contains the result ZZ of the deciphering operation (see Figure 7-4)

Table 7-6. Response data field for PSO DECIPHER

'xx'..'xx'	ZZ
'9000'	SW1 SW2 (or specific status bytes)

Note: The use of the key SK.CH.KE requires, that the global PIN (SK.CH.ADMIN) has been successfully presented.

7.4 Algorithm Identifier for DECIPHER

Table 7-7 shows the AlgIDs relevant for DECIPHER function

Table 7-7. AlgIDs decipherment.

AlgID	Meaning
'0x'	x ≥ A, Padding method not indicated
'0A'	RSA, PI = '00'
'1A'	RSA, PI ≠ '00', see Table 7-8.
'0B'	DH Key Agreement

Table 7-8. Key encipherment input formats

PI	KEI	Specification
'00'	No further information	ISO/IEC 7816-4
'81'	'02' || RND (all bytes ≠ 0, number key length dependent) || '00' || document cipher key	PKCS #1, Version 2.1, Chapter 9.1.2 "EME-PKCS1-v1_5"

8 Signature verification

Signature verification is an optional service of the signature card. To verify a digital signature under a signer's document the following data elements are necessary from the point of view of the ICC:

- the hash value
- the public key of the signatory
- the digital signature
- the DSI format (relevant only for RSA signatures)

As ICCs with today's technology cannot interpret X509-certificates, this function is only of value, when the PK of the signatory is already present in the ICC (this is not the usual situation). The actual storage of the public key is out of the scope of this specification.

8.1 Signature verification execution flow

Table 8-1 shows the execution flow of the signature verification

Table 8-1. Signature verification execution flow

Step	IFD	Transmission	ICC
1	PSO: HASH deliver hash value to be verified	→ ←	 OK
2	MSE: SET (signer's public key, DSI format, ...)	→ ←	 OK
3	PSO:VERIFY DIGITAL SIGNATURE	→ ←	 signature verification return result of verification

The data to be hashed or the final digest repectively are sent to the ICC with the PSO: HASH command as described in ISO/IEC 7816-8 [14].

Note: The alternatives for presenting the hash input and its computation correspond identically to those described in Part 1. For the convenience of the reader, however, these steps are repeated below.

Execution Flow 1a of 2 (all hashing in ICC)

```
PSO( INS = '2A',          // PERFORM SECURITY OPERATION
     P1 = '90',           // return hash code if required by Le
     P2 = '80',           // plain value
     data = plain data to be hashed)
```

Execution Flow 1b of 2 (partial hashing in ICC)

```
PSO( INS = '2A',          // PERFORM SECURITY OPERATION
     P1 = '90',           // return hash code if required by Le
     P2 = 'A0'            // input template for hash computation
     data =               '90' L90 <intermediate hash result> ||
                          '80' L80 <data to be hashed> ;
```

Execution Flow 1c of 2 (all hashing outside ICC)

```
PSO( INS = '2A',          // PERFORM SECURITY OPERATION
     P1 = '90',           // return hash code if required by Le
     P2 = 'A0'            // input template for hash computation
     data = '90' L90 <hash value>
```

For the structure and content of the APDU refer to ISO/IEC 7816-8, chapter 5.2..

For the content of the command data field or DO 'A0' resp. (HASH Input) refer to 13.2 "Hash Input-Formats" in Part 1 page 13-99.

8.1.1 Step 1 : Receive Hash

First the hash value for the signature to be verified needs to be sent to the ICC

Execution Flow:

```
PSO:HASH(  INS = '2A'          // PERFORM SECURITY OPERATION
           P1 = '90',          // return hash code if required by Le
           P2 = 'A0',          // input template for hash computation
           data = <see below>);
```

If additional data is to be hashed, the command APDU data field contains two data objects:

Table 8-2. Command APDU data field for HASH with data

T	L	V	
'90'	L_{90}	'xx...xx'	= Intermediate hash result
'80'	L_{80}	'xx...xx'	= Data to be hashed

If the final hash value shall be transmitted the APDU command data consists only of the hash value

Table 8-3. Command APDU data field for HASH without data

T	L	V	
'90'	L_{90}	'xx...xx'	= Intermediate hash result

The response data field of the command is empty.

8.1.2 Step 2: Select verification key

Before the command PSO: VERIFY SIGNATURE can be used, the PK of the signatory has to be selected via the command MSE.

Execution Flow:

```
MSE:SET (INS = ‘22’              // MSE
         P1 = ‘81’,              // SET for verification
         P2 = ‘B6’,              // DST
         data = <see below>);
```

For the structure and content of the APDU refer to ISO/IEC 7816-4, chapter 7.5.11.

Table 8-4. Command APDU data field for MSE:SET(USER.PK)

T	L	V	
‘80’	L_{80}	‘xx..xx’	= DO Alg reference (for DSI format)
‘83’	L_{83}	‘xx...xx’	= DO KeyRef of the user’s PK

The response data field is empty.

The value of the status bytes SW1-SW2 is

- ´9000´ = PK set
- ‘6xxx’ = PK not available or other error condition

8.1.3 Step 3: Verify digital signature

The hash value delivered in step 1 is verified with the selected key from step 2.

Execution Flow

```
PSO( INS = ‘2A’,     // VERIFY DIGITAL SIGNATURE
     P1 = ‘00’,      //
     P2 = ‘A8’,      // Input template for signature verification
     data = <see below>);
```

For the structure and content of the APDU refer to ISO/IEC 7816-8, chapter 5.7.

Table 8-5. Command APDU data field for VERIFY DIGITAL SIGNATURE

T	L	V
‘9E’	L_{9E}	Digital signature, DSI format to be interpreted by the ICC according to the information provided in Step 2.

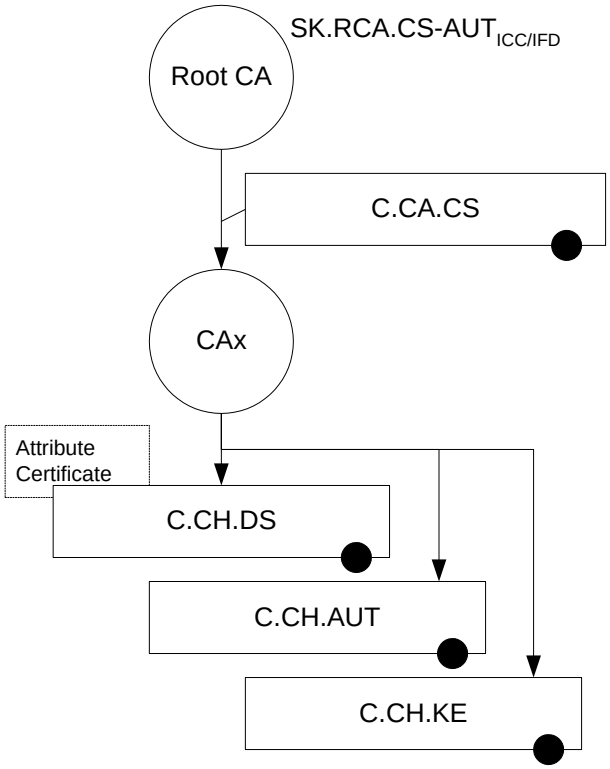
The response data field is empty.

The status bytes indicate the (un)successful verification of the signature.

9 Certificates for additional services

The additional services require the following types of end user certificates:

- AUTC/S certificate (typically X.509v3 authentication certificate) for proving access rights to servers; key usage flag is set to "digital signature"
- KE certificate (typically X.509v3 key encipherment certificate); key usage flag is set to "key encipherment or key agreement".



Digital Signature Certificate (C.CH.DS) may have associated Attribute Certificates; an Attribute Certificate may also be used together with C.CH.AUT and C.CH.KE. The certificate encoding scheme is typically X.509, but also another encoding scheme may be used.

Figure 9-1. CAs and certificates (example)

Note: The root CA for the digital signature service and for the other security services may be not the same. In case that there is a common root, it may be that different keys are used for signing DS-related certificates and certificates related to the other security services.

9.1 File structure

The appropriate files, holding the certificates, are stored in DF.ESIGN. Refer to 15.1 “File structure” in Part 1 page 15-116. The actual location and file identifier, are out of the scope of this specification and may be retrieved from the information coded in DF.CIA(16 “Cryptographic Information Application” in Part 1 page 16-122)

9.2 EF.C.CH.AUT

The EF.C.CH.AUT contains the C/S certificate(s) of the cardholder. The format of such certificate(s) is described in other standards and is out of scope of this standard.

Presence

The presence of this file is optional. If another format than X.509 is required by the application, then the content of this file might be replaced by an appropriate certificate.

Access conditions

Read: Always

Update: Typically the content of certificates is not subject to changes. As an exception, writing this file might be allowed for administrational purposes in order to update the certificate. The update of certificates is considered application dependent and not described in this specification

9.3 EF.C.CH.KE

The EF.C.CH.KE contains the data encryption certificate of the card holder. The format of this certificate is described in other standards and is out of scope of this standard.

Presence

The presence of this file is optional. If another format than X.509 is required by the application, then the content of this file might be replaced by an appropriate certificate.

Access conditions

Read: Always

Update: Typically the content of certificates is not subject to changes. As an exception, writing this file might be allowed for administrational purposes in order to update the certificate. The update of certificates is considered application dependent and not described in this specification

9.4 Reading Certificates and the public key of CAs

For reading the global data objects and certificates the ISO/IEC 7816-4 commands SELECT and READ BINARY are used.

For the structure and coding of the SELECT APDU refer to ISO/IEC 7816-4, chapter 7.1.1.

For the structure and coding of the READ BINARY APDU refer to ISO/IEC 7816-4, chapter 7.2.3.

10 APDU data structures

This chapter describes the data structures used in the ESIGN application.These data structures may describe elements being found in files or as part of a command APDU. The appropriate chapters will refer to these descriptions.

10.1 Algorithm Identifiers

The following algorithm identifiers are to be used for the appropriate functions

10.1.1 AlgIDs for INTERNAL AUTHENTICATION.

Table 10-1. AlgID for hash-functions and signature algorithms for INTERNAL AUTHENTICATE used for device authentication and client/server authentication

AlgID	Meaning
'12'	SHA-1 RSA with DSI acc. to PKCS #1

10.1.2 Algorithm Identifier for DECIPHER

Table 10-2. AlgIDs for decipherment

AlgID	Meaning
'0x'	' x ≥ A, Padding method not indicated
'0A'	'RSA , PI = '00'
'1A'	'RSA , PI ≠ '00', see tab. A.4
'0B'	'DH Key Agreement

Table 10-3. Key cipherment Input Formats

PI	KEI	Specification
'00'	'No further information	ISO/IEC 7816-4
'81'	'02' || RND (all bytes ≠ 0, number key length dependend) || '00' || document cipher key	PKCS #1, Version 2.1, Chapter 9.1.2 "EME-PKCS1-v1_5"

10.2 CRTs

The MANAGE SECURITY ENVIRONMENT command (refer to ISO/IEC 7816-4, chapter 7.5.11.) is used in applications to reference keys or to set security environments. The appropriate key reference information is coded in CRTs (Control Reference Templates). The following paragraphs describe the contents of relevant CRTs.

10.2.1 CRT DST for selection of ICC's private client/server auth. key

If used with the COMPUTE DIGITAL SIGNATURE command (6.4.2 "Step 2 - Set signing key for client/server internal authentication" on page 6-17) this CRT refers the private key of the ICC and is submitted in the data field of the MANAGE SECURITY ENVIRONMENT command to specify the signing key. The actual CRT tag for digital signature 'B6' is submitted in the parameter P2 of the command.

Table 10-4. Structure and content of CRT to select ICC's private C/S authentication key

'B6'[1]	L_{B6} [2]	CRT for digital signature			
		'84'	L_{84}	'xx..xx'	= KeyRef of SK.CH.AUT

1 sent in P2 of the APDU
2 sent in Lc of the APDU

Typically the corresponding public key is retrieved from a X.509 certificate.

10.2.2 CRT AT for selection of ICC's private client/server auth. key

If used with the INTERNAL AUTHENTICATE command (6.4.2 "Step 2 - Set signing key for client/server internal authentication" on page 6-17) this CRT refers the private key of the ICC and is submitted in the data field of the MANAGE SECURITY ENVIRONMENT command to specify the signing key. The actual CRT tag for authentication 'A4' is submitted in the parameter P2 of the command.

Table 10-5. Structure and content of CRT to select ICC's private C/S authentication key

'A4'[1]	L_{A4} [2]	CRT for authentication			
		'84'	L_{84}	'xx'	= KeyRef of SK.CH.AUT

1 sent in P2 of the APDU
2 sent in Lc of the APDU

Typically the corresponding public key is retrieved from a X.509 certificate.

10.2.3 CRT CT for selection of ICC's private key

This CRT refers the private key of the ICC and is submitted in the data field of the MANAGE SECURITY ENVIRONMENT command to specify the ICC's private key. The actual CRT tag for confidentiality 'B8' is submitted in the parameter P2 of the command to indicate the use of the key as deciphering key. (7.2.1.1 "Step 1 - Set deciphering key" on page 7-22)

Table 10-6. Structure and content of CRT to select ICC's private key

'B8'[1]	L_{B8} [2]	CRT for confidentiality			
		'84'	L_{84}	'xx..xx'	= KeyRef of SK.CH.KE
		'80'	L_{80}	'xx..xx'	= AlgId, see Table 10-2. "AlgIDs for decipherment" on page 10-33 (optional)

1 sent in P2 of the APDU
2 sent in Lc of the APDU

10.2.4 CRT CT for selection of ICC's DH encryption key

This CRT refers the public DH key of the ICC and is submitted in the data field of the MANAGE SECURITY ENVIRONMENT command to specify the private key. The actual CRT tag for confidentiality 'B8' is submitted in the parameter P2 of the command to indicate the use of the key as deciphering key. (7.3.1.1 "Step 1 : Select DH encryption key" on page 7-25)

Table 10-7. Structure and content of CRT to select ICC's public DH key

'B8'[1]	L_{B8} [2]	CRT for confidentiality			
		'84'	L_{84}	'xx..xx'	= KeyRef of PK.DH.CH.KE
		'80'	L_{80}	'xx..xx'	= AlgId, see Table 10-2. "AlgIDs for decipherment" on page 10-33 (optional)

1 sent in P2 of the APDU
2 sent in Lc of the APDU

10.2.5 CRT DST for selection of IFD's public key (signature verification)

This CRT refers the public key of the IFD and is submitted in the data field of the MANAGE SECURITY ENVIRONMENT command to specify signature verification key. The actual CRT tag for digital signature. 'B6' is submitted in the parameter P2 of the command to indicate the use of the key as verifying key. (See "Step 2: Select verification key" on page 8-28.).

Table 10-8. Structure and content of CRT to select IFDs public verification key

'B6'[1]	L_{B6} [2]	CRT DST			
		'83'	L_{83}	'xx..xx'	= KeyRef of PK.IFD.KE
		'80'	L_{80}	'xx..xx'	= AlgId, see 13.1 "Algorithm Identifiers and OIDs" in Part 1 page 13-98 (optional)

1 sent in P2 of the APDU
2 sent in Lc of the APDU

Annex A. Security Service Descriptor Templates (normative)

The use of Security Service Descriptors is conditional. If it is not implicitly known by the IFD how to use the card commands, the ICC may provide SSD templates. If the ICC provides SSD templates they shall be coded according to this annex.

Security Service descriptor templates provide optional information for the external world. They follow a simplified coding scheme and have the advantage over the information stored in DF.CIA, that no particular interpreter is required in order to ‘understand’ the coding.

Security Service descriptor templates serve to cover the interoperability issue, i.e. allow a terminal to learn about command sequences and parameters required to be used in order to accomplish a functionality according to Table A-1. The information given in SSD’s may reflect or enhance the information coded in DF.CIA. The idea of security service descriptors is derived from [41].

If a terminal is realized through another smart card, the advantage of SSD’s over DF.CIA to be readable without an interpreter becomes relevant.

A.1 Security Service Descriptor Concept

For supporting interoperability and coexistence of chipcards with differences, e.g. in command sequences, as well as to facilitate migration in an easier way, an SSD file should exist in the SigG application. The information about available security services are provided in SSD templates to be interpreted by the terminal. The SSD file contains either

- One or more SSD-Templates for each security service (see Table E.1) or
- a DO 'SSD Profile identifier' (see DO with tag '8D').

Table A-1. SSD Templates

Tag	Meaning
'A0	User verification service
'A1'	C/S internal authentication service
'A2'	External authentication service (sym.)
'A3'	External / Mutual authentication service (asym.)
'A4'	Digital signature computation service
'A5'	Digital signature verification service
'A6'	Certificate verification service
'A7'	Checksum computation service
'A8'	Checksum verification service

Table A-1. SSD Templates

Tag	Meaning
'A9'	Encipherment service
'AA'	Decipherment service
'AB'	File management service
‘AC’	Key generation service (may include implicit public key export)
‘AD’	Key import service
‘AE’	Public key export service

A.2 SSD Data Objects

The SSD templates have context-specific tags, whereby the context is given by the SSD file. The DO 'Instruction set mapping' is mandatory in the value part of each SSD template. All other DOs are optional.

A.2.1 DO Extended Header List, tag ‘4D’

This DO can be used to indicate an extended header list e.g. for the use in the Key Import process (PUT DATA) acc. to 10.3.1.3 “Using Header Lists to Import a Private Key” in Part 1 page 10-78.

A.2.2 DO Instruction set mapping (ISM), tag '80'

This DO contains the regular command to provide the respective security service. If the security service requires a sequence of commands, then this DO is repeated. The DO contains at least the CLA-byte and the INS-byte of the command as defined in main part of this specification. If appropriate, P1 and P2 or further parts of the command APDU may be present. It is assumed that the outside world knows the command in its complete form and which data - if any - have to be inserted in the body part of the command.

A.2.3 DO Command to perform (CTP), tag '52' (see ISO/IEC 7816-6)

This DO - if present - informs the outside world which command is supported by the card to provide the same security service as achievable with the command indicated in the DO ISM. If several commands are necessary then the DO CTP is repeated. It shall follow - if used - immediately behind the last DO ISM.

Usually the CLA byte is set to 00 independent of any channel usage. If the command shall be send in SM mode then the SM bits shall be set in CLA

A.2.4 DO Algorithm object identifier (OID), tag '06' (see ISO/IEC 7816-6)

The value field of this DO contains the object identifier of the related crypto algorithm available in the card. The encoding rule of the OID is according to ISO 8825 as follows:

- first subidentifier is multiplied by 40 and coded as binary number in one byte; the value of the second subidentifier is added to this byte

- the following subidentifiers are each represented as a binary number in a sequence of bytes from which bit b8 is the chaining bit (1 = not last byte, 0 = last or only byte)

A.2.5 DO Algorithm reference, tag '81'

The value field of this DO contains the algorithm reference as used in the card for the algorithm denoted by the DO OID or specified in an application context.

A.2.6 DO Key reference, tag '82'

The value field contains the key reference of the key to be applied by the card when performing the related security operation. If this DO is present, then the value has to be used in the command related to this security service.

A.2.7 DO FID key file, tag '83'

The value field contains the file id of a file containing the key to be applied by the card when performing the related security operation. If this DO is present, a SELECT FILE command with the respective FID has to be sent as first command before using a security service.

A.2.8 DO Key group, tag '84'

This DO is only relevant for symmetric encipherment algorithms which work with individual keys and master keys. When this DO is used, then the value denotes the entity using the master key (e.g. '01' = physician, '02' = pharmacist, i.e. the values can be considered as group ids)

A.2.9 DO FID base certificate file, tag '85'

The value field of the DO contains the file id of a file containing the base certificate related to the respective security service.

A.2.10 DO FID adjoint certificate file, tag '86'

The value field contains the file id of a file containing an adjoint certificate related to the base certificate of the same security service template.

A.2.11 DO Certificate reference, tag '87'

If no FID for a certificate file is given, then this DO - if present - contains a reference to the related certificate which is not stored in the card. The reference may be a key identifier (tag '88') and/or the distinguished name (or the ID) of the CA issuing the certificate (tag '89') followed by the certificate serial number (tag '8A').

A.2.12 DO Certificate qualifier, tag '88'

The value field contains the information whether the certificate is a non-ICC certificate (e.g. a X.509 certificate) to be verified outside a card (value '00') or an ICC certificate, which may be interpreted by a card (value '01'). The default value is '00'.

A.2.13 DO FID for file with public key of the certification authority PK(CA), tag '89'

This DO - if present - contains the FID of the file in which the DO public key of the certification authority is stored (tag '5F4A').

A.2.14 DO PIN usage policy, tag '5F2F'

(see ISO/IEC 7816-6)

This DO - if present - indicates the PIN usage policy. The value filed of this DO (2 bytes) is application specific.

For DF.ESIGN the following values are defined:

First byte:

'80' i.e. PIN relevant for application, see clause 6.4 of ISO/IEC 7816-6

Second byte:

- msn: Coding scheme
- lsn: Limitation

'0x': ASCII coding (default for PIN reference [1] '80')

'1x': Format 2 PIN block (default for PIN reference = '80')

'x0': after PIN presentation no limitation

'x1'-'xF': the number indicates the amount of signatures possible with one PIN presentation

Other values RFU

Note: If only one byte is present in the value field, then only the coding scheme and limitation indication are indicated.

A.2.15 DO PIN reference, tag '8A'

The value field of this DO contains one byte coding the qualifier of the reference for the cardholder verification data, if the specified value is not used (i.e. for the OIC the values '01' and '81').

If bit b8 of the value field is set to 1, the DO contains the PIN reference for the specific PIN (i.e. for the OIC the DS-PIN). If bit b8 of the value field is set to 0, the DO contains the PIN reference for the global PIN (i.e. for the OIC the AUT+KE-PIN).

A.2.16 DO Application identifier (AID), tag '4F' (see ISO/IEC 7816-6)

This DO indicates, to which application the related template belongs and shall only be used, if an SSD file on the MF level is needed (e.g. if a PIN presentation before selection of the application with the specified AID is required).

A.2.17 DO CLA coding, tag '8B'

The value field of this DO contains the CLA coding which has to be used instead of the CLA coding given in the DO ISM.

A.2.18 DO Status information (SW1-SW2), tag '42' (see ISO/IEC 7816-6)

If relevant status bytes (e.g. from the VERIFY command) differ from those of the command described in the DO ISM, then the mapping shall be given, i.e. a SW1-SW2 of the ISM command is followed by SW1-SW2 coding send by the card.

A.2.19 DO Discretionary data, tag '53' (see ISO/IEC 7816-6)

The contents of this DO - if present - is defined by the application provider.

A.2.20 DO SE number, tag '8C'

The value field of this DO contains the security environment number.

A.2.21 DO SSD profile identifier, tag '8D'

The value of this DO identifies for the respective application an SSD set available in the IFD. The externally stored SSD set describes the security service supported by the card.

A.2.22 DO FID mapping, tag '8E'

The value of this DO contains one or more pairs of FIDs: the first FID gives the value assigned in this specification, the second FID indicates the FID to be used.

A.3 Location of the SSD templates

The SSD templates are stored in a transparent file EF.SSD.

If EF.SSD is used, for interoperability reasons it shall be found under the DF.ESIGN. The file identifier is application specific.

A.4 Examples for SSD templates

Table A-2. Example for key generation

'AC 13' '80 04 002241A4' || '80-04 004602xx' || '80 05 004683xx00'	Key generation and export using SK.ICC.AUT: MSE SET SK.ICC.AUT for authentication GENERATE AKP, response empty, xx = Keyref of SK.CH.DS GENERATE AKP, PK export response data field
'AC 07' '80 05 0C4682xx00'	Key generation and export with dynamic SM: GENERATE AKP, xx = Keyref of SK.CH.DS, response according to EHL

Table A-3. Example for signature generation

'a4 20'	signature generation template
'80 08' '00 22 41b6' '03' 80 01 12'	command to perform CLA INS P1 P2 indicating MSE Set DST for computation Lc indicating length of command data field data field with TLV-object, tag='80' indicates algorithm identifier
'80 08' '00 22 41aa' '03 80 01 10'	command to perform CLA INS P1 P2 indicating MSE Set HT for computation Lc indicating length of command data field data field with TLV-object, tag='80' selects hash algorithm
'80 04' '00 2a 90a0'	command to perform CLA INS P1 P2 indicating PSO Hash, because no Lc is given, implicitly hash of last round is meant.
'80 04' '00 2a 9e9a'	command to perform CLA INS P1 P2 indicating PSO Compute Digital Signature

If for example a hash calculation shall be done completely outside the Smart Card the according SSD entry would look like:

Table A-4. Alternative hash computation of row 4 in the above table

'80 07' '00 2a 90a0' '16' '90 14'	command to perform CLA INS P1 P2 indicating PSO Hash Lc indicating length of command data field tag='90' and length='14'=20 indicates hash value (rather than intermediate hash value which would have a length of 28 bytes)

Please note that algorithm identifiers shown above are interpreted by the card only, if several different algorithm identifiers are given for one service, then the IFD could choose one of them on the convenience of the IFD. To do so the IFD also has to know what value refers to what algorithm.

Annex B. (informative) Security environments

A security environment defines a set of security parameters eg. key IDs, algorithm IDs and modes of secure messaging operations. Files and data objects contain access definitions which can either refer to those security environments or code the security parameters directly.

Security environments consist of a set of Control Reference Templates (CRT) that code the specific values of security parameters.

Any security environment is coded as data object '7B'. Multiple security environments may be organised in a file EF_SECENV which can be referred to in the file control information of a file with the tag '8D'[1]. The appropriate security environment number is referred to in the actual access conditions of a file / data object.

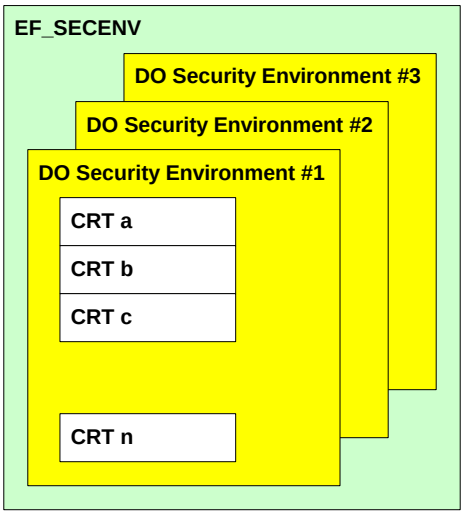


Figure B-1. Security environments and CRTs

Three security environments are defined in this chapter. The first security environment is used as default security environment.

If the additional services from this part are used, the different services may require different security environments. In particular if the same command is used for different services (e.g. COMPUTE DS for Signature generation and Client/Server authentication) an SE selection is mandatory to distinguish between these cases.

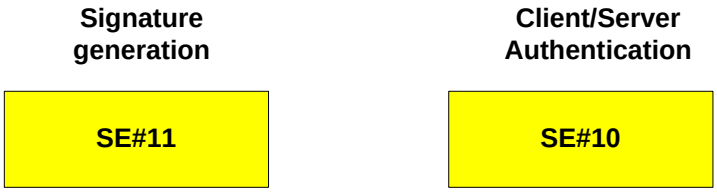


Figure B-2. Security services and their corresponding security environments

1 according to ISO-7816-4

The following codings serve as a ‘language’ to express the functionality to be achieved for the security environments. Only the functionality is mandatory, however, the same functionality may be achieved by another coding.

The definitions given in the following chapters A.1 ... A.3 specify a particular example of a security environment and the appropriate access rule codings. According to Figure C-1 on page C-56 the following choices have been made for these examples

- key transport protocol
- knowledge based user verification
- The PSO: COMPUTE DIGITAL SIGNATURE command in the context of client/ server authentication.

B.1 Definition of CRTs (examples)

For the ESIGN application interindustry types of Control Reference Templates (CRT) according to ISO/IEC 7816-4 and ISO/IEC 7816-9 shall be used in the context of Security Environments.

The CRTs with the contained data objects of the following examples exist in Security Environments as shown in Figure B-1 on page B-42.

Table B-1. Control Reference Templates for security environments

Tag	Value
‘A4’	CRT valid for authentication (AT)
‘B4’	CRT valid for cryptographic checksum (CCT)
‘B6’	CRT valid for digital signature (DST)
‘B8’	CRT valid for confidentiality (CT)

The contents of the CRT DOs referenced by these tags consists of DOs from the following list:

Table B-2. Data objects in CRTs

Tag	Length	Value
‘95’	‘01’	Usage Qualifier Byte (UQB)
‘83’	variable	Reference of public or secret (symm. case) key
‘84’	variable	Reference of a private key
‘80’	variable	Algorithm ID

The Usage Qualifier Byte (UQB) may specify the usage of the template, either as a security condition or in compliance with the MSE command.

The DO with tag ‘83’ references a secret (e.g. session[1]) or a public key, the tag ‘84’ references a private key, see ISO/IEC 7816-4. Key identifiers are not prescribed by the E Sign K specification. The key identifiers are specified in the CIA application.

If the algorithm ID is present it shall be compliant to the E Sign K specification.

B.1.1 CRT for Authentication (AT)

The UQB is optional as well as the algorithm ID. The key reference is mandatory.

The following authentication templates are required:

- AT_1 for knowledge based user verification using the global PIN:

  PIN.CH.ADMIN

- AT_2 for knowledge based user verification using the PIN PIN.CH.DS

- AT_3 / 4 for external device authentication and for internal device authentication (can be combined)

These templates are encoded as follows

1 ISO 7816-4 specifies tag ‘84’ for a key, that computes a session key, however in this example the already computed session key is referred by tag ‘83’.

AT_1

Table B-3. Structure and content of AT_1

'A4'	L_{A4}	CRT authentication (AT)		
		'95'	'01'	'08' = UQB: user verification, knowledge based
		'83'	L_{83}	reference to global password PIN.CH.ADMIN
		... other DO's, if required		

AT_2

Table B-4. Structure and content of AT_2

'A4'	L_{A4}	CRT authentication (AT)		
		'95'	'01'	'08' = UQB: user verification, knowledge based
		'83'	L_{83}	reference to PIN.CH.DS
		... other DO's, if required		

AT_3

Table B-5. Structure and content of AT_3

'A4'	L_{A4}	CRT authentication (AT)		
		'95'	'01'	'80' = UQB: external authentication
		'83'	L_{83}	reference to PK.IFD.AUT
		'84'	L_{84}	reference to SK.ICC.AUT
		'80'	'01'	'xx' = algorithm reference, see Table 13-2 in volume 1 page 13-99
		... other DO's, if required		

AT_4

Table B-6. Structure and content of AT_4

'A4'	L_{A4}	CRT authentication (AT)		
		'95'	'01'	'40' = UQB: internal authentication
		'83'	L_{83}	reference to PK.IFD.AUT
		'84'	L_{84}	reference to SK.ICC.AUT
		'80'	'01'	'xx' = algorithm reference, see Table 13-2 in volume 1 page 13-99
		... other DO's, if required		

B.1.2 CRT for Cryptographic Checksum (CCT)

The UQB is optional, the key reference is mandatory, the algorithm ID shall not be used. The following CCT templates are required:

- CCT_1 /_2 for Secure Messaging in the command data fields and in the response data fields (can be combined)

These templates are encoded as follows:

CCT_1

Table B-7. Structure and content of CCT_1

'B4'	L_{B4}	CRT cryptographic checksum (CCT)		
		'95'	'01'	'10' - UQB: SM in command data field
		'83'	L_{83}	reference to macing session key K(MAC)
		... other DO's, if required		

CCT_2

Table B-8. Structure and content of CCT_2

'B4'	L_{B4}	CRT cryptographic checksum (CCT)		
		'95'	'01'	'20' - UQB: SM in response data field
		'83'	L_{83}	reference to macing session key K(MAC)
		... other DO's, if required		

B.1.3 CRT for Digital Signature (DST)

The UQB is optional as well as the algorithm ID. The key reference is mandatory.

The following DST template is required:

- DST_1 for a digital signature using SK.CH.DS
- DST_2 for certificate verification (in the context of device authentication) using the root CA key PK.RCA.AUT
- DST_3 for certificate verification (in the context of device authentication) using a public key of the certificate chain

These templates are encoded as follows:

DST_1

Table B-9. Structure and content of DST_1

'B6'	L_{B6}	CRT digital signature (DST)		
		'95'	'01'	'40' = UQB: computation (DST)
		'84'	L_{84}	references to SK.CH.DS
		'80'	'01'	algorithm reference (optional)
		... other DO's, if required		

DST_2

Table B-10. Structure and content of DST_2

'B6'	L_{B6}	CRT digital signature (DST)		
		'95'	'01'	'80' = UQB: Verification (DST)
		'83'	L_{83}	Reference PK.RCA.AUT
		'80'	'01'	algorithm reference (optional)
		... other DO's, if required		

DST_3

Table B-11. Structure and content of DST_2

'B6'	L_{B6}	CRT digital signature (DST)		
		'95'	'01'	'80' = UQB: Verification (DST)
		'83'	L_{83}	Reference to public key in certificate chain
		'80'	'01'	algorithm reference (optional)
		... other DO's, if required		

B.1.4 CRT for confidentiality (CT)

The UQB is optional as well as the algorithm ID. For the present example, the key identifier is mandatory.

The following CT template are required:

- CT_1 for secure messaging in the command data fields and CT_2 for SM in the response data fields (can be combined)
- CT_3 for key decipherment

These templates are encoded as follows:

CT_1

Table B-12. Structure and content of CT_1

'B8'	L_{B8}	CRT confidentiality (CT)		
		'95'	'01'	'10' = UQB: SM in command data field
		'83'	L_{83}	reference to encryption session key K(ENC)
		... other DO's, if required		

CT_2

Table B-13. Structure and content of CT_2

'B8'	L_{B8}	CRT confidentiality (CT)		
		'95'	'01'	'20' = UQB: SM in response data field
		'83'	L_{83}	reference to encryption session key K(ENC)
		... other DO's, if required		

Note: CT_1 and CT_2 can be combined in a common CT_12 in the case command and response data field are to be protected with the same key.

CT_3

Table B-14. Structure and content of CT_3

'B8'	L_{B8}	CRT confidentiality (CT)		
		'95'	'40'	'40' = UQB: key parameters for message decryption[1]
		'84'	L_{84}	reference to decrypting SK.CH.KE
		... other DO's, if required		

1 This decryption is additional to a possible encryption in the SM layer.

B.2 Security Environments (example)

The following Security Environments are specified (SE#10 is related to this part 2, SE#11 is related to Part 1).

Table B-15. Security environments

SE#	Meaning
SE#1	Default Security Environment
SE#10	User verification with **global** reference data (PIN PIN.CH.ADMIN) and client/ server authentication
SE#11	Mutual device authentication, secure messaging, user verification (PIN.CH.DS) and/or biometrics) and Digital Signature
SE#10 and SE #11 can be enhanced to key decipherment.	

B.2.1 Security Environment #10

The security environment #10 is used in the context of client server authentication

Table B-16. Security Environment #10

'7B'	L_{7B}	Security environment		
		'80'	'01'	'0A' = SE #10
		AT_1		
		DST_2		

B.2.2 Security Environment #11

The security environment #11 is used in the context of signature generation in untrusted environment.

Table B-17. Security Environment #11[1]

'7B'	L_{7B}	Security environment		
		'80'	'01'	'0B' = SE #11
		AT_2:		user authentication, knowledge based (PIN.CH.DS)
		AT_3:		external authentication
		AT_4:		internal authentication
		CT_1:		SM in command data field
		CT_2:		SM in response data field
		CCT_1:		SM in command data field
		CCT_2:		SM in response data field
		DST_1:		digital signature with SK.CH.DS

1 The security environment #11 may be enhanced by the parameters for key decryption as specified in “CT_3” on page B-48.

Note: A combined approach of asymmetric/symmetric device authentication could be used, if the session keys after an asymmetric device authentication are stored persistently. In the following sessions, the available (stored) session keys may be used with a symmetric authentication scheme as described in 8.7 “Symmetric authentication scheme” in Part 1 page 8-59 to derive the new session keys.

If this approach is used, the CRTs of SE #11 are appropriate for use.

B.3 Definition of File Control Information Templates (example)

In this chapter templates for file control information are presented using a subset of the interindustry templates according to ISO/IEC 7816-4. The ESIGN application may return a template according to this specification in response to the SELECT command for every file.

The following interindustry templates for file information are used:

- File Control Parameter (FCP) template denoted by tag ‘62’
- File Management Data (FMD) template denoted by tag ‘64’
- File Control Information (FCI) template denoted by tag ‘6F’

The access conditions are returned as part of the file control information (FCI) on a SELECT command.

Note: The examples below use the expanded format according to ISO/IEC 7816-4, Chapter 5.4.3.3. The use of the compact format is allowed. However, with the compact format the expression of the PSO command requires a specification of the proprietary bits which is out of the scope of this specification. Therefore only the expanded format is referenced below.

B.3.1 Access Conditions

In the following table an example for the ACs of the E Sign K specification is summarised:

Table B-18. Access Conditions of EFs

Filename	Access Condition
EF.DIR	**Read**: always, **Update**: never
EF.SN.ICC[1]	**Read**: always, **Update**: never
EF.C.ICC.AUT	**Read**: always, **Update**: never
EF.C.CA_{ICC}.CS-AUT	**Read**: always, **Update:** never

Table B-18. Access Conditions of EFs

Filename	Access Condition
EF.C.CH.DS	**Read**: always, **Update**: never
EF.C.CA.DS	**Read**: always, **Update**: never
EF.DM	**Read**: successful device authentication using SM **Update**: after successful presentation of PIN.CH.ADMIN or **Update**: successful device authentication (using SM)[2]

1 For the privacy device authentication protocol this file shall not be readable until the IFD has authenticated successfully.
2 If used with device authentication only, a **denial of service attack** is possible by allowing an attacker to change the display message using an authenticating terminal.

Note: In this example the **Update** condition is never, because the replacement of a certificate could be performed by administrative functionality which is out of the scope of this document.

Please note that not all files are mandated (see 15 “Files” in Part 1 page 15-116).

Access (write, update) to the certificate files and the EF.DIR file is granted for administrative purposes and not in the scope of the present document.

B.3.2 Access rule references

The definitions in these paragraphs are in accordance with the specification of security attributes in expanded format in ISO/IEC 7816-4, Chapter 5.4.3.3.

In order to link these access conditions to security environments the access rule references are used according to Table B-19.

Table B-19. Access rule references

Tag	Length	Value
‘8B’	L_{8B}	< Expanded format bytes to follow>
		fileID
		SEID byte
		ARR byte
		SEID byte
		ARR byte
		....

The **file identifier (file ID)** (2 byte) is optional and only required in the case that the file EF.ARR is not used. For every SE denoted by the corresponding Security Environment Identifier (SEID, 1 byte) the Security Attribute is referenced by means of the Access Rule Reference byte (ARR, 1 byte) that denotes a record in the EF.ARR file (or the file given by the file ID).

Access Conditions can be valid for a DF (referenced in the FCI of DF.ESIGN or for files respectively).

For the DF.ESIGN the security attributes are defined by

Table B-20. DF.ESIGN security attributes

Tag	Length	Value
'8B'	L_{8B}	<list of SEID# ARR# to follow>
		fileID (optional, see above)
		SEID '0A'
		ARR = 4
		SEID '0B'
		ARR = 5

These records are referenced in the expanded format of the security attributes, valid unter the respective SE as :

SE#10 under Record ARR = 4 in the ARR File

SE #11 under Record ARR = 5 in the ARR File

and coded as

ARR = 04 (client server auth.):
AM_DO: COMPUTE DST ||
SC_DO: CRT_DST (User_AUT with PIN_CH.ADMIN)

or

ARR = 05 (digital signature service):
AM_DO: INS_VERIFY [PIN.CH.DS]
SM_DO: EXTERNAL_AUT (and SM)

respectively.

B.3.3 Access conditions for EF.ARR

The access conditions for the EF.ARR are the following:

Table B-21. ARR record for ACs of files containing SEs

Tag	Length	Value	Meaning
'80'	'01'	'01'	Read: Always
'90'	'00'	-	
'80'	'01'	'7E'	All other commands: Never
'97'	'00'		

The first access mode data object (tag '80') denotes read access. The following security condition data object (tag '90') denotes that this access is always permitted. The second

access mode DO denotes write and update access (as well as file-management commands, (see DELETE FILE, TERMINATE EF, ACTIVATE FILE and DEACTIVATE FILE), the following security condition DO (tag ‘97’) denotes that this access is never permitted. In the following it is assumed that this access rule is stored in the first record of the EF.ARR file.

B.3.4 EF.ARR records

In the case of the file EF.DM the access rules depend on the Security Environment.

Using the same abbreviations for the CRTs as in the specification of the SEs (see B.2 “Security Environments (example)” on page B-49), this leads to the following EF.ARR records (the numbering of the records is up to the implementation):

Table B-22. EF_ARR

ARR record #	**Record content (one or more acess rules)**			**Meaning**
1	‘80’	‘01’	‘01’	**Read** Always
	‘90’	‘00’	-	
	‘80’	‘01’	‘7E’	**Write/Update:** Never **File management commands**: Never
	‘97	‘00’		
2	‘80’	‘01’	‘03’	**Read/Update**: Device authentication (external auth.) SM (encryption and MACing of command and response message.
	AT_3			
	CCT_1			
	CT_1			
	CCT_2			
	CT_2			
	‘80’	‘01’	‘7C’	**File management commands**: Never
	‘97’	‘00’	-	
3	‘80’	‘01’	‘01’	**Read:** Device authentication (external auth.) SM (encryption and MACing of command and response message.
	AT_3			
	CCT_1			
	CT_1			
	CCT_2			
	‘80’	‘01’	‘7C’	**File management commands**: Never
	‘97’	‘00’	-	

Table B-22. EF_ARR

ARR record #	Record content (one or more acess rules)			Meaning
4	'87'	'03'	'2A 9E 9A'	Execution of PSO: Compute Digital Signature for Client-Server Authentication: **Precondition :** Verification of the global PIN PIN.CH.ADMIN
	AT_1			
	'80'	'01'	'7C'	**File management commands**: Never
	'97'	'00'	-	
5	'87c'	'03'	'2A9E9A'	Execution of PSO: Compute Digital Signature for the digital signature service **Precondition:** Verification of the local PIN PIN.CH.DS using secure messaging
	AT_2			
	AT_3			
	CCT_1			
	CT_1			
	CCT_2			
	CT_2			
	'80'	'01'	'7C'	**File management commands**: Never
	'97'	'00'	-	

For the definition of Security Attributes for Files in correspondence to the ARR record see B.3.2 "Access rule references" on page B-51.

Table B-23. ARR as function of SE and files

Filename	SE=10	SE=11
DF.ESIGN	4	5
EF.DIR	1	1
EF.SN.ICC	1	1
EF.C.ICC.AUT	1	1
$EF.C.CA_{ICC}$.CS-AUT	1	1
EF.C.CH.DS	1	1
EF.C.CA.CS	1	1
EF.DM	3	2[1]

1 If used with device authentication only, a denial of service attack is possible by allowing an attacker to change the display message using an authenticating terminal.

Annex C. Interoperability aspects (informative)

Part 1 of this specification covers different alternatives or options for device authentication, user verification etc. This variety had to be added in order to comply with the demand of covering most of the existing implementations and to satisfy demands of existing infrastructure (e.g. symmetric/asymmetric infrastructure). On the other hand, a variety of technical choices spoils the aspect of interoperability.

Currently the ICC might support only one choice of each the described methods, however the IFD would have to implement all possible choices that are likely to be used in ICC's.

Figure C-1 shows the flow of selectable options for signature generation with the selection of several options.

C.1 Choosing device authentication

Unless there are technical conditions to choose a particular device authentication, the following recommendations are given

Key transport protocol

May be used, if

- privacy protection of the ICC is not required and compatibilty to existing standards (e.g. DIN 66291) is required.
- the ICC is to be authenticated first. If the application environment requires this order, then the key transport protocol might be choosen.

Privacy protocol

May be used, if

- additional privacy protection for the ICC is required. This protocol is modularity based on the key transport protocol, additional protocol steps provide the privacy protection of the ICC until the IFD is successfully authenticated. This protocol was newly added with the ESIGN specification and hence existing systems do currently not provide this protocol.
- the IFD is to be authenticated first. If the application environment requires this order, then the privacy protocol might be choosen.

Symmetric protocol

May be used

- if a symmetric environment is present
- device authentication can be based on symmetric keys

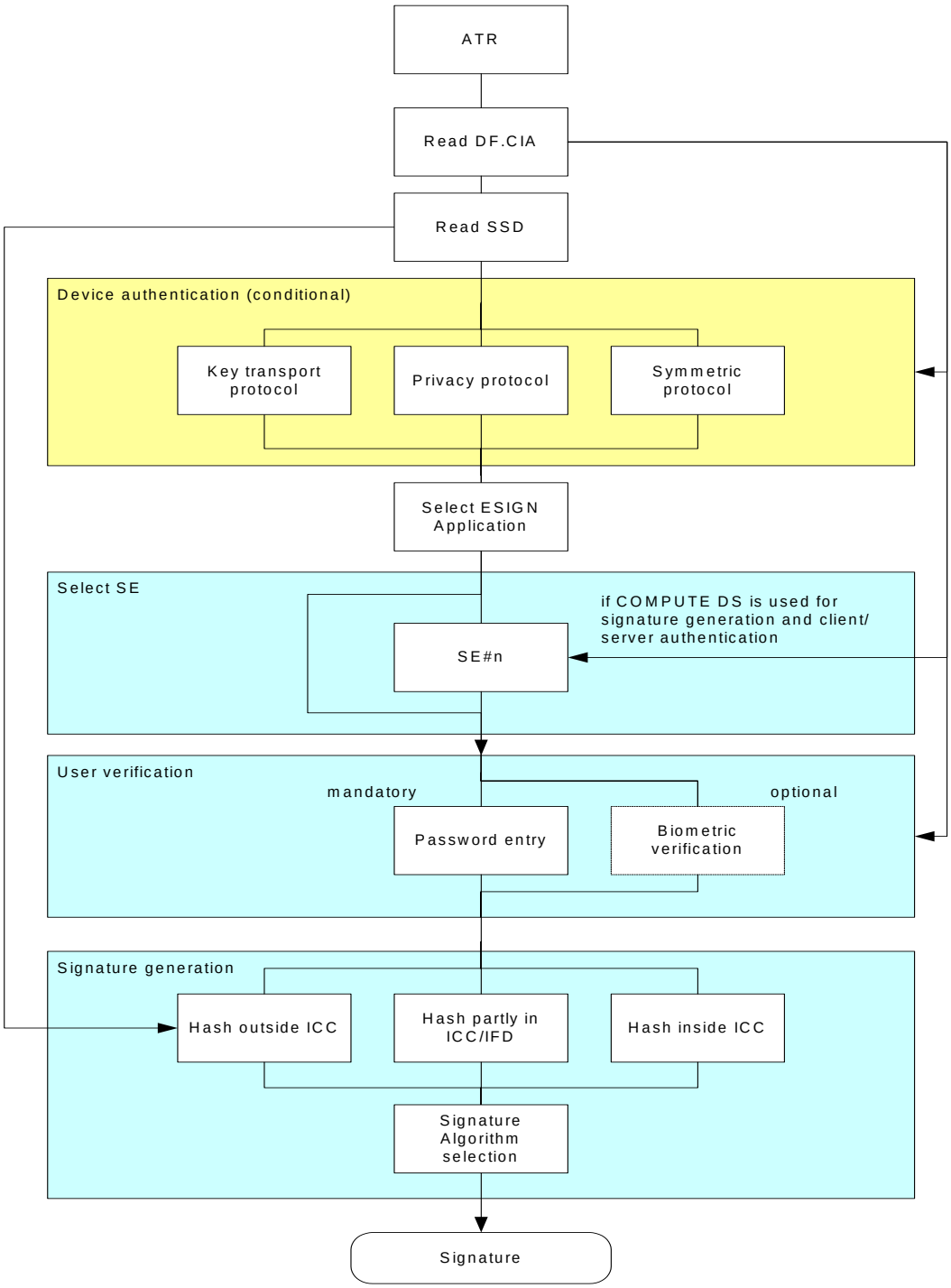


Figure C-1. Flow of selectable options for signature generation

C.2 Choosing User verification method

User verification shall always be done with a password. If biometric user verification is available in the IFD and ICC then in addition to the password verification the biometric user verification can be performed additionally, however it cannot be used instead of password verification.

Annex D. Example of DF.CIA

This example demonstrates the additional coding as part of DF.CIA described in 16 “Cryptographic Information Application” in Part 1 page 16-122.

-- The following header is required by the ASN.1 compiler in order to find the definitions of the following source.

```
ESIGN

DEFINITIONS ::= BEGIN
IMPORTS
   CardInfo, CIOChoice, AuthenticationObjectChoice,
   PrivateKeyChoice, CertificateChoice,
   PublicKeyChoice, SecretKeyChoice,
   DataContainerObjectChoice
    FROM
   CryptographicInformationFramework;

-- Definition of ISO7816-15 Directory Files
EFOD ::= SEQUENCE OF CIOChoice
EFAOD ::= SET OF AuthenticationObjectChoice
EFPrKD ::= SEQUENCE OF PrivateKeyChoice
EFCD ::= SEQUENCE OF CertificateChoice
EFPuKD ::= SEQUENCE OF PublicKeyChoice
EFSKD ::= SEQUENCE OF SecretKeyChoice
EFDCOD ::= SEQUENCE OF DataContainerObjectChoice
...............
```

-- The following is the actual source to be coded into the appropriate files

```
-- EF.PrKD
EFPrKD EFPrKD ::=
{
...............

-- SK.CH.AUT   Client-Server-Authentication Key
  privateRSAKey :
  {
    commonObjectAttributes
    {
      label "SK.CH.AUT",
      flags { private },
      accessControlRules
      {
        accessMode { execute },
        securityCondition and:
        {
          authId: '9B'H, -- Cross-Reference to PIN.CH.ADMIN
          authReference:
          {
            authMethod { userAuthentication },
            seIdentifier 10 -- reference to security environment #10
          } -- authReference
```

```
        } -- securityCondition and:
      } -- access ControlRules
    } -- commonObjectAttributes
  } -- privateRSAKey

  classAttributes
  {
    iD '05'H, -- shall share the same iD with Public Key and certificate
    usage { sign },
    native TRUE,
    accessFlags { sensitive, alwaysSensitive, neverExtractable,
                                                cardGenerated },
    keyReference 133, -- 0x85 KID in the card
  },

  typeAttributes
  {
    value indirect : path : {efidOrPath '3F 00 45 00'H },
    modulusLength 1024
  }
.................
}

-- EF.CD
EFCD EFCD ::=
{
.............

-- C.CH.AUT   Client-Server-Authentication Certificate
  x509Certificate :
  {
    commonObjectAttributes
    {
      label "C.X509.CH.AUT"
    },

    -- shall share the same id with the Private Key
    classAttributes
    {
      iD '05'H,
      authority FALSE
    },

    -- Path to EF.C.CH.AUT
    typeAttributes
    {
      value indirect : path :
      {
        efidOrPath '3F 00 40 02'H
      }
    } -- typeAttributes
  } -- x509Certificate
}, // closes file
```

```
-- EF.AOD
EFAOD EFAOD ::=
 {
  -- CV Certificate of the ICC for mutual device authentication
  -- and Secure Messaging
   external :
    {
      commonObjectAttributes
       { label "C_CV.ICC.AUT" },

      classAttributes
       { authId '07'H },

      typeAttributes
       certBasedAttributes :
      { cha '.....'H -- CHA of the C_CV.ICC.AUT certificate
      }
    }
 }


-- EF.DCOD
EFDCOD EFDCOD ::=
{
  -- Display Message
  opaqueDO :
  {
    commonObjectAttributes
    {
      label "Display Message",
      flags {private, modifiable},
      accessControlRules
      {
        {
          accessMode { read }, securityCondition
or:
          -- the following access condition is an optional
          -- application specific feature according to part 1 of
          -- the present specification:
          -- read after presenation of the related password
          -- in this example the global PIN and SE#10 are chosen
          {
and:        {
              authId: '9B'H,
              -- Cross-Reference to PIN.CH.ADMIN
              authReference:
              {
                authMethod { userAuthentication },
                seIdentifier 10
              }
            },
             -- the following access condition is mandatory
            -- according to part 1 of the present specification:
            -- read encrypted under secure messaging
```

```
            -- in this example an asymmetric device authentication
            -- and SE#11 are chosen
and:
            {
              authId: '07'H,
              -- Cross-Reference to CV certificate of
              -- the ICC in EF.AOD
              authReference:
              {
                authMethod
                {
                  secureMessaging,
                  extAuthentication},
                  seIdentifier 11
                }
              }
            }
          },
          -- the following access condition is mandatory
          -- according to part 1 of the present specification:
          -- update after presentation of the related password
          -- in this example the global PIN and SE#10 are chosen
          {
          accessMode { update }, securityCondition
and:
          {
            authId: '9B'H, -- Cross-Reference to PIN.CH.ADMIN
            authReference:
            {
              authMethod
              {
                userAuthentication
              },
              seIdentifier 10
            }
          }
        }
      }
    },

    -- applicationName
    classAttributes
    {
      applicationName "A0000000167455349474E"
    },

    typeAttributes
      indirect : path : {efidOrPath '3F 00 45 00 C0 00'H }
  },

 -- Security Service Descriptor
  opaqueDO :
  {
    commonObjectAttributes
    {
```

```
      label "Security Service Descriptor"
    },

    classAttributes
    {
      applicationName "A0000000167455349474E"
    },

    typeAttributes
    -- Path to EF.SSD
    indirect : path : { efidOrPath '3F 00 45 00 E4 00'H }
  }
}

END
```