INTERNATIONAL STANDARD

ISO/IEC 7816-8

Fourth edition
2019-08

Identification cards — Integrated circuit cards —

Part 8: Commands and mechanisms for security operations

Cartes d'identification — Cartes à circuit intégré —

Partie 8: Commandes et mécanismes pour les opérations de sécurité

Reference number
ISO/IEC 7816-8:2019(E)

ISO copyright office
CP 401 • Ch. de Blandonnet 8
CH-1214 Vernier, Geneva
Phone: +41 22 749 01 11
Fax: +41 22 749 09 47
Email: copyright@iso.org
Website: www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular, the different approval criteria needed for the different types of document should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO and IEC shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents) or the IEC list of patent declarations received (see http://patents.iec.ch).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation of the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see www.iso.org/iso/foreword.html.

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 17, *Cards and personal identification*.

This fourth edition cancels and replaces the third edition (ISO/IEC 7816-8:2016), which has been technically revised.

The main changes compared to the previous edition are as follows:

- the meaning of P1 in PSO command (even INS) has been improved by defining it as output data of the security operation, which was originally defined as expected response data;
- the meaning of P2 in PSO command (even INS) has been improved by defining it as input data to the security operation, which was originally defined as command data;
- where there is a choice from several values for P1 or P2 in PSO command (even INS), the meaning of choice '00' has been marked as to be used only for legacy reasons;
- in Table A.10, the format of the certificate content template DO’7F4E’ conveyed by PSO command has been corrected;
- in Table A.10, P1P2 value of PSO command has been corrected;
- in Annex D, the tag value of file reference DO has been corrected;
- in Annex D, the value of extended header list has been corrected.

A list of all parts in the ISO/IEC 7816 series can be found on the ISO website.

Any feedback or questions on this document should be directed to the user’s national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Introduction

ISO/IEC 7816 is a series of standards specifying integrated circuit cards and the use of such cards for interchange. These cards are identification cards intended for information exchange negotiated between the outside world and the integrated circuit in the card. As a result of an information exchange, the card delivers information (computation result, stored data) and/or modifies its content (data storage, event memorization).

- Five parts are specific to cards with galvanic contacts and three of them specify electrical interfaces:
  - ISO/IEC 7816-1 specifies physical characteristics for cards with contacts;
  - ISO/IEC 7816-2 specifies dimensions and location of the contacts;
  - ISO/IEC 7816-3 specifies electrical interface and transmission protocols for asynchronous cards;
  - ISO/IEC 7816-10 specifies electrical interface and answer to reset for synchronous cards;
  - ISO/IEC 7816-12 specifies electrical interface and operating procedures for USB cards.
- All the other parts are independent from the physical interface technology. They apply to cards accessed by contacts and/or by radio frequency:
  - ISO/IEC 7816-4 specifies organization, security and commands for interchange;
  - ISO/IEC 7816-5 specifies registration of application providers;
  - ISO/IEC 7816-6 specifies interindustry data elements for interchange;
  - ISO/IEC 7816-7 specifies commands for structured card query language;
  - ISO/IEC 7816-8 specifies commands for security operations;
  - ISO/IEC 7816-9 specifies commands for card management;
  - ISO/IEC 7816-11 specifies personal verification through biometric methods;
  - ISO/IEC 7816-13 specifies commands for handling the life cycle of applications;
  - ISO/IEC 7816-15 specifies cryptographic information application.

ISO/IEC 10536 (all parts) specifies access by close coupling. ISO/IEC 14443 (all parts) and ISO/IEC 15693 (all parts) specify access by radio frequency. Such cards are also known as contactless cards.

Identification cards — Integrated circuit cards —

Part 8: Commands and mechanisms for security operations

1 Scope

This document specifies interindustry commands which can be used for security operations. This document also provides informative directives on how to construct security mechanisms with commands defined in ISO/IEC 7816-4.

The choice and conditions of use of cryptographic mechanism in security operations can affect card exportability. The evaluation of the suitability of algorithms and protocols is outside the scope of this document. It does not cover the internal implementation within the card and/or the outside world.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 7816-4:—[1], *Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
asymmetric key pair
pair of elements belonging to cryptographic techniques that use two related operations: a public operation defined by public numbers or by a public *key* (3.4) and a private operation defined by private numbers or by a private key

Note 1 to entry: The two operations have the property that, given the public operation, it is computationally infeasible to derive the private operation.

3.2
certificate
digital signature (3.3) binding a particular person or object and its associated public *key* (3.4) (the entity issuing the certificate also acts as tag allocation authority with respect to the data elements in the certificate)

[SOURCE: ISO/IEC 7816-4:—, 3.11]

[1] Under preparation. Stage at the time of publication: ISO/IEC DIS 7816-4:2019.

3.3
digital signature

data appended to, or cryptographic transformation of, a data string that proves the origin and the integrity of the data string and protects against forgery, e.g. by the recipient of the data string

[SOURCE: ISO/IEC 7816-4:—, 3.20]

3.4
key

sequence of symbols controlling a cryptographic operation (e.g. encipherment, decipherment, a private or a public operation in a dynamic authentication, signature production, signature verification)

[SOURCE: ISO/IEC 7816-4:—, 3.31]

3.5
non-self-descriptive certificate

certificate (3.2) consisting of a concatenation of data elements associated to a header list or extended header list, describing the structure of the certificate

3.6
self-descriptive certificate

certificate (3.2) consisting of a concatenation of data objects

3.7
secure messaging

set of means for cryptographic protection of (parts of) command-response pairs

[SOURCE: ISO/IEC 7816-4:—, 3.50]

4 Abbreviated terms

BCD	binary-coded decimal
BER	basic encoding rules of ASN.1 (see ISO/IEC 8825-1)
CA	certification authority
CCT	control reference template for cryptographic checksum
CRT	control reference template
CT	control reference template for confidentiality
CVCA	country verifying CA
DG3	data group 3
DO	BER-TLV data object
DO'...'	BER-TLV data object, the tag of which is a hexadecimal value given between single quotation marks
DSA	digital signature algorithm
DST	control reference template for digital signature
DV	document verifier
ECC	elliptic curve cryptography

ECDH	elliptic curve Diffie–Hellman key exchange
ECDSA	elliptic curve digital signature algorithm
EF	elementary file
GQ2	modified Guillou-Quisquater protocol for zero knowledge proof
HT	control reference template for hash-code
ICAO	international civil aviation organization
ICC	integrated circuit card
IS	inspection system
KAT	control reference template for key agreement
LDS	logical data structure
MRTD	machine readable travel document
MSE	MANAGE SECURITY ENVIRONMENT command
OID	object identifier, as defined by ISO/IEC 8825-1
PSO	PERFORM SECURITY OPERATION command
RFU	reserved for future use for ISO/IEC JTC 1/SC 17
RSA	Rivest, Shamir, Adleman
SE	security environment
SEID	security environment identifier
TLV	tag, length, value

5 Interindustry commands for security operations

5.1 General

An ICC compliant with this document may support any of the commands and/or options provided in the following clauses and subclauses.

NOTE In addition to the use of logical channels, there are other alternatives that can be used for switching the security context. Annex D provides information about this functionality.

5.2 GENERATE ASYMMETRIC KEY PAIR command

The GENERATE ASYMMETRIC KEY PAIR command, which shall be as specified in Table 1, initiates either

— the generation and storing of an asymmetric key pair, i.e. a public key and a private key, in the card,

— the generation, storing of an asymmetric key pair and extracting generated public key, or

— the extracting previously generated public key.

The command may be preceded by a MANAGE SECURITY ENVIRONMENT command in order to set key generation related parameters (e.g. algorithm reference). The command may be performed in one or several steps, possibly using command chaining (see ISO/IEC 7816-4).

Table 1 — GENERATE ASYMMETRIC KEY PAIR command-response pair

CLA	As defined in ISO/IEC 7816-4
INS	'46' or '47'
P1	See Table 2
P2	'00' (no information provided) or reference of the private key to be generated coded according to ISO/IEC 7816-4:—, Table 102
L_c field	Absent for encoding $N_c = 0$, present for encoding $N_c > 0$
Data field	Absent, or Proprietary data if P1-P2 set to '0000', or One or more CRTs associated to the key generation if P1-P2 different from '0000' (see notes) A CRT may include an extended header list
L_e field	Absent for encoding $N_e = 0$, present for encoding $N_e > 0$

Data field	Absent, or Public key as a sequence of data elements (INS = '46'), or Public key as a sequence of data objects (INS = '47'), or Public key as a sequence of data objects according to an extended header list (INS = '47')
SW1-SW2	See ISO/IEC 7816-4:—, Tables 6 and 7 where relevant, e.g. 6985

NOTE 1 Several CRTs are present when the key pair is generated for several uses. In the command data field, a CRT possibly has a zero length.

Table 2 — P1 coding

b8	b7	b6	b5	b4	b3	b2	b1	Value
0	0	0	0	0	0	0	0	No information given
1	—	—	—	—	x	x	x	Additional information given
1	—	—	—	—	—	—	x	**Key generation**
1	—	—	—	—	—	—	0	— Generate asymmetric key pair
1	—	—	—	—	—	—	1	— Access to an existing public key
1	—	—	—	—	—	x	—	**Format of returned public key data**
1	—	—	—	—	—	0	—	— Proprietary format
1	—	—	—	—	—	1	—	— Output format according to extended header list
1	—	—	—	—	x	—	—	**Output indicator**
1	—	—	—	—	0	—	—	— Public key data in response data field
1	—	—	—	—	1	—	—	— No response data if Le field absent or proprietary if Le field present
—	x	x	x	x	—	—	—	0000, other values are RFU

NOTE 2 The private key can be stored in an internal EF the reference of which is known before issuing the command or in a DO'7F48' as cardholder private key template.

NOTE 3 The public part can be stored for example in a DO'7F49' as cardholder public key template.

For extracting a previously generated public key (i.e. no generation), the command data field shall be empty or shall contain a CRT, possibly including an extended header list.

NOTE 4 In those cases when only access to a previously generated public key is requested, P2 is either '00' or references the private key.

The response data field shall be either

- absent,
- a public key as a sequence of data elements (INS = '46'),
- a public key as a sequence of data objects (INS = '47') from Table 3, or

— a public key as a DO'7F49' (INS = '47') nesting data objects from Table 3.

If the command data field does not indicate any format of public key data, it shall be implicitly known before issuing the command (e.g. as part of the security environment). When the command data field indicates an extended header list within a CRT, it covers public key data objects and other requested data object.

EXAMPLE Annex E provides a set of examples on the use of this command.

If the algorithm is not indicated in the command, then the algorithm is known before issuing the command. In the public key template, the context-specific class (first byte from '80' to 'BF') is reserved for public key data objects.

Table 3 — Public key data objects

Tag		Value
'7F49'		Interindustry template for nesting one set of public key data objects with the following tags
	'06'	Object identifier of any further information, optional
	'80'	Algorithm reference as used in control reference data objects for secure messaging, optional
		Set of public key data objects for RSA
	'81'	Modulus (a number denoted as n coded on x bytes)
	'82'	Public exponent (a number denoted as v, e.g. 65 537)
		Set of public key data objects for DSA
	'81'	First prime (a number denoted as p coded on y bytes)
	'82'	Second prime (a number denoted as q dividing p-1, e.g. 20 bytes)
	'83'	Basis (a number denoted as g of order q coded on y bytes)
	'84'	Public key (a number denoted as y equal to g to the power x mod p where x is the private key coded on y bytes)
		Set of public key data objects for ECC
	'81'	Prime (a number denoted as p coded on z bytes)
	'82'	First coefficient (a number denoted as a coded on z bytes)
	'83'	Second coefficient (a number denoted as b coded on z bytes)
	'84'	Generator (a point denoted as PB on the curve, coded on 2z + 1 or 2z or z + 1 bytes)
	'85'	Order (a prime number denoted as q, order of the generator PB, coded on z bytes)
	'86'	Public key (a point denoted as PP on the curve, equal to x times PB where x is the private key, coded on 2z + 1 or 2z or z + 1 bytes)
	'87'	Co-factor
		Set of public key data objects for GQ2
	'81'	Modulus (a number denoted as n coded on x bytes)
	'83'	Number of basic numbers (a number denoted as m coded on 1 byte. If tag '83' is present, then tag 'A3' shall be absent and the m basic numbers denoted as g, g_2..g_m are the first m prime numbers 2, 3, 5, 7, 11...)
	'84'	Verification parameter (a number denoted as k coded on 1 byte)
	'A3'	Set of m basic numbers denoted as g, g_2..$g_{m,}$ each one coded on 1 byte with tag '80' (If tag 'A3' is present, then tag '83' shall be absent)
		Set of public key data objects for RSA Okamoto-Schnorr signature scheme[a]

NOTE In this context, ISO/IEC JTC 1/SC 17 reserves any other data object of the context-specific class (first byte in the range '80' to 'BF').

[a] The RSA Okamoto-Schnorr signature scheme, is considered a blind signature process, which is an interactive procedure between a signer and a recipient. It allows a recipient to obtain a signature of a message of the recipient's choice without giving the signer any information about the actual message or the resulting signature[8][9][10][11]. DO'73' may be used in the data field for returning a multi-part digital signature response comprised of concatenation of context-specific data objects defined by the application.

Table 3 *(continued)*

Tag		Value
	'81'	p the first large prime number
	'82'	q the second large prime number such that q|(p − 1), with q a divisor of (p − 1)
	'83'	Zp* the set of integers U modulo p such as 0 < U < p and gcd (U,p) = 1, gcd() being the greatest common divisor
	'84'	Zq* the set of integers U' modulo q such as 0 < U' < q and gcd (U',q) = 1
	'85'	g the first element of Zp* of order q such as g is a generator of Gq and Gq a cyclic group of prime order q
	'86'	h the second element of Zp* of order q different from g
	'87'	y the public key, an integer denoted as $y=g^{-r} h^{-s}$ mod p where (s,r) is the secret key, and s and r are two elements of (Zq*), and h of (Zp*)

NOTE In this context, ISO/IEC JTC 1/SC 17 reserves any other data object of the context-specific class (first byte in the range '80' to 'BF').

[a] The RSA Okamoto-Schnorr signature scheme, is considered a blind signature process, which is an interactive procedure between a signer and a recipient. It allows a recipient to obtain a signature of a message of the recipient's choice without giving the signer any information about the actual message or the resulting signature[8][9][10][11]. DO'73' may be used in the data field for returning a multi-part digital signature response comprised of concatenation of context-specific data objects defined by the application.

NOTE 5 For other Blind Signature schemes, e.g. Blind RSA signature (with data objects related to RSA), Blind Schnorr signature (with data objects related to DSA and/or ECDSA), Okamoto-Guillou-Quisquater blind signature scheme (with data objects related to GQ2), the OID under template '7F49' determines the nature and meaning of any further or different data objects, i.e. the following indications are possibly denoted by the OID

- blind signature type, e.g. RSA, Schnorr, Okamoto-Schnorr, Okamoto-Guillou-Quisquater),
- cryptographic Hash function,
- generic description of the token/credential (message) to be signed,
- attributes generic structure, and/or
- type of control upon signed message, i.e. partially blind, fully blind or restrictive blind signature (in some mechanisms, the signer does not totally lose control over the signed message since the signer can include explicit information in the resulting signature based on some agreement with the recipient. Such blind signatures are called partially blind signatures. Other mechanisms allow a recipient to receive a blind signature on a message not known to the signer but the choice of the message is restricted and conforms to certain rules. Such schemes are called restrictive blind signature mechanisms).

For the coding of the DO stating information about the private part of the key pair, Table 4 applies.

Table 4 — Private key data objects

Tag		Value
'7F48'		Interindustry template for nesting one set of private key data object with the following tags
	'82'	public exponent (optional)
	'92'	parameter p
	'93'	parameter q
	'94'	parameter 1/q mod p
	'95'	parameter d mod (p – 1)
	'96'	parameter d mod (q –1)
'7F48'		Interindustry template for nesting one set of ECDSA/ECDH private key data object with the following tags

NOTE In this context, ISO/IEC JTC 1/SC 17 reserves any other data object of the context-specific class (first byte in the range '80' to 'BF').

Table 4 *(continued)*

Tag		Value
	'92'	Private key
	'06'	object identifier of related curve (optional)
	or	
		curve information (optional):
	'93'	— p is the prime specifying the base field;
	'94'	— A 1st coefficient of the equation y^2 = x^3 + A*x + B mod p defining the elliptic curve;
	'95'	— B 2nd coefficient of the equation y^2 = x^3 + A*x + B mod p;
	'96'	— G = (x,y) base point, i.e., a point in E of prime order, with x and y being its x- and y-coordinates;
	'97'	— q prime order of the group generated by G;
	'98'	— h cofactor of G in E, i.e. #E[GF(p)]/q.
NOTE In this context, ISO/IEC JTC 1/SC 17 reserves any other data object of the context-specific class (first byte in the range '80' to 'BF').		

Annex C provides examples of exporting a public key and importing a private key.

5.3 PERFORM SECURITY OPERATION command

5.3.1 General

The PERFORM SECURITY OPERATION command, which shall be as specified in Table 5, initiates the following security operations:

- computations, such as
  - computation of a cryptographic checksum,
  - computation of a digital signature, or
  - computation of a hash-code;
- verifications, such as
  - verification of a cryptographic checksum,
  - verification of a digital signature, or
  - verification of a certificate;
- encipherment; or
- decipherment.

P1 defines output data of the security operation (see Table 6). P2 defines input data to the security operation (see Table 7). Values of tag of SM data object defined in ISO/IEC 7816-4 are used for P1 and P2.

P1 and P2 also define operation of this command. It depends on each operation defined in subsequent subclauses which value is used for P1 and P2. If the security operation requires several commands to complete, then command chaining may apply (see ISO/IEC 7816-4).

The PERFORM SECURITY OPERATION command may be preceded by a MANAGE SECURITY ENVIRONMENT command.

For example, the security object reference as well as the cryptographic mechanism reference shall be either implicitly known or specified in a CRT in a MANAGE SECURITY ENVIRONMENT command.

NOTE A security object reference is a reference of a secret key, a reference of a public key, a reference data, a reference for computing a session key or a reference of a private key. See ISO/IEC 7816-4.

Such a command can be performed only if the security status satisfies the security attributes for the operation. The successful execution of the command may be subject to successful completion of prior commands (e.g. VERIFY before the computation of a digital signature).

If present (e.g. implicitly known by the card or because it is part of the command data field), a header list or an extended header list defines the order and the data items that form the input for the security operation.

For this command, when a verification related operation is considered, SW1-SW2 set to '6300' or '63CX' indicates that a verification failed, 'X' ≥ '0' encodes the number of further allowed retries.

Table 5 — PERFORM SECURITY OPERATION command-response pair with INS = '2A'

CLA	As defined in ISO/IEC 7816-4
INS	'2A'
P1	See Table 6
P2	See Table 7
L_c field	Absent for encoding N_c = 0, present for encoding N_c > 0
Data field	Absent or value of the data object specified in P2
L_e field	Absent for encoding N_e = 0, present for encoding N_e > 0

Data field	Absent or value of the data object specified in P1
SW1-SW2	See ISO/IEC 7816-4:—, Tables 6 and 7 where relevant, e.g. 6985

Table 6 — P1 coding for output data of the security operation

Value	Meaning
'00'[a]	No output data
'80'	Plain value not encoded in BER-TLV
'82'	Cryptogram (plain value encoded in BER-TLV DO and including SM DOs)
'84'	Cryptogram (plain value encoded in BER-TLV DO, but not including SM DOs)
'86'	Padding-content indicator byte followed by cryptogram (plain value not encoded in BER-TLV DO)
'8E'	Cryptographic checksum
'90'	Hash-code
'9E'	Digital signature

NOTE Any other value is reserved for future use by ISO/IEC JTC 1/SC 17.

[a] P1 = '00' may be used for legacy reasons to indicate that output data is stored in the card and not returned in the response, i.e. output data is not present in the response data field.

Table 7 — P2 coding for input data to the security operation

Value	Meaning
'00'[a]	No input data
'80'	Plain value not encoded in BER-TLV

NOTE Any other value is reserved for future use by ISO/IEC JTC 1 SC 17.

[a] P2 = '00' may be used for legacy reasons to indicate that input data is already stored in the card and therefore not present in the command data field.

Table 7 *(continued)*

'82'	Cryptogram (plain value encoded in BER-TLV DO and including SM DOs)
'84'	Cryptogram (plain value encoded in BER-TLV DO, but not including SM DOs)
'86'	Padding-content indicator byte followed by cryptogram (plain value not encoded in BER-TLV DO)
'92'	Certificate (data not encoded in BER-TLV DO)
'9A'	Input data element for the computation of a digital signature
'A0'	Input template for the computation of a hash-code (the template is hashed)
'A2'	Input template for the verification of a cryptographic checksum (the template is integrated)
'A8'	Input template for the verification of a digital signature (the template is signed)
'AC'	Input template for the computation of a digital signature (the concatenated value fields are signed)
'AE'	Input template for the verification of a certificate (the concatenated value fields are certified)
'BC'	Input template for the computation of a digital signature (the template is signed)
'BE'	Input template for the verification of a certificate (the template is certified)

NOTE Any other value is reserved for future use by ISO/IEC JTC 1 SC 17.

[a] P2 = '00' may be used for legacy reasons to indicate that input data is already stored in the card and therefore not present in the command data field.

The PSO command with INS = '2B', which shall be as specified in Table 8, allows security operation commands with extensions. The functions are distinguished by function numbers in P1, defined in Table 9. Input DOs are conveyed in the command data field.

Optionally, the last DO is an extended header list describing the output.

Table 8 — PERFORM SECURITY OPERATION command-response pair with INS = '2B'

CLA	As defined in ISO/IEC 7816-4:—, 5.4.1	
INS	'2B'	
P1	'XX'	function number, to distinguish the different variants of PSO, see Table 9
P2	'00', output DO defined in command data field	
L_c field	present for encoding $N_c > 0$	
Data field	Either input DO(s) (see Table 7) or DO'73', and optionally an extended header list describing the output (primitive or constructed) (see Table 6)	
L_e field	Absent for encoding $N_e = 0$, present for encoding $N_e > 0$	

Data field	Absent or output either according to the extended header list or DO'73'
SW1-SW2	See ISO/IEC 7816-4:—, Tables 6 and 7 where relevant, e.g. 6985

Table 9 — Function numbers for PSO command

Value	Meaning
'01'	Compute cryptographic checksum
'02'	Compute digital signature
'03'	Hash operation
'04'	Verify cryptographic checksum
'05'	Verify digital signature
'06'	Verify certificate
'07'	Encipher
'08'	Decipher

NOTE Any other value is RFU.

5.3.2 through 5.3.9 provide further specifications for the case of the even INS code.

5.3.2 COMPUTE CRYPTOGRAPHIC CHECKSUM operation

The COMPUTE CRYPTOGRAPHIC CHECKSUM operation, which shall be as specified in Table 10, initiates the computation of a cryptographic checksum.

Table 10 — Parameters and data fields for COMPUTE CRYPTOGRAPHIC CHECKSUM operation

P1	'8E'
P2	'80'
Command data field	Data for which the cryptographic checksum is computed
Response data field	Cryptographic checksum

5.3.3 COMPUTE DIGITAL SIGNATURE operation

The COMPUTE DIGITAL SIGNATURE operation, which shall be as specified in Table 11, initiates the computation of a digital signature. The algorithm may be either a digital signature algorithm or a combination of a hash algorithm and a digital signature algorithm. Annex A provides examples of digital signature operations.

For the computation of a digital signature, the data to be signed or integrated in the signing process are transmitted in the command data field or submitted in a previous command, e.g. PSO command with hash operation. In P2, the digital signature is specified with tag values of SM data object '9A', 'AC' or 'BC' according to Table 6.

Table 11 — Parameters and data fields for COMPUTE DIGITAL SIGNATURE operation

P1	'9E' [a]
P2	'9A', 'AC' or 'BC' [b]
Command data field	Absent (data already in the card), or If P2 = '9A', data to be signed or integrated in the signature process, or If P2 = 'AC', data objects, the concatenated value fields of which are signed or integrated in the signature process, or If P2 = 'BC', data objects to be signed or integrated in the signature process
Response data field	Absent (digital signature stored in the card), or digital signature

[a] P1 = '00' may be used for legacy reasons to indicate that output data will be stored in the card and not returned in the response, i.e. output data is not present in the response data field.

[b] P2 = '00' may be used for legacy reasons to indicate that input data is already stored in the card and therefore not present in the command data field.

5.3.4 HASH operation

The HASH operation, which shall be as specified in Table 12, initiates the computation of a hash-code by performing either

— the complete computation inside the card, or

— a partial computation inside the card.

The HT (i.e. DO'AA' or DO'AB') indicates the algorithm by a cryptographic mechanism reference for computing a hash-code (see ISO/IEC 7816-4).

The input data shall be presented to the card in successive input blocks (one or more at a time), the length of which is algorithm dependent. Depending on the hash algorithm, the last input data have a length equal or shorter than the block length. The padding mechanism, if appropriate, is part of the definition of the hash algorithm.

Even if no data are transmitted (i.e. empty command data when P2 = '00'[2] or '80'), P1 shall be set to '90'. This is applicable, for example, when data is already in the card.

For the resulting hash-code, the following two cases shall be distinguished either

— the card stores the hash-code for a subsequent command; then the L_e field is not present, or

— the card delivers the hash-code in the response; then the L_e field shall be set to the appropriate length.

Table 12 — Parameters and data fields for HASH operation

P1	'90' [a]
P2	'80' or 'A0' [b]
Command data field	If P2 = '80', data to hash, or absent (e.g. for initialization or data already in the card), or If P2 = 'A0', data objects relevant for hashing (e.g. '90' for intermediate hash-code, '80' for data to hash)
Response data field	Absent (hash-code stored in the card), or hash-code

[a] P1 = '00' may be used for legacy reasons to indicate that output data will be stored in the card and not returned in the response, i.e. output data is not present in the response data field.

[b] P2 = '00' may be used for legacy reasons to indicate that input data is already stored in the card and therefore not present in the command data field.

5.3.5 VERIFY CRYPTOGRAPHIC CHECKSUM operation

The VERIFY CRYPTOGRAPHIC CHECKSUM operation, which shall be as specified in Table 13, initiates the verification of a cryptographic checksum.

Table 13 — Parameters and data fields for VERIFY CRYPTOGRAPHIC CHECKSUM operation

P1	'00'
P2	'A2'
Command data field	Data objects relevant to the operation (e.g. plain value DO'80' and / or cryptographic checksum DO'8E')
Response data field	Absent

NOTE The value field of DO'80' contains data or data elements covered by the value field of DO'8E' as cryptographic checksum.

5.3.6 VERIFY DIGITAL SIGNATURE operation

The VERIFY DIGITAL SIGNATURE operation, which shall be as specified in Table 14, initiates the verification of a digital signature delivered as a data object in the command data field. Other verification relevant data are either transmitted in a command chaining process or present in the card. The algorithm may be either a digital signature algorithm or a combination of a hash algorithm and a digital signature algorithm. Annex A provides examples of digital signature operations.

The public key as well as the algorithm may be either

— implicitly known,

— referenced in a DST (DO'B6') of a MANAGE SECURITY ENVIRONMENT command, or

— available as a result from a previous VERIFY CERTIFICATE operation.

If the algorithm reference in the card declares a signature only algorithm, then the data consists of a hash-code, or the signature is of message recovery type [see ISO/IEC 9796 (all parts)]. Otherwise,

[2] P2 = '00' may be used for legacy reasons.

the hash-code calculation is performed in the card and the algorithm reference additionally contains a reference to a hash algorithm.

Table 14 — Parameters and data fields for VERIFY DIGITAL SIGNATURE operation

P1	'00'
P2	'A8'
Command data field	Data objects relevant to the operation (e.g. either '9A', 'AC' or 'BC', and '9E')

Response data field	Absent

If the command data field contains an empty data object, then the card is expected to know all data relevant for verification.

5.3.7 VERIFY CERTIFICATE operation

The VERIFY CERTIFICATE operation, which shall be as specified in Table 15, verifies a certificate. For the verification of a certificate, the digital signature of a certificate to be verified is delivered as a data object in the command data field. Annex B provides relevant examples of how to implement this operation, which may help to better understand this subclause.

The public key of the certification authority to be used in the verification process is either implicitly selected or may be referenced in a DST using the MANAGE SECURITY ENVIRONMENT command. The algorithm to apply is implicitly known or may be referenced in a DST. If other data objects are to be used in the verification process (e.g. hash-code), then these data objects shall be present in the card or shall be transmitted using the command chaining process.

It is recommended for the public key of the certification authority to be on the card.

The following two cases shall be distinguished:

- if the certificate is self-descriptive (P2 = 'BE'), then the card retrieves a public key identified by its tag in the (recovered) certificate content;
- if the certificate is not self-descriptive (P2 = 'AE'), then the card retrieves a public key in the certificate either implicitly or explicitly by using the public key tag in a header list describing the content of the certificate.

If the retrieved public key is stored in the card, that key may be the default key for the subsequent operation (e.g VERIFY DIGITAL SIGNATURE).

Practical implementations recommend that tag '7F21' is not used in the data field, on behalf of the contained templates/DOs of the card verifiable certificates. The next edition of this document would possibly deprecate the use of '7F21' in this operation.

Table 15 — Parameters and data fields for VERIFY CERTIFICATE operation

P1	'00'
P2	'92', 'AE' or 'BE'
Command data field	Data elements or data objects relevant to the operation. If P2 = '92', data element to be used in the certificate verification process, (see ISO/IEC 7816-4:—, Table 50) If P2 = 'AE' or 'BE', data objects to be used in the certificate verification process. The allowed DOs may be those of a Card Verifiable Certificate (see ISO/IEC 7816-4:—, Table 50)

Response data field	Absent

If a partial message recovery scheme is used and part of the information is already stored in the card, then the data object for auxiliary data is expected to be sent empty, with the data to be inserted later by the card.

5.3.8 ENCIPHER operation

The ENCIPHER operation, which shall be as specified in Table 16, enciphers data transmitted in the command data field or data in a card. The usage of this operation may be restricted.

NOTE The operation can be used for generating diversified keys.

Table 16 — Parameters and data fields for ENCIPHER operation

P1	'82', '84', '86' (cryptogram according to ISO/IEC 7816-4:—, Table 50)
P2	'80' [a]
Command data field	Absent (data already in the card) or data to be enciphered

Response data field	Enciphered data as mandated by the P1 value

[a] P2 = '00' may be used for legacy reasons to indicate that input data is already stored in the card and therefore not present in the command data field.

5.3.9 DECIPHER operation

The DECIPHER operation, which shall be as specified in Table 17, deciphers data transmitted in the command data field. The usage of this operation may be restricted.

Table 17 — Parameters and data fields for DECIPHER operation

P1	'80' [a]
P2	'82', '84', '86' (cryptogram according to ISO/IEC 7816-4:—, Table 50)
Command data field	Data to be deciphered as mandated by the P2 value

Response data field	Absent (deciphered data remains in the card) or deciphered data

[a] P1 = '00' may be used for legacy reasons to indicate that output data will be stored in the card and not returned in the response, i.e. output data is not present in the response data field.

Annex A
(informative)

Examples of operations related to digital signature

A.1 General

This annex provides examples of how to operate with digital signatures.

A.2 Sequences of commands for managing a security environment

Table A.1 represents a sequence of MANAGE SECURITY ENVIRONMENT commands to SET DST, CCT and CT components of the current SE and finally, to STORE the current SE under a SEID indicated in P2.

Table A.1 — Setting of security environment components

Command	Operation	P1-P2	Command data field
MSE	SET DST	'41B6'	{'84' – L – <Key reference>} {'91 00'}
MSE	SET CCT	'41B4'	{'83' – L – <Key reference>} {'87' – L – <Initialization value>}
MSE	SET CT	'41B8'	{'83' – L – <Key reference>}
MSE	STORE (SEID = 1)	'F201'	Absent

The SET DST operation references the private key to use in the signature computation and specifies the integration of a random number in the digital signature input. The SET CCT operation references a secret key and an initial value to use for the computation of a cryptographic checksum. The SET CT operation references a secret session key to use for confidentiality.

A.3 Sequences of commands for digital signature computation

Table A.2 shows the syntax for producing a digital signature by using a signature scheme with appendix. The input is a hash-code completed with padding bytes. This example illustrates the calculation of a digital signature with combined algorithm including a hash operation. In this example, the hash input is delivered to the card.

Table A.2 — First example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	RESTORE	'F301'	Absent	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Hash-code with padding bytes	Digital signature

NOTE 1 This example is purely illustrative and its value is limited in terms of implementation as a result of possible export controls that can apply and indeed for general security reasons (avoidance of repeat signatures is desirable in some circumstances).

Table A.3 shows the syntax for producing a digital signature by using a signature scheme with appendix. The digital signature input consists of the hash-code without padding bytes.

Table A.3 — Second example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	RESTORE	'F301'	Absent	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Hash-code without padding bytes	Digital signature

NOTE 2 In order to avoid export restrictions, a combined signature and hash algorithm can be used.

NOTE 3 In some circumstances, avoidance of repeat signatures, although desirable, cannot be achieved.

Table A.4 shows a signature scheme with appendix. The digital signature input contains a hash-code without padding bytes delivered to the card and the card is requested to generate a random number as required in the extended header list of the DST in the command data field of the MSE command. As specified by tag 'BC' in P2, a concatenation of data objects (hash-code provided to the card and random number provided by the card) is signed.

Table A.4 — Third example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	SET	'41B6'	{'4D' – L – ('90' – L – '91' – L=0)}{'84' – L – <Key reference>}	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9EBC'	{'90' – L – <Hash-code>}	Digital signature

Table A.5 shows the syntax for digital signature with limited message recovery. The data to be signed are configured in accordance with a signature scheme giving limited message recovery using data objects presented in the command data field, whereby the digital signature counter is used as internal message provided by the card.

Table A.5 — Fourth example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	RESTORE	'F302'	Absent	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9EAC'	{'90' – L – <Hash-code>}	Digital signature

NOTE 4 Padding for computing the hash-code as well as the digital signature are according to ISO/IEC 9796-2.

In Table A.6, the card performs the hashing (or the last round of the hash computation). The digital signature input is empty in the COMPUTE DIGITAL SIGNATURE operation, since all input data are present in the card.

Table A.6 — Fifth example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	RESTORE	'F301'	Absent	Absent
PSO	HASH	'9080'	Data to hash	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Absent	Digital signature

In Table A.7, the card performs partly the hashing. The digital signature input is empty in the COMPUTE DIGITAL SIGNATURE operation since the intermediate hash is conveyed in the HASH operation.

Table A.7 — Sixth example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	SET	'41B6'	{'84' – L – <Key reference>}	Absent
PSO	HASH	'90A0'	{'90' – L – <intermediate hash followed by a bit counter> } {'80' – L – <data to be hashed (limited to one block)>}	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Absent	Digital signature

In Table A.8, the hashing is completely worked out outside the card. The digital signature input conveys the hash value computed off-card in the COMPUTE DIGITAL SIGNATURE operation.

Table A.8 — Seventh example of digital signature scheme with appendix

Command	Operation	P1-P2	Command data field	Response data field
MSE	SET	'41B6'	{'84' – L – <Key reference>}	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Hash value computed off-card	Digital signature

A.4 Sequences of commands for digital signature verification

In Table A.9, an extended header list specifies the construction of a non-self-descriptive certificate (see Annex B): the digital signature input consists of data elements. The VERIFY CERTIFICATE operation uses command chaining.

Table A.9 — First example of digital signature verification

Command	Operation	P1-P2	Command data field
MSE	SET DST	'41B6'	{'4D' – L – {'42' – L – '5F20' – L – '5F49' – L)} {'83' – L – <Key reference>}
PSO	VERIFY CERTIFICATE (CLA = '1X')	'00AE'	{'5F4E' – L – <Certificate content>}
PSO	VERIFY CERTIFICATE (CLA = '0X')	'00AE'	{'5F37' – L – <Digital signature of certificate>}
PSO	HASH	'9080'	Hash input
PSO	VERIFY DIGITAL SIGNATURE	'00A8'	{'9E' – L – <Digital signature>}

— As the first step, the certificate content data object is presented [concatenation of the data elements: issuer identification number (tag '42'), cardholder name (tag '5F20'), and cardholder public key (tag '5F49')]. The card performs the hashing using the certificate content as hash input.

— As a second step, the digital signature belonging to the certificate is re-transformed and the result is compared with the hash-code computed before. Then, the HASH operation is performed. For verifying the digital signature, the public key has been retrieved and verified by the previous VERIFY CERTIFICATE operation. The hash input is dependent on the hash algorithm, either the plain value, possibly presented in chained commands, or a pre-processed hash-code if the card performs only the last round of hash-code computation.

— As the final step, the VERIFY DIGITAL SIGNATURE operation is performed.

Table A.10 shows the verification of a self-descriptive certificate (see Annex B): the digital signature input consists of data objects. The VERIFY CERTIFICATE operation uses command chaining. In the first step, the data objects integrated in the certificate are presented (e.g. a concatenation of the data objects: certification authority reference, cardholder name and cardholder public key). The card uses this concatenation as hash input. Further steps are identical to those of the previous example.

Table A.10 — Second example of digital signature verification

Command	Operation	P1-P2	Command data field
MSE	SET DST	'41B6'	{'83' – L – <Key reference>}

PSO	VERIFY CERTIFICATE (CLA='1X')	'00BE'	{'7F4E' – L – <Certificate content template>
			{'5F29' – L – <Certificate profile identifier>}
			{'42' – L – <Issuer identification number>}
			{'5F20' – L – <Cardholder name>}
			{'5F49' – L – <cardholder public key>}
			}

PSO	VERIFY CERTIFICATE (CLA='0X')	'00BE'	{'5F37' – L – <Digital signature of certificate>}
PSO	HASH	'9080'	Hash input
PSO	VERIFY DIGITAL SIGNATURE	'00A8'	{'9E' – L – <Digital signature>}

Table A.11 shows the usage of a public key previously installed in the card.

Table A.11 — Third example of digital signature verification

Command	Operation	P1-P2	Command data field
MSE	SET DST	'41B6'	{'83' – L – <Key reference>}
PSO	HASH	'90A8'	Hash input
PSO	VERIFY DIGITAL SIGNATURE	'00A8'	{'9E' – L – <Digital signature>}

A.5 CHA-Certificate Holder Authorization Data Object (CHA-DO)

Table A.12 — Certification Holder Authorization Data Object

Tag	'5F4C'
Purpose	Encodes the role of the holder (i.e. CVCA, DV, IS) and assigns read/write access rights to data groups storing sensitive data
Format	Refer to below

The "Certificate Holder Authorization" (CHA) shown in Table A.12 describes access rights of the certificate holder. It is a general DO in a CV certificate, which is used to identify (i.e. access) rights of the certificate holder. The meaning of CHA can be compared with a role based key identifier.

The CHA consists of the application reference/ID for which the authorization is valid, and the role identifier.

The CHA is formatted as shown in Table A.13.

Table A.13 — CHA format

Prefix	Value
AID of related DF (6 most significant bytes)	Role Identifier

— AID of the related DF (6 most significant bytes) specifies the AID of the DF where the authorization is valid.

— Role Identifier indicates the privileges granted within the application whose AID was specified before.

Table A.14 shows a coding example of the role identifier implemented by CEN TC 224.

Table A.14 — Coding example of the role identifier for non-self-descriptive certificates by CEN TC 224

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	x	x	x	x	x	x	x	Bits b7 to b1 code the value as described
1	0	0	0	0	0	—	—	Role Identifier coded in two bytes
—	—	—	—	—	—	0	0	Role Identifier coded as number in the subsequent byte
—	—	—	—	—	—	0	1	Role Identifier coded as bit map in the subsequent byte; the "effective" Role identifier is calculated by a logical AND of the Role-ID of CA public key and the Role-ID given in the certificate
—	—	—	—	—	—	1	0	Role Identifier coded as bit map in the subsequent byte; the bits set in the subsequent byte are correlated to order further Data Elements with 'FF' separator
Other values								Assigned by CEN TC 224

Table A.15 shows a coding example of the role identifier implemented by CEN TC 224 if two bytes are indicated through the first byte value '82'.

Table A.15 — Coding Example of role identifier with two-byte header by CEN TC 224

Byte 1	Byte 2								Meaning
	b8	b7	b6	b5	b4	b3	b2	b1	
10 000 010	—	—	—	—	—	—	—	1	Community ID verification
	—	—	—	—	—	—	1	—	Age verification with finite value
	—	—	—	—	—	1	—	—	Age verification with inclusive range
	Other values								Assigned by CEN TC 224

Table A.16 shows an example of the coding of the payload if bits b3 to b1 in Table A.15 are all set to 1.

Table A.16 — Role Identifier's further data elements (example with inclusive range)

Data Element	separator	Data Element	separator	Data Element
Community ID verification value	'FF'	Age lower limit	'FF'	Age upper limit

In this case, the role ID for public key of CA should be set to '00'.

When the certificate is used in a certificate verification chain to certify a public key of a CA, it is coded as described in the following paragraph.

A.6 COMPUTE DIGITAL SIGNATURE with two subsequent signature commands

Table A.17 shows the syntax for producing a digital signature by using two commands. The final signature, e.g. a DSA- or a EC-DSA-Signature (r,s), consists of two parts which are computed separately by the two signature commands.

Table A.17 — Example of signature creation by two subsequent signature commands

Command	Operation	P1-P2	Command data field	Response data field
MSE	SET DST	'41B6'	{'84' – L – <Key reference>} – {'91 00'}	Absent
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Absent	Pre-signature, e.g. part r

Table A.17 *(continued)*

Command	Operation	P1-P2	Command data field	Response data field
PSO	COMPUTE DIGITAL SIGNATURE	'9E9A'	Randomized hash-code value	Digital signature, e.g. part s

This is applicable, e.g. for randomized hashing according to NIST/SP 800-106; if the randomizer of the signature is used in the computation of the hash value of the message.

A.7 Sequence of commands for self-descriptive card verifiable certificate verification

The most common use case of the VERIFY CERTIFICATE operation is using it together with GENERAL AUTHENTICATE or EXTERNAL AUTHENTICATE, as indicated in ISO/IEC 7816-4.

The verification of a card verifiable certificate requires two steps: selection of the verification key and verification of the certificate together with the import of the public key. Generally, the top key of a certificate chain available in the ICC is the public key of the root CA. The two steps shall be performed repeatedly until the correct terminal public key is available in the ICC. The ICC verifies this certificate with the public key set in the previous step. If the verification fails, the ICC responds with an appropriate status word. When successful, the ICC stores the public key contained in the certificate and allows a further import of certificates along the certificate chain. This process is called multi-stage verification.

The role of the certificate holder should comply with its signer. The selection of root CA public key can always be accepted to verify a chained certificate. Any other key that was imported with a certificate may be applied only if the role of its certificate matches the purpose of verification of the next incoming certificate.

The MSE command selects the key to be used for verification of the certificate to be received in the consecutive PSO command as shown in Table A.18.

Table A.18 — Example for certificate verification

Command	Operation	P1-P2	Command data field	Response data field
MSE	SET DST	'81B6'	{'83' – L – <Issuer identification number>} Key reference of root CA public key or other key in chain	Absent
PSO	VERIFY CERTIFICATE	'00BE'	{'7F4E' – L – <certificate content template>} – {'5F37' – L – <digital signature>}	Absent

The key reference of root CA public key or any other key in the chain that is temporarily imported is the certificate authority reference CAR coded in the certificate.

An example of a certificate structure applied in the Extended Access Control (EAC) protocol is given in B.5. The transmission of the self-descriptive certificate is done without the tag '7F21', i.e. only the relevant data objects from the value field of the certificate will be transmitted.

The response data field is absent.

Annex B
(informative)

Examples of certificates interpreted by the card

B.1 General

This annex defines examples of how certificates can be interpreted by the card.

B.2 Data objects for card-verifiable certificates

Table B.1 shows data objects relevant for card-verifiable certificates. These data objects are extracted from ISO/IEC 7816-6, which supersedes their definition in the following table. The definitions are copied in this table just for improving the readability.

Table B.1 — Interindustry data objects relevant for card-verifiable certificates (non-exhaustive list)

Tag	Data element
'06'	Object identifier
'42'	Issuer identification number
'5F20'	Cardholder name
'5F24'	Certificate expiration date
'5F25'	Certificate effective date
'5F29'	Certificate profile indicator
'5F37'	Static internal authentication (signature of a certificate, produced by the issuer)
'5F49'	Cardholder public key
'5F4C'	Certificate holder authorization
'5F4E'	Certificate content
'65'	Cardholder related data (e.g. certificate extensions)
'7F21'	Cardholder certificate
'7F49'	Public key template
'7F4C'	Certificate holder authorization template (CHAT)
'7F4E'	Certificate content template

NOTE In order to provide certificate extension, the DO'73' (i.e. Discretionary Data Template) can be used directly without being encapsulated into the DO'65'. See Table B.7.

The issuer may specify further data objects such as certificate serial number, version number, expiration date, etc.

Two different structures of card-verifiable certificates shall be distinguished:

- a self-descriptive card-verifiable certificate consists of a concatenation of BER-TLV data objects;
- a non-self-descriptive card-verifiable certificate consists of a concatenation of data elements.

B.3 Self-descriptive card-verifiable certificates

For the signature creation of a certificate, a digital signature scheme with or without message recovery is used. Table B.2 shows an example of a self-descriptive card-verifiable certificate with a digital signature scheme without message recovery.

Table B.2 — Construction for self-descriptive card-verifiable certificate of a cardholder

'7F21'	Length	Value of subsequent data objects	

		{'7F4E' – L – <Certificate content template>
		{'5F29' – L – <Certificate profile identifier>}
		{'42' – L – <Issuer identification number>}
		{'5F20' – L – <Cardholder name>} –
		{'5F49' – L – <Cardholder public key>}
		}
			{'5F37' – L – <Digital signature>}

Tag of the certificate (constructed)	Length of the certificate	DO certificate content template	DO'7F4E' (including tag-field and length-field of DO'7F4E') is signed

NOTE 1 The identification data of the certification authority can reference this public key.

NOTE 2 The identification data of the cardholder can be used for controlling access rights to data stored in the card.

NOTE 3 The public key of the cardholder can be used in a subsequent VERIFY DIGITAL SIGNATURE operation.

B.4 Non-self-descriptive card-verifiable certificates

An extended header list data object may be present in the card to verify this type of certificate; otherwise, it should be protected when delivered to the card. An extended header list data object (tag '4D', see ISO/IEC 7816-4) describes the concatenation of data elements by tag/length pairs in the same order as in the digital signature (See Table B.3).

Table B.3 — Construction for non-self-descriptive card-verifiable certificate of a cardholder

'7F21'	Length	Value of subsequent data objects		
		{'4D' – L – ('42' – L – '5F20' – L – '5F49' – L)}	{'5F4E' – L – <Issuer identification number – Cardholder name – Cardholder public key>}	{'5F37' – L – <Digital signature>}

Tag of the certificate (constructed)	Length of the certificate	Extended header list (present only if the certificate structure is not implicitly known)	Certificate content data object integrated in the signature (present only in the absence of message recovery, it contains the data elements according to the extended header list)	The data elements are signed:
				— Issuer identification number;
				— Cardholder name;
				— Cardholder public key.

B.5 Self-descriptive card verifiable certificates

B.5.1 General

This subclause provides an example of a self-descriptive certificate as it is typically used in the protocol Extended Access Control (EAC), see ISO/IEC 7816-4. The certificate is card verifiable and is delivered as a certificate content template followed by a signature data object. Only the shaded parts shown in Table B.4 are transmitted to the ICC. The transmission of the certificate is done without the tag '7F21'.

Table B.4 — Structure and content of a self-descriptive CV certificate (example)

'7F21'	**'82 01 BA'**	CV certificate					
		'7F4E'	**'82 01 7A'**				
				'5F29'	'01'	'00'	Certificate Profile Identifier
				'42'	'0E'	'44 45 43 ...'	Certificate Authority Reference
				'7F49'	'81 FD'	See Table B.5	Public key
				'5F20'	'0E'	'64 65 63 ...'	Certificate Holder Reference
				'7F4C'	'0E'	See Table B.6	Certificate Holder Authorization Template (CHAT)
				'5F25'	'06'	'00 07 00 04 00 01'	Certificate Effective Date: 2007-APR-01
				'5F24'	'06'	'00 09 00 04 00 01'	Certificate Expiration Date: 2009-APR-01
				'65'	'2F'	See Table B.7 (optional data object)	
		'5F37'	**'38'**	See B.5.10			Digital Signature: ECDSA

B.5.2 Certificate profile identifier

For self-descriptive certificates, the certificate profile identifier defines the list of data objects used to construct the data to be signed and the order of the data objects, but not their length.

B.5.3 Certification authority reference

The "Certification Authority Reference" (CAR) has the purpose of identifying the certificate issuing CA in such a way that the DO can be used at the same time as an authority key identifier.

B.5.4 Public key

The encoding of the public key and the domain parameters contained in the certificate is explained in Table B.5.

Table B.5 — Structure and content the public key template

'7F49'	**'81 FD'**	Public key template			
		'06'	**'0A'**	'04 00 7F 00 07 02 02 02 02 02'	**Object Identifier**: *Terminal Auth. with ECDSA-SHA-224*
		'81'	**'1C'**	'D7 C1 34 ... C8 C0 FF'	Prime modulus **p**
		'82'	**'1C'**	'68 A5 E6 ... D2 9F 43'	First coefficient **a**
		'83'	**'1C'**	'25 80 F6 ... 6C 40 0B'	Second coefficient **b**
		'84'	**'39'**	'04 0D 90 ... 14 02 CD'	Base point **G** (uncompressed format)
		'85'	**'1C'**	'D7 C1 34 ... A7 93 9F'	Order of the base point **r**
		'86'	**'39'**	'04 39 3E ... 13 9E 14'	Public point **Y**
		'87'	**'01'**	'01'	Cofactor **f**

B.5.5 Certificate holder reference

This data object provides a unique name for the public key nested within the certificate.

B.5.6 Certificate holder authorization template

In the example, the discretionary data object contained in the CHAT identifies a CVCA that allows read access to DG3 according to the ICAO LDS (See Table B.6).

Table B.6 — Structure and content of certificate holder authorization template

'7F4C'	**'0E'**	Certificate holder authorization template			
		'06'	**'09'**	'04 00 7F 00 07 03 01 02 01'	**Object identifier**: *Inspection system to access MRTDs*
		'53'	**'01'**	'C1'	**Role and access rights**: *CVCA with right 'Read DG3*

B.5.7 Certificate effective date

The certificate effective date (CED) defines the date from which the certificate is valid. The date is coded in 6 bytes using unpacked BCD as follows: 0Y 0Y 0M 0M 0D 0D. Other formats are out of the scope of B.5.

B.5.8 Certificate expiration date

The certificate expiration date CXD defines the date after which the certificate expires. The date is coded in 6 bytes using unpacked BCD as follows: 0Y 0Y 0M 0M 0D 0D.

B.5.9 Certificate extension

Optionally, a certificate may contain certificate extensions (see Table B.7). This gives the opportunity to import arbitrary data in an authentic way. In the example, the discretionary data object contained in the certificate extension authenticates auxiliary data by its hash value to be used in subsequent protocol steps that are out of scope of this document.

Table B.7 — Structure and content of the certificate extension

'65'	**'2F'**	Certificate extension				
		'73'	**'2D'**	Discretionary data template		
				'06'	**'09'**	Object identifier
				'80'	**'20'**	HASH value (SHA-256)

B.5.10 Digital signature (ECDSA)

For self-descriptive certificates, the data to be signed is the complete data object denoted by tag '7F4E' including this tag and the corresponding length field.

ECDSA signatures in plain format are given as a direct concatenation of two byte strings R||S. For ECDSA-224, each byte string has length 28 (decimal) and the signature is of length 56 bytes (decimal).

Annex C
(informative)

Examples of asymmetric key transfer

C.1 Usage of the GET DATA command for public key export

It is assumed that the data objects describing a public key are present in the card coded in a form as shown in Table C.1.

Table C.1 — Coding for public key data objects present in the card

'A8'	L	T-L pair to indicate a template for digital signature verification				
		'B6'	L	DST		
				'83'	L	Key reference of public key
		'7F49'	L	Public key data object		
				'81'	L	Modulus
				'82'	L	Public exponent
		'9E'	L	Digital signature (all bytes of the digital signature verification template preceding tag '9E' are signed)		

With the MSE command, the public key to be retrieved is selected. Then the GET DATA command (odd INS, P1-P2 = '3FFF') is used in three steps, whereby the data fields shown in Table C.2 to Table C.7 occur at the card interface.

Table C.2 — Data field of the GET DATA command, step 1 of 3

'4D'	'0B'	Extended header list						
		'A8'	'09'	T-L pair to indicate a template for digital signature verification				
				'B6'	'02'	T-L pair that indicates a DST data object		
						'83'	'00'	T-L pair that indicates a public key reference
				'7F49'	'02'	T-L pair that indicates the public key data object		
						'81'	'00'	T-L pair that indicates the modulus

Table C.3 — Data field of the GET DATA response, step 1 of 3

'A8'	L					
		'B6'	L	DST		
				'83'	L	Key reference of exported public key
		'7F49'	L	Public key		
				'81'	L	Modulus

Table C.4 — Data field of the GET DATA command, step 2 of 3

'4D'	'07'	Extended header list						
		'A8'	'05'	T-L pair to indicate a template for digital signature verification				
				'7F49'	'02'	T-L pair that indicates the public key data object		
						'82'	'00'	T-L pair that indicates the public exponent

Table C.5 — Data field of the GET DATA response, step 2 of 3

'A8'	L					
		'7F49'	L	Public key		
				'82'	L	Public exponent

Table C.6 — Data field of the GET DATA command, step 3 of 3

'4D'	'04'	Extended header list				
		'A8'	'02'	T-L pair to indicate a template for digital signature verification		
				'9E'	'00'	T-L pair that indicates digital signature data object

Table C.7 — Data field of the GET DATA response, step 3 of 3

'A8'	L			
		'9E'	L	Digital signature

C.2 Usage of the PUT DATA command for private key import

C.2.1 Example for referencing the corresponding private key

Initially, an MSE command should be sent in order to reference the corresponding private key (i.e. the key reference is already known to the ICC). Then the PUT DATA command (odd INS, P1-P2 = '3FFF') is sent for import of a private key with a command data field shown in Table C.8 or Table C.9. Both Table C.8 and Table C.9 show data formats that convey an extended header list followed by key parameters with a digital signature.

Table C.8 — Data field of the PUT DATA command for RSA private key object import

'4D'	L	Extended header list						
		'A8'	L	T-L pair to indicate a template for digital signature verification				
				'B6'	L	T-L pair to indicate a DST		
						'84'	L	T-L pair to indicate a key reference to SK.CH.DS
				'7F48'	L	T-L pair to indicate a private key data object		
						'92'	L	T-L pair for parameter p
						'93'	L	T-L pair for parameter q
						'94'	L	T-L pair for parameter 1/q mod p
						'95'	L	T-L pair for parameter d mod (p-1)
						'96'	L	T-L pair for parameter d mod (q-1)
				'9E'	L	T-L pair to indicate a digital signature		
'5F48'	L	Concatenation of the key parameter data elements according to the extended header list. Data elements that are associated to filler tags '00' in the extended header list are read but ignored.						
'9E'	L	Digital signature						

Table C.9 — Data field of the PUT DATA command for ECDSA private key object import

'4D'	L	Extended header list						
		'A8'	L	T-L pair to indicate a template for digital signature verification				
				'B6'	L	T-L pair to indicate a DST		
						'84'	L	T-L pair to indicate a key reference of private key
				'7F48'	L	T-L pair to indicate a private key data object		
						'92'	L	T-L pair to indicate a private key
						'06'	L	T-L pair to indicate an OID (optional)
				'9E'	L	T-L pair to indicate a digital signature		
'5F48'	L	Concatenation of the key parameter data elements according to the extended header list. Data elements that are associated to filler tags '00' in the extended header list are read but ignored.						
'9E'	L	Digital signature						

C.2.2 Example of private key import under secure conditions

Alternative to the private key import where the empty DO structure of the key is delivered to the ICC along with key value and related signed data elements, this subclause describes a data field example incorporating the key parameters values directly within their DO structure (see Table C.10).

Table C.10 — Data field of the PUT DATA command for RSA private key import under secure conditions

'A4' or 'B6'	'03'	Key usage: Authentication template ('A4') or Digital Signature template ('B6')		
		'84'	'01'	Key ID
'7F48'	L	RSA private key data objects		
		'92'	L	parameter p
		'93'	L	parameter q
		'94'	L	parameter 1/q mod p
		'95'	L	parameter d mod (p – 1)
		'96'	L	parameter d mod (q –1)

Annex D
(informative)

Alternatives to achieve the reversible change of security context

In addition to the use of logical channels to switch reversely the security context, further alternatives may be used as follows.

a) During a secured session, the ICC receives from a first off-card entity an MSE SET command with a CRT containing a further DO'E0' or 'E1' which is a private class tag intended respectively to instruct the ICC application about

- the references of the dynamic data generated during the running course of the transaction with the current security environment, e.g. session keys, send sequence counter and that should be saved i.e. maintained by the ICC in volatile or non-volatile memory so that to be recovered and re-used later (template 'E0');
- the security context that should be established next through the references provided in the CRT, e.g. Algorithm identifier (DO'80'), key identifier as DO'83' or '84' (template 'E1').

Once such MSE SET with a CRT containing 'E0' is received by the ICC, the ICC turns ready to expect an incoming command from a second off-card entity that is assumed to initiate a secure channel according to the security context established by the previous MSE SET.

Once the transaction is ended with the second off-card entity and the secure context needs to reverse to its former situation, the second off-card entity sends an MSE SET command containing the template 'E1'.

Upon reception of such an MSE SET command containing the template 'E1', the ICC should expect from the first off-card entity to resume the secure transaction that was temporarily held on, with the same dynamic data, e.g. session keys, send sequence counter, etc. Accordingly, from now on, as soon as an incoming command is received, it should be processed by the ICC application with the cryptographic secrets shared/involved with the first off-card entity, thus assuming this off-card entity is resuming the transaction. To this end, the ICC should recover the dynamic data stored at the first step, i.e. when MSE SET with CRT nesting DO'E0' was performed.

DO'E0' and DO'E1' may nest context-specific DO for which the interpretation is application-specific, e.g. the identifier of the session context assigned by the local terminal.

b) To achieve the same goal as described in alternative a), this document can rely on existing ISO commands (refer to ISO IEC 7816-4) as follows:

- a SELECT DATA command with a data field as {'4F' – L – <AID>} {'51' – L – <file reference>} {'4D' – '04' – ('7B' – '02' – 'A4' – '00')} pointing as example on a CRT-AT (DO'A4') that occurs in the current SE followed with a PUT DATA command initiating a security context shift with a data field of this kind: {CRT-CC} {CRT-CCT} setting as example the CRT for integrity and confidentiality in the current SE;
- once such PUT DATA is received by the ICC application, it should behave as indicated in alternative a);
- once the transaction is ended with the second off-card entity, a PUT DATA command may reverse the security context with a data field of this kind: {CRT-CC initial} {CRT-CCT initial} setting as example the CRTs involved with the second off-card entity to their initial value.