INTERNATIONAL STANDARD

ISO/IEC 18013-3

Second edition
2017-04

Information technology — Personal identification — ISO-compliant driving licence —

Part 3: Access control, authentication and integrity validation

Technologies de l'information — Identification des personnes — Permis de conduire conforme à l'ISO —

Partie 3: Contrôle d'accès, authentification et validation d'intégrité

Reference number
ISO/IEC 18013-3:2017(E)

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

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. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

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).

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

For an explanation on 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 the following URL: www.iso.org/iso/foreword.html.

The committee responsible for this document is ISO/IEC JTC 1, *Information technology*, SC 17, *Cards and personal identification*.

This second edition cancels and replaces the first edition (ISO/IEC 18013-3:2009), which has been technically revised. It also incorporates the Amendments ISO/IEC 18013-3:2009/Amd 1:2012 and ISO/IEC 18013-3:2009/Amd 2:2014, and the Technical Corrigenda ISO/IEC 18013-3:2009/Cor 1:2011 and ISO/IEC 18013-3:2009/Cor 2:2013.

The most significant changes are the following:

— In the interest of interoperability of cards used for personal identification, the authentication protocols for the IDL are simplified. Active Authentication is harmonised with other ISO standards and thus BAP configurations 2, 3 and 4, as well as EAP are no longer supported by this document.

— Replacing BAP, the optional EACv1 protocol is defined for the IDL, enabling access control to sensitive biometric data stored on an integrated circuit. EACv1 may be used in conjunction with either BAP configuration 1 or PACE.

— The optional PACE protocol enables access control to the data stored on an integrated circuit. The PACE protocol is a password authenticated Diffie Hellman key agreement protocol based on a (short) input string that provides secure communication between a secure integrated circuit on an IDL and a terminal and allows various implementation options (mappings, input strings, algorithms). The PACE protocol implementation for the IDL is restricted to Elliptic Curve Diffie Hellman (ECDH) generic mapping and can be used as a stand-alone protocol or in combination with the EACv1 protocol.

A list of all the parts in the ISO/IEC 18013 series can be found on the ISO website.

Introduction

This document prescribes requirements for the implementation of mechanisms to control access to data recorded in the machine-readable technology on an ISO-compliant driving licence (IDL), verifying the origin of an IDL, and confirming data integrity.

One of the functions of an IDL is to facilitate international interchange. While storing data in machine-readable form on the IDL supports this function by speeding up data input and eliminating transcription errors, certain machine-readable technologies are vulnerable to being read without the knowledge of the card holder and to other means of unauthorized access by unintended persons that is other than driving licence or law enforcement authorities. Controlling access to IDL data stored in machine-readable form protects the data on the card from being read remotely by electronic means without the knowledge of the card holder.

Identifying falsified driving licences or an alteration to the human-readable data on authentic driving licences present a major problem for driving licence and law enforcement authorities, both domestically and in the context of international interchange. Verifying the authenticity of an IDL and confirming the integrity of the data recorded on an IDL provide driving licence and law enforcement authorities with a means to identify an authentic IDL from a falsified or altered one in the interests of traffic law enforcement and other traffic safety processes.

Information technology — Personal identification — ISO-compliant driving licence —

Part 3:
Access control, authentication and integrity validation

1 Scope

ISO/IEC 18013 establishes guidelines for the design format and data content of an ISO-compliant driving licence (IDL) with regard to human-readable features (ISO/IEC 18013-1), machine-readable technologies (ISO/IEC 18013-2), and access control, authentication and integrity validation (ISO/IEC 18013-3). It creates a common basis for international use and mutual recognition of the IDL without impeding individual countries/states to apply their privacy rules and national/community/regional motor vehicle authorities in taking care of their specific needs.

This document

— is based on the machine-readable data content specified in ISO/IEC 18013-2;

— specifies mechanisms and rules available to issuing authorities (IAs) for:

  — access control (i.e. limiting access to the machine-readable data recorded on the IDL),

  — document authentication (i.e. confirming that the document was issued by the claimed IA), and

  — data integrity validation (i.e. confirming that the data has not been changed since issuing).

This document does not address issues related to the subsequent use of data obtained from the IDL, e.g. privacy issues.

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 1831:1980, *Printing specifications for optical character recognition*

ISO/IEC 7816-4:2013, *Identification cards — Integrated circuit cards — Part 4: Organization, security and commands for interchange*

ISO/IEC 8859-1:1998, *Information technology — 8-bit single-byte coded graphic character sets — Part 1: Latin alphabet No. 1*

ISO 9796-2, *Information technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Integer factorization based mechanisms*

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

ISO/IEC 10118-3:2004, *Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash-functions*

1) ISO/IEC 9797-1:1999 is withdrawn and replaced by the 2011 version.

ISO/IEC 11770-2:1996[2], *Information technology — Security techniques — Key management — Part 2: Mechanisms using symmetric techniques*

ISO/IEC 11770-2:1996/Cor.1:2005, *Information technology — Security techniques — Key management — Part 2: Mechanisms using symmetric techniques — Technical Corrigendum 1*

ISO/IEC 18013-1, *Information technology — Personal identification — ISO-compliant driving licence — Part 1: Physical characteristics and basic data set*

ISO/IEC 18013-2, *Information technology — Personal identification — ISO-compliant driving licence — Part 2: Machine-readable technologies*

ISO/IEC 18033-3:2005[3], *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers*

ISO/IEC 18033-3:2005/Cor1:2006, *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers — Technical Corrigendum 1*

ISO/IEC 18033-3:2005/Cor2:2007, *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers — Technical Corrigendum 2*

ANSI X9.62:2005, *Public Key Cryptography for the Financial Services Industry: The Elliptic Curve Digital Signature Algorithm (ECDSA)*

BSI Technical Guideline TR-03110-1: *Advanced Security Mechanisms for Machine Readable Travel Documents — Part 1 — eMRTDs with BAC/PACEv2 and EACv1 — Version 2.10 — 2012-03-20*

BSI Technical Guideline TR-03110-3: *Advanced Security Mechanisms for Machine Readable Travel Documents — Part 3 — Common Specifications — Version 2.10 — 2012-03-20*

FIPS 186-2 (including Change Notice), *Digital Signature Standard (DSS), Federal Information Processing Standards Publication, National Institute of Standards and Technology, 27 January 2000*

ICAO Technical Report – *Supplemental Access Control for Machine Readable Travel Documents, v1.01, 2010 [TR-PACE*]

NIST/SP 800-38B, *Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication*, *May 2005*

RFC 2631, E. Rescorla, *Diffie-Hellman Key Agreement Method*, *June 1999*[4]

RFC 3279, *W. Polk et al., Algorithms and Identifiers for the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, April 2002* [4]

RFC 3280, *R. Housley et al., Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, April 2002* [1]

RFC 3369, *R. Housley, Cryptographic Message Syntax, August 2002* [1]

RFC 4055, *J. Schaad, B. Kaliski, R. Housley, Additional Algorithms and Identifiers for RSA Cryptography for use in the Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile, June 2005* [1]

RFC 5639, *M. Lochter, J. Merkle, Elliptic Curve Cryptography (ECC) Brainpool Standard Curves and Curve Generation, March 2010* [1]

2) ISO/IEC 11770-2:1996 is withdrawn and replaced by the 2008 version.

3) ISO/IEC 18033-3:2005 is withdrawn and replaced by the 2010 version.

4) http://www.ietf.org/rfc.html

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 18013-1, ISO/IEC 18013-2 and the following apply.

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

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

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

3.1
active authentication
mechanism that uses information stored in a secure area of a *secure integrated circuit (SIC)* (3.17) to confirm that the SIC and the other machine-readable data were issued together

Note 1 to entry: See 8.2.

3.2
basic access protection
BAP
mechanism to confirm that an inspection system (IS) has physical access to a proximity integrated circuit card (PICC) before the IS is allowed access to the data stored on the PICC and to ensure that communication between the IS and the PICC (once access is authorized) is protected

Note 1 to entry: See 8.5 and Annex B.

3.3
chip authentication
ephemeral-static key agreement protocol that provides authentication of the *secure integrated circuit* (3.17) and strong secure messaging

Note 1 to entry: See 8.6.

3.4
clone
unauthorized exact copy of a document that has the same security characteristics as the original document and that cannot be distinguished from the legitimate one

3.5
eavesdropping
unauthorized interception and interpretation of information-bearing emanations

[SOURCE: ISO/IEC 2382-8:2015, 08.05.25, modified]

3.6
extended access control v1
EACv1
protocol used to limit access to optional signature and biometric data groups

Note 1 to entry: See 8.6 and Annex D.

3.7
input string
string of characters printed on an ISO-compliant driving licence [as human-readable text, optionally (or by specification) accompanied by or consisting of a machine-readable rendering thereof] used as input (either manually or automatically through the use of suitable equipment) for the non-match alert and *BAP* (3.2) or *PACE* (3.10) mechanisms

3.8
issuing authority
IA
licensing authority (or issuing country if separate licensing authorities have not been authorized) which applies a digital signature to an ISO-compliant driving licence and is responsible for the associated key management

[SOURCE: ISO/IEC 18013-1]

3.9
non-match alert
mechanism to detect any differences between the machine-readable information and (some of) the human-readable information on an ISO-compliant driving licence

Note 1 to entry: See 8.4.

3.10
PACE
alternative mechanism to BAP to confirm that an inspection system (IS) has physical access to a *secure integrated circuit (SIC)* (3.17) on a driving licence card before the IS is allowed to access to the data stored on the SIC and to establish a secure communication channel between the IS and SIC once access is authorised

Note 1 to entry: As stated in TR-PACE, PACE refers to PACE v2.

Note 2 to entry: See 8.7 and Annex C.

3.11
passive authentication
mechanism to confirm that machine-readable data on an ISO-compliant driving licence (IDL) has not been changed since the IDL was issued

Note 1 to entry: See 8.1.

3.12
pseudo issuing authority
PIA
authority that does not issue ISO-compliant driving licences [but that is similar to an *issuing authority (IA)* (3.8) in all other respects] and which does not issue document keys, but which does have a root key pair with which it can sign documents of other IAs or PIAs that it trusts

3.13
public key infrastructure
PKI
technologies and products using public key (asymmetric) cryptography

Note 1 to entry: Both *passive authentication* (3.11) and extended access protection use this technology.

3.14
reading authority
RA
authorized entity reading the machine-readable data on an ISO-compliant driving licence (IDL)

Note 1 to entry: Driving licence authorities other than the authority that issued the IDL and law enforcement authorities are examples of reading authorities.

3.15
reference string
string of characters used as a reference against which to compare the *input string* (3.7) when using the *non-match alert* (3.9) mechanism, and used for session key calculation purposes by the *secure integrated circuit* (3.17) during execution of the *basic access protection* (3.2) mechanism

3.16
scanning area identifier
SAI
one or more graphical elements that demarcate an *input string* (3.7)

3.17
secure integrated circuit
SIC
integrated circuit that includes both a security feature (or security features), and memory and/or a central processing unit

Note 1 to entry: An integrated circuit card with contacts and a proximity integrated circuit card (PICC) are examples of a SIC.

Note 2 to entry: A SIC can be embedded in different solutions, for example in ID-1 sized cards (as used for the ISO-compliant driving licence) and in a booklet (as found in passports).

3.18
secure memory
integrated circuit (IC) memory of which the content [once populated by an *issuing authority* (3.8) during the personalization process] is accessible only by the IC operating system for internal use, and cannot be made available by the operating system to any reading device

3.19
skimming
reading data from a proximity integrated circuit card (PICC) without the card holder's awareness

3.20
trust chain
sequential set of *trust points* (3.23) that a *verifying authority* (3.26) references to verify a specific *issuing authority's* (3.8) public root key

3.21
trust model
description of the functional and logical aspects of a traditional *public key infrastructure* (3.13), specifically excluding technical implementation details

3.22
trust network
component of a *trust model* (3.21) that describes the trust relationships and chains between issuing authorities

3.23
trust point
issuing authority (3.8) or *pseudo issuing authority* (3.12) that publishes a trust list (and the related public root keys) that verifying entities can reference

3.24
twinning
copying the data and/or integrated circuit of a physically and/or biometrically similar driver to the attacker's integrated circuit or ISO-compliant driving licence

3.25
unpacked BCD
binary coding of a sequence of integers using 4 bits for each integer (where the bit weights are 8421) and encoding one integer in the least significant bits of each byte

Note 1 to entry: Only unsigned BCD is used in this document.

3.26
verifying authority
VA
verifying entity (3.27) that is part of a *trust network* (3.22), i.e. that also is an *issuing authority* (3.8) or a *pseudo issuing authority* (3.12)

Note 1 to entry: Not all verifying entities are VAs. A car rental company can be a verifying entity, but is not a VA as it is not part of the trust network.

Note 2 to entry: VAs can be divided into *immediate VAs* (3.26.1) and *non-immediate VAs* (3.26.2).

3.26.1
immediate VA
VA (3.26) that acquired the public root key of the *issuing authority* (3.8) via out-of-band means

3.26.2
non-immediate VA
VA (3.26) that acquired the public root key of the *issuing authority* (3.8) from another VA

3.27
verifying entity
entity that tries to determine if a digital signature is valid (i.e. if the data to which a certificate has been applied has not been changed, and if the signature was generated by the *issuing authority* (3.8) the verifying entity expects)

4 Abbreviated terms

APDU	application protocol data unit
BAC	basic access control
BAP	basic access protection
BCD	binary coded decimal
BER-TLV	basic encoding rules – tag-length-value (see ISO/IEC 8825-1:2002[a])
CA	certification authority
CBC	cipher block chaining
DER	distinguished encoding rules (see ISO/IEC 8825-1:2002)
DF	dedicated file
DG	data group
DO	data object
DST	control reference template for digital signature (see ISO/IEC 7816-4:2013)
EACv1	extended access control v1
EAP	extended access protection
EF	elementary file
IA	issuing authority
IC	integrated circuit

ICC	integrated circuit card
IDL	ISO-compliant driving licence
IFD	interface device
IS	inspection system
IV	initialization vector
KAT	control reference template for key agreement (see ISO/IEC 7816-4:2013)
LDS	logical data structure
MAC	message authentication code
MF	master file
MRTD	machine readable travel document
MRZ	machine readable zone
MSE	manage security environment (see ISO/IEC 7816-4:2013)
OCR	optical character recognition
OID	object identifier
PACE	password authenticated connection establishment
PIA	pseudo issuing authority
PIC	proximity integrated circuit
PICC	proximity integrated circuit card
PKI	public key infrastructure
PSO	perform security operation (see ISO/IEC 7816-4:2013)
RA	reading authority
RFU	reserved for future use
SAI	scanning area identifier
SIC	secure integrated circuit
SM	secure messaging
SOD	document security object
SSC	send sequence counter
TRCA	trust root certificate authority
UTC	coordinated universal time

VA verifying authority

2D two-dimensional

[a] ISO/IEC 8825-1:2002 is withdrawn and replaced by the 2015 version.

5 Conformance

A driving licence is in conformance with this document if it meets all mandatory requirements specified directly or by reference herein. Compliance with ISO/IEC 18013-2 is required for compliance with this document.

Compliance with ISO/IEC 18013-1 is not required for compliance with this document. Conversely, the incorporation of a machine-readable technology which is not compliant with this document does not render the IDL non-compliant with ISO/IEC 18013-1.

6 Functional requirements

6.1 Access control

Access control can be broken down into the following functional requirements:

a) prevent skimming of machine-readable data on a PICC by ensuring that physical access to the IDL is acquired prior to reading;

b) prevent unnoticed alteration of communication between a reader and a SIC;

c) prevent eavesdropping between a reader and a SIC;

d) selectively restrict access to specific optional machine-readable data groups for specific reading authorities.

6.2 Document authentication

Document authentication can functionally be established by allowing for verification of the origin of an IDL.

6.3 Data integrity validation

Data integrity validation can be broken down into the following functional requirements:

a) Verify that the IDL (including the machine-readable data) is not a clone of another IDL. A cloning attempt can schematically be illustrated as shown in Figure 1.

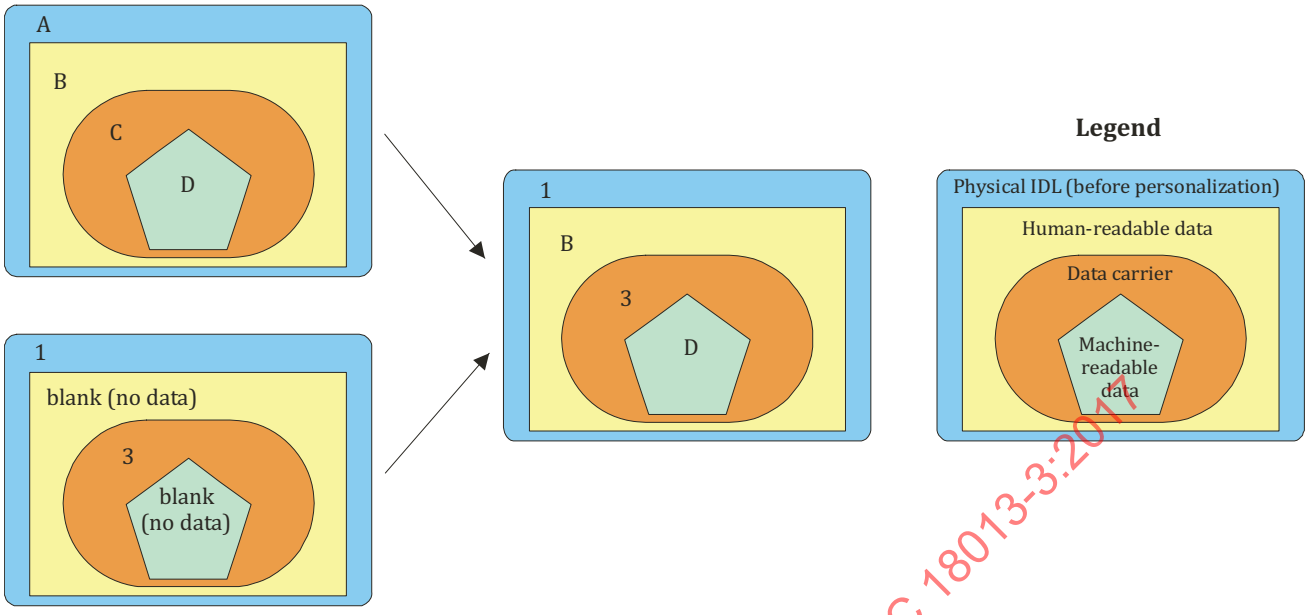


Figure 1 — Data integrity validation: IDL cloning

b) Protect against the exchange of machine-readable data carriers between otherwise authentic IDLs. This type of attack can schematically be illustrated as shown in Figure 2.

NOTE This guards (among others) against an IC "twinning" attack. This type of attack is of particular concern in inspection environments where machine-readable data and human-readable data is not compared (or only cursorily compared by an operator using, for example, a portrait image). Finding a biometrically similar driver is possible by skimming the data of a few thousand IDL PICs.

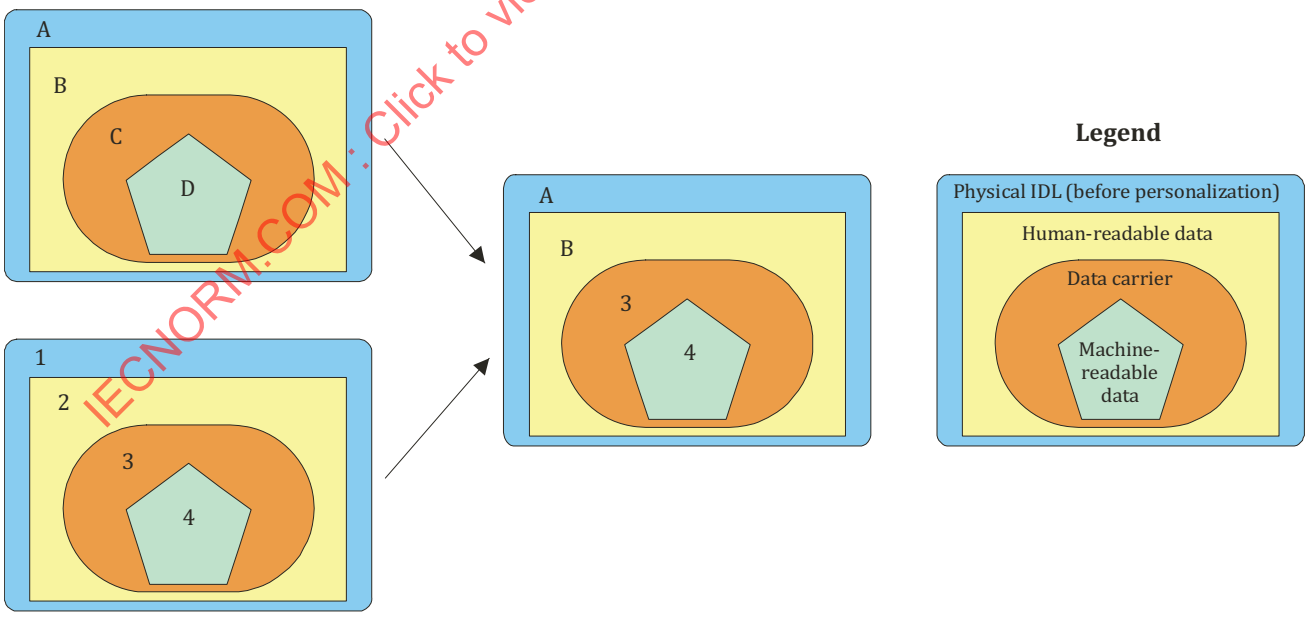


Figure 2 — Data integrity validation: Data carrier exchange or twinning

c) Verify that the physical IDL and the machine-readable data thereon were issued (belong) together. This type of attack can schematically be illustrated as shown in Figure 3.

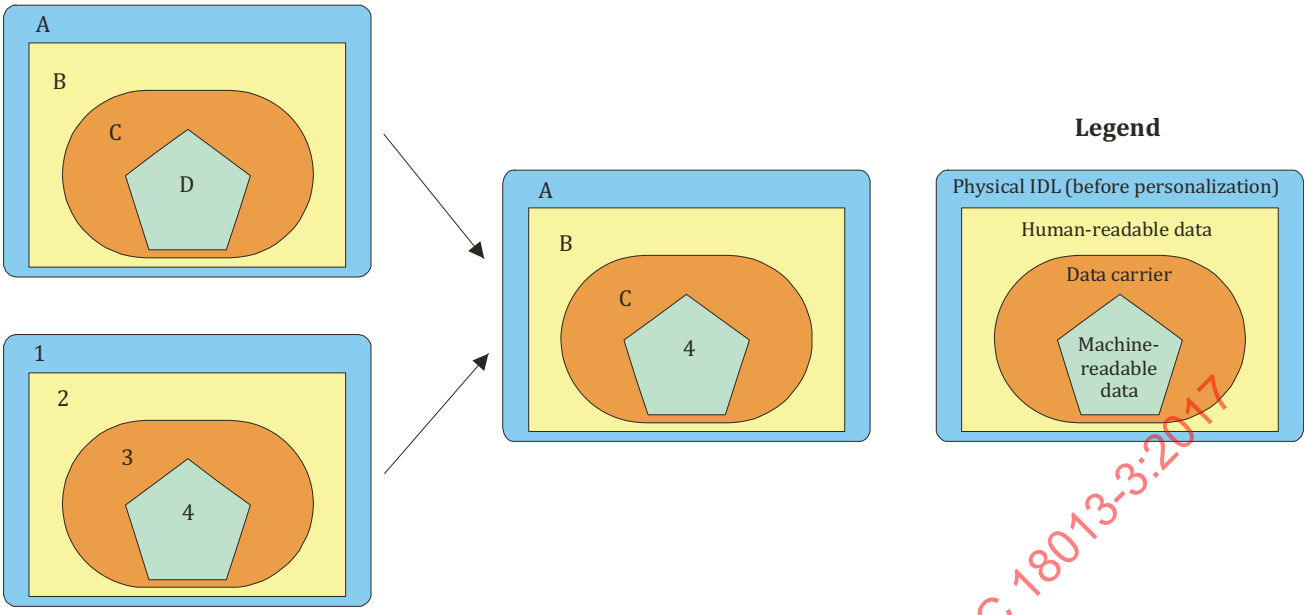


Figure 3 — Data integrity validation: Machine-readable data exchange

d) Validate the integrity of the human readable data (i.e. confirm that the human-readable data has not changed since issuing). This type of attack can schematically be illustrated as shown in Figure 4.

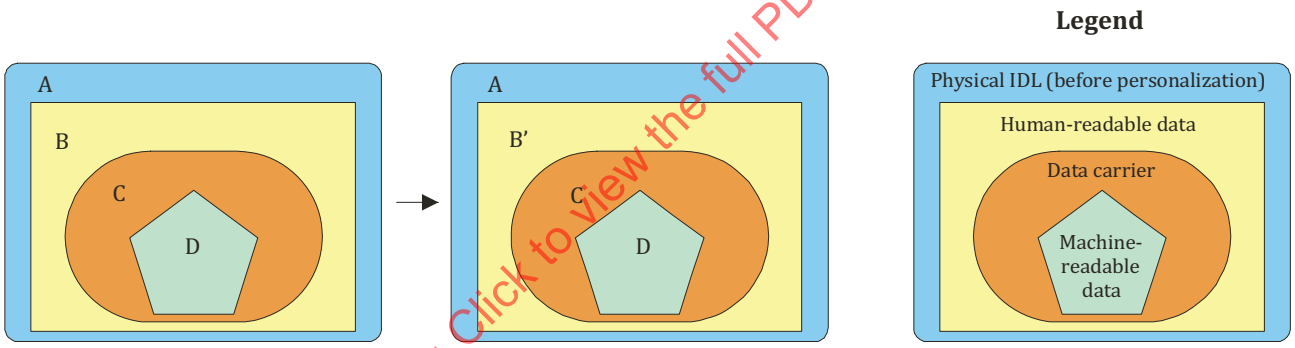


Figure 4 — Data integrity validation: Human-readable data alteration

e) Validate the integrity of the machine-readable data (i.e. confirm that the machine-readable data has not changed since issuing). This can schematically be illustrated as shown in Figure 5.

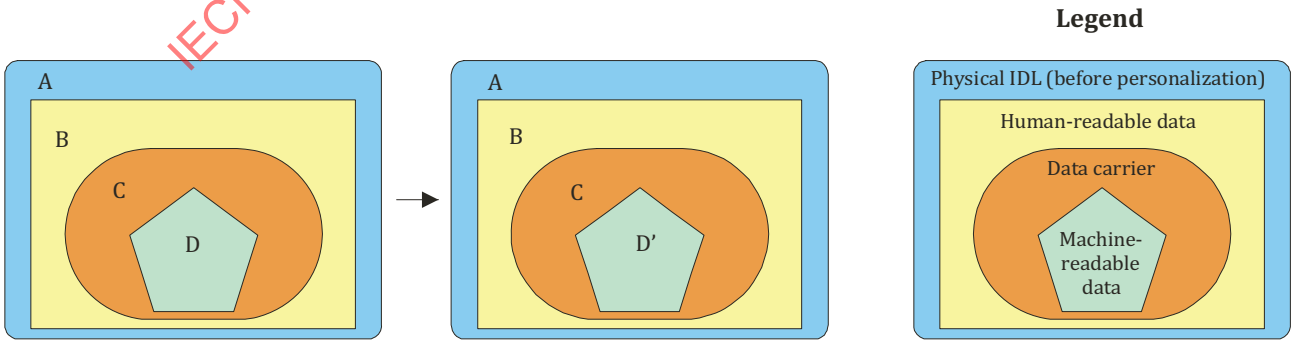


Figure 5 — Data integrity validation: Machine-readable data alteration

7 Mapping of mechanisms to requirements and technologies

Mechanisms that, when used individually, can address specific functional requirements are specified in Clause 8. Mechanisms can also be used together to address a broader range of functional requirements. Not all machine-readable technologies have the same functional requirements, and some mechanisms are applicable only to specific machine-readable technologies. Table 1 specifies the relationships between the mechanisms, machine-readable technologies, and functional requirements.

Table 1 — Applicable mechanisms

Functional requirement	Mechanisms		
	2D bar code	ICC with contacts	PICC
Prevent skimming	Physical access to the IDL is required to read the machine-readable data	Physical access to the IDL is required to read the machine-readable data	BAP or PACE
Prevent unnoticed alteration of communication between a reader and an IDL	N/A	Chip authentication (EACv1) in combination with BAP or PACE secure messaging	BAP or PACE Chip authentication (EACv1) in combination with BAP or PACE secure messaging
Prevent eavesdropping	N/A	Chip authentication (EACv1) in combination with BAP or PACE secure messaging	BAP[a] or PACE Chip authentication (EACv1) in combination with BAP or PACE secure messaging
Selectively restrict access to specific optional machine-readable data groups for specific reading authorities	N/A	Terminal authentication (EACv1)	Terminal authentication (EACv1)
Allow for verification of the origin of an IDL	Step 1: Verify trustworthiness of/obtain a trustworthy asymmetric/public key (via a PKI and a protected distribution communication channel, or via alternative trust building methods) Step 2: Use trusted key to perform passive authentication, followed by use of the non-match alert mechanism, or followed by visual comparison of the human-readable data printed on the document with the authenticated machine-readable data	Step 1: Verify trustworthiness of/obtain a trustworthy asymmetric/public key (via a PKI and a protected distribution communication channel, or via alternative trust building methods) Step 2: Use trusted key to perform passive authentication; if the passive authentication was preceded by BAP or PACE, the IDL is authentic while if BAP or PACE is not implemented, passive authentication can be followed by use of the non-match alert mechanism, or followed by visual comparison of the human-readable data printed on the document with the authenticated machine-readable data	Step 1: Verify trustworthiness of/obtain a trustworthy asymmetric/public key (via a PKI and a protected distribution communication channel, or via alternative trust building methods) Step 2: Use trusted key to perform passive authentication; if the passive authentication was preceded by BAP or PACE, the IDL is authentic while if BAP or PACE is not implemented, passive authentication can be followed by use of the non-match alert mechanism, or followed by visual comparison of the human-readable data printed on the document with the authenticated machine-readable data
Verify that the IDL (including the machine-readable data) is not a clone of another IDL	2D bar code: Cannot be verified through available international standards	Active authentication via challenge response Chip authentication (EACv1)	Active authentication via challenge response Chip authentication (EACv1)
Protect against the exchange of machine readable data carriers between otherwise authentic IDLs	Non-match alert Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data

Table 1 *(continued)*

Functional requirement	Mechanisms		
	2D bar code	ICC with contacts	PICC
Verify that the physical IDL and the machine-readable data thereon were issued (belong) together (i.e. that the machine-readable data has not been copied from another IDL)	Non-match alert Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data Active authentication via challenge response[a]	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data Active authentication via challenge response[a]
Validate the integrity of the human readable data (i.e. confirm that the human-readable data has not changed since issuing)	Non-match alert Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data Visual inspection of the security features incorporated into the IDL (see ISO/IEC 18013-1)	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data Visual inspection of the security features incorporated into the IDL (see ISO/IEC 18013-1)	Non-match alert BAP or PACE Passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data Visual inspection of the security features incorporated into the IDL (see ISO/IEC 18013-1)
Validate the integrity of the machine-readable data (i.e. confirm that the machine-readable data has not changed since issuing)	Passive authentication	Passive authentication	Passive authentication

NOTE Visual comparison of human-readable information printed on an IDL with machine-readable information obtained from the same IDL is not formally specified as a mechanism in this document; it is assumed that such comparison can be implemented by any RA in a manner that meets local needs.

[a] The entropy of the reference string used has to be commensurate with the IA's assessment of the threat.

IA's may selectively implement the mechanisms identified above, subject to any dependencies noted in Clause 8.

8 Mechanisms

8.1 Passive authentication

8.1.1 Purpose

The purpose of passive authentication is to confirm that machine-readable data has not been changed since the IDL was issued.

8.1.2 Applicability

Passive authentication is applicable to all machine-readable technologies.

8.1.3 Description

Passive authentication is implemented by way of a digital signature over specified machine-readable data on the IDL, using a public-private (asymmetric) key pair.

In the case of standard encoding, a separate message digest is calculated for each data group and included in the machine-readable data. The collection of message digests is then digitally signed (using a private key that is kept secret by the IA) and the digital signature is added to the machine-readable data.

In the case of compact encoding, no message digests are calculated separately. The contents of the data groups present is directly signed (using a private key that is kept secret by the IA) and the digital signature is added to the machine-readable data.

NOTE A message digest has the following properties:

a) It is very small in size compared to the IDL data.

b) The probability of finding any two (different) IDL data sets that lead to the same message digest is negligible. This has the following implications:

   1) The probability of finding an IDL data set A that produces the same message digest as a given IDL data set B is negligible.

   2) The probability that a message digest (for the data on an IDL) remains the same upon a change in the data is negligible.

When the IDL is presented to a RA, the RA uses the IA's public key to verify the digital signature. The RA also computes the message digests of each of the data groups that it is interested in and compares them to the corresponding message digests stored in the machine-readable data. If the following conditions are met, the RA can consider the data groups that it is interested in to be authentic:

a) the digital signature verifies;

b) the calculated message digests are the same as the message digests stored in the machine-readable data;

c) the RA is confident that the public key used to verify the digital signature belongs to the claimed IA.

If the digital signature does not verify, either an incorrect public key was used or the data on the IDL has been changed. Depending on the digital signature method used, it may be possible to further narrow down the cause of the non-verification.

This document does not prescribe methods to obtain and/or to establish trust in public keys. It is the responsibility of each RA to obtain and/or to establish trust in the public keys used to verify a digital signature on an IDL. However, informative methods and approaches to establish such trust are provided in Annex A, which describes the principles for a PKI that may be used for public key distribution in the absence of one global certification authority.

This document does not prescribe methods for the generation, administration and safekeeping of key pairs. It is the responsibility of each issuing jurisdiction to ensure that keys are generated, administered and protected as necessary.

8.1.4 Hash function

8.1.4.1 Standard encoding

For standard encoding, IA's shall choose the SHA-1, SHA-224, SHA-256, SHA-384 or the SHA-512 hash function.

SHA-256 is recommended. SHA-1 remains for compatibility with ICAO Doc 9303-1.

A message digest is calculated separately for each data group present and stored in the machine-readable data (see 8.1.5.1). The same hash function is used for all data groups. A message digest for a data group is calculated on the concatenation of those data elements present in the data group in the order specified in ISO/IEC 18013-2.

NOTE This approach allows reading authorities to read only those data groups it is interested in.

8.1.4.2 Compact encoding

For compact encoding, IA's shall not calculate separate message digests for each data group. Therefore, no hash function is specified for compact encoding (except as required as part of any digital signature mechanism; see 8.1.5.2).

8.1.5 Signing method

8.1.5.1 Standard encoding

An IDL digital signature is generated over the concatenation of the message digests of the data groups present.

IA's may use either ECDSA or RSA as digital signature methods for standard encoding.

IA's using RSA shall use RFC 4055. RFC 4055 specifies two signature mechanisms, RSASSA-PSS and RSASSA-PKCS1-v1_5. It is recommended to generate signatures according to RSASSA-PSS, but reading authorities shall also be prepared to verify signatures according to RSASSA-PKCS1-v1_5. The minimum size of the modulus, *n*, shall be 1024 bits.

IA's implementing ECDSA shall use ANSI X9.62. The elliptic curve domain parameters used to generate the ECDSA key pair shall be described explicitly in the parameters of the public key, i.e. parameters shall be of type ECParameters (no named curves, no implicit parameters) and shall include the optional cofactor. ECPoints shall be in uncompressed format. The minimum size for the base point order shall be 160 bits.

In addition to EF.COM and the data groups specified in ISO/IEC 18013-2, IA's shall add the SOD to accommodate the hashes of the individual data groups (see 8.1.4.1) and the digital signature of the data on the IDL. The SOD is implemented as a SignedData Type, as specified in RFC 3369 (including processing rules). The SOD shall be produced in DER format.

Table 2 — SignedData Type

Value	Type	Comments
`SignedData`	m	
`Version`	m	
`digestAlgorithms`	m	
`encapContentInfo`	m	
`eContentType`	m	id-icao-`ldsSecurityObject`
`eContent`	m	The encoded contents of an `ldsSecurityObject`
`certificates`	o	
`Crls`	x	
`signerInfos`	m	
`SignerInfo`	m	
`version`	m	
`Sid`	m	
`issuerandSerialNumber`	c	It is recommended that IA's support this field over `subjectKeyIdentifier`.
`subjectKeyIdentifier`	c	

m = mandatory (the field shall be present);

x = do not use (the field shall not be populated);

o = optional (the field may be present);

c = choice (the field contents is a choice from alternatives)

Table 2 *(continued)*

Value	Type	Comments
`digestAlgorithm`	m	The algorithm identifier of the algorithm used to produce the hash value over `encapsulatedContent` and `SignedAttrs`.
`signedAttrs`	m	IA's may wish to include additional attributes for inclusion in the signature, however these do not have to be processed by a RA except to verify the signature value.
`signatureAlgorithm`	m	The algorithm identifier of the algorithm used to produce the signature value and any associated parameters.
`signature`	m	The result of the signature generation process.
`unsignedAttrs`	o	IA's may wish to use this field, but it is not recommended and reading authorities may choose to ignore them.
m = mandatory (the field shall be present); x = do not use (the field shall not be populated); o = optional (the field may be present); c = choice (the field contents is a choice from alternatives)		

ASN.1 sequence

```
LDSSecurityObject { joint-iso-itu-t(2) international-organizations(23) icao(136) mrtd(1)
security(1) ldsSecurityObject(1)}

DEFINITIONS IMPLICIT TAGS ::=
BEGIN

-- Imports from RFC 3280 [PROFILE], Appendix A.1
AlgorithmIdentifier FROM
PKIX1Explicit88 { iso(1) identified-organization(3) dod(6)
internet(1) security(5) mechanisms(5) pkix(7)
id-mod(0) id-pkix1-explicit(18) }

-- Constants
ub-DataGroups INTEGER ::= 16

-- Object Identifiers
id-icao OBJECT IDENTIFIER ::= {2.23.136}
id-icao-mrtd OBJECT IDENTIFIER ::= {id-icao 1}
id-icao-mrtd-security OBJECT IDENTIFIER ::= {id-icao-mrtd 1}
id-icao-ldsSecurityObject OBJECT IDENTIFIER ::= {id-icao-mrtd-security 1}

-- LDS Security Object
LDSSecurityObjectVersion ::= INTEGER {V0(0)}
DigestAlgorithmIdentifier ::= AlgorithmIdentifier
LDSSecurityObject ::= SEQUENCE {
   version LDSSecurityObjectVersion,
   hashAlgorithm DigestAlgorithmIdentifier,
   dataGroupHashValues SEQUENCE SIZE (2..ub-DataGroups) OF DataGroupHash }

DataGroupHash ::= SEQUENCE {
dataGroupNumber DataGroupNumber,
dataGroupHashValue OCTET STRING }
DataGroupNumber ::= INTEGER {
dataGroup1 (1),
dataGroup2 (2),
dataGroup3 (3),
dataGroup4 (4),
dataGroup5 (5),
dataGroup6 (6),
dataGroup7 (7),
dataGroup8 (8),
dataGroup9 (9),
dataGroup10 (10),
```

```
dataGroup11 (11),
dataGroup12 (12),
dataGroup13 (13),
dataGroup14 (14),
dataGroup15 (15),
dataGroup16 (16)}
END
```

NOTE 1 The field `dataGroupHashValue` contains the calculated hash over the complete contents of the Data Group EF, specified by `dataGroupNumber`.

NOTE 2 Data Groups 15 and 16 may be defined in future.

8.1.5.2 Compact encoding

An IDL digital signature is generated over the full data stream from the start of DG1 (including the data group delimiter between DG1 and the header) to the end of DG12 (including the data group delimiter that terminates DG12, but excluding DG.SOD, if present). The digital signature is stored in DG.SOD.

NOTE The header is not included in the information to be signed as the length information in the header pertains to DG11 as well.

IA's shall use ECDSA (as defined in ANSI X9.62) as a signing method for compact encoding, and in order to reduce the storage requirements for the domain parameters, shall use one of the curves specified in FIPS 186-2, Appendix 6.

DG.SOD shall consist of a concatenation of two Type 2 data groups and one Type 1 data group, storing the following information:

a) DG.SOD.1: digital signature, shall be the DER encoded ASN.1 sequence of two integers, *r* and *s*: `SEQUENCE ::= { r INTEGER, s INTEGER };`

b) DG.SOD.2: public key; octet string representation of the public point in uncompressed form according to X9.62;

c) DG.SOD.3: curve identifier.

DG.SOD shall be added after DG12, as follows:

[header] × [Data Group 1] × [Data Group 2] × [Data Group 3] × [Data Group 4] × [Data Group 7] × [Data Group 11] × [Data Group 12] × [DG.SOD.1 length] [digital signature] × [DG.SOD.2 length] [public key] × [DG.SOD.3: named curve]

The inclusion of DG.SOD.2 and DG.SOD.3 (as a pair) is optional. Length information is encoded using ASN.1 rules.

NOTE Reading authorities should be able to verify (or obtain) a public key (and domain parameters) from the IA using the data in other data groups (specifically the ISO issuer ID number and document discriminator in DG3 and the licence number and date of issue in DG1).

The curve identifier shall consist of one byte, as shown in Table 3.

Table 3 — Curve identifiers

Curve identifier	Curve name in FIPS 186-2	Curve identifier	Curve name in RFC 5639
'01'	P-192	'11'	brainpoolP160r1
'02'	P-224	'12'	brainpoolP160t1
'03'	P-256	'13'	brainpoolP192r1
'04'	P-384	'14'	brainpoolP192t1
'05'	P-521	'15'	brainpoolP224r1
'06'	Reserve for future use	'16'	brainpoolP224t1

Table 3 *(continued)*

Curve identifier	Curve name in FIPS 186-2	Curve identifier	Curve name in RFC 5639
'07'	Reserve for future use	'17'	brainpoolP256r1
'08'	Reserve for future use	'18'	brainpoolP256t1
		'19'	brainpoolP320r1
		'1A'	brainpoolP320t1
		'1B'	brainpoolP384r1
		'1C'	brainpoolP384t1
		'1D'	brainpoolP512r1
		'1E'	brainpoolP512t1

EXAMPLE Suppose that a compact encoded data string contains the following data groups: DG1, DG2, DG7 and DG.SOD.1. A digital signature is included, but the public key and curve identifier are not included. The sequence of data groups and data group delimiters will be as follows:

[header] × [DG1] × [DG2] × × × [DG7] × × × [DG.SOD.1] ¶

8.2 Active authentication

8.2.1 Purpose

Active authentication uses information stored in a secure area of an SIC to confirm that the SIC and the other machine-readable data were issued together.

8.2.2 Applicability

This mechanism is limited to SICs.

8.2.3 Description

A challenge-response protocol matches the private and public keys of an SIC-individual key pair. The private key is stored in the SIC's secure memory and cannot be copied. The public key is stored as part of the signed data on the SIC (the challenge-response protocol thus has to be preceded by passive authentication). The SIC (or more specifically, the SIC operating system) can prove knowledge of the SIC-individual private key in a challenge-response protocol. In this protocol, the SIC digitally signs a challenge randomly chosen by the IS. The IS is convinced that the SIC is genuine if and only if the returned signature is correct.

NOTE Active authentication using a challenge-response protocol is open to a "traceability" attack. The challenge-response protocol does not place any limitation on the format or content of the challenge. If the challenge includes time, date, location, and a secret key, a log of challenge responses can be used to track IDLs and to prove presence (due to the inclusion of the secret key). The counterargument is that basic access protection would safeguard against unknown read attempts and that a driver's location is known in any case in those instances where an IDL is handed over for inspection.

8.2.4 Mechanism

8.2.4.1 General

The SIC-individual public key shall be stored in DG13, an additional data group specifically intended for use with the active authentication mechanism. The format of the structure (`SubjectPublicKeyInfo`) is specified in RFC 3280. `SubjectPublicKeyInfo` shall be produced in DER format as follows:

```
ActiveAuthenticationPublicKeyInfo ::= SubjectPublicKeyInfo

SubjectPublicKeyInfo ::= SEQUENCE {
   algorithm AlgorithmIdentifier,
   subjectPublicKey BIT STRING }

AlgorithmIdentifier ::= SEQUENCE {
   algorithm OBJECT IDENTIFIER,
   parameters ANY DEFINED BY algorithm OPTIONAL }
```

Allowed algorithm object identifiers and parameters are specified in RFC 3279.

Active authentication shall be performed using the ISO/IEC 7816 INTERNAL AUTHENTICATE command as defined by ISO/IEC 7816-4:2013. The input shall be a terminal-generated nonce (RND.IFD) that shall be 8 bytes in length. The SIC computes a signature which shall be sent back to the IS. The IS shall check the response and verify the signature.

8.2.4.2 Signature generation using RSA

The signature shall be computed according to ISO/IEC 9796-2 Digital Signature Scheme 1. M shall consist of M1 and M2, where M1 is an SIC-generated nonce of c-4 bits and M2 is RND.IFD. The result of the signature generation shall be the signature Σ without the recoverable message part M2.

8.2.4.3 Signature generation using ECC

An IDL supporting ECDSA shall use a prime curve with uncompressed points for this computation. In this case the output of the computation shall be the concatenation of the values r and s (r||s). The input for this computation shall be compressed with a hash algorithm with an output length shorter or equal to the length of the signature key.

NOTE The plain signature format (r||s) for the ECDSA is according to TR-03111[6].

Based on a suitable algorithm catalogue, the length of the key for ECDSA should be chosen to provide the desired security level for the physical life of the IDL.

If ECDSA based signature algorithm is used for Active Authentication by the SIC, the SecurityInfos in LDS Data Group 14 of the IDL application shall contain following `SecurityInfo` entry:

```
ActiveAuthenticationInfo ::= SEQUENCE {
    protocol id-icao-mrtd-security-aaProtocolObject,
    version INTEGER, -- MUST be 1
    signatureAlgorithm OBJECT IDENTIFIER
}

id-icao-mrtd-security-aaProtocolObject OBJECT IDENTIFIER ::=
                                        { id-icao-mrtd-security 5 }
```

The object identifiers for `signatureAlgorithm` are defined in TR-03111[6]. `ecdsa-plain-SHA1` and `ecdsa-plain-RIPEMD160` shall not be used.

NOTE 1 See 8.1.5.1 for the definition of the Object Identifier `id-icao-mrtd-security`.

NOTE 2 `SecurityInfos` may contain entries for other protocols, e.g. Chip Authentication, Terminal Authentication, PACE.

8.2.4.4 Security mechanism indicator

SICs which implement active authentication may optionally indicate this in the security mechanism indicator (see Clause 9). The object identifier shall be `id-sm-AA`.

```
id-sm-AA OBJECT IDENTIFIER ::= {
  iso(1) standard(0) driving-licence(18013) part-3(3)
  security-mechanisms(2) 3}
```

The parameters are mandatory and shall be of type `param-AA`.


```
param-AA ::= SEQUENCE {
  version INTEGER,
  publicKeyDG INTEGER}
```

The version shall be set to v1(0) for this version of active authentication. The `publicKeyDG` field shall be set to the number of the data group of the active authentication public key, i.e. 13.

8.3 Scanning area identifier

8.3.1 Applicability

This mechanism is used with the non-match alert, BAP or PACE mechanisms, which collectively are applicable to SICs and 2D barcode (see 8.4 and 8.5).

8.3.2 Description

8.3.2.1 General

The primary purpose of this mechanism is to facilitate machine assisted automatic or interactive verification procedures that match machine-readable data to human-readable data. This mechanism is not a solution on its own, but is intended for use as a component of either of the following mechanisms:

a) non-match alert (see 8.4), which checks if the machine-readable and human-readable data belong together;

b) BAP (see 8.5) or PACE (see 8.7), which ensures that a PICC cannot be read without physical access to the IDL.

NOTE The BAP mechanism also performs the function of the non-match alert mechanism, but potentially at a different level of security.

The SAI demarcates the input string. The input string can be entered manually by an operator or can be read automatically using machine-readable technologies.

The SAI and any machine-readable information demarcated by it shall be adapted for reading in the B900 infrared band defined in ISO 1831:1980. Under such illumination, the printing background and any overlays within the SAI will backscatter light in a homogeneous way (i.e. the SAI shall contain no optically variable device, local deviations in properties, security features, or surface gloss exceeding natural transparent overlay gloss, which will interfere with readability). In addition, machine-readable printed information (e.g. bar codes and OCR characters) shall comply with the associated standards. The brightness of the printing background (including the effects of overlays) will not be less than 40 % compared to an ideal white surface under the same illumination. Human-readable information inside the SAI shall be rendered such that it can be read without difficulty.

The contrast ratio between character lines and the background shall be a minimum of 4:1. Any reflective layer covering the SAI area shall not further deteriorate this contrast value when illuminated under angles of incidence up to 45° and looked at under angles more than 10° outside of the nominal glance angle.

The presence of the SAI is identified as follows:

a) If PICC access is required and the access conditions are unsatisfied (i.e. BAP or PACE is in place), the reader searches for a SAI on the IDL.

b) If access to DG12 is available, BAP or PACE is not applicable and the non-match alert mechanism is present. DG12 may indicate the location of the SAI, alternatively, the reader searches for a SAI on the IDL.

Not more than one SAI shall be present on a single IDL.

The reference string (and the input string, when stored in a barcode) shall be encoded in accordance with ISO/IEC 8859-1:1998.

The content of the SAI can follow one of three standards, i.e. based on an existing text field, containing a dedicated text field, or containing a barcode.

8.3.2.2 SAI content based on existing text field

The SAI may be constructed around an existing field on an IDL. In this case, the SAI shall consist of a double lined rectangle as illustrated by the example in Figure 6.

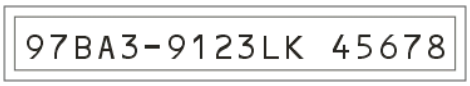


Figure 6 — SAI around an existing field (not to scale)

Each line shall have a thickness of 0,2 mm ± 0,1 mm, and the distance between the centre of each line shall be 0,6 mm ± 0,1 mm. The lines shall be black in colour. The clear distance between the inside line and the extremities of the input string shall be at least 1 mm.

The input string shall be printed using OCR-B size 1 or a character set with the same symbols and character shapes but using a reduced size of either 25 % or 50 % reduction of all linear dimensions compared to OCR-B size 1.

To prevent confusion about the input sequence of the characters, text within the rectangle shall be limited to one line. The data element name (if reflected on the IDL) and/or the data field reference code shall not be included within the rectangle.

In order to facilitate manual entry of the input string when needed, the input string may contain only Latin capital letters (hexadecimal range 41 to 5A of ISO/IEC 8859-1:1998), Latin small letters (hexadecimal range 61 to 7A of ISO/IEC 8859-1:1998), N characters, and the S characters shown in Table 4.

NOTE 1 Latin capital letters (hexadecimal range 41 to 5A of ISO/IEC 8859-1:1998) are recommended.

NOTE 2 References to ISO/IEC 8859-1:1998 are only for purposes of identifying the characters. Encoding methods are specified elsewhere.

Table 4 — S characters allowed for SAI content based on existing field

S character	Hexadecimal number in ISO/IEC 8859-1:1998[a]
<space>	20
–	2D
<	3C

[a] Reference to ISO/IEC 8859-1:1998 is only for the purpose of identifying the characters. Encoding methods are specified elsewhere.

The reference string shall not contain any space characters (i.e. hexadecimal character '20' of ISO/IEC 8859-1:1998). Thus, the IS shall delete all space characters (hexadecimal code 20 of ISO/IEC 8859-1:1998) from the input string before further processing.

IA's should carefully consider the level of entropy of existing fields before use within the SAI. Any rules followed in the construction of the field will decrease the entropy.

IA's considering the use of an SAI based on an existing field should keep in mind that a check digit is not provided for in this case.

EXAMPLE See Figure 7.

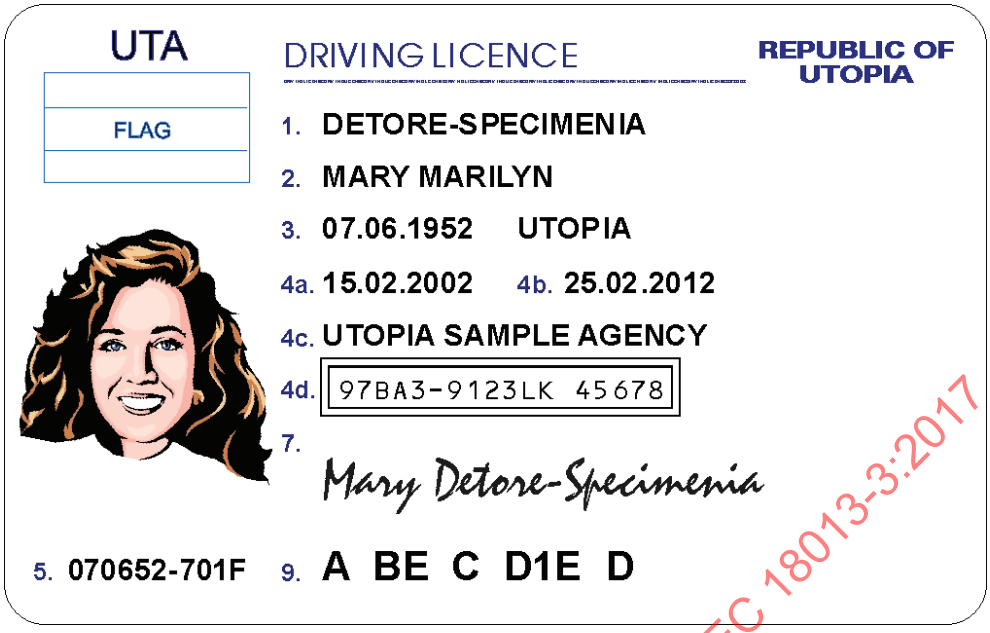


Figure 7 — Existing data field on portrait side of IDL designated as input string (not to scale)

8.3.2.3 SAI content consisting of a dedicated text field

A dedicated field (i.e. a field designed specifically for this purpose) may be used within a SAI. In this case, the SAI shall consist of a rectangle constructed by a black single line with a thickness of 0,65 mm ± 0,1 mm. The clear distance between the line and the extremities of the printed input string shall be at least 1 mm.

EXAMPLE See Figure 8.

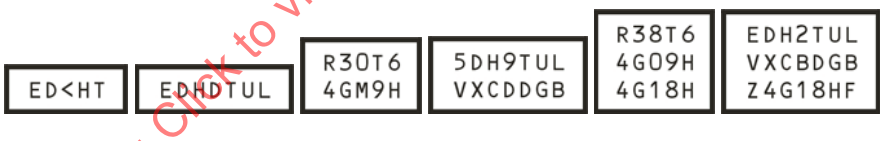


Figure 8 — SAI around a dedicated field (not to scale)

The content of the SAI shall comply with the following requirements:

a) the text shall be limited to the set of ICAO MRZ characters, i.e. 0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <;

b) an odd number of characters per line shall be used;

c) if characters are printed in more than one line, each line shall have the same number of characters;

d) a line shall not have more than seven characters;

e) no more than three lines shall be used;

f) the text shall be printed using OCR-B font at 100 % of OCR-B size 1;

g) the middle character of each line shall be reserved for a check digit. Before adding the check digit(s), the input string may have the following lengths:

   1) for one line only, string lengths of 4 or 6 characters are permitted;

   2) for two lines, string lengths of 8 or 12 characters are permitted;

3) for three lines 12 or 18 characters are permitted.

To construct an input string (with check digits) consisting of i lines, the input string (before check digits are added) is parsed into i parts of equal length. The leftmost part is used to form line 1, the following parts each to form an additional line. Each part is split in the middle into a left and a right subsection. The left subsection forms the leftmost part of the corresponding line, the right subsection forms the rightmost part of the line. An additional character position is inserted in the middle of each line (for the check digit). The value of the check digit of each line will be calculated as explained further below.

For the purpose of check digit calculation, each character in the input string (before the check digits are added) represents a numerical value V_C defined by the formula $V_C = E_C$, where E_C is a unique number allotted to each character value, as defined in Table 5.

Table 5 — Character numerical equivalents

Char	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F	G	H	I
E_C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
Char	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	<	
E_C	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	

If line 1 is the only line, then the unique check digit will be calculated as follows:

a) A check sum S is calculated as $S = \Sigma V_{Ci}$, where i runs from 0 to n–1 and V_{Ci} indicates the respective value V_C for the character in the i-th position in the input string of length n, counted from right to left, starting with position 0 for the rightmost character.

b) From S, the corresponding check digit c is calculated as $c = S$ modulo 37; (integer remainder when dividing S by 37).

If there are two or three lines, then the check digits c_1 for line one and c_2 for line two will be calculated as follows:

a) A check sum S_V is calculated over the entire input string (before it is split into lines and before adding the check digits) as $S_V = \Sigma V_{Ci}$, where i runs from 0 to n–1 and indicates the character in the i-th position in the input string of length n, counted from right to left, starting with position 0 for the rightmost character.

b) A check sum S_P is calculated over the entire input string (before it is split into lines, and before adding the check digits) as $S_P = \Sigma(V_{Ci}*i)$, where i runs from 0 to n–1 and i indicates the i-th position number in the input string of length n, counted from right to left, starting with position 0 for the rightmost character, and V_{Ci} indicates the respective value for the character in the i-th position.

c) Check digits c_1 and c_2 are calculated as $c_1 = (S_V$ modulo 37) and $c_2 = (S_P$ modulo 37), respectively.

In the case of exactly three lines, the inserted character c_3 in the centre of line three will represent the version number and will be set to "1" for this version of the standard. Version numbers are not supported for less than three lines.

When printed on an IDL, lines are numbered from top to bottom, as shown in Figure 9.

Figure 9 — Line configuration (not to scale)

EXAMPLE Input string for Figure 9 is EDHTULVXCDGBZ4G8HF.

Input string	E	D	H	T	U	L	V	X	C	D	G	B	Z	4	G	8	H	F
E_C	14	13	17	29	30	21	31	33	12	13	16	11	35	4	16	8	17	15
i	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
V_{Ci}*i	238	208	255	406	390	252	341	330	108	104	112	66	175	16	48	16	17	0

$S_V = 335$

$S_P = 3082$

Check digit	Numeric value	Code value
c1	2	2
c2	11	B

c3 = 1

The input string submitted as input to the non-match alert or BAP mechanisms is the input string inclusive of the check digit, i.e. the concatenation s1 + s2 + s3 where s*i* is the string in line *i* of the corresponding input string, including the respective check digit and read from the left to the right, with *i* ranging from 1 to a maximum of 3.

NOTE Mathematically, the check digit scheme allows identifying any single digit reading error in the case of a one-line input string and to correct any single digit reading error, as well as identifying any double error in the case of a two-line or three-line input string. Less than 3 % of any other statistical reading error may pass inadvertently. This is only true if adequate measures are taken on the side of the reading equipment to make use of the information offered by the check digit scheme.

It is recommended that the minimum entropy of the input string be 40 bits or more (before addition of the check digits). The use of random data is recommended.

NOTE IA's may consider imposing restrictions on allowable content of input strings to meet local requirements, e.g. to address the "nasty word" problem, or to limit the value range of practically existing check digits (i.e. numerical characters only). Methods exist that allow enforcement of such restrictions without violation of the rules set out above. The use of such methods is left to the discretion of the IA's. Attention is drawn to the fact that such restrictions will reduce the entropy of the input string.

EXAMPLE See Figure 10.

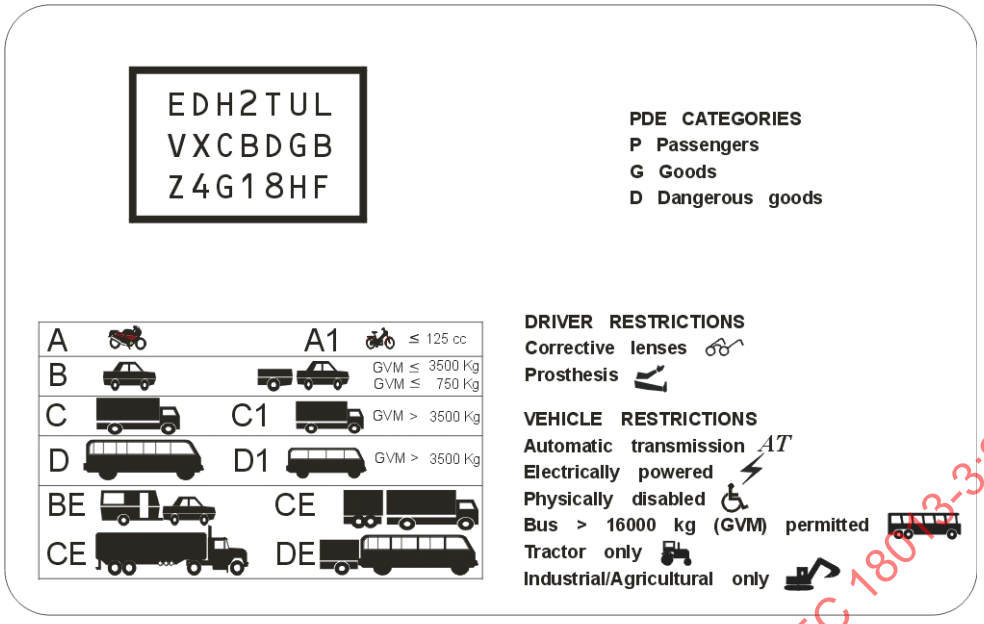


Figure 10 — Dedicated data field on non-portrait side of IDL designated as input string (not to scale)

8.3.2.4 SAI content consisting of a barcode

A barcode may be used within a SAI as the primary means of input. The SAI's corners shall be clearly shown. Each corner indicator shall be constructed of two perpendicular lines joined at the end points, with each line having a thickness of 0,5 mm ± 0,3 mm and a length of 4 mm ± 2 mm.

EXAMPLE See Figure 11.

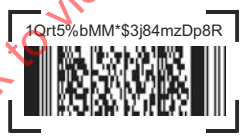


Figure 11 — SAI around a dedicated field (not to scale)

The input string may contain any A, N or S characters, and is not limited in length. However, it is recommended that the input string be limited to the characters specified in 8.3.2.3 to facilitate manual entry when required. The barcode may be accompanied by a human-readable version of the input string (in accordance with ISO/IEC 8859-1:1998), printed in a font not smaller than 6pt.

EXAMPLE See Figure 12.

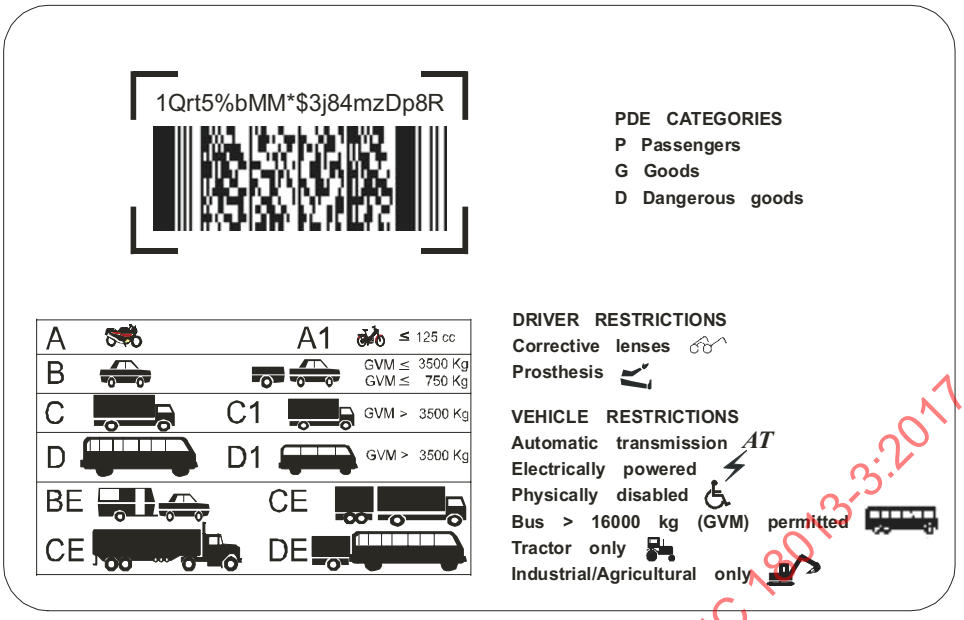


Figure 12 — Barcode on non-portrait side of IDL designated as input string (not to scale)

8.3.2.5 SAI consisting of an IDL MRZ

8.3.2.5.1 General

This mechanism is a 1 line MRZ and is used with the non-match alert, BAP or PACE mechanisms, which collectively are applicable to SICs and 2D barcode (see 8.4 and 8.5).

The IDL MRZ differs from the manner in which an input string is identified in 8.3, in that, its position is fixed and consequently does not require one or more graphical elements that demarcate the input string as an SAI.

The data elements shall be printed in machine readable form, in the MRZ, beginning with the leftmost character position in each field in the sequence indicated in the data structure specifications for the IDL MRZ.

NOTE The content of the IDL MRZ is not to be confused with data elements in the passport MRZ.

Check digits are used within the machine readable zone to provide verification that the entered data are correctly interpreted.

The presence of the IDL MRZ is identified as follows:

a) If SIC access is required and the access conditions are unsatisfied (i.e. BAP or PACE is in place), the reader searches for the IDL MRZ on the IDL.

b) If access to DG12 is available, BAP or PACE is not applicable and the non-match alert mechanism is present, DG12 may confirm the existence of the IDL MRZ, alternatively, the reader searches for the IDL MRZ on the IDL.

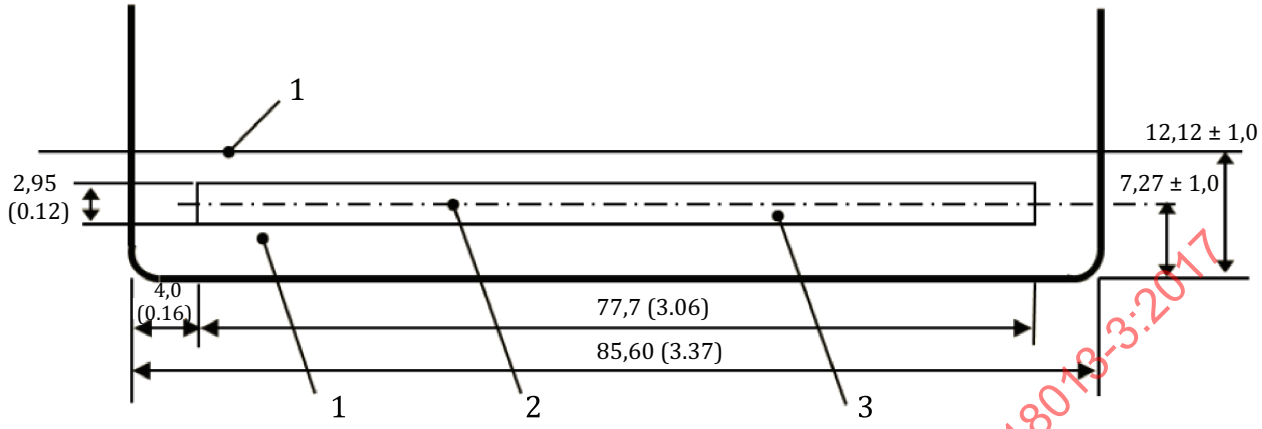
Not more than one IDL MRZ shall be present on a single IDL.

The reference string shall be encoded in accordance with ISO/IEC 8859-1:1998.

The IDL MRZ may be located on the portrait side or the non-portrait side of the IDL, except for the case where the reference string for the non-match alert is stored in a bar code. In such cases, the IDL MRZ shall not be located on the same side as the barcode.

Figure 13 shows the location of the 1 line MRZ at the bottom of the IDL, the nominal dimensions and position of the IDL MRZ data.

Dimensions in millimetres (inches)



Key

1 machine reading zone
2 reference centre line
3 printing zone

Figure 13 — Location of 1 line MRZ at the bottom of the IDL (not to scale)

8.3.2.5.2 Printing

The position of the left edge of the first character of the OCR line shall be 4,0 mm ± 1,0 mm (0.16 in ± 0.04 in) from the left edge of the IDL. The reference centre line for the OCR line shall be 7,27 mm ± 1,0 mm (0.29 in ± 0.04 in) from the bottom edge of the IDL. The OCR line shall not exceed the printing zone length of 77,7 mm ± 1,0 mm (3.06 in ± 0.04 in). No other printing in the B900 infrared band defined in ISO 1831:1980 may appear in the machine reading zone within 12,12 mm ± 1,0 mm (0.48 in ± 0.04 in) from the bottom edge of the IDL.

Machine readable data shall be printed in OCR-B type font, size 1, constant stroke width. The MRZ shall be printed with a horizontal printing density of nominally 10 characters per 25,4 mm.

The text shall be limited to the set of ICAO MRZ characters, i.e. 0 1 2 3 4 5 6 7 8 9 A B C D E F G H I J K L M N O P Q R S T U V W X Y Z <, excluding the space character.

The IDL MRZ shall be adapted for reading in the B900 infrared band defined in ISO 1831:1980. Under such illumination, the printing background and any overlays will backscatter light in a homogeneous way (i.e. the IDL MRZ shall not be obstructed by an optically variable device, local deviations in properties, security features, or surface gloss exceeding natural transparent overlay gloss that will interfere with readability). The brightness of the printing background (including the effects of overlays) will not be less than 40 % compared to an ideal white surface under the same illumination.

The contrast ratio between character lines and the background shall be a minimum of 4:1. Any reflective layer covering the SAI area shall not further deteriorate this contrast value when illuminated under angles of incidence up to 45° and looked at under angles more than 10° outside of the nominal glance angle.

8.3.2.5.3 Use of the IDL MRZ as a BAP or PACE input string

The IDL MRZ can be used for establishing BAP as specified in 8.5 or PACE as specified in 8.7. The input string shall be composed of characters 2 to 29 of the IDL MRZ.

The IDL MRZ shall contain the elements given in Table 6.

Table 6 — IDL MRZ data elements

IDL MRZ character positions	Data element	Specification	Number of characters
1	Identifier	The first character shall be “D”.	1
2	Configuration	The second character shall designate the configuration as follows. “1” for IDL SIC protected with BAP configuration 1 “P” for IDL SIC protected with PACE only (i.e. without BAP support) “N” for IDL SIC protected with non-match alert “<” for an MRZ does not contain a reference string Any other characters are reserved for future use. NOTE If the second character of the MRZ is set to 1, PACE may be supported in addition to BAP configuration 1.	1
3 to 29	Discretionary data	The contents of this field are to be determined at the discretion of the Issuing Authority. Unused character positions shall be completed with filler characters (<) repeated up to position 29, as required.	27
30	Check digit	The check digit allows verification of all the data in the OCR line. Check digit is calculated over the characters in positions 1 to 29 according to requirements set out below	1
The IA should ensure that the entropy of the input string is commensurate with its purpose.			

8.3.2.5.4 Check digit calculation

The IDL MRZ check digit shall be calculated on modulus 10 with a continuously repetitive weighting of 731 731 ..., as follows:

— **Step 1:** Going from left to right, multiply each digit of the pertinent numerical data element by the weighting figure appearing in the corresponding sequential position.

— **Step 2:** Add the products of each multiplication.

— **Step 3:** Divide the sum by 10 (the modulus).

— **Step 4:** The remainder shall be the check digit.

For data elements in which not all available character positions are occupied, the symbol < shall be used to complete vacant positions and shall be given the value of zero for the purpose of calculating the check digit.

When the check digit calculation is applied to data elements containing alphabetic characters, the characters A to Z shall have the values 10 to 35 consecutively, as shown in Table 7.

Table 7 — Check digit calculation

Character	A	B	C	D	E	F	G	H	I	J	K	L	M	
Value	10	11	12	13	14	15	16	17	18	19	20	21	22	
Character	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	<
Value	23	24	25	26	27	28	29	30	31	32	33	34	35	0

EXAMPLE Using an IDL with the following elements in the IDL MRZ, as shown in Table 8.

Table 8 — Example IDL MRZ data elements

IDL MRZ element	Value	Meaning
Identifier	"D"	In accordance with this document
Configuration	"1"	BAP configuration 1
Discretionary data	"23T09PJ3Y8478FSD<<<<<<<<<<"	16 random characters followed by special characters to fill vacant positions
Check digit	"?"	Check digit to be calculated over characters 1 to 29

Check digit is to be calculated over the following data: D123T09PJ3Y8478FSD<<<<<<<<<<

Step 1: Multiplication:

Data	**D**	**1**	**2**	**3**	**T**	**0**	**9**	**P**	**J**	**3**	**Y**	**8**	**4**	**7**	**8**	**F**	**S**	**D**	**<**	**<**	**<**	**<**	**<**	**<**	**<**	**<**	**<**	**<**
Value	13	1	2	3	29	0	9	25	19	3	34	8	4	7	8	15	28	13	0	0	0	0	0	0	0	0	0	0
Weight	7	3	1	7	3	1	7	3	1	7	3	1	7	3	1	7	3	1	7	3	1	7	3	1	7	3	1	7	3
Product	91	3	2	21	87	0	63	75	19	21	102	8	28	21	8	105	84	13	0	0	0	0	0	0	0	0	0	0

Step 2: Sum of products:

91 +3 +2 +21 +87 +0 +63 +75 +19 +21 +102 +8 +28 +21 +8 +105 +84 +13 +0 +0 +0 +0 +0 +0 +0 +0 +0 +0 = 751

Step 3: Division by modulus: $\frac{751}{10}$ = 75, remainder 1.

Step 4: Check digit is the remainder, 1. The resulting MRZ including check digit shall be written as:

D123T09PJ3Y8478FSD<<<<<<<<<<1

Table 9 — Example IDL MRZ data elements and check digit

IDL MRZ element	Value	Meaning
Identifier	"D"	In accordance with this document
Configuration	"1"	BAP configuration 1
Discretionary data	"23T09PJ3Y8478FSD<<<<<<<<<<"	16 random characters followed by special characters to fill vacant positions
Check digit	"1"	Check digit calculated over characters 1 to 29

The corresponding hex value of the above input string (Configuration and Discretionary data) given in Table 9 encoded in accordance with ISO/IEC 8859-1:1998 is:

'31 32 33 54 30 39 50 4A 33 59 38 34 37 38 46 53 44 3C 3C 3C 3C 3C 3C 3C 3C 3C 3C'

8.3.2.5.5 Use of IDL MRZ for non-match alert

The IDL MRZ can be used with the non-match alert mechanism as described in 8.4.

In order to describe such use in DG12, byte 1 of the `SAI_inputmethod` field shall be assigned a value of '41'. The four most significant bits (upper nibble) of this byte identifies the IDL MRZ as the input method. The four least significant bits (lower nibble) of byte 1 is assigned in accordance with Table 10.

If byte 2 of the `SAI_inputmethod` is present, it shall have the value 'FF'.

If the presence of the non-match alert mechanism is indicated in the EF.COM file and if the optional parameters are provided, the `version` field shall be set to v1(1) for this version of the non-match alert mechanism.

EXAMPLE See Figure 14.

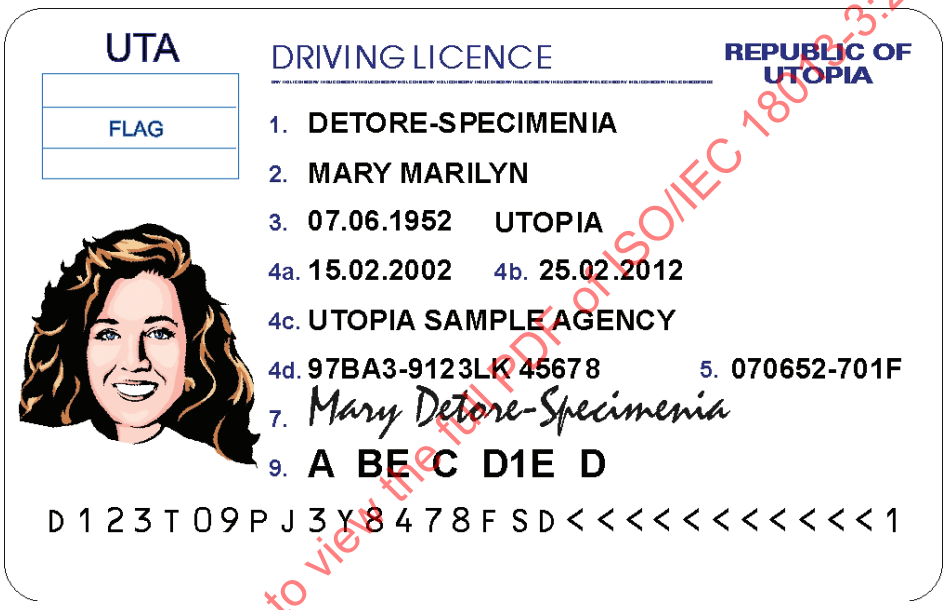


Figure 14 — IDL MRZ on portrait side of IDL designated as input string (not to scale)

EXAMPLE See Figure 15.

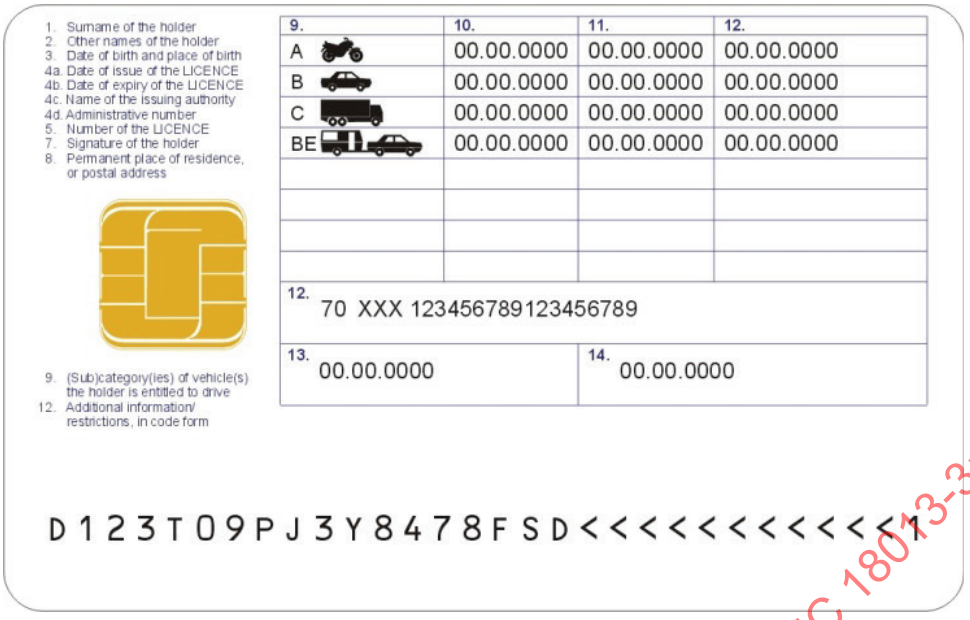


Figure 15 — IDL MRZ on non-portrait side of IDL designated as input string (not to scale)

8.4 Non-match alert

8.4.1 Purpose

This mechanism allows detection of any differences between the machine-readable information and (some of) the human-readable information (printed on an IDL). This mechanism may be of value in the presence of machine-readable technologies with data content not accessible to visual inspection and on data carriers or physical support that may be transferred completely from one IDL to another without strong visual evidence of alteration.

8.4.2 Applicability

This mechanism is applicable to SICs and 2D barcode.

8.4.3 Description

Conceptually, the reference string is verified via a digital signature, after which the reference string is compared to the input string. If the comparison is not successful, it means that the machine-readable data and the (human-readable) printed data are not consistent.

NOTE 1 It is assumed that the RA trusts the public key used to verify the digital signature.

NOTE 2 If the comparison is successful, the RA may conclude that the machine-readable data and the physical document are consistent if it can be assumed that alteration to any part of the printed data (i.e. not just the area that contains the input string) would be visually apparent.

The reference string is not separately verified with a dedicated digital signature, but is verified during passive authentication (the digital signature used for passive authentication already covers the reference string).

NOTE 3 The function provided by the non-match alert mechanism can also be achieved by performing passive authentication followed by a visual comparison of the human-readable data printed on the document with the authenticated machine-readable data.

8.4.4 Mechanism

8.4.4.1 General

The manner in which the non-match alert mechanism is implemented is specified in DG12, using the parameters and values as shown in Table 10.

Table 10 — Non-match alert parameters

Name, Fixed (F) or Variable (V), Mandatory (M) or Optional (O)	Field format/length/type	Example
SAI_referencestring , V, M	Byte 1: '00' if the input string follows '01' if a reference to where the input string can be obtained follows Subsequent bytes: If byte 1 = '00', the input string follows from byte 2 (inclusive). The input string is encoded in accordance with ISO/IEC 8859-1:1998. If byte 1 = '01', the reference to the field that contains the input string is constructed as aabb where aa is the data group and bb is the sequence number of the referenced data element, with aabb encoded as unsigned BCD	An input string of ABC4DEF contained in the SAI_referencestring field will be coded as '00 41 42 43 34 44 45 46', where '41 42 43 34 44 45 46' is the encoded form of ABC4DEF. If the licence number is used as the input string, this will be coded as '01 01 08'. If the 16th data element of Data Group 12 is used as the input parameter, the input string will be coded as '01 12 16'
SAI_inputmethod, V, O	Byte 1: SAI standard and input method. The four most significant bits (upper nibble) of byte 1 can take on any of the following values: — '0x' if the input string is based on an existing field; — '1x' if the input string is based on a dedicated field; — '2x' if the input string is stored in a barcode. The four least significant bits (lower nibble) of byte 1 denotes the input method, and can take on any of the following values: — 'x0' if the input string is intended for manual input; — 'x1' if the input string is intended for OCR interpretation; — 'x2' if the input string is stored as a barcode. Byte 2: Barcode standard. If the first byte is of the form 'x2', byte 2 is mandatory, taking on any of the following values: — '00' for PDF417; — '01' for Code 39 (ISO/IEC 16388); — '02' for Code 128 (ISO/IEC 15417); — '03' for data matrix (ISO/IEC 16022); — 'FE' for other barcode standards not provided for above; — 'FF' no barcode.	If the licence number is used as the input string (i.e. a SAI is constructed around the existing licence number field on the IDL), the value of SAI_inputmethod will be '00'. If, in addition, the input string is printed in OCR-B font, the input string will be '00 01', or alternatively '00 01 00'. If, in addition, the SAI is located on the portrait side of the IDL, with the top left corner of the SAI at 29 mm from the left edge of the card and 24 mm from the bottom edge of the card, and the right bottom corner of the SAI at 59 mm from the left edge of the card and 14 mm from the bottom edge of the card, the input string will be '00 01 00 00 29 24 59 14'
	Byte 2 is also mandatory if Bytes 3 to 7 are present. Bytes 3 to 7: Position of the SAI, expressed as 'aa bb cc dd ee', where 'aa' is the side of the card on which the SAI appears ('00' for portrait side, and '01' for non-portrait side), 'bb cc' is the top left corner of the SAI (where 'bb' is the distance from the left edge of the IDL and 'cc' is the distance from the bottom edge of the card), and 'dd ee' is the bottom right corner of the SAI (where 'dd' is the distance from the left edge of the IDL and 'ee' is the distance from the bottom edge of the card), with all distances measured in millimetres, and encoded as BCD. The bytes are progressively mandatory, i.e. SAI_inputmethod can consist only of byte 1, or only of bytes 1 and 2, or of bytes 1 to 7.	

8.4.4.2 Compact encoding

When used with compact encoding, DG12 shall be constructed as a Type 1 data group as follows:

... × [SAI_referencestring] ÷ [SAI_inputmethod] × ...

8.4.4.3 Standard encoding

When used with standard encoding, the parameters shall be stored in DG12 as shown in Table 11.

Table 11 — Non-match alert parameters for standard encoding

Tag	Length	Value
'82'	X	SAI_referencestring as defined in Table 10.
'81'	X	SAI_inputmethod as defined in Table 10.

IA's may optionally indicate the presence of the non-match alert mechanism in the EF.COM file using the security mechanism indicator as specified in Clause 9.

The object identifier for non-match alert shall be `id-sm-NMA`.

```
id-sm-NMA OBJECT IDENTIFIER ::= {
  iso(1) standard(0) driving-licence(18013) part-3(3)
security-mechanisms(2) 4
}
```

The parameters are optional and shall be of type `param-NMA`.

```
param-NMA ::= SEQUENCE {
  version INTEGER,  SAI_inputmethod OCTET STRING
}
```

The `version` field shall be set to v1(0) for this version of the non-match alert mechanism.

The `SAI_inputmethod` field shall be set to the corresponding value in DG12. If the `SAI_inputmethod` field is not present in DG12, the field shall also not be included in EF.COM.

8.5 Basic access protection

8.5.1 Purpose

BAP confirms that an IS has physical access to a PICC before the IS is allowed access to the data stored on the PICC. In addition, BAP ensures that communication between the IS and the PICC (once access is authorized) is protected.

NOTE Although BAP was designed for PICCs, it can also be used to satisfy a variety of functional requirements for ICs with contacts as listed in Table 1.

8.5.2 Applicability

This mechanism is limited to PICCs and ICs with contacts.

8.5.3 Description

For a SIC that is accessed via a contactless interface and protected by the BAP, mechanism shall deny access to its content by the contactless interface unless the IS can prove that it is authorized to access the SIC. This proof shall be given in a challenge-response protocol, where the IS proves knowledge of the PICC-individual document basic access keys (K_{ENC} and K_{MAC}) that are derived from the input string.

After the IS has been authenticated successfully, the SIC shall enforce encryption and message authentication of the communication channel between the IS and the SIC by Secure Messaging techniques.

8.5.4 Mechanism

The document basic access keys are stored in the internal elementary file (see ISO/IEC 7816-4:2013).

A SIC that supports BAP shall respond to unauthenticated read attempts (including selection of files in the logical data structure) with 'Security status not satisfied' (0x6982). The presence of BAP thus is determined by an IS from the response to read attempts (see below) and the subsequent success in locating the SAI.

BAP is specified in Annex B. The following shall be used in the application of Annex B:

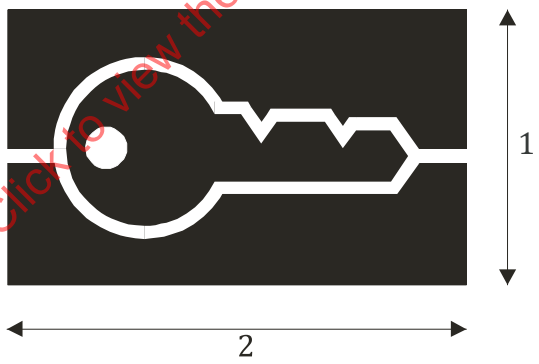
a) the reference string shall be used as the document keying material (K_{doc});

b) the IS can automatically read the input string or can allow an operator to manually enter the input string (if present) into the IS;

c) the first byte of the input string shall identify the BAP configuration used.

NOTE BAP is intended mainly as an anti-skimming mechanism. However, provided that the entropy of the reference string is commensurate with the IA's assessment of the threat, this mechanism can also serve as protection against eavesdropping.

IA's may optionally add the BAP logo to an IDL to indicate that the SAI contains a BAP input string. Figure 16 shows the BAP logo while Figure 17 shows the BAP logo on IDL. The minimum dimensions of the A-dimension of the logo shall be 5 mm, and the B-dimension shall be scaled proportionally in a ratio of 5:3. It is recommended that the BAP logo be placed in close proximity to the associated SAI.



Key

1 B-dimension

2 A-dimension

Figure 16 — Basic access protection logo (not to scale)

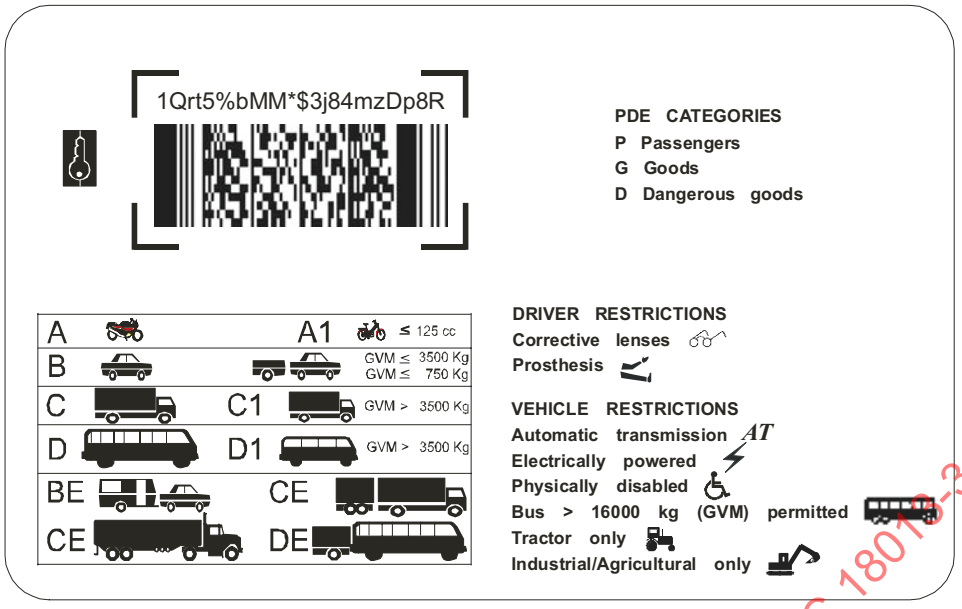


Figure 17 — Basic access protection logo on IDL (not to scale)

EXAMPLE Suppose the following barcode is printed on the IDL:

This barcode encodes the string "14624833454341156544340341183612848170 41", whose hexadecimal rendering is '31 34 36 32 34 38 33 33 34 35 34 33 34 31 31 35 36 35 34 34 33 34 30 33 34 31 31 38 33 36 31 32 38 34 38 31 37 30 34 31'

The first byte ('31') of the input string indicates BAP configuration 1. The entire input string is used as K_{doc}. The resulting derived document access keys are:

— K_{enc}: 'E4 80 0E 99 54 FF 39 EB F6 09 15 FD 46 C4 3F DD'

— K_{mac}: 'E7 EA 15 4F ED 39 38 77 A1 12 9D CB 7A 54 93 50'

The barcode in this example contains 40 numeric digits. As the first byte conveys predictable information, only the following 39 bytes may contribute entropy. Assuming they were generated randomly, the resulting K_{doc} contains $\log_2(10^{39}) \approx 129$ bits of entropy.

8.6 Extended Access Control v1

8.6.1 Purpose

EACv1 consists of:

a) chip authentication, which provides for authentication of the SIC and strong secure messaging, and

b) terminal authentication, which provides for conditional authenticated access to data groups.

8.6.2 Applicability

This mechanism is applicable only to SICs.

8.6.3 Description and mechanism

EACv1 is defined in Annex D.

This document only supports EACv1 Chip Authentication with the ECDH mechanism.

This document only supports EACv1 Terminal Authentication with the ECDSA mechanism.

The following rules shall be used in the application of EACv1 for an IDL:

a) The SIC's key agreement public key(s) shall be stored in DG14, formatted in accordance with BSI/TR 03110-3.

b) If BAP is performed before EACv1, the driving licence number, as it appears in DG1, shall be used as SIC identifier (ID_{SIC}). If PACE is performed before EACv1, the SIC identifier (ID_{SIC}) is computed from the SIC's ephemeral PACE public key, i.e. IDSIC=Comp(PKSIC).

c) Strong secure messaging (established using chip authentication as described in BSI/TR 03110) shall be active before terminal authentication can take place.

d) DG14 shall be accessible without terminal authentication.

8.7 PACE

8.7.1 Purpose

The PACE protocol confirms that an IS has physical access to a SIC before the IS is allowed to access to the data stored on the chip. Once access is authorized, PACE protects the subsequent communication by a secure channel between SIC and IS. The PACE protocol can be used as an alternative to BAP and allows various implementation options (mappings, input strings, algorithms).

8.7.2 Applicability

This mechanism is applicable only to SICs.

8.7.3 Description and mechanism

PACE is defined in Annex C. This document only supports PACE with ECDH generic mapping. The first byte of the input string shall be '50' ("P") if PACE is used as a stand-alone protocol, i.e. not used in conjunction with BAP configuration 1.

If PACE is used to gain access to the SIC, the SIC shall deny access to the content of the IDL application by its interface unless the IS can prove that it is authorized to access the SIC. This proof is given in a password authenticated Elliptic Curve Diffie Hellman key agreement protocol where the IS proves its knowledge of a SIC-specific key K_{π}, which is derived from the input string.

After the IS has been authenticated successfully, the SIC shall enforce encryption and message authentication of the communication channel between the SIC and the IS by Secure Messaging techniques.

PACE shall be performed in the master file (MF) of the SIC. The SIC provides the relevant `SecurityInfos` in a transparent EF.CardAccess contained in the MF (and additionally in DG14 contained in the IDL application). Due to the execution on MF level, PACE provides an application independent authentication between IS and SIC that may also be used to get access to potential other domestic applications on the SIC.

NOTE See TR-PACE sections 2.2 and 3.1.5.

8.7.4 PACE relative to BAP

PACE differs from BAP in the following respects:

a) PACE introduces a mandatory master file (MF) structure.

b) PACE requires an EF.CardAccess file within the MF.

c) Security conditions are established at MF level for PACE.

d) The IDL application is selected by secure messaging using session keys derived in accordance with the PACE procedure.

In relation to BAP, the PACE protocol has the following advantages:

a) Strong session keys are provided independent of the strength of the input string.

b) The entropy of the input string(s) used to authenticate the IS can be very low (e.g. 6 digits are sufficient in general).

c) The binding between PACE and a Terminal Authentication is universal and does not depend on the input string.

The BAP logo in 8.5 may be used to denote the presence of an input string for PACE.

9 Security mechanism indicator

The security mechanism(s) deployed on an IDL with an SIC can be identified to reading authorities by the addition of tag '86' to EF.COM. Tag '86' identifies the data groups subject to each mechanism used.

NOTE Use of the security mechanism indicator is optional unless a mechanism mandates the security mechanism to be used.

Tag '86' shall have the following DER-encoded ASN.1 TLV structure:

```
SecurityMechanisms ::= SET OF SecurityMechanism

SecurityMechanism ::= SEQUENCE {
   mechanism MechanismIdentifier,
   datagroups SET OF INTEGER}

MechanismIdentifier ::= SEQUENCE {
   mechanism OBJECT IDENTIFIER,
   parameters ANY DEFINED BY mechanism OPTIONAL}
```

EXAMPLE Assume an implementation using LDS version 1.0 having the following data content – mandatory data (DG 1), optional licence holder data (DG 2), and optional finger biometric template (DG7). The optional finger biometric template is protected using EACv1 (see Annex D). The implementation makes use of the file identifier '55 66' for EF.CVCA and does not specify a short EF identifier.

EF.COM would be encoded as follows (in order to improve clarity, tags are printed in *ITALIC*, lengths are UNDERLINED and values are printed in normal text):

```
'60'  '36'
      '5F 01'  '02'
          '01 00'
      '5C'  '03'
          '61 6B 63'
      '86'  '2A'
          '31 28 30 21 30 1F 06 08 04 00 7F 00 07 02 02 02
          30 13 06 08 04 00 7F 00 07 02 02 02 02 01 01 30
          04 04 02 55 66 31 03 02 01 07'
```

where '31 28 30 21 ... 01 07' is the following DER-encoded ASN.1 structure:

```
SET
   SEQUENCE
      SEQUENCE
         OBJECT IDENTIFIER: 0.4.0.127.0.7.2.2.2
         SEQUENCE
            OBJECT IDENTIFIER: 0.4.0.127.0.7.2.2.2
            INTEGER: 01
               SEQUENCE
```



```
            OCTET STRING:55 66
SET
    INTEGER: 07
```

10 SIC LDS

10.1 General

In addition to the data groups identified in ISO/IEC 18013-2, Figure C.1, this document defines the data groups shown in Table 12 and Figure 18. The file structure for an IDL which supports PACE is shown in Figure 19. All tags used are listed in Annex F.

Table 12 — Assignment of file identifiers and Data Group Tags

Elementary file	Name	Short EF Identifier	EFID	Tag
EF.SOD	Document security object	'1D'	'001D'	'77'
EF.DG12	Non-match alert	'0C'	'000C'	'71'
EF.DG13	Active authentication	'0D'	'000D'	'6F'
EF.DG14	EACv1	'0E'	000E	'6E'

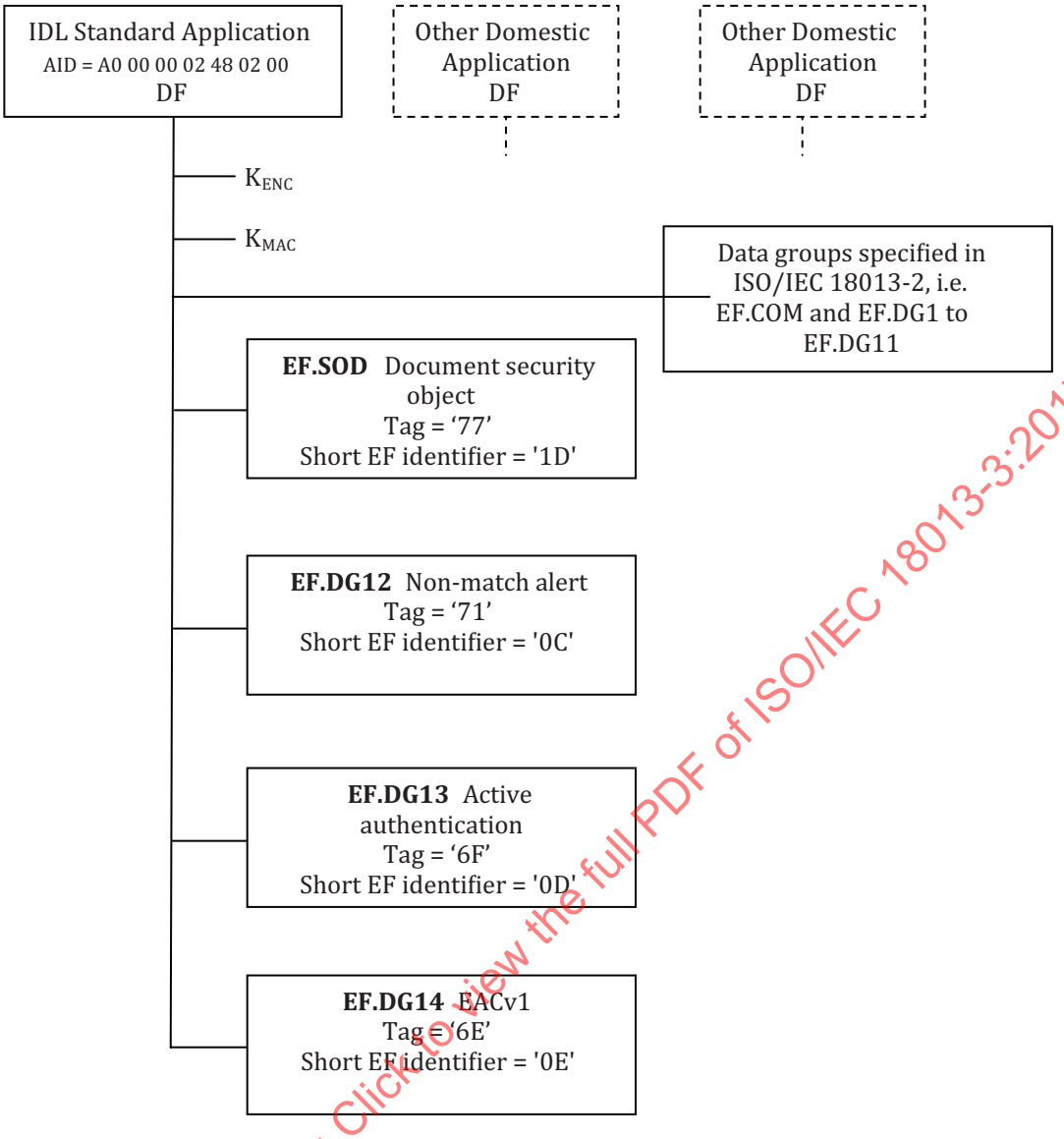
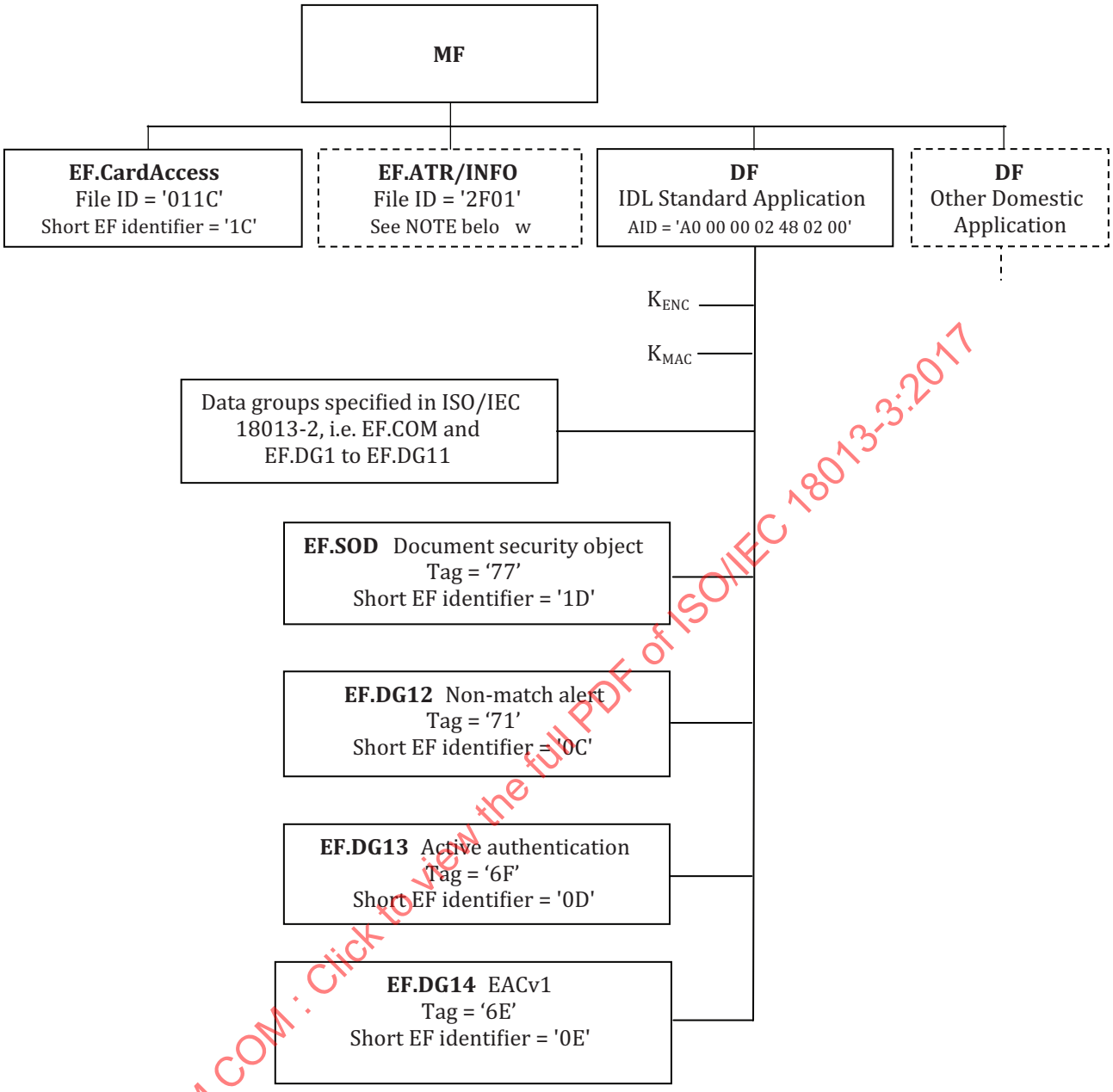


Figure 18 — Data groups



NOTE This EF may contain software functions supporting especially command chaining and extended Lc/Le field.

Figure 19 — File structure for an IDL supporting PACE

10.2 EF.SOD – Document security object (short EF identifier = '1D', Tag = '77')

EF.SOD is defined in 8.1.5.1.

10.3 EF.DG12 Non-match alert (short EF identifier= '0C', Tag = '71')

DG12 is defined in 8.4.4.

10.4 EF.DG13 Active authentication (short EF identifier = '0D', Tag = '6F')

DG13 is defined in 8.2.4.1.

10.5 EF.DG14 EACv1 (short EF identifier = '0E', Tag = '6E')

DG14 is defined in Annex D for EACv1.

If PACE is supported, additional DG14 content is defined in Annex C.

NOTE See TR-PACE section 3.1.5.

10.6 EF.CardAccess if PACE is supported (short EF identifier = '1C')

EF.CardAccess is defined in Annex C for PACE.

NOTE See TR-PACE section 3.1.5.

Annex A
(informative)

Public key infrastructure (PKI)

A.1 General

This annex suggests a structure for a public key infrastructure specifically in respect of IDLs that are used internationally. Such a use environment poses unique challenges, primarily the infeasibility of any one issuing jurisdiction to conclude, implement and maintain key sharing agreements and infrastructures with all other issuing jurisdictions (given the assumption that a "super certification authority" that controls and issues keys globally is not available).

The concepts in this annex were developed from first principles, and explore the functional and logical aspects of a PKI. Standardisation of technical aspects relating to implementation (e.g. certificate formats) are outside the scope of this annex, although some issues that would need standardisation are pointed out. Consequently, this annex describes the mechanisms to establish a "trust model" more than it describes a traditional PKI.

The trust model described in this annex is based on a "trust by proxy" principle, i.e. A trusts C because A trusts B and B trusts C. Consequently, the ultimate responsibility to establish trust in a public key remains with the RA (also see Annex B dealing with trust establishment). The trust model should not be seen as infallible confirmation of the origin and integrity of a public key.

A.2 PKI Design principles

In addition to the principle stated in A1, the trust model was designed to comply with the following principles:

a) No central CA is available.

b) The trust model should exploit existing trust relationships.

c) The rules for setting up and maintaining the trust network should not require any one entity to manage the trust network, but should allow for the trust network to essentially manage itself.

d) Use a two-level key approach, consisting of a root key-pair and a document key pair. The document key pair is used to sign and verify an IDL, and the root key pair is used to sign and verify the document key and other information as described below. Public root keys are to be exchanged by out-of-band means.

NOTE 1 A one-level key approach is possible but less appropriate given that keys used to sign IDLs are replaced periodically.

NOTE 2 In this context, "sign" means that a private key is used to create a digital signature for a certificate that contains a public key and/or other information.

NOTE 3 Given the autonomy of each country, and the political sensitivities that exist between some, a trust model that uses one global certification authority is considered infeasible.

NOTE 4 Existing trust relationships are an ideal starting point to build a trust network. For example, the American Association of Motor Vehicle Administrators (AAMVA) already has trust relationships established with most IA's in North America. In Southern Africa, the Southern Africa Development Community (SADC) has relationships set up with all the countries in Southern Africa, with a similar function performed by the Economic Community of West African States (ECOWAS) in West Africa, and the Common Market for Eastern and Southern Africa (COMESA) in Eastern and Southern Africa. In Europe, the European Commission has relationships with numerous IA's.

A.3 General description

NOTE 1 The trust model is somewhat similar to the user-centric trust model employed by PGP, and as discussed in Reference [4].

Each IA generates two sets of key pairs, a root key pair, and a document key pair. The IA signs its own public root key using its private root key. This means that the IA certifies that the public root key is associated with the IA. In a more traditional environment, a CA would sign e.g. IA A's public root key, thus certifying the association between the public root key and IA A. If IA B trusts the CA, IA B would then also trust the association between IA A and its public root key. If IA A self-signs its public root key, and IA B trusts IA A, a CA becomes superfluous (at least for the purpose of associating IA A with its public root key). The public root key is distributed out-of-band only. The public document key is also signed with the private root key. For standard encoding, the public document key may optionally be distributed with the IDL.

NOTE 2 The root key as used in this document is similar to the Country Signing CA Key in the ICAO PKI Specification (see Reference [9]).

NOTE 3 The document key as used in this document is similar to the Document Signer Key in the ICAO PKI Specification (see Reference[9]).

NOTE 4 ICAO's PKI Specification (see Reference[9]) mentions the out-of-band *confirmation* of a root public key, thus inferring that the actual distribution of such keys may take place separately from the out-of-band confirmation, possibly using "in-band" communication.

As noted above, it is infeasible to expect each IA's public root key to be distributed (out of band) to every other IA. Consequently, if IA A trusts IA B, then IA A would also like to know who else IA B trusts. IA B thus needs to publish (see A.4.1 for a discussion of the term "publish".) a list (signed with IA B's private root key) of the IAs (including IA A, if applicable) that it trusts. IA B also signs the public root key of each IA in IA B's trust list, and publishes all the signed public root keys. B thus publishes the documents (or certificates) shown in Figure A.1.

NOTE 5 IA B would presumably (but not necessarily) only directly trust those IAs for which it has received the public root key in an out-of-band fashion.

NOTE 6 In addition, IA B would also publish information on IA B's public keys used to sign IDLs. This however does not directly pertain to the trust network, and is only discussed later.

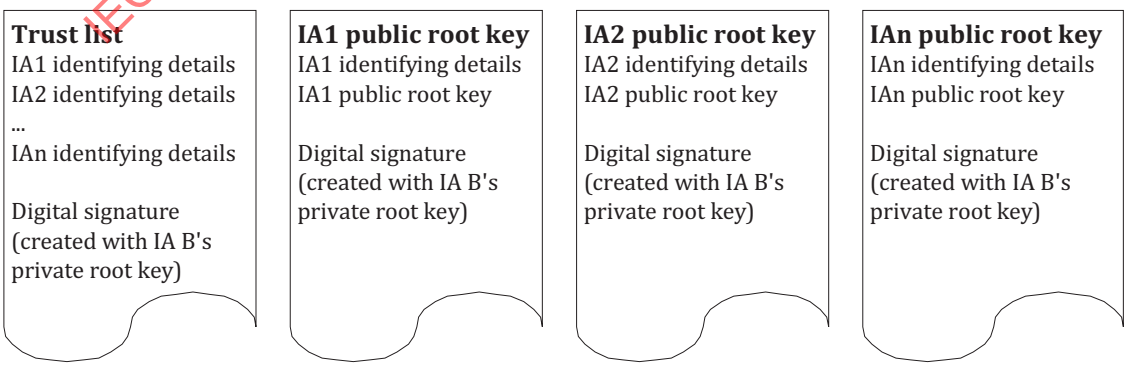


Figure A.1 — Certificates published by an IA

If every IA publishes the same items that IA B publishes (i.e. a signed trust list, and signed public root keys of each authority on the trust list), it enables IA A to construct a diagram of the complete trust network. For example, if IA B trusts IA C, IA A uses IA C's public root key (signed by IA B), to verify IA C's trust list. If IA C trusts IA D, IA A uses IA C's public root key to verify IA D's public root key, and then verifies IA D's trust list, and so on. IA A thus compiles a trust network diagram which can be used to verify any IDL presented.

NOTE 7 The trust network diagram includes all the public keys, lists and other information required to verify an IDL in an off-line manner. Typically, the trust network diagram is updated periodically (e.g. daily), and used as reference for all internal verification actions.

The above allows IA A to verify the public root key of each IA (in the trust network). The public root key then is used to verify the public document keys published by each IA.

Public root keys distributed between IAs in an out-of-band fashion essentially become trust anchors, and should be stored, protected and used in an appropriately secure manner.

NOTE 8 The proposed trust model requires the unrestricted availability of the public root key. The ICAO PKI Specification (see Reference[9]) however appears to discourage such availability of ICAO's equivalent of the public root key.

It is anticipated that the global trust network will eventually consist of a collection of "local" trust networks, with the different local trust networks connected by a limited number of trust "links". For example, in the Southern Africa Development Community (SADC), South Africa may act as an aggregator for the SADC countries. That is, each SADC country exchanges public root keys (out-of-band) only with South Africa, and vice versa. South Africa then signs each public root key, and publishes same along with the list of all SADC countries. In the European Union (EU), the European Parliament may decide to act as a central trust broker, or PIA. That is, the EU obtains the public root key for each IA in the EU, signs it with the EU private root key, and publishes same. The American Association of Motor Vehicle Administrators (AAMVA) may fulfil a similar function in North America. Figure A.2 illustrates an example of a trust network.

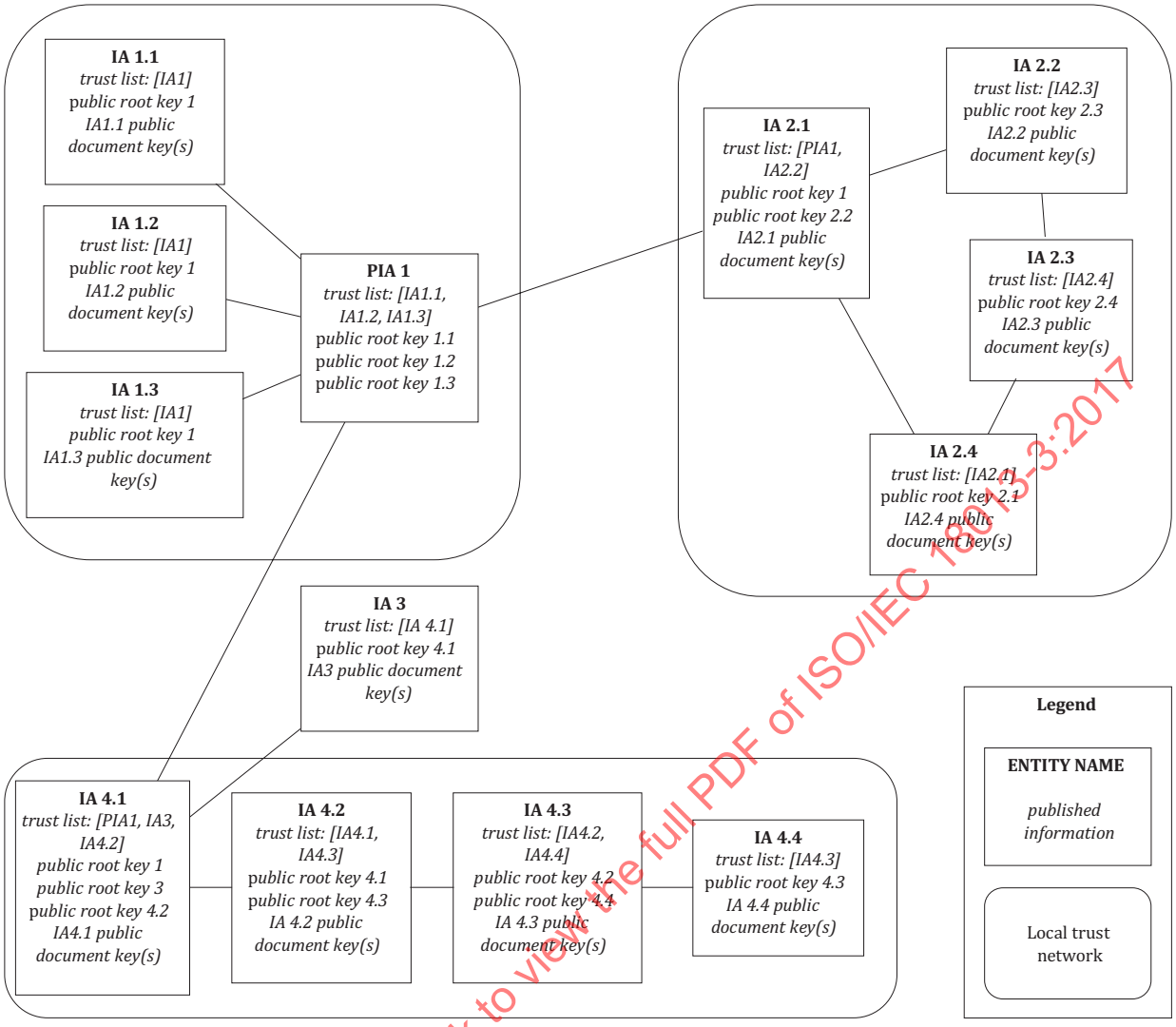


Figure A.2 — Trust network example

A.4 Implementation

A.4.1 Publication mechanism

It is suggested that the information to be published, be published (as signed certificates) in a location that is easily accessible by all verifying entities. Consequently, each IA's identifying details should include directions on how and where published information can be accessed.

NOTE The Internet would be a location that complies with the above accessibility requirements. However, security concerns have been expressed about using the Internet. If the Internet is used, a requirement to use server side authenticated SSL (similar to the ICAO requirement) for all communication can be added to enhance security. This implies that each IA will have to obtain a single server key from a commercial party. The disadvantage of alternative publication locations in general is that they may not be readily accessible to all verifying entities (with the potential for consequential difficulties for the IA's card holders). In the end, it remains the responsibility of the IA to select a suitable publication location.

The security of the publication location is the responsibility of each IA. However, the authenticity of all published information can be verified using the IA's public root key, thus adding an additional layer of security to published information.

It follows that in cases where an IA may not have the necessary infrastructure or expertise or may not want to publish and/or maintain published information in-house, such publication may be performed (under agreement) by other IAs or trusted third parties.

The availability of published information is crucial. If published information is not available, a verifying entity cannot update its trust network diagram, and thus will be unaware of any changes in the trust network diagram such as new document keys issued, document keys revoked, or IAs removed from the trust network. Thus, this approach is vulnerable to a denial of service attack (see A.7.3).

A.4.2 Information published

As discussed in A.3, each IA (in its role as VA) publishes a trust list and the public root key of each IA on the trust list. Each of these documents is signed with the publishing VA's private root key.

Each VA may optionally also publish the network trust diagram that it has constructed. Such a published network trust diagram can potentially be referenced by local industry (e.g. airlines and car hire companies) as a primary source to verify foreign driving licences.

NOTE In a sense, a published trust network diagram is similar to the public key directory defined in the ICAO PKI Specification (see Reference [9]), with the difference that it is intended for local use only. However, nothing prevents a company in e.g. Australia to use (because it trusts) a network trust diagram published by e.g. Germany.

The VA (if it is also an IA) also publishes its own public document keys.

A standard format for each of the above documents (or certificates) has to be specified.

A.5 Certificate content

In considering certificate content, cognizance was taken of industry experts' recommendations that X.509 certificates should only be used if absolutely necessary (see Reference [5] BSI Technical Guideline TR-03110-1 and BSI Technical Guideline TR-03110-3). Consequently, and in keeping with the intent of this annex (see A.1), the certificate content proposed below is based solely on the functional requirements pertaining to the IDL environment and the proposed trust model. The actual certificate profile eventually specified may or may not be X.509 compliant.

NOTE X.509 compliancy has advantages and disadvantages, which need to be carefully considered by an IA prior to implementation. The disadvantages are discussed in the noted references, and in essence involve unneeded complexity and questionable "fitness for purpose". The disadvantage to using a non-X.509 certificate is that IAs' existing environments may already be set up to accommodate only X.509 certificates.

A.5.1 Verification processes

As mentioned above, the functional processes drive the certificate contents. This clause discusses the processes involved.

When a public document key certificate needs to be verified, the following steps are executed:

a) Identify the IA that signed the certificate. Primary identification is based on the ISO Issuer Number in Data Group (DG) 3 (this field thus becomes mandatory if a digital signature is used). Additional identification information includes directions on how and where information published by the IA can be accessed.

b) Using a trust network diagram, identify the IA's public root key(s). If the IA has published more than one root key, identify the public root key used associated with the public document key certificate.

   NOTE If the IA does not appear on the VA's trust network diagram, the public document key certificate cannot be verified. The IA first has to be added to the VA's trust network diagram via the out-of-band exchange of the IA's public root key with any of the existing IAs on the VA's trust network diagram.

   The following options exist to identify this particular public root key:

1) Include a public root key identifier in the public document key certificate and include the same identifier in the public root key certificate.

2) Use the date that the public document key certificate was signed to identify the public root key that was used for signing certificates during this period. This requires that the public root key certificate contain "valid for signing from" and "valid for signing until" dates, and limits the IA to the use of only one key at any given time.

c) Use the IA's public root key to check the public document key certificate's digital signature. This requires knowledge of the digital signature algorithm (and accompanying parameters) used. The algorithm and parameters are specified in the public document key certificate.

d) Verify that the public document key is still valid, that is, that the document key has not been revoked yet. This can be set up in either of the following ways:

1) Use certificate revocation lists.

2) Use a "Revoked Y/N" field in the public document key certificate (this field would always be "N" for a public key certificate included on the IDL; only public keys published by the IA can have a "Y" value for this field.). This is unusual in that it implies that a certificate for the same key can be issued twice. However, it also negates the need for certificate revocation lists (refer to A.6.2 for more information), thus reducing the complexity of the overall solution.

When an IDL needs to be verified, the following actions are involved.

a) Identify the IA. The same identification process as for the public document key certificate verification is followed.

b) Obtain the correct public document key. Regardless of whether or not the public document key is included on the IDL, an attempt should be made to obtain the public document key from the appropriate public document key certificate on the IA's publication area (or alternatively as included in the VA's most recent trust network diagram).

For compact encoding, the appropriate public document key is identified by comparing the issue date of the IDL with the "valid for signing from" and "valid for signing until" dates of the available public document keys of the IA. For standard encoding, the appropriate public document key can be identified using a key identifier.

c) Verify the public document key certificate, as described above.

d) Check the digital signature on the IDL using the public document key. This requires knowledge of the digital signature algorithm and parameters used to sign the IDL. This information is stored in DG.SOD (for both compact encoding and standard encoding).

A.5.2 Document key

Based on the discussion in A.5.1, a public document key certificate should contain the following information:

a) IA identifying details;

b) public document key;

c) public document key identifier (if dates are not used to identify the public document key);

d) identifier of the public root key used to sign the document public key certificate (if dates are not used to identify the public root key);

e) beginning (and ending) issuing dates for which the key is valid (at the time of signing the certificate) (if dates are used to identify the public document key);

f) digital signature algorithm and parameter information;

g) revocation indicator (Yes/No);

h) revocation date (mandatory if revocation indicator is Yes) (if dates are used to identify the public document key);

i) notes (optional), which can be used to add additional information regarding the revocation, in English, French or Spanish if required;

j) date of signing (if dates are used to identify the public root key), which is used to identify the private key that was used to sign the certificate; this is optional if the public key is included on the IDL, as it can be assumed that the date of signing is the same as the IDL issue date;

k) digital signature (assuming a digital signature with appendix scheme).

A.5.3 Root key

Based on the discussion in A.5.1, a root key certificate should consist of the following information:

a) IA identifying details;

b) public root key;

c) public root key identifier (if dates are not used to identify the public root key);

d) beginning (and ending) signing dates for which the key is valid (at the time of signing the certificate);

e) date of signing (optional);

f) digital signature (assuming a digital signature with appendix scheme).

Note that whenever a private document key is compromised, all the documents signed with the private key become suspect (and not only those documents signed prior to the compromise).

A.6 Key revocation

A.6.1 Root key compromise

A private root key compromise has the following far-reaching consequences:

a) none of the documents signed with the private root key can be considered “safe” anymore;

b) the IA (whose private root key has been compromised) is effectively deleted from the trust network.

Due to the above, notification via publication (as defined in A.4.1) is infeasible. The fact that the private root key has been compromised can be published, but technically no VA will be able to verify such information, since there is no way in which the compromised IA can sign such publication.

In case of a compromise, the compromised IA thus has to notify all the immediate VAs of the compromise in an out-of-band fashion. Following such notification, the immediate VAs have to immediately remove the compromised IA from their trust lists (and their trust network diagrams). The compromised IA now has to create a new root key pair and distribute the new public root key to the immediate VAs. At the same time, the compromised IA has to resign all other information that it wishes to publish, e.g. the trust list, public root keys for the IAs on the trust list, and the compromised IA’s existing public document keys.

It thus is in each IA’s best interest to communicate any compromise as expediently as possible to all immediate VAs. To this end, each IA should keep a list of all its immediate VAs. It is also recommended that the compromised IA re-obtain (out-of-band) the public root keys of the IAs in the compromised IA’s

trust list, to ensure that the public keys re-signed and re-published by the compromised IA (with the IA's new root private key) are in fact the correct keys.

NOTE The list of an IA's immediate VAs is not necessarily the same as the IA's trust list. An IA may not trust all the VAs to which it provided a copy of its public root key (by out-of-band means).

For non-immediate VAs, the compromised IA's public root key is essentially revoked when the compromised IA is removed from the immediate VAs' trust lists. Each VA thus needs to carefully consider the frequency with which it updates its trust network diagram.

A.6.2 Document key compromise

The compromise of a private document key is easier to communicate and process than in the case of a private root key. When a private document key is compromised, the compromised IA (i.e. the IA whose private document key was compromised) simply re-publishes the public document key associated with the compromised private document key with the value of the "Revocation indicator" field set to "Y", and (if necessary) the value of the ending issuing date associated with the compromised public key set to the date the private document key was last used to issue an IDL.

This approach does away with the need to maintain certificate revocation lists. The compromised key is noted as such by any other VA the moment the VA updates its trust network diagram.

A.7 Trust model weak points

A.7.1 Overview

The trust model presented in this annex is to a large extent predetermined by the design principles and constraints stated in A.2. However, since no trust model is perfect, it is important to also take note of the weak points of the model, so that these can be adequately addressed. This subclause points out some of the weak points of the trust model. Weak points concerning other areas (e.g. testing, issuing, document security) are not discussed here.

A.7.2 General

One of the inherent weaknesses of the "trust by proxy" approach is that VAs may not all have the same criteria for measuring trustworthiness. If IA A trusts IA B and IA B trusts IA C, but IA A and IA B do not use the same criteria to determine trustworthiness then IA C does not necessarily comply with IA A's trustworthiness criteria, even though the "trust by proxy" approach would imply that IA A can trust IA C.

The trust chain can also become rather long, introducing ever more opportunity for a trust breakdown. A solution to this conundrum is for a VA to adapt its trust network (within the constraints imposed by cost and logistical considerations) so that higher volume IDL verifications "flow" over shorter trust chains. That is, a VA could establish direct trust relationships in such a manner that it minimizes the sum of chain lengths for all IDLs verified.

NOTE This approach does not guarantee the lowest probability of letting a problematic IDL slip through. Knowing about the above approach, criminals can specifically target the end points of long trust chains in a VA's trust network in their attempts to circumvent the system.

If a breakdown in trust occurs or is suspected (i.e. it is determined or suspected that an IDL is verified when it shouldn't have been, or vice versa), the point where it is discovered may be several trust points removed from the IA. The longer the trust chain involved, the more cumbersome it becomes to confirm a trust breakdown and to identify the point of compromise. As described above, minimizing the sum of chain lengths for all IDLs verified can minimize this problem.

A VA may include an IA in a trust list for political reasons, even if the VA does not trust the IA internally. Even though such conduct would effectively sabotage the trust model, it cannot be ruled out. Consequently, it is important that VAs augment the trust model with other methods of establishing trust in an IDL issued by an IA.

A.7.3 Attacks

In general terms, (at least) the following attacks are possible against any two-level PKI:

a) obtain private root key;

b) replace private root key;

c) replace trusted root key;

d) obtain private document key;

e) replace private document key;

f) replace trusted document key;

g) denial of service attack.

The above attacks can take many shapes and forms, some of which are unique to the trust model used, and others that would be applicable regardless of the trust model. The paragraphs that follow discuss some of the attacks that are specific to the proposed trust model.

The distributed nature of the proposed trust model requires each IA to take responsibility for the safekeeping of its own private keys. The general level of security which an IA is capable of or willing to employ, or the conscientiousness with which it follows its security procedures, may be less than would have been the case with one central (or even more than one) commercial certification authority. This can create a weak spot in the proposed trust model.

Trust in a public root key is established by exchanging such keys by out-of-band means. The main attack on such an exchange would be to replace the public key somewhere in the process. IAs should be aware of this risk, and implement appropriate procedures to secure their out-of-band key exchanges.

Several attacks against published information (e.g. trust lists, public root and document keys) are possible. The appeal of some of these attacks, and the expected duration before the attacks are uncovered, are related to the frequency with which a trusted root public key is used to verify certificates that were supposed to be signed by the trusted root public key (essentially the frequency with which the trust network diagram is verified). Other attacks (on published information) are not influenced by the frequency of trust network verification. These attacks are however likely to be uncovered over time, when it is realized that IDLs that do not verify is due to incorrect public keys being used to try and verify authentic IDLs.

A variant on the published information attack is to try and sneak the details of a fictitious IA into the trust list and public keys published by an existing IA. Although this would require some inside help (as the information has to be signed using the existing IA's private root key), this subterfuge can potentially remain undetected.

Denial of service attacks could be used on their own or in conjunction with some of the attacks mentioned above. A denial of service attack will prevent a VA from updating its trust network diagram, creating opportunities for many other attacks.

The ultimate decision on which IAs to trust and which IAs not to trust lies with the VA. Setting up and maintaining a trust network diagram thus requires active involvement from the VA. Any VA that does not take this responsibility seriously or does not allocate sufficient resources can become a liability for the whole trust network.

The bigger and more complex a trust network diagram becomes, the more opportunity there is for situations arising that require manual decision-making.

EXAMPLE 1 IA A has immediate VAs B and C. VA D's trust network diagram has paths to both VA B and VA C. For some reason, VA B removes IA A from its trust list. Does VA D now also remove IA A from its trust network diagram?

EXAMPLE 2 After investigating IA B's security procedures and practices, VA A decides not to include IA B in its trust list. However, VA A does include IA C in its trust list, and IA C includes IA B in its trust list. VA A thus has to manually ensure that IA B is not included in its trust network diagram.

Annex B
(normative)

Basic access protection

B.1 General

BAP is a mechanism and protocol to protect identity documents with a SIC against skimming attacks.

This protection is achieved by requiring the establishment of a secure channel using pre-defined key material which should only be revealed by closer physical inspection of the document, before access is granted to information stored in the SIC.

The secure channel protects the integrity and authenticity of authorized communication between an IS and an identity document. If the entropy of the key material is high enough, a certain amount of protection against eavesdropping attacks is also achieved.

NOTE 1 Because the same pre-defined key material is used for all communication sessions with a given document, this protocol does not give forward-secrecy. This means that, if knowledge about the keying material is gained, it can be used to decrypt any past sessions. However, knowledge of a particular session's session keys does not enable the decryption of past or future sessions.

NOTE 2 This protocol is based on ICAO's Basic Access Control.

B.2 Parameters

Referring specifications shall specify the following when referencing BAP:

a) the source of K_{doc};

b) the method(s) by which K_{doc} is entered into the IS.

NOTE This document uses information printed on the IDL in machine and/or human-readable form as K_{doc}. The SAI demarcates the information used. The first byte of the input string indicates the use of BAP and has the value "1". For further details, see 8.3 and 8.5.

B.3 Protocol

BAP comprises the following steps:

a) Document basic access keys are established using the key derivation mechanism described in B.4.

b) The IS and the SIC mutually authenticate and derive session keys. The authentication and key establishment protocol described in B.5 is used.

c) After successful authentication, subsequent communication is protected by Secure Messaging as described in B.6. Access shall only be granted as long as secure messaging is active.

B.4 Key derivation mechanism

The following key derivation mechanism is used to derive keys from a key seed (K_{seed}) for both the establishment of the document basic access keys and the establishment of the session keys for secure messaging.

A 32-bit counter c is used to allow for deriving multiple keys from a single seed. Depending on whether a key is used for encryption or MAC computation, the following values shall be used:

— c = 1 (i.e. '00 00 00 01') for encryption;

— c = 2 (i.e. '00 00 00 02') for MAC computation.

The following steps are performed to derive a key K from the seed K_{seed} and c using the cryptographic hash function *h*:

a) Let D be the concatenation of K_{seed} and c ($D = K_{seed} || c$).

b) Using *h*, calculate the hash H of D ($H = h(D)$).

c) The *k* leftmost bits of H form the key K.

The document basic access keys K_{enc} and K_{mac} are derived using the mechanism described above, with c = 1 and 2, respectively. In addition, the most significant 16 bytes of the $h(K_{doc})$ is used as the value for K_{seed} . *h* is the selected cryptographic hash function defined the BAP configuration.

K_{doc} should be different for every document and care should be taken to ensure that K_{doc} is sufficiently random for the intended application.

NOTE The entropy of K_{doc} is the upper bound on available entropy for the secure messaging keys. For example, if K_{doc} only provides 30 bits of entropy, the derived keys cannot contain more – even if the key size is larger.

B.5 Authentication and key establishment

Authentication and key establishment is provided by a three pass challenge-response protocol according to ISO/IEC 11770-2:1996 key establishment mechanism 6 using the selected block cipher *e*. A cryptographic checksum according to the selected MAC algorithm *m* is calculated over and appended to the cipher texts. The modes of operation described in B.7 shall be used. Exchanged nonces shall have a size of 64 bits, exchanged keying material shall be *k* bits long. Distinguishing identifiers shall not be used.

In more detail, the IS and SIC perform the following steps.

NOTE The abbreviations IS and SIC used here are equivalent to IFD and ICC, respectively as used in ISO/IEC 7501-1 (ICAO Doc 9303-1).

a) The IS requests a challenge RND.ICC by sending the GET CHALLENGE command.

b) The SIC generates and responds with a random nonce RND.ICC.

c) The IS performs the following operations:

   1) Generate a random nonce RND.IFD and random keying material K.IFD.

   2) Generate the concatenation S = RND.IFD || RND.ICC || K.IFD.

   3) Compute the cryptogram $E\_IFD = e[K_{enc}](S)$.

   4) Compute the checksum $M\_IFD = m[K_{mac}](E\_IFD)$.

   5) Send a MUTUAL AUTHENTICATE command using the data E_IFD || M_IFD.

d) The SIC performs the following operations:

   1) Check the checksum M_IFD of the cryptogram E_IFD.

   2) Decrypt the cryptogram E_IFD.

   3) Extract RND.ICC from S and check if the IS returned the correct value.

4) Generate random keying material K.ICC.

5) Generate the concatenation R = RND.ICC || RND.IFD || K.ICC

6) Compute the cryptogram E_ICC = $e[K_{enc}](R)$.

7) Compute the checksum M_ICC = $m[K_{mac}](E\_ICC)$.

8) Send the response using the data E_ICC || M_ICC.

e) The IS performs the following operations:

1) Check the checksum M_ICC of the cryptogram E_ICC.

2) Decrypt the cryptogram E_ICC.

3) Extract RND.IFD from R and check if the SIC returned the correct value.

B.6 Secure messaging

After a successful execution of the authentication protocol, both the IS and the SIC compute session keys KS_{enc} and KS_{mac} using the key derivation mechanism described in B.4 with (K.ICC ⊕ K.IFD) as key seed. All further communication shall be protected by secure messaging (SM) as described in ISO/IEC 7816-4:2013 according to the requirements below. The modes of operation described in B.7 shall be used.

B.6.1 Message structure of SM APDUs

The SM data objects shall be used according to Table B.1 in the following order:

— command APDU: [DO‘87’] [DO‘97’] DO‘8E’;

— response APDU: [DO‘87’] DO‘99’ DO‘8E’.

All SM data objects shall be encoded in BER-TLV as specified in ISO/IEC 7816-4:2013. The command header shall be included in the MAC calculation, therefore the class byte CLA shall be ‘0C’.

The actual value of Lc will be modified to Lc’ after application of secure messaging. In the protected command APDU, the *new Le* byte shall be set to ‘00’, while the value of the original Le byte may be conveyed in the appropriate data object.

Table B.1 — Usage of SM Data Objects

	DO‘87’	DO‘97’	DO‘99’	DO‘8E’
Content	Padding-content indicator byte (shall be ‘01’) followed by the cryptogram	Le (1 or 2 bytes)	Processing status (SW1-SW2)	Cryptographic checksum (MAC)
Command APDU	Mandatory if data is sent, otherwise absent.	Mandatory if data is requested, otherwise absent.	Not used.	Mandatory.
Response APDU	Mandatory if data is returned, otherwise absent.	Not used.	Mandatory, only absent when a SM error occurs.	Mandatory if DO‘87’ and/or DO‘99’ are present.

Figure B.1 shows the transformation of an unprotected command APDU to a protected command APDU in the case that data and Le are available (case 4). If no data is available (case 1 and 2), leave building DO‘87’ out. If Le is not available (case 1 and 3), leave building DO‘97’ out.

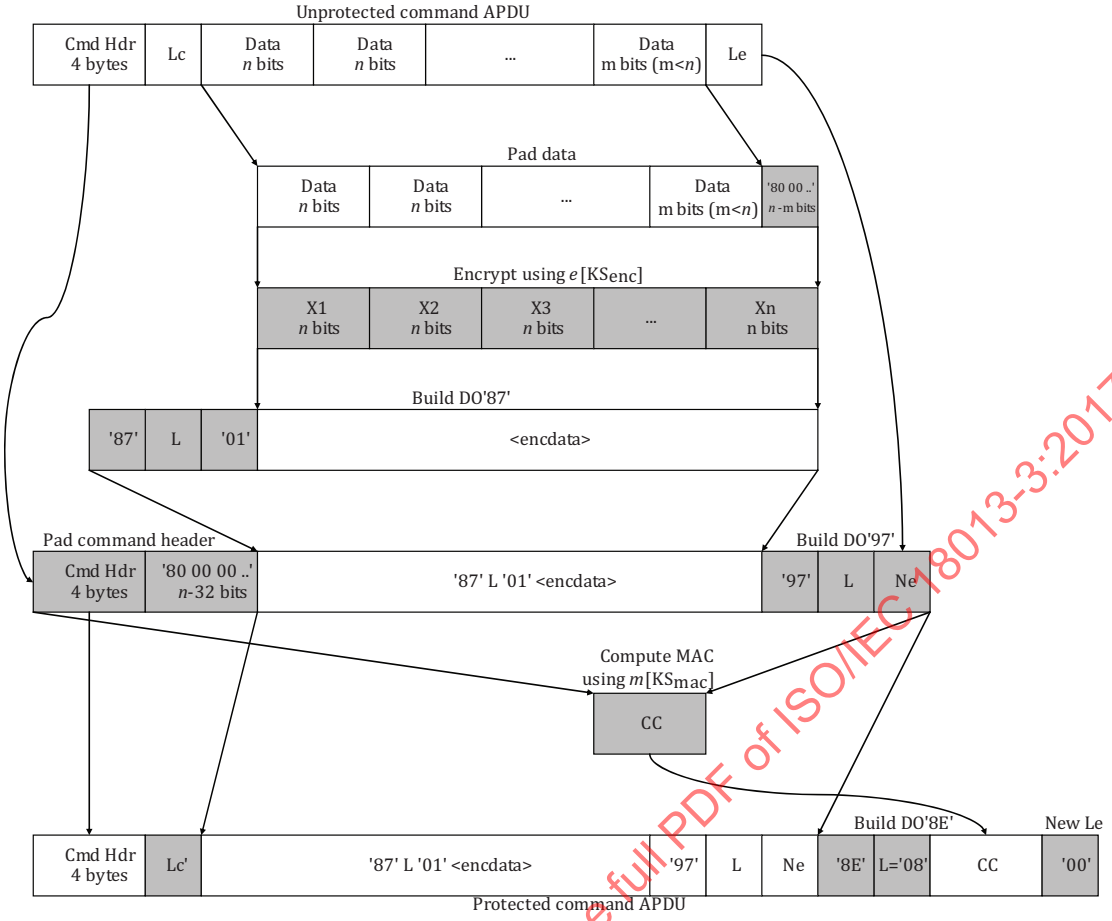


Figure B.1 — Computation of a SM command APDU

Figure B.2 shows the transformation of an unprotected response APDU to a protected response APDU in case data is available. If no data is available, leave building DO‘87’ out.

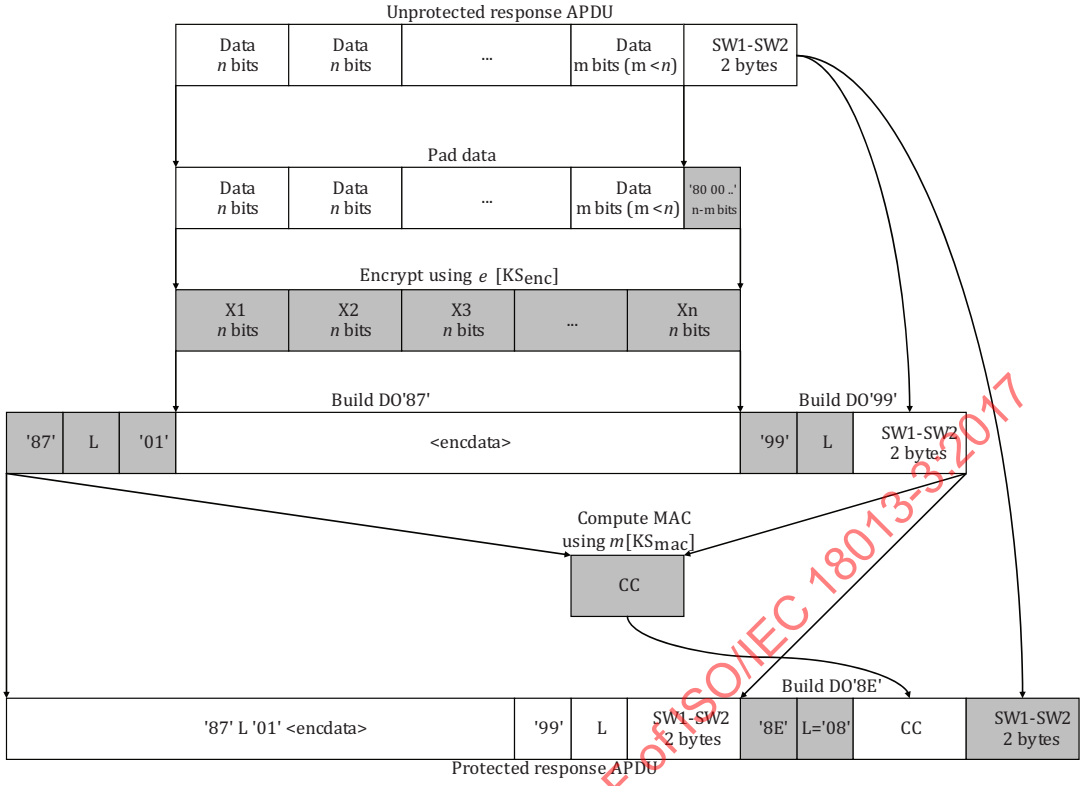


Figure B.2 — Computation of a SM response APDU

B.6.2 SM errors

When the integrated circuit in the document recognizes an SM error while interpreting a command, then the secure messaging session shall be aborted and the status words returned in plain. ISO/IEC 7816-4:2013 defines the following status bytes to indicate SM errors:

a) '6987' Expected SM data objects missing

b) '6988' SM data objects incorrect

B.6.3 Other errors

In the application context, other errors (i.e. status words other than '90 00') may occur that are protected under SM. Under these conditions SM shall not be aborted.

B.7 Modes of operation

B.7.1 Encryption

During encryption, the selected block cipher shall operate in cipher block chaining (CBC) mode with an initialization vector (IV) of *n* '0' bits.

During authentication, the data to be encrypted shall only be padded if it is not a multiple of the block cipher's block length *n*. During the computation of SM APDUs, data shall always be padded.

Padding according to ISO/IEC 9797-1:1999 padding method 2 shall be used.

B.7.2 Message authentication

Cryptographic checksums are calculated using the selected MAC algorithm with an initialization vector (IV) of *n* '0' bits. The MAC length shall be 8 bytes.

After a successful authentication, the datagram to be MACed shall be prepended with an 8 byte send sequence counter (SSC). If the block length n is larger than 64 bits, the SSC shall be prepended by n-64 zero bits to form a full block. The SSC is incremented every time before a MAC is calculated, i.e. if the starting value is x, in the first command the value of SSC is x+1. The value of SSC for the first response is then x+2. The initial value of the SSC is computed by concatenating the four least significant bytes of RND.ICC and RND.IFD, respectively:

SSC = RND.ICC (4 least significant bytes) || RND.IFD (4 least significant bytes)

B.8 Basic access protection configuration

When selecting *h*, *e*, *n*, *k* and *m*, IA's shall pick a valid combination, herein called "configuration", from the choices in this subclause.

This document supports one configuration, which uses the following algorithms:

— SHA-1 according to ISO/IEC 10118-3:2004;

— TDEA according to ISO/IEC 18033-3:2005 ("Triple DES");

— ISO/IEC 9797-1:1999 MAC algorithm 3.

NOTE The first edition of ISO/IEC 18013-3:2005 supported multiple BAP configurations. This edition supports BAP configuration 1 only. See Table B.2.

Table B.2 — BAP configuration 1

One-byte identifier	'31'
OID	`bap-config-1`
Hash algorithm, *h*	SHA-1
Block cipher, *e*	TDEA using keying option 2. The leftmost 64 bits of the 128-bit key form K1, while the right-most 64 bits form K2.
Block length, *n*	64 bits
Key length, *k*	128 bits Note that only 112 bits are effectively used by this block cipher as keying material; certain implementations may require adjustment of the remaining parity bits.
MAC algorithm, *m*	ISO/IEC 9797-1:1999 MAC algorithm 3 with block cipher TDEA and padding method 2. TDEA is used with the keying option that K1=K2=K3 (reduces to DEA). For MAC calculation, the leftmost 64 bits of the 128-bit key form K, while the right-most 64 bits form K'. The resulting MAC algorithm is also known as "Retail MAC".

BAP configuration 1 is equivalent to Basic Access Control (BAC) as described in ICAO Doc 9303 (ISO/IEC 7501-1), Annex A, Appendix 5.

The following ASN.1 object identifier is used to refer to the BAP configuration 1:

```
   bap-config-1 OBJECT IDENTIFIER ::= {
      iso(1) standard(0) driving-licence(18013) part-3(3) security-mechanisms(2) id-sm-
BAP(1) 1
   }
```

B.9 Card commands

B.9.1 GET CHALLENGE

The GET CHALLENGE command in Table B.3 receives a (true) random challenge from the card for authentication in the subsequent MUTUAL AUTHENTICATE command in Table B.4.

Table B.3 — Command APDU: GET CHALLENGE

CLA	As defined in ISO/IEC 7816-4:2013
INS	0x84 GET CHALLENGE
P1	0x00 No information given
P2	0x00 (any other values reserved for future use)
Lc field	Absent
Data field	Absent
Le field	0x08

Table B.4 — Response APDU: GET CHALLENGE

Data field	8-byte random challenge (RND.ICC)
SW1-SW2	'9000' Normal processing Other Operating system dependent error

B.9.2 MUTUAL AUTHENTICATE

The MUTUAL AUTHENTICATE command in Table B.5 is used to submit the host cryptogram to the card and receive the card cryptogram in the Response MUTUAL AUTHENTICATE in Table B.6.

Table B.5 — Command APDU: MUTUAL AUTHENTICATE

CLA	As defined in ISO/IEC 7816-4:2013
INS	0x82 MUTUAL AUTHENTICATE
P1	0x00 reference algorithm implicitly known
P2	0x00 qualifier reference implicitly known
Lc field	Length of subsequent data field.
Data field	Host cryptogram including MAC (E_IFD || M_IFD).
Le field	0x28

Table B.6 — Response APDU: MUTUAL AUTHENTICATE

Data field	Card cryptogram including MAC (E_ICC || M_ICC)
SW1-SW2	'9000' Normal processing '6300' Verification failed. Host cryptogram or MAC verification failed Other Operating system dependent error

B.10 Worked example (informative)

This subclause provides a worked example for BAP configuration 1. Note that not all steps are explicitly shown.

Static document keying material:

K_{doc} = '31239AB9CB282DAF66231DC5A4DF6BFBAE'

Computation of basic access keys:

Input: $K_{seed} = H_{SHA-1}(K_{doc})$

K_{seed} = 'BFE25204D0A589510CD9C397C064CC2DAF5E952F'

Encryption Key (K_{enc}) computation:

1. Concatenate K_{seed} and c (c = 1):

   D = 'BFE25204D0A589510CD9C397C064CC2D
   00000001'

2. Calculate the hash of D:

   H_{SHA-1}(D) = 'AE161CC6AFB5FB766BD20016CAC3F181
   E77D9428'

3. Form key:

   K_{enc} = 'AE161CC6AFB5FB766BD20016CAC3F181'

   $K_1 = K_3$ = 'AE161CC6AFB5FB76'

   K_2 = '6BD20016CAC3F181'

Message Authentication Key (K_{mac}) computation:

4. Concatenate K_{seed} and c (c = 2):

   D = 'BFE25204D0A589510CD9C397C064CC2D
   00000002'

5. Calculate the hash of D:

   H_{SHA-1}(D) 24F522867731552B72533F5D25CC4806
   777D5953'

6. Form key:

   K_{mac} = '24F522867731552B72533F5D25CC4806'

   K = '24F522867731552B'

   K' = '72533F5D25CC4806'

Authentication and Establishment of Session Keys:

IS:

1. Request an 8 byte random challenge from the document's SIC:

   Command APDU:

CLA	INS	P1	P2	Le
'00'	'84'	'00'	'00'	'08'

Document SIC:

2. Generate random challenge and return it to IS:

RND.ICC = '4608F91988702212'

Response APDU:

Response Data Field	SW1	SW2
RND.ICC	'90'	'00'

IS:

3. Generate an 8-byte random challenge and 16-byte random keying material:

RND.IFD = '781723860C06C226'

K.IFD = '0B795240CB7049B01C19B33E32804F0B'

4. Concatenate RND.IFD, RND.ICC and K.IFD:

S = '781723860C06C2264608F91988702212
0B795240CB7049B01C19B33E32804F0B'

5. Encrypt S using TDEA with key K_{enc}:

E_IFD = '861D8A36082E38FB1F699FFDFAF7F903
ADF74AA79E8459E50080F43ACB096B52'

6. Compute "Retail MAC" over E_IFD with key K_{mac}:

M_IFD = '20498D845BE458C3'

7. Construct command data for MUTUAL AUTHENTICATE and send command to the document's SIC:

cmd_data = '861D8A36082E38FB1F699FFDFAF7F903
ADF74AA79E8459E50080F43ACB096B52
20498D845BE458C3'

Command APDU:

CLA	INS	P1	P2	Lc	Command Data Field	Le
'00'	'82'	'00'	'00'	'28'	cmd_data	'28'

Document SIC:

8. Generate 16-byte random keying material:

K.ICC = '0B4F80323EB3191CB04970CB4052790B'

9. Calculate XOR of K.IFD and K.ICC:

K_{seed} = '0036D272F5C350ACAC50C3F572D23600'

10. Derive session keys:

KS_{enc} = '969EC03B1CBFE9DDD11AB1FED206EBE4'

KS_{mac} = 'F0CA1E1EB5ADF208816B88DD579CC1F8'

11. Initialize send sequence counter:

SSC = '887022120C06C226'

12. Concatenate RND.ICC, RND.IFD and K.ICC:

R = '4608F919887022127817238 60C06C226
0B4F80323EB3191CB04970CB4052790B'

13. Encrypt R using TDEA with key K_{enc}:

E_ICC = 'C8F977C50533BE2104E68A844040310A
11362AF11EC09D972CE8AD3FDCB9164B'

14. Compute "Retail MAC" over E_ICC with key K_{mac}:

M_ICC = '9E8E43F7B5CEDB06'

15. Construct response data and send response APDU to the IS:

resp_data = 'C8F977C50533BE2104E68A844040310A
11362AF11EC09D972CE8AD3FDCB9164B
9E8E43F7B5CEDB06'

Response APDU:

Response Data Field	SW1	SW2
resp_data	'90'	'00'

IS:

16. Calculate XOR of K.IFD and K.ICC:

K_{seed} = '0036D272F5C350ACAC50C3F572D23600'

17. Derive session keys:

KS_{enc} = '969EC03B1CBFE9DDD11AB1FED206EBE4'

KS_{mac} = 'F0CA1E1EB5ADF208816B88DD579CC1F8'

18. Initialize send sequence counter:

SSC = '887022120C06C226'

Secure Messaging:

IS

1. SELECT EF.COM (file identifier = ’01 1E’):

Unprotected command APDU:

CLA	INS	P1	P2	Lc	Command Data Field
‘00’	‘A4’	‘02’	‘00’	‘02’	’01 1E’

a) Mask class byte and pad command header:

cmd_header = ‘0CA4020C80000000’

b) Pad data:

p_data= ‘011E800000000000’

c) Encrypt p_data using TDEA with KS_{enc}:

enc_data = ‘6375432908C044F6’

d) Build DO‘87’:

DO87 = ‘8709016375432908C044F6’

e) Concatenate cmd_header and DO87:

M = ‘0CA4020C800000008709016375432908
C044F6’

f) Compute “Retail MAC” of M with KS_{mac}:

— Increment SSC:

SSC = ‘887022120C06C227’

— Concatenate SSC and M:

N = ‘887022120C06C2270CA4020C80000000
8709016375432908C044F6’

— Compute MAC:

CC = ‘BF8B92D635FF24F8’

g) Build DO‘8E’:

DO8E = ‘8E08BF8B92D635FF24F8’

h) Construct command data:

cmd_data = ‘8709016375432908C044F68E08BF8B92
D635FF24F8’

Protected command APDU:

CLA	INS	P1	P2	Lc	Command Data Field	Le
'0C'	'A4'	'02'	'0C'	'15'	cmd_data	'00'

Document SIC:

2. Set EF.COM as the currently selected file and send affirmative response to IS:

Unprotected response APDU:

a) Build DO'99':

DO99 = '99029000'

b) Compute "Retail MAC" of DO99 with KS_{mac}:

— Increment SSC:

SSC = '887022120C06C228'

— Concatenate SSC and DO99:

N = '887022120C06C22899029000'

— Compute MAC:

CC = 'FA855A5D4C50A8ED'

c) Build DO'8E':

DO8E = '8E08FA855A5D4C50A8ED'

d) Construct response data:

resp_data = '990290008E08FA855A5D4C50A8ED'

SW1	SW2
'90'	'00'

Protected response APDU:

Response Data Field	SW1	SW2
resp_data	'90'	'00'

IS:

3. READ BINARY of the first 4 bytes:

Unprotected command APDU:

CLA	INS	P1	P2	Le
'00'	'B0'	'00'	'00'	'04'

a) Mask class byte and pad command header:

cmd_header = '0CB0000080000000'

b) Build DO '97':

DO97 = '970104'

c) Concatenate cmd_header and DO97:

M = '0CB0000080000000970104'

d) Compute "Retail MAC" of M with KS_{mac}:

— Increment SSC:

SSC = '887022120C06C229'

— Concatenate SSC and M:

N = '887022120C06C2290CB0000080000000
970104'

— Compute MAC:

CC = 'ED6705417E96BA55'

e) Build DO'8E':

DO8E = '8E08ED6705417E96BA55'

f) Construct command data:

cmd_data = '9701048E08ED6705417E96BA55'

Protected command APDU:

CLA	INS	P1	P2	Lc	Command Data Field	Le
'0C'	'B0'	'00'	'00'	'0D'	cmd_data	'00'

Document SIC:

4. Return 4 bytes of EF.COM starting at offset 0:

data = '600D5F01'

Unprotected response APDU:

Response Data Field	SW1	SW2
data	'90'	'00'

a) Pad data:

p_data= '600D5F0180000000'

b) Encrypt p_data using TDEA with KS_{enc}:

enc_data = 'F9435D056E27C52E'

c) Build DO'87':

DO87 = '870901F9435D056E27C52E'

d) Build DO'99':

DO99 = '99029000'

e) Concatenate DO‘87’ and DO‘99’:

M = ‘870901F9435D056E27C52E99029000’

f) Compute “Retail MAC” of M with KS_{mac}:

— Increment SSC:

SSC = ‘887022120C06C22A’

— Concatenate SSC and M:

N = ‘887022120C06C22A870901F9435D056E
27C52E99029000’

— Compute MAC:

CC = ‘0C15238078E0A4C9’

g) Build DO‘8E’:

DO8E = ‘8E080C15238078E0A4C9’

h) Construct response data:

resp_data = ‘870901F9435D056E27C52E990290008E
080C15238078E0A4C9’

Protected response APDU:

Response Data Field	SW1	SW2
resp_data	‘90’	‘00’

IS:

5. READ BINARY of the remaining 11 bytes:

Unprotected command APDU:

CLA	INS	P1	P2	Le
‘00’	‘B0’	‘00’	‘04’	‘0B’

a) Mask class byte and pad command header:

cmd_header = ‘0CB0000480000000’

b) Build DO ‘97’:

DO97 = ‘97010B’

c) Concatenate cmd_header and DO97:

M = ‘0CB000048000000097010B’

d) Compute “Retail MAC” of M with KS_{mac}:

— Increment SSC:

SSC = ‘887022120C06C22B’