Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
Premier ministre
ANSSI

Bundesamt
für Sicherheit in der
Informationstechnik

Technical report

Signature creation and administration for eIDAS token

Part 1: Functional Specification

Version 1.0

Date: 2015/07/21

Foreword

This technical report specifies an autonomous signature creation application embedded on an eIDAS token. Unless explicitly mentioned, this specification is compliant with all referenced standards.

This document has been written by ACSIEL (www.acsiel.fr) in close relation with ANSSI (www.ssi.gouv.fr) and ANTS (www.ants.interieur.gouv.fr).

TABLE OF CONTENTS

TABLE OF FIGURES

Glossary

ADF	Application dedicated file
AES	Advanced encryption standard
AID	Application identifier
APDU	Application protocol data unit
AT	Authentication Terminal
CAN	Card access number
CHAT	Certificate holder authorization template
DF	Dedicated file
DTBS	Data to be signed
EC	Elliptic curve
ECDSA	Elliptic curve digital signature algorithm
EFID	Elementary file identifier
EIDAS	Electronic Identification, Authentication and trust Services
GAP	General authentication procedure
IFD	Interface Device
LCS	Life Cycle State
MF	Master file
PACE	Password authenticated connection establishment
PIN	Personal identification number
PUK	PIN unlocking key
QES	Qualified electronic signature
RSA	Rivest Shamir Adleman
SCA	Signature creation application
SFI	Short file identifier
SHA	Secure hashing algorithm
SMT	Signature management terminal

SSCA	Secure signature creation application
SSCD	Secure signature creation device
ST	Signature Terminal
UD	User device

1 Definitions

This chapter contains definition of concepts discussed all along this document.

1.1 User

Natural person holding the eIDAS token.

1.2 eIDAS Token

The eIDAS token is a device based on a secure element that may have various form factors (smartcard, µSD,...). It MAY contain several applications (electronic ID, travel document,…), and in the scope of this document contains an application enabling the creation of electronic signature according to [EC_Regulation].

1.3 Electronic Signature Key

Private portion of an asymmetric key, used to create electronic signature in the sense of the [EC_Regulation]. This key is identified in the signature application with the [ISO/IEC 7816-15] structure.

The key usage shall be compliant with the certificate policy of the issuer.

1.4 General Authentication procedure (GAP)

General authentication procedure as defined in [TR03110-2].

1.5 User Credential

User credentials are defined in [TR03110-2]. Unless otherwise specified, the user credentials are the PIN, PUK, physical user credential and CAN. For more information about physical user credential, refer to [TR_PhysicalAuthentication]

1.6 Global Personal Identification Number (PIN)

The global PIN is a user credential global to the whole eIDAS token. Depending on the configuration, it may be absent. When present, it is shared by the [eSign application] and the other applications.

1.7 Local User Credential

A local user credential is specific to the [eSign application]. It is not shared with other applications on the eIDAS token. Depending on the configuration, it may be absent.

1.8 Password Unblocking Key (PUK)

The PUK may be used to unblock a user credential: signature user credential, or the global PIN.

1.9 Qualified Electronic Signature Key

Private portion of an asymmetric key, used to create qualified electronic signature in the sense of the [EC_Regulation]. This key is identified in the signature application with the [ISO/IEC 7816-15] structure.

A qualified electronic signature key is associated with a qualified electronic certificate issued by a qualified authority.

The key usage shall be compliant with the certificate policy of the issuer.

1.10 Secure Signature Creation Application (SSCA)

The secure signature creation application ([eSign application]) is the signature application contained in the eIDAS token enabling the creation of electronic signature. More details about the secure signature creation application can be found in §3.4.

1.11 Signature Creation Application (SCA)

The signature creation application ([SCA]) is the application requesting the creation of a digital signature from the eIDAS token. An external application (text editor, mail software,…) – executed on a service provider, local computer,…- authorized to request a signature creation and for which the user has expressed its consent.

1.12 Signature Management Terminal (SMT)

See [TR03110-2], §2.1.2.2

1.13 Signature Terminal (ST)

See [TR03110-2], §2.1.3

1.14 User device (UD)

The user device is the device the user directly interacts with. It is always local, hence physically accessible by the user. This device is used to enter a user credential or PUK (user credential) and provides an IFD (interface device) to communicate with the eIDAS token. It may be a computer, a mobile phone or a pad.

2 Introduction

The [eSign application] is designed to be embedded in an eIDAS token (smartcard, SIM, µSD,…). Using the [eSign application], the following services are provided:

a. The signature creation can be performed either via contact or contactless interface;

b. The signature creation is under control of the holder independently of the communication interface;

c. The received data to be signed only comes from the intended signing application;

d. The user consent performed with user authentication shall be done in a secure way, to ensure protection of the authentication data. User authentication via PIN code ensures authentication of the given agreement;

This Technical Report focuses on the recommended configuration for the eIDAS interoperable signature application. All configurations are based on protocols described in [TR03110-2]. Only one of the configurations must be implemented. The basic configuration is primarily meant for compatibility with existing systems and described in chapter 4. The advantage of the extended configuration, described in chapter 3, is that it provides increased flexibility and additional features:

a. User consent via Biometric data (physical user credential);

b. All operations are performed by a unique terminal. The Signature Management Terminal is an extension of an authentication terminal, resulting in a simplified PKI with the deployment of a single certificate chain for both signature management and eID access.

c. Easy deployment with cost-effective readers.

d. Multiple PIN code management for user consent;

e. The [eSign application] does not require the presence of any other applications (e.g. eID application).

f. Batch mode for multiple signature creation

3 Extended Signature Application

This chapter specifies extended configuration of the eSign application (to be used with Signature Management Terminals).

3.1 Environment of the eIDAS token

The eIDAS token SHALL

a. Manage the authorization of an external entity;

b. Manage on card key generation and keep the private key secret;

c. Ensure user consent through the signature user credential;

d. Compute the digital signature on card;

To compute an electronic signature, three entities are involved:

a. **Signature Creation Application [SCA]**: application (that can be local or remote) requesting a signature, which use [UD] to send/receive APDU;

b. **User Device [UD]**: manages the local transmission and reception of APDUs with the eIDAS token;

c. **The eIDAS token**, holding the embedded electronic signature application named the [eSign application]

This specification focuses on end to end communications between the [eSign application] and:

a. The [SCA] through a dedicated protected channel [CH_SCA];

b. The [UD] through a dedicated protected channel [CH_UD];

The communication channel between the [SCA] and [UD] is out of the scope of this specification.

The [UD] includes an [IFD] in charge of the communication with the eIDAS token. The [UD] MAY operate with a keypad and/or display, with a local or remote [SCA].

The [SCA] and [UD] are complementary sub-parts of an eIDAS compliant terminal. Thus, this terminal is able to authenticate with the eIDAS token and the underlying [eSign application]. The [SMT] SHALL use the General Authentication Procedure. The [SMT] MAY use other authentication procedures.

The following figure summarizes the environment of the eIDAS token:

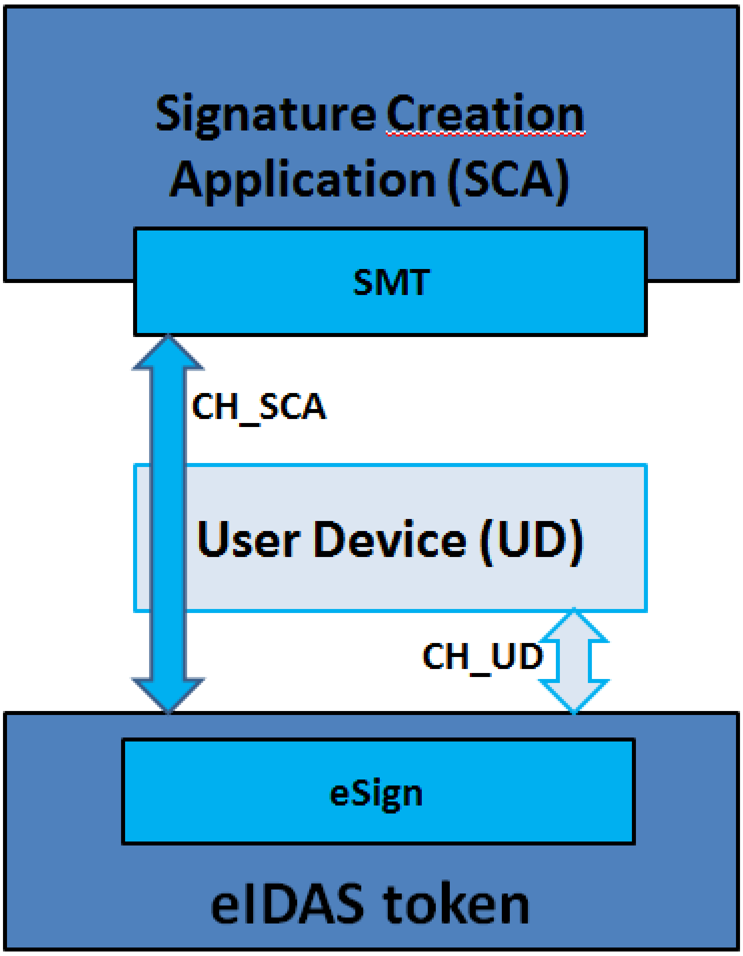

Figure 1 - eIDAS token environment for signature

Note:

110 The channel between the User Device and SMT is out of scope.

3.1.1 Signature management terminal

Depending on the rights the [SMT] has been granted, it can request the [eSign application] to:

a. create electronic signature;

b. manage its elements (user credentials, keys and files);

115 c. manage the [eSign application] itself;

As such, the [SMT] MAY be used by the following entities:

a. [SCA] when the [SMT] is entitled to create electronic signature;

b. Certificate provider for the signature service when the [SMT] is entitled to generate signature keys and upload certificates;

120 c. Identity provider of the user when the [SMT] is entitled to update the content of the file EF.INFO4CERT (if present);

d. Administrator of the [SCA] when the [SMT] is entitled to [eSign application] management;

The [SMT] is an authentication terminal as defined in [TR03110-2]. The certificate SHALL contain the signature authorization extension of the [SMT] and MAY contain the signature attribute extension.

For more information about certificate extensions management, refer to [TR03110-3] §C.3.

The effective authorization of the [SMT] is computed as described in [TR03110-3] §2.7.

The effective maximum number of consecutive signature the [eSign application], also called [NSIGN], is allowed to perform is computed by taking the smallest value amongst all the values received in the certificate chain "Maximum number of consecutive signatures". When the attribute is missing in one certificate, the effective value is '01'.

The [SMT] MAY request up to [NSIGN] consecutive signatures regardless of the signature keys used. The [eSign application] SHALL not exceed the effective maximum number of consecutive signature obtained during the GAP. Once the number of consecutive signature has been met, the [eSign application] SHALL require new user consent.

3.1.1.1 Signature authorization extension

The following Object Identifier SHALL be used for this extension:

Id-SMT OBJECT IDENTIFIER :: = {1.2.250.1.223.1001.1.1}

This field SHALL be present in the certificate extensions. The authorization of [SMT] is encoded as described below - the authorization bit mask SHALL be evolving as described [TR03110-3] §C.3 - :

8 7 6 5 4 3 2 1	8 7 6 5 4 3 2 1	Description
x x x x x x x x	x x x x x x x x	Access Rights
1- - - - - - -	- - - - - - - - -	Generate Electronic signature • PSO CDS on a key • Read EF.CD, EF.cert associated
- 1 - - - - - -	- - - - - - - - -	Generate Qualified Electronic signature • PSO CDS on a key • Read EF.CD, EF.cert associated
- -1 - - - - -	- - - - - - - - -	Install electronic Certificates • Read EF.INFO4CERT • Generate keys • UpdateEF.CD[x] (dedicated) • [optional] write EF.cert associated
- - -1 - - - -	- - - - - - - - -	Install Qualified electronic Certificates • Read EF.INFO4CERT • Generate keys

		• UpdateEF.CD[y] (qualified one) • [optional] write EF.cert associated
- - - - 1 - - -	- - - - - - - - -	SSCA Management • Life Cycle
- - - - - 1 - -	- - - - - - - - -	Signature Keys management • Life Cycle • Update [ISO/IEC 7816-15] files
- - - - - - - 1 -	- - - - - - - - -	File EF.INFO4CERT Management • Update
- - - - - - - - 1	- - - - - - - - -	User credential management • Unblock user credential
- - - - - - - - -	1 - - - - - - -	User credential management • Update user credential value
- - - - - - - - -	- 1 - - - - - -	User credential management • Life Cycle • Update [ISO/IEC 7816-15] file
- - - - - - - - -	- - 1 - - - - -	User credential management • Initialization

3.1.1.2 Signature attribute extension

The following Object Identifier SHALL be used for this extension:

Id-SMT OBJECT IDENTIFIER :: = {1.2.250.1.223.1001.1.4}

This extension MAY contain the maximum number of consecutive signature.

3.2 Communication, protocol and security mechanisms

In contact and contactless, the [eSign application] SHALL mandate secure messaging ensuring integrity, authenticity and confidentiality for the communication with the [UD], once PACE authentication is established.

3.3 Communication protocols

The eIDAS token SHALL support the communication protocols defined in [TR03110-2].

3.3.1 Security protocols

The [eSign application] mandates global authentication services to be provided in the master file (MF) of the eIDAS token. These global authentication services are:

a. PACE as defined in [TR03110-2]. In the scope of this technical report, PACE SHALL be performed under the MF, and before selecting the [eSign application];

b. GAP as defined in [TR03110-2]; This protocol allows to update internal date and to validate the effective authorization of the remote terminal through the CHAT on the [eSign application];

3.3.2 Secure messaging

After a successful authentication using PACE or GAP (as defined in [TR03110-2]), the secure messaging SHALL comply with the one specified in [TR03110-3].

The [eSign application] supports switching of session context. This is mandatory to perform user consent operations through the [UD] as it requires local action from the user. The session context of the eSign application consists of the secure messaging context (keys and SSC), the current DF and EF, as defined in [TR03110-2]. The current DF is the eSign application ADF.

This is the typical execution flow to perform local operation when CH_SCA (GAP Secure Messaging) is active:

a. MSE SET AT (template for PACE session context identifier): switches session context to CH_UD, MF becomes current DF

b. SELECT eSign application (under CH_UD)

c. Perform user credential operation (VERIFY, CHANGE REFERENCE DATA, RESET RETRY COUNTER)

d. MSE SET AT (template for GAP session context identifier): switches session context to CH_SCA, MF becomes current DF (this context was saved during GAP flow)

e. SELECT eSign Application (under CH_SCA)

f. Resume operations (under CH_SCA)

Notes

A user credential’s validated state is not part of the session context

3.3.3 User consent

Depending of the configuration, the global PIN and/or local user credential are considered as the user consent.

3.3.3.1 User consent to access to the eIDAS token

The holder must express his agreement for the physical use of the eIDAS token. This operation is performed using PACE with a global credential.

PACE SHALL be performed in the MF as described in [TR03110-1] and [TR03110-2] with the global user credentials available in the eIDAS token.

The different PACE configurations supported are indicated in the file EF.CardAccess (as defined in [TR03110-3]). This file SHALL be present.

3.3.3.2 User consent to access to the [eSign application]

In order to use the [eSign application], the user needs to express his agreement. This operation is performed by using a user credential. Depending on the configuration of the [eSign application], it SHALL be one of the following:

a. a local user credential(s) stored in the [eSign application];

b. the global PIN stored in the MF (global PIN of the eIDAS token);

The verification of this user credential SHALL be performed using:

a. VERIFY command for validation of local user credential(s) or the global PIN. It allows submitting the user credential. It MAY be performed in the [eSign application], to submit the local user credential, or the global PIN. This command SHALL be executed under secure messaging;

b. PACE for validation of the global PIN;

> ***Notes***
>
> When the global PIN is also the signature PIN, the user consent for the physical access to eIDAS token and logical access to [eSign application] is merged. As a result, a successful PACE performed with the global PIN grant access to both the eIDAS token and the [eSign application].

3.3.3.3 User consent and operations

The table below indicates for each use case the user credential that is mandated to get access to the said service.

Operation	Type of data required	Mode of operation
• Generate electronic signature • Generate qualified electronic signature	It SHALL be the user credential protecting the signature creation function of the key. Depending on the configuration of the eIDAS token, one of the following user credential SHALL be used: -the global PIN(1) -the local user credential (2) Note: the user credential (the Global PIN & local user credential) to use is the one protecting the access to such operations (several user credentials MAY be present in	**Case 1** -PACE + VERIFY Global PIN -PACE with Global PIN In case the user consent is lost (see §3.6.2.1 and §1.1.1.1), additional VERIFY Global PIN may be performed. **Case 2** -PACE + VERIFY local user credential

	the eIDAS token).	
- Install electronic certificate - Install qualified electronic certificate	It SHALL be the user credential protecting the signature key generation. Depending on the configuration of the eIDAS token, one of the following user credential SHALL be used: -the global PIN(1) -the local user credential (2) Note: the user credential (the Global PIN & local user credential) to use is the one protecting the access to such operations (several user credentials MAY be present in the eIDAS token).	**Case 1** -PACE + VERIFY Global PIN -PACE with Global PIN In case the user consent is lost (see §3.6.2.1 and §1.1.1.1), additional VERIFY Global PIN may be performed. **Case 2** -PACE + VERIFY local user credential
- [eSign application] management - User credential management (Change)	CAN, PUK or the Global PIN Note: the user credential (the Global PIN) to use is the one protecting the access to such operations (several user credentials MAY be present in the eIDAS token).	PACE
- Signature keys management - EF.INFO4CERT management - User credential management (Initialization)	Global PIN Note: the user credential (the Global PIN) to use is the one protecting the access to such operations (several user credentials MAY be present in the eIDAS token).	-PACE with Global PIN -PACE + VERIFY Global PIN
- User credential management (Unblock)	PUK Note: the user credential (PUK) to use is the one protecting the access to such operations (several user credentials MAY be present in the eIDAS token).	PACE with PUK
- User credential management (Life cycle)	Global PIN or PUK Note: the user credential (the Global PIN & PUK) to use is the one protecting the access to	-PACE with Global PIN -PACE + VERIFY Global PIN

	such operations (several user credentials MAY be present in the eIDAS token).	-PACE with PUK

Figure 2 - User consent and operations

3.4 Secure signature creation application [eSign application]

The [eSign application] is an Application Dedicated File (ADF) as defined in [ISO/IEC 7816-4], located under the master file (MF) of the eIDAS token.

3.4.1 Prerequisite

The [eSign application] does not require any other application to be present on the eIDAS token. It can be the sole application present on the eIDAS token.

The eIDAS token MAY have an MRZ, but the [eSign application] SHALL NOT be accessible using PACE with MRZ.

3.4.2 Selection of the application

The selection of the [eSign application] SHALL be made under secure messaging. The AID of [eSign application] used for selection SHALL be the following, as defined in [EN419212-1]:

A0 00 00 01 67 45 53 49 47 4E

Notes

The [eSign application] selection MAY be restricted to Authentication terminals. For privacy reasons, the eIDAS token MAY return “file not found” for unauthorized selection. Returning 'security status not satisfied' is also possible.

3.4.3 Life cycle of the [eSign application]

[eSign application] has a life cycle compliant with [ISO/IEC 7816-9] and supports the following states:

a. Operational state – activated : in this state the [eSign application] is usable. It contains all the data objects and files as defined in §3.5;

b. Operational state – deactivated: in this state the [eSign application] is created and selectable. In response to the selection, it SHALL indicate that its state is deactivated. The [eSign application] services allowed are switching to Operational state – activated or Termination state.

c. Termination state: in this state, the [eSign application] is irreversibly unusable;

The [eSign application] may be in state “Operational state-activated” or “Operational state – deactivated” at issuance.

3.4.4 Initialization after issuance

The [eSign application] SHALL manage all elements required for signature creation, including the global PIN, PUK, and local user credential. [eSign application] SHALL be able to handle empty objects and initialized ones.

The signature creation service of the [eSign application] is not available if:

a. the private key(s) for electronic signature or the corresponding user credential has not been initialized : the content of the object is empty and SHALL be assigned first;

b. the private key(s) for electronic signature, or the corresponding user credential is not in state “operational – activated”;

c. the user credential is blocked (the number of tries has reached the maximum number authorized);

3.5 Data elements of the [eSign application]

The [eSign application] manages the following types of data:

a. PACE credentials

These data are credentials that can be used in a PACE

i. The Global PIN;

ii. The CAN;

iii. The PUK;

b. User credentials

These data are credentials used to authenticate the user and unlock access to the [eSign application]. Depending on the configuration of the [eSign application], they MAY unlock the access to the application (global PIN), some function of it, or unlock the usage of the signature creation function;

i. [eSign application] PIN (either a local PIN or the global PIN);

ii. [eSign application] physical user credential;

They may be used either through a VERIFY command, or with a PACE protocol. These credentials are described in §3.5.1.

c. Signature keys

They are cryptographic keys that may be of RSA or EC type depending on the configuration of the [eSign application]. They may be used to create qualified electronic signature or electronic signature in the sense of [EC_Regulation]. The usage of the signature key is indicated in the [ISO/IEC 7816-15] structure;

i. private key(s) for electronic signature;

ii. private key(s) for qualified electronic signature;

These credentials are described in §3.5.3.

d. Transparent files as defined in [ISO/IEC 7816-4]. These files are used to store the [ISO/IEC 7816-15] structure as well as other information. They are described in §3.5.4.

All the elements listed above SHALL be identified in the [ISO/IEC 7816-15] structure as described in §3.8.

3.5.1 User credentials

User credentials are user credential data used by eIDAS token to:

a. Authenticate the user;

b. Express the consent of user before creating an electronic signature;

When delivered, the data container dedicated to store the user credential in the [eSign application] SHALL be in one of the following state:

c. Operational state. The user credential is already loaded.

d. Initialization state. The user SHALL initialize the user credential prior any usage.

Notes

The current technical report does not limit the number of user credential contained in the eIDAS token. Depending on the configuration of the eIDAS token, one or several user credential may be present (local and/or global), and each of them may protect the access to one or several operations.

3.5.1.1 Available operations

The following operations may be performed on a user credential:

a. **Verification**: this operation submits a candidate user credential to the eIDAS token that compares it against the reference user credential. Upon success the following actions are performed

   - the user credential verification status is set;
   - the corresponding access rights are granted;
   - the retry counter is restored to its initial value;

   Upon failure, the following actions are performed:

   - the user credential verification status is reset;
   - the corresponding access rights are denied;
   - the retry counter is decremented by one;

b. **Change**: this operation changes the reference user credential values stored in the eIDAS token. If successful, the user credential verification status is reset”

c. **Devalidation**: this operation resets the user credential verification status.

d. **Unblocking** : this operation consists in unblocking the user credential, namely restoring its retry counter to the initial value (described below), resetting its verification status, and changing its reference value.

Notes

In order to identify the user credential on which the operation SHALL be performed, the identifier of the user credentials SHALL be provided in the field “reference data” of the command. All user credentials share the same range of identifiers. For example, if a PIN credential has the identifier #1, no physical user credential SHOULD have the identifier #1. Identifiers SHALL be in range 1 to 31 included.

3.5.1.2 Life cycle state

User credentials have a life cycle compliant with [ISO/IEC 7816-9] and support the four following states:

a. **Initialization state**: in this state the data container is created, but the data (user credential) has not been initialized yet. The user credential usage is restricted. The global PIN MAY be used for PACE. Other restrictions are described in §3.6.3.3.

b. **Operational state – activated** : in this state the data container is created, filled with a user credential, and usable;

c. **Operational state – deactivated**: in this state the data container is created, filled with a user credential, and its usage is restricted;

d. **Termination state**: in this state, the data container is irreversibly unusable;

eIDAS token allows transition between these states for each of the object it contains. The transitions and the command used to perform these transitions are compliant with [ISO/IEC 7816-9].

3.5.1.3 Attributes

User credentials SHALL contain the two following attributes:

a. **Retry counter**: counter indicating the number of remaining tries for the verification of the user credential. This counter is persistent, meaning it is not reset upon reset or eIDAS token selection. It is decremented on a wrong verification, and restored to its initial value (described below) upon successful verification of the user credential. When this counter has reached ’zero’, the user credential becomes unusable. Prior any other use, it SHALL be unblocked (see §3.6.3.5).

b. **Initial value of retry counter**: value indicating the maximum number of incorrect verification allowed by the user credential. The retry counter is reset to the initial value in case of successful verification/unblock/change.. The initial value may take any value between ’1’ (decimal) and ‘15’ (decimal) and its value is indicated in the [ISO/IEC 7816-15] structure. No modification SHALL be possible once this value is set.

3.5.2 PIN unblocking keys (PUK)

PIN unblocking key (PUK) aims at unblocking user credentials once their retry counter has reached zero. A PUK may be either a PIN or physical user credential and SHALL be located in the MF. The PUK may be a blocking or unblocking PIN or physical user credential.

> ***Notes***
>
> The current technical report does not limit the number of PUK contained in the eIDAS token. Depending on the configuration of the eIDAS token, one or several PUK may be present, and each of them may unlock a given user credential.

3.5.2.1 Available operations

The following operation may be performed on PUK:

a. **Verification**: this operation submits a candidate PUK to the eIDAS token that compares it against the reference PUK.

   Upon success the following actions are performed

   - the PUK verification status is set;
   - the corresponding access rights are granted;
   - if exists, the retry counter is restored to its initial value;

   Upon failure, the following actions are performed:

   - the PUK verification status is reset;
   - the corresponding access rights are denied;
   - if exists, the retry counter is decremented by one;

> ***Notes***
>
> In order to identify the PUK on which the operation SHALL be performed, the identifier of the PUK SHALL be provided in the field “reference data” of the command.

3.5.2.2 Life cycle state

PUK has a life cycle compliant with [ISO/IEC 7816-9] and supports the following state:

a. **Operational state – activated** : in this state the data container is created, filled with a value, and usable;

b. **Termination state**: in this state, the data container is irreversibly unusable;

eIDAS token allows transition between these states for each of the object it contains. The transitions and the command used to perform these transitions are compliant with [ISO/IEC 7816-9].

3.5.2.3 Attributes

PUK MAY contain the two following attributes:

a. **Retry counter**: counter indicating the number of remaining tries for the verification of the PUK. This counter is persistent, meaning it is not reset upon reset or eIDAS token selection. It is decremented on a wrong verification, and restored to its initial value (described below) upon successful verification of the PUK. When this counter has reached ’zero’, the PUK becomes unusable.

b. **Initial value of retry counter**: value indicating the maximum number of incorrect verification allowed by the PUK. The retry counter is reset to the initial value in case of successful verification. The initial value may take any value between '1' (decimal) and '15' (decimal) and its value is indicated in the [ISO/IEC 7816-15] structure. No modification SHALL be possible once this value is set.

a. .

3.5.3 Signature keys

Signature keys are cryptographic keys that may be of RSA or EC type depending on the configuration of the [eSign application].

Notes

The current technical report does not limit the number of signature keys contained in the [eSign application]. Depending on the configuration, one or several signature key may be present, each of them having its own usage: qualified electronic signature or electronic signature.

The [ISO/IEC 7816-15] structure MAY declare the signature keys present in the eIDAS token in the following ways:

- the certificate(s) associated to the signature keys should be stored in EF.CD[x] files;
- the attribute "key usage" of a signature key used for non qualified signature should be set to "sign";
- the attribute "key usage" of a signature key used for qualified signature should be set to "non repudiation";

3.5.3.1 Available operations

In the scope of the current technical report, signature key import is not considered. The signature keys MAY be generated on board, several times over the life time of the [eSign application].

The signature keys can be used to create qualified electronic signature or electronic signature in the sense of [EC_Regulation]. The usage of the signature key is indicated in the [ISO/IEC 7816-15] structure. For security reasons, a signature key SHALL have a unique usage, i.e. it SHALL not be usable to create a qualified electronic signature and an electronic signature.

When delivered, the object(s) stored in the [eSign application] SHALL be in one of the following state (see also §3.5.3.2):

a. Operational state. The key value has been set.

b. Initialization state. The user SHALL set the key prior any usage through a key generation.

Notes

In order to identify the signature key on which the operation SHALL be performed, the identifier of the signature key SHALL be provided. Signature key share the same range of identifier whether they are RSA or EC keys. For example, if a RSA signature key has the identifier #1, no EC signature key should have the identifier #1. Identifiers SHALL be in range from 1 to 31 included.

3.5.3.2 Life cycle state

Signature keys have a life cycle compliant with [ISO/IEC 7816-9] and support the four following states:

a. **Initialization state**: in this state the data container is created, but the data (signature key) has not been initialized yet. The signature key is not usable;

b. **Operational state – activated** : in this state the data container is created, filled with a signature key, and usable;

c. **Operational state – deactivated**: in this state the data container is created, filled with a signature key, and its usage is restricted;

d. **Termination state**: in this state, the data container is irreversibly unusable;

The [eSign application] manages transitions between these states for the objects it contains. The transitions and the command used to perform these transitions are compliant with [ISO/IEC 7816-9].

3.5.4 Files

The [eSign application] supports elementary files with transparent structure as defined in [ISO/IEC 7816-4].

Notes

The current technical report only considers the [eSign application] which is an ADF as defined in [ISO/IEC 7816-4], and elementary files stored under this ADF. The support of other types of files is out of the scope of the current technical report.

3.5.4.1 File attributes

The transparent files may be selected either by EFID, or SFI. While the support of EFID is mandatory, the support of SFI is optional.

a. The EFID SHALL be indicated in the [ISO/IEC 7816-15] structure for each file.

b. The SFI MAY be indicated in the [ISO/IEC 7816-15] structure for file.

3.5.4.2 Available operations

The [eSign application] supports the following operations on files:

a. Selection : the current technical report only envisions file selection by EFID or SFI as defined in [ISO/IEC 7816-4];

b. Reading : the file reading is compliant with [ISO/IEC 7816-4];

c. Updating : the file updating is compliant with [ISO/IEC 7816-4];

Notes

When importing a certificate (at signature key initialization or regeneration), the file dedicated to store the certificate in the [ISO/IEC 7816-15] structure, SHALL be large enough to allow storage of the data. The management of file resizing, in case the size of the certificate exceeds the size of the file is out of the scope of this technical report. Therefore, it is recommended to create the file dedicated to store the signature key certificate large enough.

3.5.4.3 Life cycle state

Files only have one life cycle compliant with [ISO/IEC 7816-9]:

a. Operational state – activated;

3.5.5 File Structure

When delivered, the [eSign application] SHALL contain a minimum set of files.

The files are also used to store the [ISO/IEC 7816-15] structure that allows the [UD] to discover the content of the [eSign application] and its features.

Four types of information can be present in the eIDAS token:

- Configuration information ;
- Public information ;
- Private information ;
- Other information ;

Configuration information

The following files are used to store the configuration information:

a. The file EF.DIR SHALL be present;

b. The files EF.CardAccess SHALL be present. For more details refer to [TR03110-3].

c. The files EF.CardSecurity and EF.ChipSecurity MAY be present. For more details refer to [TR03110-3].

d. The file EF.ATR/INFO (defined in [ISO/IEC 7816-4]) MAY be present.

This information is public information that is used by the [UD] to discover the capabilities of the eIDAS token. As such, it does not hinder the privacy of the holder. Any other data that could hinder the user privacy SHALL not be stored within these files.

Public information

The following files are used to store the public information:

a. The file EF.CIAInfo SHALL be present;

b. The file EF.OD SHALL be present;

c. The file EF.AOD for user credentials and PUK SHALL be present;

d. The file EF.PrKD for electronic signature keys SHALL be present;

e. The file EF.CD[x] for signature certificate SHALL be present;

f. The file EF.DCOD for the description of other files and objects MAY be present. This file MAY reference EF.INFO4CERT flagged using the string “EF.INFO4CERT”, or contain an URL.

This information is public information that is used to discover the content of the eIDAS token. As such, it may hinder the privacy of the holder. In order to protect the privacy of the holder, it SHALL be readable after access to [eSign application] is granted and MAY require further access conditions (user consent or dedicated role).

Notes

The eIDAS token SHALL hold one EF.CD[x] file per certificate.

The file EF.CD[x] MAY contain the signature certificate or a reference to it (e.g. URL).

Private information

The following files are used to store the private information:

a. The file EF.INFO4CERT, which is also a transparent file, MAY be present. It is intended to store all the data related to the user that are needed to generate signature key certificate (name, surname,…). This file can be omitted if this information can be found somewhere else, or if no signature key generation (certificate generation step) is required during the eIDAS token life.

b. The certificates files MAY be present

This information is private information that contains information about the eIDAS token and/or its holder. In order to protect the privacy of the holder, it SHALL be readable after access to [eSign application] is granted.

Other information

The [eSign application] MAY also contain other files. In such case, a great attention SHALL be paid to the type of information stored in these files, and the access conditions applied to them, in order not to endanger the privacy of the user.

For more information about the [ISO/IEC 7816-15] structure, refer to 3.8.

Notes

Warning: the files protected by the [eSign application] user credentials SHOULD be read first before signature creation

3.5.6 PIN format

PIN SHALL be used for the user credentials and the PUK. This chapter provides information about their format and structure. The [ISO/IEC 7816-15] structure MAY provide information about PIN in the EF.AOD file.

Notes

The current technical report does not impose any specific format or encoding for the PIN.

The PIN length, the PIN policy, the PIN format, when using PIN is out of the scope of the current technical report. It is up to the issuer to set up its own recommendations.

3.5.7 Physical user credential format

Refer to [TR_Physical_Autentication] §1.3.1

3.6 Available [eSign application] services

The terminal should read the EF.DIR from the [ISO/IEC 7816-15] structure to discover which security mechanism is applied on the [eSign application]. Execute the GAP process to open the access condition.

3.6.1 Life cycle management of the [eSign application]

The [eSign application] enables to manage its life cycle, namely by locking/unlocking (suspension of its usage) and termination. The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Manage the life cycle of the [eSign application](through commands ACTIVATE, DEACTIVATE and TERMINATE);

PACE SHALL be performed according to [TR03110-2] and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege "[eSign application] management" as defined in §3.1.1.1.

In the state "Operational – Deactivated", the [eSign application] SHALL NOT execute any of the operational command except "ACTIVATE" or "TERMINATE".

The state "Operational – Deactivated" is reversible.

The state "Terminated" is irreversible as described in [ISO/IEC 7816-8].

Notes

Information about the state of the [eSign application] is indicated by the status word returned by the SELECT command, as defined in [ISO/IEC 7816-4].

3.6.2 Signature creation

A digital signature operation can only be executed provided the following conditions are met:

a. The user has given his consent;

b. There is an active secure messaging;

User consent may be expressed to the [eSign application] as described in §3.3.3.3.

The secure messaging results from the protocols described in §3.3.1

Once the user has given the consent for creating a signature, the [eSign application] can process the data and return the computed signature.

The signature is created using the following APDU commands:

a. MSE SET – DST to select the signature key, and implicitly the hashing and signature algorithm;

b. PSO - HASH (conditional) to compute the message digest;

c. PSO - COMPUTE DIGITAL SIGNATURE to compute digital signature;

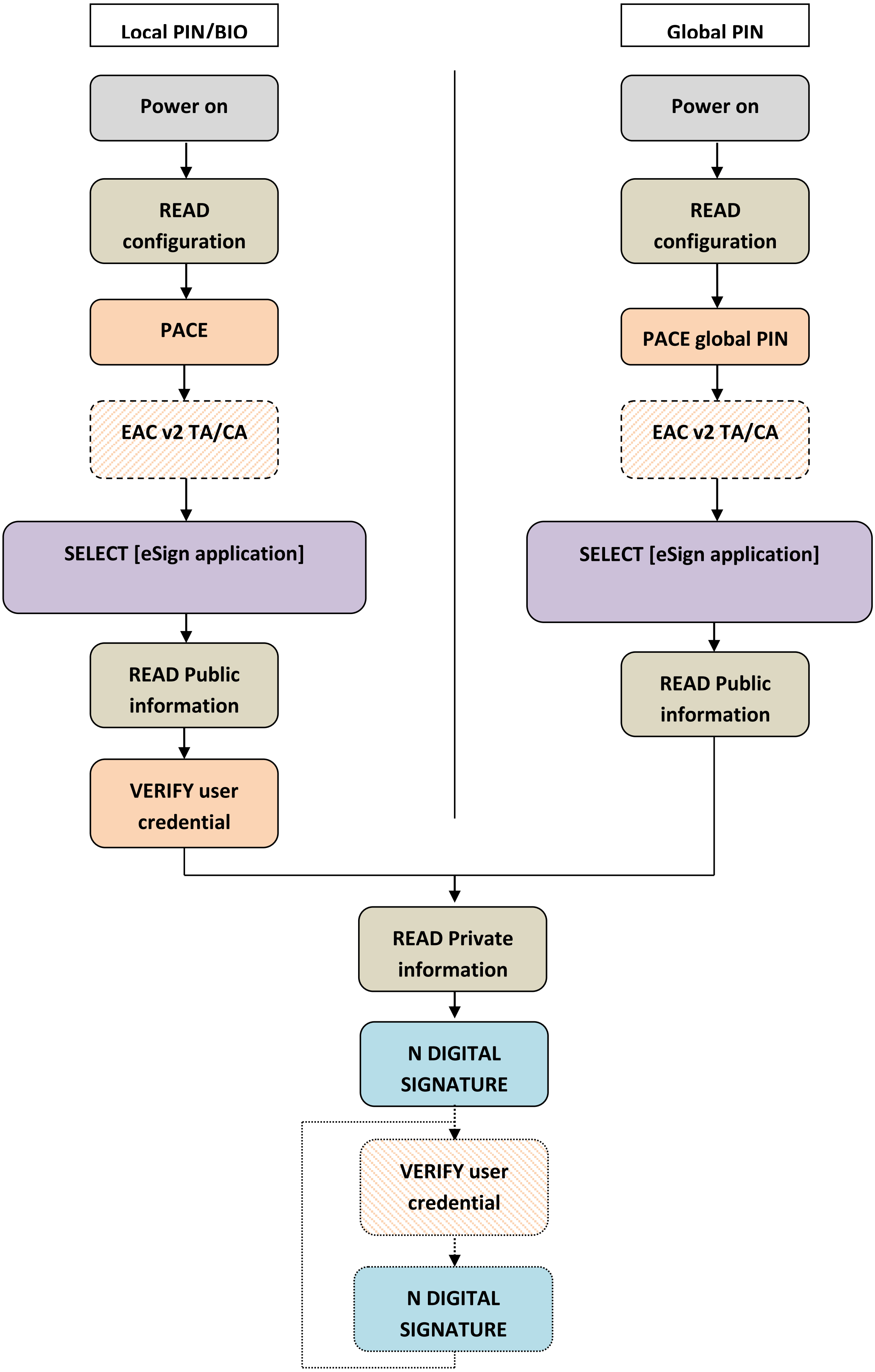

Figure 3 - Signature Creation sequence

3.6.2.1 Discovery mechanism

When accessed, the eIDAS token provides an [ISO/IEC 7816-15] based card discovery mechanism.

The access conditions shall be fulfilled before accessing any [ISO/IEC 7816-15] data containing privacy related data

As the configurations of [eSign application] requiring or not authentication of the [SMT] are exclusive, the terminal SHALL discover the configuration of the [eSign application] in order to request an electronic signature creation.

The terminal SHALL apply the following discovery mechanism:

a. Read the EF.DIR from the [ISO/IEC 7816-15] structure. EF.DIR indicates which global authentication protocols (PACE or GAP) have to be performed prior to accessing applications.

b. Perform PACE or GAP protocol to be performed using the global PIN or the CAN (if present). After successful completion of the protocol, the terminal SHALL select [eSign application].

c. After successful selection of the [eSign application], the [SMT]/terminal SHALL explore the [ISO/IEC 7816-15] structure, and:

    i. Discover the list and location of signature key available in the [eSign application], as well as their life cycle and usage

    ii. Choose one signature key among the list that is in operational-activated state, matching the required usage;

    iii. Identify the user credential required to create electronic signature with the signature key it has chosen, and check its life cycle;

At this stage, three cases (case 1a, case 1b and case 2) SHALL be sorted out, depending on whether the global PIN is the signature PIN or not.

3.6.2.2 Signature creation with authentication of the [SMT]

The sequence of operation described in §3.6.2.1 SHALL be applied. In this case, the eIDAS token SHALL require a successful GAP before signature creation.

Depending on the configuration of the [eSign application], the signature creation is controlled by a user credential that is either the Global PIN, or the local user credential.

In any case, PACE SHALL be performed according to [TR03110-2] and SHALL contain the confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege “Generate qualified electronic signature” or “Generate electronic signature” as defined in [TR03110-4].

The terminal SHALL have the privilege matching the usage of the electronic signature key it intends to select to create an electronic signature:

a. The [SMT] SHALL have the privilege “Generate qualified electronic signature” to use a qualified electronic signature key;

b. The [SMT] SHALL have the privilege “Generate electronic signature” to use an electronic signature key;

3.6.2.2.1 Case 1: the user credential signature is the global PIN

Case 1a): If PACE with the global PIN has been performed, under secure messaging the [SMT]/[UD]:

a. SHALL select the signature key to be used;

b. SHALL send the data to be signed to the [eSign application];

For each subsequent signature creation required, under secure messaging the [SMT]/[UD] SHALL:

a. Verify the global PIN: [CONDITIONAL] to usage of consecutive signature mode (see §3.1.1.2).

   i. switch the session context to local secure messaging CH_UD

   ii. select the [eSign Application]

   iii. Submit the global PIN using the VERIFY command

   iv. switch back the session context to CH_SCA

   v. select the [eSign Application]

b. select the signature key to be used;

c. send the data to be signed to the [eSign application];

Case 1b): If PACE with the CAN has been performed, under secure messaging the [SMT]/[UD] SHALL:

a. Verify the global PIN

   i. switch the session context to local secure messaging CH_UD

   ii. select the [eSign Application]

   iii. Submit the global PIN using the VERIFY command

   iv. switch back the session context to CH_SCA

   v. select the [eSign Application]

b. select the signature key to be used;

c. send the data to be signed to the [eSign application];

For each subsequent signature creation required, under secure messaging the [SMT][UD] SHALL:

a. Verify the global PIN: [CONDITIONAL] to usage of consecutive signature mode (see §3.1.1.2).

   i. switch the session context to local secure messaging CH_UD

   ii. select the [eSign Application]
   iii. Submit the global PIN using the VERIFY command
   iv. switch back the session context to CH_SCA
   v. select the [eSign Application]

b. select the signature key to be used;

c. send the data to be signed to the [eSign application];

3.6.2.2.2 Case 2: the user credential signature is NOT the global PIN

Under secure messaging, the [SMT]/[UD] SHALL:

a. Verify the local user credential
   i. switch the session context to local secure messaging CH_UD
   ii. select the [eSign Application]
   iii. Submit the global PIN using the VERIFY command
   iv. switch back the session context to CH_SCA
   v. select the [eSign Application]

b. select the signature key to be used;

c. send the data to be signed to the [eSign application];

For every subsequent signature, under secure messaging the [SMT]/ [UD] SHALL:

a. Verify the local user credential: [CONDITIONAL] to usage of consecutive signature mode (see §3.1.1.2).
   i. switch the session context to local secure messaging CH_UD
   ii. select the [eSign Application]
   iii. Submit the global PIN using the VERIFY command
   iv. switch back the session context to CH_SCA
   v. select the [eSign Application]

b. select the signature key to be used;

c. send the data to be signed to the [eSign application];

Notes

The [SMT]/terminal SHALL iterate the two last operations to create signature as long as required and allowed by the [eSign application] (before the user credential status is not verified)

3.6.2.3 Signature creation without authentication of the [SMT]

This mode of operation is available only if the [eSign application] has been configured to create signature without the authentication of the [SMT]. As such, the [eSign application] SHALL be used in a trusted environment. The user and the issuer SHALL take all appropriate measures to ensure a sufficient security level is met.

> ***Notes***
>
> This configuration is exclusive with the one described in §3.6.2.2

The sequence of operation described in §3.6.2.1 SHALL be applied. In this case, the eIDAS token SHALL require a successful PACE before signature creation.

Depending on the configuration of the [eSign application], the signature creation is controlled by a user credential that is either the Global PIN, or the local user credential.

PACE with the global PIN SHALL be performed. In any case, PACE SHALL be performed according to [TR03110-2] and SHALL contain confined authorization (in the CHAT),, ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

3.6.2.4 Management of user consent's internal state

The user's consent for signing a document is materialized by the submission (and authentication) of the [eSign application] user credentials. These credentials are either local or global, when the global PIN is used as a signature PIN.

The management of the user's consent internally stored in [eSign application] differs depending on the type of key used for the signature key:

a. Electronic signature key;

b. Qualified electronic signature key;

By default, [SMT] can produce only one signature before the user's consent is reset. However, the [SMT] can create up to the number of signatures specified in the certificate extension before the user's consent is reset. When no external authentication is mandated, the number of signatures SHALL be configured in the [eSign application].

Once the maximum number of consecutive signature(s) has been met, the user consent SHALL be set through the VERIFY command.

3.6.3 Data management

3.6.3.1 Local user credential vs. global user credential

eIDAS token MAY handle global and local credentials:

- global user credentials are located in the MF;
- local user credentials are located in the ADF;

Both local and global user credentials SHALL be managed according to the current document.

Global user credentials that grant access to the signature creation function and local user credentials SHALL only be managed according to the security policies defined herein.

In case the eIDAS token also contains other applications, the global user credentials MAY also be managed according to other security policies defined by these supplemental application, provided they do not grant access to the signature creation function. In particular, if the eIDAS token also supports an eID application as defined in [TR03110-2], global user credentials SHALL also be managed as described in [TR03110-2].

3.6.3.2 Specific issues for global user credentials and PUK

The suspension mechanism as stated in [TR03110-2] SHALL be supported for the PUK(s) and global user credential(s) (user credential). As such, the support of CAN is mandatory.

3.6.3.3 User credential initialization

The [eSign application] enables to initialize the user credential (user credential) used to access the signature function, whether it is global or local. The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Retrieval of the [ISO/IEC 7816-15] data;

d. Initialization of the user credential;

e. Update of [ISO/IEC 7816-15] data

PACE SHALL be performed according to [TR03110-2] and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The [SMT], after successful completion of GAP is allowed to initialize the user credential and to update the content of [ISO/IEC 7816-15] files in the [eSign application]. In particular the [SMT] is responsible for updating accordingly the [ISO/IEC 7816-15] structure.

The terminal performing the GAP SHALL be a [SMT] with the privilege “User credential initialization”. to initialize a user credential;

Prior the user credential initialization, the [SMT] SHALL retrieve the [ISO/IEC 7816-15] structure to discover the entire user credential available in the eIDAS token as well as their internal state: “initialization”, “operational – activated”, “operational – deactivated” or “terminated” (see §3.5.1.1).The [eSign application] SHALL only accept initialization on user credential whose internal life cycle state is set to “initialization state”.

The user credential SHALL be initialized using CHANGE REFERENCE DATA command with P1=’01’, P2 containing the identifier of the user credential and the data field containing the user credential. This command is available only once. After a successful completion, the user credential life cycle status is automatically set to “operational – activated”.

The [SMT] SHALL also update the [ISO/IEC 7816-15] structure to reflect the change of the life cycle state of the user credential. The [SMT] SHALL use the UPDATE BINARY command to update the [ISO/IEC 7816-15].

3.6.3.4 User credential update

The [eSign application] enables to update the value of the user credential, whether it is global or local. The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Retrieval of the [ISO/IEC 7816-15] data;

d. Update the user credential value;

PACE SHALL be performed according to [TR03110-2] and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege "User credential value update" as defined in §3.1.1.1.

3.6.3.5 Prior the user credential value update, the [SMT] SHALL retrieve the [ISO/IEC 7816- the entire user credential available as well as their internal state: "initialization", "operational – deactivated" or "terminated" (see Available operations

The following operations may be performed on a user credential:

e. **Verification**: this operation submits a candidate user credential to the eIDAS token that compares it against the reference user credential. Upon success the following actions are performed
   - the user credential verification status is set;
   - the corresponding access rights are granted;
   - the retry counter is restored to its initial value;

   Upon failure, the following actions are performed:
   - the user credential verification status is reset;
   - the corresponding access rights are denied;
   - the retry counter is decremented by one;

f. **Change**: this operation changes the reference user credential values stored in the eIDAS token. If successful, the user credential verification status is reset"

g. **Devalidation**: this operation resets the user credential verification status.

h. **Unblocking** : this operation consists in unblocking the user credential, namely restoring its retry counter to the initial value (described below), resetting its verification status, and changing its reference value.

Notes

In order to identify the user credential on which the operation SHALL be performed, the identifier of the user credentials SHALL be provided in the field "reference data" of the command. All user credentials share the same range of identifiers. For example, if a PIN credential has the identifier #1, no physical user credential SHOULD have the identifier #1. Identifiers SHALL be in range 1 to 31 included.

Life cycle state§3.5.1.1).The [eSign application] SHALL only accept update on user credential whose internal life cycle state is set to "operational – activated".

Case of PIN credential update

The old PIN, and the new PIN SHALL be submitted together in the same incoming CHANGE REFERENCE DATA command with P1=’00’ and reference indicated in P2.

Updating the global PIN can also be performed according to [TR03110-3], §B.7.1.

Case of physical user credential update

Refer to [TR_Physical_Authentication] §1.3.4

3.6.3.6 User credential unblocking

Once the retry counter of the user credential has reached zero, it is locked and does not allow further usage. The [eSign application] enables to unblock and modify the value of the user credential, whether it is global or local. The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Retrieval of the [ISO/IEC 7816-15] data;

d. Unblock the user credential;

PACE SHALL be performed according to [TR03110-2], and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege “User credential unblock” as defined in §4.1.

Prior the user credential unblocking, the [SMT] SHALL retrieve the [ISO/IEC 7816-15] structure to discover the entire user credential available as well as their internal state: “initialization”, “operational – activated”, “operational – deactivated” or “terminated” (see §3.5.1.1).The [eSign application] SHALL only unblock user credential whose internal life cycle state is set to “operational – activated”.

A user credential is unblocked with RESET RETRY COUNTER command with ‘02’ set in P1, reference indicated in P2 and new value set in data field. Retry counter is reset to its initial value n in case of success.

Unblocking the global PIN can also be performed according to [TR03110-3], §B.7.1.

3.6.3.7 Signature key and certificate generation

The [eSign application] enables to generate an electronic signature key on board, whether it is a qualified electronic signature key or not. The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Retrieval of the [ISO/IEC 7816-15] data;

d. Generation of a signature key;

e. Certificate building;

f. Update EF.CD[x] in [ISO/IEC 7816-15] structure;

PACE SHALL be performed according to [TR03110-2], and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege “install certificate” or “install qualified certificate” as defined in §4.1. The [SMT] SHALL have the privilege:

a. “install qualified certificate” to

   i. generate a qualified electronic signature key;

   ii. update the content of EF.CD[x] in [ISO/IEC 7816-15] structure;

b. “install certificate” to

   iii. generate an electronic signature key;

   iv. update the content of EF.CD[x] in [ISO/IEC 7816-15] structure;

Prior the key generation, the [SMT] SHALL retrieve the [ISO/IEC 7816-15] structure to discover all the electronic signature keys available as well as their internal state: “initialization”, “operational – activated”, “operational – deactivated” or “terminated” (see Life cycle state §3.5.1.1). The [eSign application] SHALL only generate keys on electronic signature keys whose internal life cycle is set to “initialization state” or “operational – Activated”.

The [SMT], after successful completion of GAP is allowed to perform the electronic signature key generation AND to update the content of [ISO/IEC 7816-15] files in the [eSign application]. In particular the [SMT] is responsible for loading the certificate in the [eSign application] and updating accordingly EF.CD[x] in the [ISO/IEC 7816-15] structure.

Key generation allow to generate EC and RSA key pair. It is performed with the GENERATE ASYMMETRIC KEY PAIR command containing:

During the generation of the electronic signature key the following operations are performed by the [eSign application]:

a. the private and the public portion are generated by the [eSign application];

b. the public portion is exported so that the [SMT] can generate the signature key certificate;

Notes

The public portion of the signature key is returned only by the GENERATE ASYMMETRIC KEY PAIR command.

Key generation can be used for electronic signature key initialization and renewal. Upon key generation, signature keys are generated using the attributes associated to the data container:

a. signature algorithm;

b. key type;

c. key length;

d. domain parameters for EC;

In case of successful renewal the key initially saved to this location is replaced by the newly generated one. In case of failure during renewal while the usage of the command was allowed, the key initially saved to this location is erased and cannot be used until the key generation is successful.

The [eSign application] does not require the user credential protecting the electronic signature key to be initialized before signature key generation.

The holder submits its user credentials to unlock the signature creation function in the [eSign application]. The user's consent internal state is reset after electronic signature keys have been generated, whether the key is a qualified electronic signature key or not.

In order to generate the signature key certificate, the [SMT] SHALL retrieve the following information:

a. attributes of the electronic signature key. Thanks to [ISO/IEC 7816-15] structure, the [SMT] can retrieve all the keys attributes (Electronic signature algorithms, Key type and size,…);

b. attributes of the user (name, surname,…). The [SMT] can either retrieve it in another application present in the eIDAS token (e.g. an eID application), or in the file EF.INFO4CERT if present.

Once the signature key certificate has been generated, the [SMT] shall update the [ISO/IEC 7816-15] structure in order to store the newly generated signature key certificate. In case the signature key was in "initialization state", the [SMT] SHALL also update the [ISO/IEC 7816-15] structure to reflect the change of the life cycle state of the signature key. The [SMT] SHALL use the UPDATE BINARY command to update the [ISO/IEC 7816-15].

3.6.3.8 Life cycle management

The [eSign application] enables to manage the life cycle of the electronic signature key (whether it is qualified or not) and user credential, namely by locking/unlocking (suspension of its usage) and termination (the object becomes irreversibly unusable). The process flow is the following:

a. GAP;

b. Selection of the [eSign application];

c. Retrieval of the [ISO/IEC 7816-15] data;

d. Manage the life cycle of the signature key or user credential;

e. Update of [ISO/IEC 7816-15] data;

PACE SHALL be performed according to [TR03110-2], and SHALL contain confined authorization (in the CHAT), ensuring the user validates the operations to be performed on the [eSign application]. Once PACE has been successfully performed, all the subsequent communication SHALL be protected using secure messaging.

The terminal performing the GAP SHALL be a [SMT] with the privilege "object management" as defined in §3.1.1.1.

The [SMT], after successful completion of GAP is allowed to manage the lifecycle of the electronic signature keys and user credential, and to update the content of [ISO/IEC 7816-15] files in the [eSign application]. In particular the [SMT] is responsible for updating accordingly the [ISO/IEC 7816-15] structure.

While the state “Operational – Deactivated” is reversible, the state “Terminated” is irreversible as described in [ISO/IEC 7816-8].

After having successfully managed the life cycle of an object, the [SMT] SHALL update the [ISO/IEC 7816-15] structure to modify the life cycle on the said object:

a. Operational – Activated

b. Operational – Deactivated

c. Terminated

3.6.4 User authentication devalidation

The [eSign application] can devalidate the verification state of user credential.

3.6.4.1 Global user credentials

The global PIN may be used for both PACE and as a signature PIN. On such a configuration, it maintains a verification state. This state is updated on following events:

a. PACE execution using the PIN. If PACE succeeds, the PIN verification state is set (and PACE authentication state is set). If PACE fails, the PIN verification state is reset (and PACE authentication state is reset).
b. A VERIFY command on the global PIN results in the PIN verification state to be set in case of success, and reset in case of failure.
c. A reset of the eIDAS token resets the PIN verification state.
d. A RESET RETRY COUNTER resets the PIN verification state.
e. a VERIFY command with P1 set to 'FF', P2 containing the identifier of the PIN and an empty data field resets the PIN verification state

Moreover, if the Global PIN is used for electronic signature (qualified or not), its verification state SHALL be reset after the following events:

a. a signature key update, whatever the procedure succeed or not;

b. the n-th signature creation, where n indicates the maximum number of electronic signature indicated in the [SMT]'s certificate extension validated during GAP (n = 1 if the field is absent from the certificate extension);

c. one signature creation when the terminal has not been authenticated;

3.6.4.2 Local user credentials

The verification state of the local user credentials SHALL be reset after the following events:

a. a reset of the eIDAS token;

b. a VERIFY command with P1 set to 'FF', P2 containing the identifier of the user credential and an empty data field;

c. a CHANGE REFERENCE DATA command on the user credential;

d. a RESET RETRY COUNTER on the user credential;

e. Selection of the MF;

The verification state is not modified during a secure context switch.

Moreover, if the user credential is used for electronic signature (qualified or not), its verification state SHALL be reset after the following events:

a. a signature key update, whatever the procedure succeed or not;

b. the n-th signature creation, where n indicates the maximum number of electronic signature indicated in the [SMT]’s certificate extension validated during GAP (n = 1 if the field is absent from the certificate extension);

c. one signature creation when the terminal has not been authenticated;

3.7 ISO/IEC 7816 mapping

3.7.1 GENERATE ASYMMETRIC KEY PAIR

The generate public key pair command initiates the generation and storing of a key pair, i.e. the private portion and the public portion (optional) are stored in the eIDAS token. The public portion SHALL be returned by the eIDAS token in order to generate a dedicated certificate and use for checking the signature. It can be performed only if the security status satisfies the security attributes for this command.

This command is operational for ECC and RSA key type.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'47'
P1	'82'
P2	'00'
L_c field	Data Length
Data field	’B6’-L-{ {‘83’- L-PublicKeyRef} {‘4D’- L-{‘7F49’-’80’}} {[‘E2’ L ‘82’- L –V]} } , for RSA key , with [] optional (default value ‘10001’) ’B6’-L-{ {‘83’- L-PublicKeyRef} {‘4D’- L-{‘7F49’-‘04’-‘06’-‘00’-‘86’-‘00’}} }, for EC key
L_e field	Present

RESPONSE PARAMETER	MEANING
Data field	Return only public elements in order to generate a dedicated certificate 7F49’-L – {‘81’L <modulus> - 82 L <exponent>}} for RSA key 7F49’-L – {‘06’L <OID> - ‘86’ L <public point> } for ELC key
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

3.7.2 MSE SET for CRT DST

This command sets or replaces the signature CRT DST hash and scheme algorithm in the current Security Environment. 960

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO '22' '41' – Set for signature creation 'B6' – CRT of DST
L_c field	Data Length
Data field	’84’-‘01’ -{ Private key Reference}
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 4 - MSE SET for CRT DST

965 3.7.3 PSO: HASH

The off-card entity is responsible for computing the intermediate hash over the first part of the data to be signed. The intermediate hash-code is transferred to the eIDAS token by the PSO:HASH command together with the remaining part of the data, the eIDAS token performs the last round hash computation.

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO '2A' '90' 'A0'
L_c field	Data Length
Data field	'90' L90 <intermediate hash-code followed by a bit counter encoded with most significant bit first> '80' L80 <data to be hashed>, limited to one block
L_e field	Absent

970

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 5 – PSO Hash

3.7.4 PSO: Compute Digital Signature

The PSO Compute Digital Signature operation initiates the computation of a digital signature. It can be performed only if the security status satisfies the security attributes for this command.

The algorithm may be either a digital signature algorithm or a combination of a hash algorithm and a digital signature algorithm.

To compute a digital signature, the data to be signed or integrated in the signing process are transmitted in the command data field or provided through a previous PSO: HASH command.

Note: in case of PSO: HASH used command, the eIDAS token SHALL guarantee this DTBS only will be signed.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'2A'
P1	'9E'
P2	'9A'
L_c field	Absent if PSO:HASH occurred before Data Length if Hash value to transfer
Data field	Absent (PSO:HASH occurred before) Data To be Signed (Hash value computed off card)
L_e field	Present

RESPONSE PARAMETER	MEANING
Data field	Digital signature
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 6 - PSO Compute Digital Signature

3.7.5 MSE SET AT for context switch

This command performs a context switch according to a given session context identifier. The session context consists of the secure messaging context (session keys and SSC), the current DF and EF, as defined in [TR03110-2]. The current DF is the eSign application ADF.

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO '22' '01' 'A4' – CRT AT for session context restore
L_c field	'05'

Data field	'E1'-'03' - { '81'-'01' – *session_ctxt_id*}
	session_ctxt_id (see also [TR03110-3]):
	'00' for CH_UD (PACE), implicitly saved during PACE execution
	'01' for CH_SCA (GAP), explicitly saved during GAP execution

L_e field	Absent

990

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 7 - MSE SET AT for context switch

3.7.6 VERIFY PIN

Knowledge based user verification requires the user to enter a password.

Note: If password is absent from the data field, the verification state of the user credential is returned through the status word. If status word is '9000', the user credential verification state is still set. This command has no impact on the state and associated rights.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'20'
P1	'00'
P2	reference data qualifier
Lc field	Data Length or absent
Data field	If Lc present, <Password> If Lc absent, absent
Le field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 8 - Verify PIN

3.7.7 VERIFY of physical user credential

Refer to [TR_Physical_Authentication] §2.1

3.7.8 VERIFY for Devalidation of User authentication PIN.

The command resets the PIN verification state.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'20'
P1	'FF'
P2	reference data qualifier
L_c field	Absent
Data field	Absent
Le field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 9 - Devalidation User authentication PIN

3.7.9 VERIFY for devalidation of physical user credential.

Ref. [TR_Physical_Authentication] §2.2

3.7.10 CHANGE REFERENCE DATA for PIN update

1015 This command intends to replace a current password value with a new one. It requires a successful comparison between the reference password and the verification data sent from the interface device. The reference data replacement can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO '24' '00' reference data qualifier
L_c field	Data Length
Data field	<old password> || <new password>
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

1020 **Figure 10 - Change Reference Data for PIN update**

3.7.11 CHANGE REFERENCE DATA for PIN initialization

The command initializes the PIN value. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'24'
P1	'01'
P2	reference data qualifier
L_c field	Data Length
Data field	<new password>
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 11 - Change Reference Data for PIN initialization

3.7.12 CHANGE REFERENCE DATA for physical user credential initialization

Ref. [TR_Physical_Authentication] §2.3

3.7.13 CHANGE REFERENCE DATA for physical user credential update

Ref. [TR_Physical_Authentication] §2.4

3.7.14 RESET RETRY COUNTER for PIN

After N (N as specified by application) wrong consecutive verification of the password, the password is locked and does not allow further usage of the protected functions. It can be performed only if the security status satisfies the security attributes for this command.

After successful completion of the command, the retry counter of the password is restored at its initial value N by providing a new password.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'2C'
P1	'02'
P2	reference data qualifier
L_c field	Data Length
Data field	<new password>
Le field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 12 - Reset Retry Counter for PIN

3.7.15 RESET RETRY COUNTER of physical user credential

Refer to [TR_Physical_Authentication] §2.5

3.7.16 ACTIVATE

This command is used to turn the state of the [eSign application], a key or a user credential to the activated state. No error occurs if [eSign application], key or user credential was already activated. It can be performed only if the security status satisfies the security attributes for this command.

Command Parameter	MEANING
CLA	ISO1
INS	'44'
P1	As indicated in Figure 13
P2	‘00’ if not denoted otherwise in Figure 13
Lc field	Absent for encoding Nc = 0, present for encoding Nc > 0
Data field	According to definitions in Figure 13
Le field	Absent for encoding Ne = 0

Response Parameter	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 13 – Activate

The table below describes only mandatory options that must be available for interoperability.

[1] This command is currently under discussion in [ISO/IEC 7816-9]. In the meanwhile, a proprietary class byte is used waiting for the outcomes of the standardization process. Based on the outcomes, this command OR Manage Data SHALL be used.

b8	b7	b6	b5	b4	b3	b2	b1	Meaning	Data field
0	0	0	0	x	x	x	x	**File operations (EF or DF)**	Out of scope
-	-	-	1	x	x	x	x	**Password operations**	
-	-	-	1	0	0	0	0	Indication of password in P2	Absent
			1	0	0	0	1		Out of scope
-	-	1	x	x	x	x	x	**Key operations**	
		1	0	0	0	0	0	Indication of key in P2	Absent
		1	0	0	0	0	1		Out of scope
-	1	x	x	x	x	x	x	**Operations for any structure**	Out of scope
— Any other value is RFU.									

Figure 14: LCS - Indication in P1

3.7.17 DEACTIVATE

1060 This command is used to turn the state of the [eSign application], a key or a user credential to the deactivated state. No error occurs if [eSign application], key or user credential was already deactivated. It can be performed only if the security status satisfies the security attributes for this command.

After a successful completion of the command the following rules shall apply:

1065
- In case of [eSign application], only the SELECT, ACTIVATE and TERMINATE commands are allowed,
- In case of a key any further cryptographic operation performed by the key is impossible,
- In case of a user credential no further operation performed on the user credential is allowed.

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO2 '04' As indicated in Figure 13 ‘00’ if not denoted otherwise in Figure 13
LC FIELD	Absent for encoding Nc = 0, present for encoding Nc > 0
DATA FIELD	According to definitions in Figure 13
LE FIELD	Absent for encoding Ne = 0

1070

RESPONSE PARAMETER	MEANING
DATA FIELD	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 15 - Deactivate

[2] This command is currently under discussion in [ISO/IEC 7816-9]. In the meanwhile, a proprietary class byte is used waiting for the outcomes of the standardization process. Based on the outcomes, this command OR Manage Data SHALL be used.

3.7.18 TERMINATE

1075 This command is used to turn the state of the [eSign application], a key or a user credential to the terminated state. No error occurs if [eSign application], key or user credential was already terminated. It can be performed only if the security status satisfies the security attributes for this command.

After a successful completion of the command, the file, key or user credential is unusable

COMMAND PARAMETER	MEANING
CLA	ISO3
INS	'E8'
P1	As indicated in Figure 13
P2	‘00’ if not denoted otherwise in Figure 13
LC FIELD	Data Length or absent
DATA FIELD	According to definitions in Figure 13
LE FIELD	Absent for encoding Ne = 0

RESPONSE PARAMETER	MEANING
DATA FIELD	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

1080

Figure 16 - Terminate

[3] This command is currently under discussion in [ISO/IEC 7816-9]. In the meanwhile, a proprietary class byte is used waiting for the outcomes of the standardization process. Based on the outcomes, this command OR Manage Data SHALL be used.

3.7.19 MANAGE DATA: OPERATIONAL CHANGE LCS User credential and Keys

This command changes the LCS (see [ISO/IEC 7816-4] Table 14) of User credential to the value of LCS indicated in the P2 parameter. It can be performed only if the security status satisfies the security attributes for this command.

The LCS MAY be changed to operational-state activated, operational-state deactivated or terminated

Note: this command applies also on user physical credential as defined in [TR_Physical_Authentication] §2.6

COMMAND PARAMETER	MEANING
CLA	ISO[4]
INS	'CF'
P1	'00'
P2	LCS to be set by the command (see [ISO/IEC 7816-4] Table 14)
L_c field	Data Length

Data field	‘7F71’- L – { ‘7F70’ – L – {‘83’ –L - <User credential Id>}}
	‘7F71’- L – { ‘7F70’ – L – {‘84’ –L - <Private key Id>}}

L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 17 – Manage Data: change LCS User credential and Keys

[4] This command is currently under discussion in [ISO/IEC 7816-9]. In the meanwhile, a proprietary class byte is used waiting for the outcomes of the standardization process. Based on the outcomes, this command will be removed and ACTIVATE/DEACTIVATE/TERMINATE commands should be used..

3.7.20 SELECT

This command selects a file (EF) or an application (MF or ADF). After a successful selection the file selected becomes the current file. After the reset the current DF is the MF and no EF is selected. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'A4'
P1	'00' - Select MF, EF
	'02' - Select EF under current DF
	'04' - Select by DF name (ADF)
P2	'0C' - no data in response field
L_c field	Data Length
Data field	If P1='00', MF or EF Identifier If P1='02', EF Identifier If P1='04', AID
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 18 - Select

3.7.21 SELECT MF

This command selects the MF as define in [ICAO 9303]. After a successful selection the file selected becomes the current file. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'A4'
P1	'00'
P2	'00'
L_c field	Absent
Data field	Absent
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 19 - Select MF

3.7.22 READ BINARY

This command reads the content of a transparent EF (on a current selected file or with implicit SFI selecting). It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'B0'
P1	As indicated in Table below Binary P1-P2 coding
P2	As indicated in Table below Binary P1-P2 coding
L_c field	Absent
Data field	Absent
L_e field	Number of bytes to read

RESPONSE PARAMETER	MEANING
Data field	Data read
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 20 - Read Binary

P1								P2							
B8	B7	B6	B5	B4	B3	B2	B1	B8	B7	B6	B5	B4	B3	B2	B1
0	Offset in the currently selected file over 15 bits '00' ≤ Offset ≤ '7FFF'														
1	0	0	Short File Identifier 1 ≤ SFI ≤ 30					Offset in the file over 8 bits							

Figure 21 - Read Binary: P1 P2 encoding

3.7.23 UPDATE BINARY

This command updates the content of a transparent EF (on a current selected file or with implicit SFI selecting). It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'D6'
P1	As indicated in Table Binary P1-P2 coding
P2	As indicated in Table Binary P1-P2 coding
L_c field	Data Length
Data field	Data to write
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 22 - Update Binary

3.7.24 ACTIVATE [eSign application]

This command is used to turn the [eSign application] to the activated state. No error occurs if [eSign application] was already activated. The command applies when [eSign application] is selected. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA INS P1 P2	ISO '44' '00' '00'
L_c field	Absent
Data field	Absent
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 23 - Activate

3.7.25 DEACTIVATE [eSign application]

This command is used to turn the [eSign application] to the deactivated state. No error occurs if the file was already deactivated. The command applies when [eSign application] is selected. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'04'
P1	'00'
P2	'00'
L_c field	Absent
Data field	Absent
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 24 - Deactivate

3.7.26 TERMINATE [eSign application]

This command is used to turn the [eSign application] to the terminated state. No error occurs if the file was already terminated. The command applies when [eSign application] is selected. It can be performed only if the security status satisfies the security attributes for this command.

COMMAND PARAMETER	MEANING
CLA	ISO
INS	'E6'
P1	'00'
P2	'00'
L_c field	Absent
Data field	Absent
L_e field	Absent

RESPONSE PARAMETER	MEANING
Data field	Absent
SW1-SW2	See [ISO/IEC 7816-4], Tables 5 and 6 where relevant

Figure 25 - Terminate

3.8 : Example of [ISO/IEC 7816-15] structure

This annex is informative and proposes an illustration of [ISO/IEC 7816-15] applied in the context of the current technical report.

3.8.1 General

This application profile describes a digital [eSign application] offering a high level of access control. This profile is e.g. suitable for a contact and contactless eIDAS token. The keys and certificates are necessary to generate digital signatures and are not installed at the time of issuance of the eIDAS token, the eIDAS token has to provide a mechanism to grant access rights to install keys and certificates. This mechanism is out of scope of this profile, but an eIDAS token profile using this application profile should describe a mechanism to install keys and certificates.

The mandatory features indicated below define the minimal set of functionality that must be supported by an eIDAS token in order to guarantee interoperability.

In order to ensure the privacy, the CIA structure SHALL be under the DF eSign, and only an authorized signature management terminal is allowed to handle the structure.

The Cryptographic Information application shall be designed according to [ISO/IEC 7816-15], this chapter specifies the use of CIA for the eSign application.

It describes:

a. Cryptographic algorithm of the [eSign application].

b. The authentication objects (e.g. Password, External authentication).

c. The private and public keys present in the eSign application.

d. The certificates associated to the keys of the eSign application.

The link is insured by the description of the CIA in the EF.DIR file. Such a class diagram summarizes the CIA description:

The goal of the CIA is to provide to the terminal all the descriptive files to gain a complete knowledge of the cryptographic objects and their use.

The CIA main entrance is the EF.DIR placed under the MF. To avoid disclosure of sensitive information, if privacy is required, it SHALL deliver only global conditions as preconditions to establish before access to any applications available on eIDAS token. All the others elements of the CIA SHALL be stored under the application: only grant access to the application SHALL allow access to the different [ISO/IEC 7816-15] files.

The flow to execute in order to get acknowledge of the [ISO/IEC 7816-15] SHALL follow after card reset:

a. Get knowledge of the content EF.DIR

Depending of the CIODDO information (but when PACE and GAP required)

b. Perform PACE

c. Perform EAC

d. Select [eSign application], where EF.OD and EF.CIAINFO (standard File ID) are found under their respective by default file identifier available through READ BINARY commands.

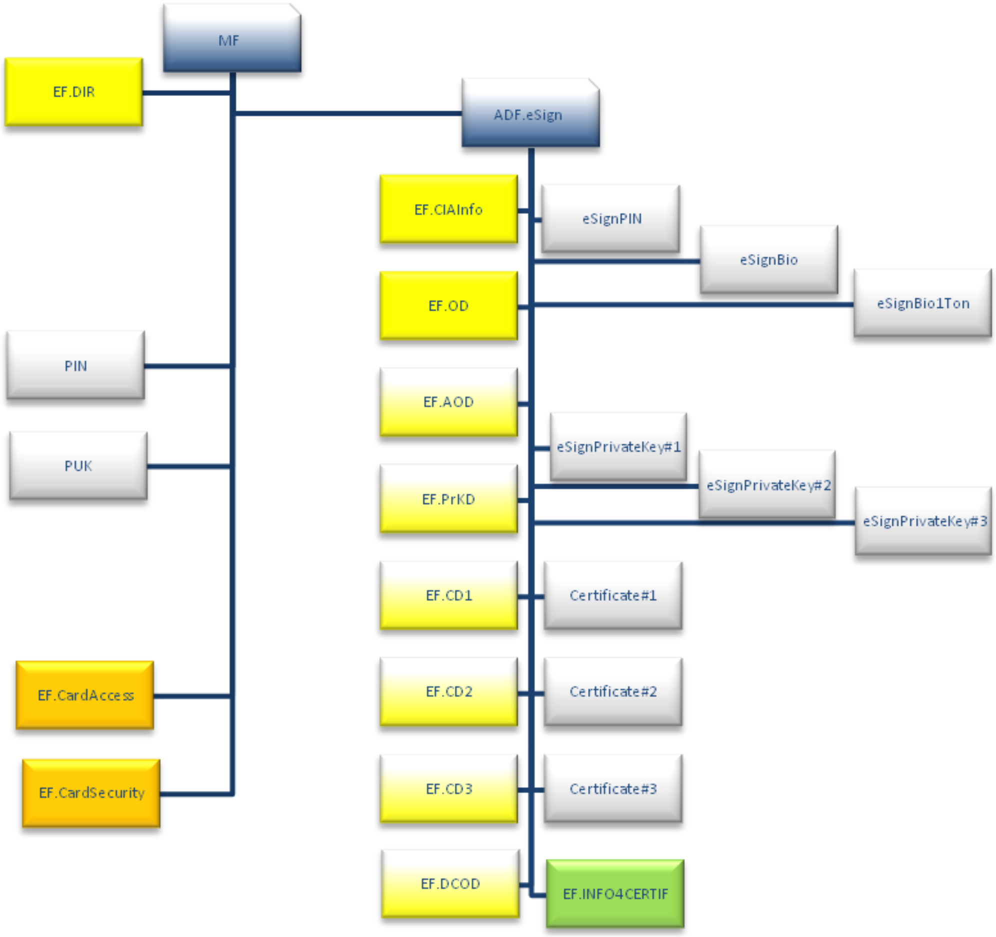

Figure 26 – ISO/IEC 7816-15 Data Structure for the [eSign application]

3.8.2 EF.DIR

This file (EFID = '2F00' and SFI = '1E') placed under the Root level and readable after the Reset of the eIDAS token. It structure can be described as follows – it SHALL contain only global access to the eIDAS token to ensure the privacy-. Set of Application template (DO’61’), each one is composed of:

```
CIODDO ::= SEQUENCE {
 securityFileOrObject SET OF SecurityFileOrObject ,
-- EF.CardAccess, e.g. 3 SecurityFileOrObject items based on CAN, PIN,
and PUK
 securityFileOrObject SET OF SecurityFileOrObject , -- EF.CardSecurity
 securityFileOrObject SET OF SecurityFileOrObject , -- EF.ChipSecurity
}

    SecurityFileOrObject ::= SEQUENCE {
     label Label ,
     fileOrObjectPath Path,
     index [0] INTEGER (0..cia-ub-index) ,
     precondition INTEGER (0..cia-ub-index),
    }
```

Note: respective ciaInfo.path for EF.CIAInfo and EF.OD are not specified here because of their file identifier (see clause [ISO/IEC 7816-15] §7.5.1)

Note: index is the unique identifier for SecurityFileOrObject and may evaluate to a key or password reference

Note: when applied to GAP, preconditions SHALL be assigned as follow ‘01’ for PACE, ‘02’ for TA2, ‘03’ for CA2

3.8.3 Data organization

The CIA application is a simple file structure. The access conditions on files should be as described in the next table, minimum representation :

Data element	Mandatory or Optional	FID	SFID	Access Conditions	
				Update:	Read:
EF.CIAInfo	M	50 32	12	Never	eSign Access Conditions
EF.OD	M	50 31	11	Never	eSign Access Conditions
EF.AOD	M	50 06	-	eSign Access Conditions and Role Admin	eSign Access Conditions
EF.DCOD	M	50 04	-	Never	eSign Access Conditions
EF.PrKD	M	50 01	-	eSign Access Conditions and Role Admin	eSign Access Conditions
EF.CD#1	M	51 01	-	eSign Access Conditions and Role CSP_Admin	eSign Access Conditions
EF.CD#2	O	51 02	-	eSign Access Conditions and Role CSP_Admin2	eSign Access Conditions
EF.CD#3	O	51 03	-	eSign Access Conditions and Role CSP_Admin3	eSign Access Conditions

Figure 27 - CIA application

Note:

- EF.CD[1…]: life cycle status of the eSign Pin SHALL be updated when on field generation occurred.

Note: eSign Access Conditions as determined by `SecurityFileOrObject` (see clause §3.8.2)

3.8.3.1 EF.CIAInfo

This file is mandatory and gives general information about the eIDAS token and about the associated eSign :

Item	Description	
version	Match the version defined in ISO/IEC 7816-15	Mandatory
serialNumber	Identifies a unique serial number for the card	Optional
Label	Provides identifying information about the application (to be displayed by host application)	Mandatory
cardFlags	Provides information about the eIDAS token itself, flags denote: whether the eIDAS token is read-only; whether there are cryptographic functions that require a user to be authenticated; whether the eIDAS token supports pseudo-random number generation	Mandatory
supportedAlgorithms	Lists all algorithms supported by the eIDAS token	Mandatory

Figure 28 - EF.CIAInfo file structure

Item	Description

AlgorithmInfo.reference	unique identifier in the CIA application of the private key.	Mandatory
AlgorithmInfo.algorithm	PKCS#11 algorithm identifier	Mandatory
AlgorithmInfo.parameters	parameters for using the algorithm if needed	Mandatory
AlgorithmInfo.supportedOperations	list of the operation that can be performed with this algorithm	Mandatory
AlgorithmInfo.objID	represent object identifier of the algorithm (OID)	Mandatory
AlgorithmInfo.algRef	indicate the associated algorithm identifier	Mandatory

Figure 29 - EF.CIAInfo: supported algorithms

3.8.3.2 EF.OD

This file is mandatory and lists all the classes of objects supported by the CIA and provides the method of retrieval of CIO files by giving the path to the file:

Files associated to each class of objects supported by the associated application:

- Reference path to the authentications objects descriptor file EF.AOD.
- Reference path to the files used as Data Container Objects Descriptor file EF.DCOD.
- Reference path to the private keys descriptor file EF.PrKD;
- Reference path to the certificates descriptor file EF.CD#1.
- Reference path to the certificates descriptor file EF.CD#2.
- Reference path to the certificates descriptor file EF.CD#3.

3.8.3.3 EF.AOD

This file is mandatory and lists the entire authentication objects; an authentication object is characterized by its attributes (e.g. user credentials and external Authentication):

- External Authenticate with Role xxx (DO 'A2')

<table>
<tr><th>Attributes</th><th colspan="2">Item</th><th colspan="13">Description</th></tr>
<tr><td>Common</td><td colspan="2">label</td><td colspan="13">An ASCII descriptive string (e.g. “Role xxx”)</td></tr>
<tr><td rowspan="3"></td><td rowspan="3">accessControlRules</td><td rowspan="2">APDU</td><td colspan="11">Accessmode</td><td rowspan="2">securityConditions</td><td rowspan="2">CommunicationMode
[OPTIONAL]</td></tr>
<tr><td>Read(0)</td><td>Update(1)</td><td>Execute(2)</td><td>Delete(3)</td><td>Attribute(4)</td><td>Pso_cds(5)</td><td>Pso_verif(6)</td><td>Pso_dec(7)</td><td>Pso_enc(8)</td><td>Int_auth(9)</td><td>Ext_auth(10)</td></tr>
<tr><td>EXTERNAL AUTHENTICATE</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>1</td><td>ALWAYS</td><td>BIT STRING {
contact (0),
contactLess (1)
}</td></tr>
<tr><td>Class</td><td>N/A</td><td colspan="14"></td></tr>
</table>

Type **(CertBased)**	cha	**Flag**	**Value**
		Defined by the content of CHAT and Extended fields e.g. value of one role ‘7F 4C’ L ‘06’ L - <id-SMT> ‘53’ L <Role> ‘65’ L ‘73’ L ‘06’ L - <id-SMT authorization> ‘53’ L <Role> [‘73’ L ‘06’ L - <id-SMT attribute > ‘53’ L <maximum consecutive signature]	

Figure 30 - EF.AOD file structure

Note: there may be as many as roles supported (e.g. example template)

- PIN authentication object is defined through the following fields :

185

<table>
<tr><th>Attributes</th><th>Item</th><th>Description</th></tr>
<tr><td rowspan="3">Common</td><td>label</td><td>An ASCII descriptive string (e.g. “GPIN”)</td></tr>
<tr><td>accessControlRules</td><td>
<table>
<tr><th rowspan="2">APDU</th><th colspan="11">Accessmode</th><th rowspan="2">securityConditions</th><th rowspan="2">CommunicationMode
[OPTIONAL]</th></tr>
<tr><th>Read(0)</th><th>Update(1)</th><th>Execute(2)</th><th>Delete(3)</th><th>Attribute(4)</th><th>Pso_cds(5)</th><th>Pso_verif(6)</th><th>Pso_dec(7)</th><th>Pso_enc(8)</th><th>Int_auth(9)</th><th>Ext_auth(10)</th></tr>
<tr><td>CHANGE REFERENCE DATA</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td><td rowspan="4">BIT STRING {
contact (0),
contactLess (1)
}</td></tr>
<tr><td>VERIFY</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>ALWAYS</td></tr>
<tr><td>RESET RETRY COUNTER</td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
<tr><td>MANAGE DATA</td><td></td><td></td><td>1</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
</table>
</td></tr>
<tr><td>currentLCS</td><td>ENUMERATED { init(1), op-activated(2), op-deactivated(3), termination(4)}</td></tr>
<tr><td>Class</td><td>authId</td><td>Unique identifier of the object in the [ISO/IEC 7816-15] structure’</td></tr>
<tr><td rowspan="6">Type</td><td>pwdFlags</td><td>
<table>
<tr><th>Flag</th><th>Value</th></tr>
<tr><td>case-sensitive</td><td>True = direct presentation</td></tr>
<tr><td>local</td><td>False = global PIN</td></tr>
<tr><td>change-disabled</td><td>False = change is allowed
True = change is not allowed</td></tr>
<tr><td>unblock-disabled</td><td>False = unblock is allowed
True = unblock is not allowed</td></tr>
<tr><td>needs-padding</td><td>False (no padding needed)</td></tr>
<tr><td>unblockingPassword</td><td>False (application PIN)</td></tr>
<tr><td>soPassword</td><td>False (not an administrator PIN)</td></tr>
<tr><td>exchangeRefData</td><td>True = change is possible with OLD || NEW reference value</td></tr>
</table>
</td></tr>
<tr><td>pwdType</td><td>Describe the type of the password always equals to ENUMERATED {bcd, ascii-numeric, utf8, half-nibble-bcd, iso9564-1, ...}</td></tr>
<tr><td>minLength</td><td>Minimum length of the password accepted by the eIDAS token</td></tr>
<tr><td></td><td></td></tr>
<tr><td>maxLength</td><td>Maximum length of the password accepted by the eIDAS token</td></tr>
<tr><td>pwdReference</td><td>Reference used in VERIFY command to reference the PIN in application</td></tr>
</table>

Figure 31 - EF.AOD: PIN authentication object structure

190 • PUK authentication object is defined through the following fields :

<table>
<tr><th>Attributes</th><th>Item</th><th colspan="14">Description</th></tr>
<tr><td>Common</td><td>label</td><td colspan="14">An ASCII descriptive string (e.g. “GPUK”)</td></tr>
<tr><td rowspan="4"></td><td rowspan="3">accessControlRules</td><td rowspan="2">APDU</td><td colspan="11">Accessmode</td><td rowspan="2">securityConditions</td><td rowspan="2">CommunicationMode
[OPTIONAL]</td></tr>
<tr><td>Read(0)</td><td>Update(1)</td><td>Execute(2)</td><td>Delete(3)</td><td>Attribute(4)</td><td>Pso_cds(5)</td><td>Pso_verif(6)</td><td>Pso_dec(7)</td><td>Pso_enc(8)</td><td>Int_auth(9)</td><td>Ext_auth(10)</td></tr>
<tr><td>VERIFY</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>ALWAYS</td><td>BIT STRING {
contact (0),
contactLess (1)
}</td></tr>
<tr><td>currentLCS</td><td colspan="14">ENUMERATED { init(1), op-activated(2), op-deactivated(3), termination(4)}</td></tr>
<tr><td>Class</td><td>authId</td><td colspan="14">Unique identifier of the object in the [ISO/IEC 7816-15] structure</td></tr>
<tr><td rowspan="13">Type</td><td rowspan="9">pwdFlags</td><td colspan="9"><b>Flag</b></td><td colspan="5"><b>Value</b></td></tr>
<tr><td colspan="9">case-sensitive</td><td colspan="5">True = direct presentation</td></tr>
<tr><td colspan="9">local</td><td colspan="5">False = global PIN</td></tr>
<tr><td colspan="9">change-disabled</td><td colspan="5">True = change is NOT allowed</td></tr>
<tr><td colspan="9">unblock-disabled</td><td colspan="5">True = unblock is NOT allowed</td></tr>
<tr><td colspan="9">needs-padding</td><td colspan="5">False (no padding needed)</td></tr>
<tr><td colspan="9">unblockingPassword</td><td colspan="5">True</td></tr>
<tr><td colspan="9">soPassword</td><td colspan="5">False (not an administrator PIN)</td></tr>
<tr><td colspan="9">exchangeRefData</td><td colspan="5">False = change is possible with OLD|| NEW reference value</td></tr>
<tr><td>pwdType</td><td colspan="14">Describe the type of the password always equals to ENUMERATED {bcd, ascii-numeric, utf8, half-nibble-bcd, iso9564-1, ...}</td></tr>
<tr><td>minLength</td><td colspan="14">Minimum length of the password accepted by the eIDAS token</td></tr>
<tr><td>maxLength</td><td colspan="14">Maximum length of the password accepted by the eIDAS token</td></tr>
<tr><td>pwdReference</td><td colspan="14">Reference used in VERIFY command to reference the PIN in application</td></tr>
</table>

Figure 32 - EF.AOD: PUK authentication object structure

- eSignPIN authentication object is defined through the following fields [OPTIONAL]:

<table>
<tr><th>Attributes</th><th>Item</th><th colspan="14">Description</th></tr>
<tr><td rowspan="8">Common</td><td>label</td><td colspan="14">An ASCII descriptive string (e.g. “eSign PIN”)</td></tr>
<tr><td rowspan="6">accessControlRules</td><td rowspan="2">APDU</td><td colspan="11">Accessmode</td><td rowspan="2">securityConditions</td><td rowspan="2">CommunicationMode [OPTIONAL]</td></tr>
<tr><td>Read(0)</td><td>Update(1)</td><td>Execute(2)</td><td>Delete(3)</td><td>Attribute(4)</td><td>Pso_cds(5)</td><td>Pso_verif(6)</td><td>Pso_dec(7)</td><td>Pso_enc(8)</td><td>Int_auth(9)</td><td>Ext_auth(10)</td></tr>
<tr><td>CHANGE REFERENCE DATA</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td><td rowspan="4">BIT STRING {
contact (0),
contactLess (1)
}</td></tr>
<tr><td>VERIFY</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>ALWAYS</td></tr>
<tr><td>RESET RETRY COUNTER</td><td></td><td></td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
<tr><td>MANAGE DATA</td><td></td><td></td><td>1</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
<tr><td>currentLCS</td><td colspan="14">ENUMERATED { init(1), op-activated(2), op-deactivated(3), termination(4)}</td></tr>
<tr><td>Class</td><td>authId</td><td colspan="14">Unique identifier of the object in the [ISO/IEC 7816-15] structure’</td></tr>
<tr><td rowspan="13">Type</td><td rowspan="9">pwdFlags</td><td colspan="7"><b>Flag</b></td><td colspan="7"><b>Value</b></td></tr>
<tr><td colspan="7">case-sensitive</td><td colspan="7">True = direct presentation</td></tr>
<tr><td colspan="7">local</td><td colspan="7">True = local PIN</td></tr>
<tr><td colspan="7">change-disabled</td><td colspan="7">False = change is allowed
True = change is not allowed</td></tr>
<tr><td colspan="7">unblock-disabled</td><td colspan="7">False = unblock is allowed
True = unblock is not allowed</td></tr>
<tr><td colspan="7">needs-padding</td><td colspan="7">False (no padding needed)</td></tr>
<tr><td colspan="7">unblockingPassword</td><td colspan="7">False (application PIN)</td></tr>
<tr><td colspan="7">soPassword</td><td colspan="7">False (not an administrator PIN)</td></tr>
<tr><td colspan="7">exchangeRefData</td><td colspan="7">True = change is possible with OLD || NEW reference value</td></tr>
<tr><td>pwdType</td><td colspan="14">Describe the type of the password always equals to ENUMERATED {bcd, ascii-numeric, utf8, half-nibble-bcd, iso9564-1, ...}</td></tr>
<tr><td>minLength</td><td colspan="14">Minimum length of the password accepted by the eIDAS token</td></tr>
<tr><td>maxLength</td><td colspan="14">Maximum length of the password accepted by the eIDAS token</td></tr>
<tr><td>pwdReference</td><td colspan="14">Reference used in VERIFY command to reference the PIN in application</td></tr>
</table>

195 **Figure 33 - EF.AOD: eSign PIN authentication object structure**

- eSignBio authentication object is defined through the following fields [OPTIONAL]

Refer to [TR_Physical_Authentication] ANNEX B

- eSignBio1Ton authentication object is defined through the following fields [OPTIONAL], on one container referencing many physical user credentials.

Refer to [TR_Physical_Authentication] ANNEX B

3.8.3.4 EF.PrKD

205 This file is mandatory and lists all the private key objects; a private key object is characterized by the following attributes:

<table>
<tr><th>Attributes</th><th>Item</th><th colspan="14">Description</th></tr>
<tr><td rowspan="9">Common</td><td>label</td><td colspan="14">An ASCII descriptive string (e.g. “eSignPrivKey1”)</td></tr>
<tr><td rowspan="2">flags</td><td colspan="8"><b>Flag</b></td><td colspan="6"><b>Value</b></td></tr>
<tr><td colspan="8">private</td><td colspan="6">True = authentication is required before access</td></tr>
<tr><td rowspan="5">accessControlRules</td><td rowspan="2">APDU</td><td colspan="11">Accessmode</td><td rowspan="2">securityConditions</td><td rowspan="2">CommunicationMode
[OPTIONAL]</td></tr>
<tr><td>Read(0)</td><td>Update(1)</td><td>Execute(2)</td><td>Delete(3)</td><td>Attribute(4)</td><td>Pso_cds(5)</td><td>Pso_verif(6)</td><td>Pso_dec(7)</td><td>Pso_enc(8)</td><td>Int_auth(9)</td><td>Ext_auth(10)</td></tr>
<tr><td>GENERATE KEY PAIR</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td><td rowspan="3">BIT STRING { contact (0), contactLess (1) }</td></tr>
<tr><td>PSO COMPUTE DIGITAL SIGN.</td><td></td><td></td><td>1</td><td></td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
<tr><td>MANAGE DATA</td><td></td><td></td><td>1</td><td></td><td>1</td><td></td><td></td><td></td><td></td><td></td><td></td><td>SM and Role xxx And UserConsent</td></tr>
<tr><td>currentLCS</td><td colspan="14">ENUMERATED { init(1), op-activated(2), op-deactivated(3), termination(4)}</td></tr>
<tr><td rowspan="12">Class</td><td>Id</td><td colspan="14">Unique identifier of the object in the [ISO/IEC 7816-15] structure</td></tr>
<tr><td rowspan="2">usage</td><td colspan="14">(2)Sign</td></tr>
<tr><td colspan="14">(9)nonRepudiation (optional)</td></tr>
<tr><td>native</td><td colspan="14">True</td></tr>
<tr><td rowspan="6">accessFlags</td><td colspan="6"><b>Flag</b></td><td colspan="8"><b>Value</b></td></tr>
<tr><td colspan="6">(0)sensitive</td><td colspan="8">true</td></tr>
<tr><td colspan="6">(1)extractable</td><td colspan="8">false</td></tr>
<tr><td colspan="6">(2)alwaysSensitive</td><td colspan="8">true</td></tr>
<tr><td colspan="6">(3)neverExtractable</td><td colspan="8">false</td></tr>
<tr><td colspan="6">(4)cardGenerated</td><td colspan="8">True = on board key generated</td></tr>
<tr><td>keyReference</td><td colspan="14">Reference used in cryptographic function to reference the private key</td></tr>
<tr><td>algReference</td><td colspan="14">contains list of applicable algorithms</td></tr>
<tr><td rowspan="5">Type</td><td colspan="15">RSA</td></tr>
<tr><td>modulus</td><td colspan="14">Length of the modulus of RSA key</td></tr>
<tr><td>keyinfo</td><td colspan="14">N/A</td></tr>
<tr><td colspan="15">EC</td></tr>
<tr><td>keyinfo</td><td colspan="14">Parameters</td></tr>
</table>

Figure 34 - EF.PrKD file structure

3.8.3.5 EF.CD

Each EF.CD (at least one is mandatory, others are operational) indicates the certificate CIOs (e.g. User Certificate associated with the signature key).

These files describe one certificate object per file, a certificate key object is characterized by attributes is provided for pre issuance personalization:

Attributes	**Item**	**Description**													
Common	label	**An ASCII descriptive string (e.g. “**eSignCertificate1**”)**													
	accessControlRules	APDU	Accessmode											securityConditions	CommunicationMode [OPTIONAL]
			Read(0)	Update(1)	Execute(2)	Delete(3)	Attribute(4)	Pso_cds(5)	Pso_verif(6)	Pso_dec(7)	Pso_enc(8)	Int_auth(9)	Ext_auth(10)		
		READ	1											ALWAYS	BIT STRING { contact (0), contactLess (1) }
		UPDATE		1										SM and Role xxx And UserConsent	
	currentLCS	ENUMERATED { init(1), op-activated(2), op-deactivated(3), termination(4)}													
Class	Id	Unique identifier of the object in the [ISO/IEC 7816-15] structure													
Type	Path	filename													

Figure 35 - EF.CD file structure

3.8.3.6 EF.DCOD

This file describes one reference to file used in the interoperability scope:

Attributes	**Item**	**Description**													
Common	label	**An ASCII descriptive string (e.g. “**EF.INFO4CERT**”)**													
	accessControlRules	APDU	Accessmode											securityConditions	CommunicationMode [OPTIONAL]
			Read(0)	Update(1)	Execute(2)	Delete(3)	Attribute(4)	Pso_cds(5)	Pso_verif(6)	Pso_dec(7)	Pso_enc(8)	Int_auth(9)	Ext_auth(10)		
		READ	1											SM and Role xxx And UserConsent	BIT STRING { contact (0), contactLess (1) }
		UPDATE		1										SM and Role xxx And UserConsent	
	currentLCS	op-activated(2)													
Class	Id	Unique identifier of the object in the [ISO/IEC 7816-15] structure													
Type	Path	filename													

Figure 36 - EF.DCOD file structure

4 : Basic Signature application

This variant of the [eSign application] provides basic signature functionality:

a. Issuance of the eIDAS token with or without signature key,
b. Key generation after issuance of the eIDAS token,
c. Signature generation.

4.1 DF.eSign

The eIDAS token SHALL contain an eSign application according to [EN419212-1]. The application SHALL contain the following data objects (see also Section 4.8):

- a local PIN object PIN.QES holding the eSign-PIN,
- a local private key object PrK.QES holding the private key of the signature key pair,
- a elementary file holding the certificate issued for the public key of the signature key pair.

The application MAY contain additional elementary files according to [EN419212-1]. The DF.eSign SHALL have the Application Identifier (see [EN419212-1])

```
A0 00 00 01 67 45 53 49 47 4E.
```

4.2 DF.CIA

The eIDAS token SHALL contain a Cryptographic Information Application (see [ISO/IEC 7816-15]) describing the objects of the eSign application, including references for the objects. The DF.CIA SHALL have the Application Identifier (see [ISO/IEC 7816-15]))

```
E8 28 BD 08 0F A0 00 00 01 67 45 53 49 47 4E.
```

4.3 PIN Management

The simple eSign application uses a dedicated (local) eSign-PIN for signature creation.

- The eSign-PIN SHALL be associated with a retry counter.
- The eIDAS token SHALL grant the right to reset the retry counter after successful performing PACE with the global PUK (see [TR03110-2]).
- The eIDAS token MAY associate the eSign-PIN with a counter restricting the number of times the retry counter can be reset. If this number is reached, the retry counter is expired.

4.4 Key Generation

If a signature key is already present, the corresponding private key and the eSign-PIN MUST be terminated before a new key pair can be generated (see Section 4.6). The generation of a new key pair comprises the following steps:

1) The terminal MUST authenticate as a Signature Terminal using the General Authentication Procedure with Effective Access Right Generate qualified electronic signature and the global PIN

(see [TR03110-2], Section 2.3) as password. If successful, the eIDAS token SHALL grant the right to set a new eSign-PIN.

2) The terminal SHALL set a new eSign-PIN (see Section 4.9.2).
3) The terminal (usually a remote terminal operated by a provider for qualified certificates) SHALL authenticate as Authentication Terminal using the General Authentication Procedure with Effective Access Right Install qualified certificate. If successful, the eIDAS token SHALL grant the right to generate a new key pair and to write the newly generated certificate to the eSign application.
4) The terminal SHALL generate a new key pair (see Section 4.9.5) and store the certificate issued for the public key of the generated key pair in the eSign application.

4.5 Signature Generation

The following procedure SHALL be used to generate a signature. As a precondition, a non-terminated eSign-PIN and a non-terminated signature key PrK.QES MUST be present on the eIDAS token.

1) The terminal MUST authenticate as a Signature Terminal using the General Authentication Procedure with Effective Access Right Generate qualified electronic signature and the CAN (see [TR03110-2], Section 2.3) as user credential.
2) The eIDAS token SHALL verify the eSign (see Section 4.9.1). If verification was successful, the eIDAS token SHALL grant the right to generate a signature to the terminal.
3) The signature is created (see Section 4.9.4).

4.6 Termination of the eSign PIN or signature keys

The following procedure SHALL be used to terminate the eSign PIN or signature keys. As a precondition, a non-terminated eSign-PIN MUST be present on the eIDAS token.

1) The terminal MUST authenticate as a Signature Terminal using the General Authentication Procedure with Effective Access Right Generate qualified electronic signature and the CAN (see [TR03110-2], Section 2.3) as user credential.
2) The eIDAS token SHALL verify the eSign (see Section 4.9.1). If verification was successful, the eIDAS token SHALL grant the right to generate a signature to the terminal.
3) To delete the eSign-PIN, the terminal terminates the eSign-PIN. To delete the signature private key, the terminal terminates the PrK.QES (see Section 4.9.6).

4.7 Signature Terminal

The following Object Identifier SHALL be used to indicate the terminal type Signature Terminal for the General Authentication Procedure (see [TR03110-2]):

```
id-ST OBJECT IDENTIFIER ::= {id-roles 3}
```

The relative authorization of the certificate holder is encoded in one byte which is to be interpreted as binary bit map as shown below. In more detail, this bit map contains a role and access rights. Both are relative to the authorization of all previous certificates in the chain.

7	6	5	4	3	2	1	0	Description
x	x	-	-	-	-	-	-	**Role**

1	1	-	-	-	-	-	-	CVCA
1	0	-	-	-	-	-	-	DV (Supervisory Body)
0	1	-	-	-	-	-	-	DV (Certification Service Provider)
0	0	-	-	-	-	-	-	Signature Terminal
-	-	x	x	x	x	x	x	**Access Rights (eSign)**
-	-	x	x	x	x	-	x	RFU
-	-	-	-	-	-	1	-	Generate qualified electronic signature

4.8 Access Conditions

The following notations are used in this section:

- PIN, CAN, PUK denote the global passwords according to [TR03110-2], Section 2.2;
- eSign-PIN denotes the local PIN of the eSign application (see Section 4.3);
- ST denotes a terminal authenticated as Signature Terminal with effective access right Generate qualified electronic signature using the General Authentication Procedure.
- AT denotes a terminal authenticated as Authentication Terminal with effective access right Install qualified certificate using the General Authentication Procedure.

The eIDAS token MUST reject selection of DF.eSign and DF.CIA unless the terminal is authenticated as ST or AT.

The objects of the eSign application SHALL have the following access conditions.

Command	**Object**	**Command Data**	**Access Conditions**			
			PACE Password	**Terminal Type**	**PrK.QES**	**PIN.QES**
CHANGE REFERENCE DATA	PIN.QES	eSign-PIN (old || new)	CAN	ST		
CHANGE REFERENCE DATA	PIN.QES	eSign-PIN (new)	PIN	ST	terminated	terminated
RESET RETRY COUNTER	PIN.QES		PUK	ST	operational	resetable*
TERMINATE	PIN.QES		PIN	ST		
TERMINATE	PrK.QES		PIN	ST		terminated
GENERATE ASYMMETRIC KEY	PrK.QES		PIN	AT	terminated	operational
VERIFY	PIN.QES	eSign-PIN	CAN	ST		operational
PSO: COMPUTE DIGITAL SIGNATURE	PrK.QES	Hash	CAN	ST	operational	verified

*The maximum number of resets of the retry counter must not be reached.

4.9 Application Protocol Datagram Units

The APDUs for the eSign application follow the specifications [ISO/IEC 7816-4], . [ISO/IEC 7816-8] and [EN419212-1].

Before using the APDUs described in this section, the terminal MUST select DF.eSign.

4.9.1 Verify

The following command SHALL be used to verify the eSign-PIN.

Command		
CLA	0x0C	Secure Messaging with authenticated header, no chaining
INS	0x20	VERIFY
P1	0x00	
P2		Reference of the eSign-PIN
Data		eSign-PIN or empty
Response		
Data		Empty
Status Bytes	0x9000	*Normal operation* Verification succesfull.
	0x63CX	*Verification failed* X denotes the number of remaining tries.
	0x6982	*Security status not satisfied* The terminal is not granted the right to perform verification.
	0x6983	*Authentication method blocked* The retry counter of the eSign-PIN has expired.
	0x6984	*Reference data not usable* The eSign-PIN is terminated.
	0x6A88	*Referenced data not found* The referenced PIN does not exist.
	Other	*Operating system specific error* Verification failed.

The eIDAS token MAY indicate a blocked or terminated eSign-PIN with status bytes 0x6982 instead of 0x6983 / 0x6984.

4.9.2 Change Reference Data

The following command SHALL be used to change the eSign-PIN.

Command			
CLA	0x0C	Secure Messaging with authenticated header, no chaining	
INS	0x24	CHANGE REFERENCE DATA	
P1	0x00 or 0x01	See below.	
P2		Reference of the eSign-PIN	
Data		old eSign-PIN || new eSign-PIN new eSign-PIN	(P1 = 0x00) (P1 = 0x01)
Response			
Data	Empty		
Status Bytes	0x9000	*Normal operation* The eSign-PIN was changed.	
	0x6982	*Security status not satisfied*	

		The terminal is not authorized to change the eSign-PIN.
	0x6983	*Authentication method blocked* The retry counter of the eSign-PIN has expired.
	0x6984	*Reference data not usable* • The private key PrK.QES is not terminated (P1 = 0x01 only) • The eSign-PIN is terminated (P1 = 0x00 only)
	0x6A80	*Incorrect parameters in data field* Incorrect data in the data field.
	0x6A88	*Referenced data not found* The referenced PIN does not exist.
	Other	*Operating system specific error* *The eSign-PIN could not be changed.*

The eIDAS token MAY indicate a blocked or terminated eSign-PIN with status bytes 0x6982 instead of 0x6983 / 0x6984.

4.9.3 Reset Retry Counter

The following command SHALL be used to reset the retry counter of the eSign-PIN.

Command		
CLA	0x0C	Secure Messaging with authenticated header, no chaining
INS	0x2C	RESET RETRY COUNTER
P1	0x03	Do not set a new PIN
P2		Reference of the eSign-PIN
Data	Empty	
Response		
Data	Empty	
Status Bytes	0x9000	*Normal operation* The eSign-PIN is unblocked.
	0x6982	*Security status not satisfied* The terminal is not authorized to unblock the eSign-PIN.
	0x6984	*Reference data not usable* The retry counter has expired.
	0x6A88	*Referenced data not found* The referenced PIN does not exist.
	Other	*Operating system specific error* *The eSign-PIN could not be unblocked.*

4.9.4 PSO:Compute Digital Signature

The following command SHALL be used to generate a signature.

Command		
CLA	0x0C	Secure Messaging with authenticated header, no chaining
INS	0x2A	PERFORM SECURITY OPERATION
P1	0x9E	COMPUTE DIGITAL SIGNATURE
P2	0x9A	„Data to be signed“
Data		Hash value of the data to be signed
Response		
Data		Signature
Status Bytes	0x9000	*Normal operation* Signature created.
	0x6982	*Security status not satisfied*

		- The terminal is not authorized to generate a signature - The eSign-PIN is not verified
	0x6984	*Reference data not usable* The signature key QES.PrK is terminated.
	0x6A80	*Incorrect parameters in data field* Incorrect data in the data field.
	Other	*Operating system specific error* *Signature creation failed.*

The eIDAS token MAY indicate a terminated signature key with status bytes 0x6982 instead of 0x6984.

4.9.5 Generate Asymmetric Key Pair

The following command SHALL be used to generate a new signature key pair PrK.QES, PuK.QES.

Command		
CLA	0x0C	Secure Messaging with authenticated header, no chaining
INS	0x47	GENERATE ASYMMETRIC KEY PAIR
P1	0x82	Generate key pair and return Public Key after the Extended Headerlist.
P2	0x00	
Data	0xB6	Digital Signature Template with reference of the signature key (tag 0x84).
	0x7F49	Parameters for the key to be generated (OPTIONAL).
Response		
Data	0x7F49	Public Key Data Object
Status	0x9000	*Normal operation* The key pair was generated.
	0x6982	*Security status not satisfied* The terminal is not authorized to generate a key pair.
	0x6984	*Reference data not usable* - The signature key PrK.QES is not terminated. - No eSign-PIN is set.
	0x6A88	*Referenced data not found* The referenced key does not exist.
	Other	*Operating system specific error* *Key pair generation failed.*

The eIDAS token MAY indicate a non-terminated signature key or a terminated eSign-PIN with status bytes 0x6982 instead of 0x6984.

4.9.6 Terminate

The following command SHALL be used to terminate the eSign-PIN or the signature key PrK.QES.

Command			
CLA	0x0C	Secure Messaging with authenticated header, no chaining	
INS	0xE6	TERMINATE	
P1	0x10 or 0x21	Reference of the eSign-PIN in parameter P2 Key	
P2	 0x00	Reference of the eSign-PIN	(P1 = 0x10) (P1 = 0x21)
Data		Empty	(P1 = 0x10)

	0xB6	Digital Signature Template with reference of the signature key (tag 0x84).	(P1 = 0x21)
Response			
Data	–	Empty	
Status Bytes	0x9000	*Normal operation* eSign-PIN / signature key terminated.	
	0x6982	*Security status not satisfied* The terminal is not authorized to terminate the eSign-PIN / the signature key	
	0x6984	*Reference data not usable* Key termination only: the eSign-PIN is not terminated.	
	0x6A88	*Referenced data not found* The referenced eSign-PIN / signature key does not exist.	
	Other	*Operating system specific error* *The termination of the eSign-PIN / the signature key failed.*	

The eIDAS token MAY indicate a non-terminated eSign-PIN with status bytes 0x6982 instead of 0x6984.

Annex A: Normative references

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

ISO/IEC 7816-6 - Identification cards -- Integrated circuit cards -- Part 6: Interindustry data elements for interchange

ISO/IEC 7816-8 - Identification cards -- Integrated circuit cards -- Part 8: Commands for security operations

ISO/IEC 7816-9 - Identification cards -- Integrated circuit cards -- Part 9: Commands for card management

ISO/IEC 7816-15 - Identification cards -- Integrated circuit cards -- Part 15: Cryptographic information application

EN 419212-1 Application Interface for smart cards used as Secure Signature Creation Devices

[TR03110-1] Technical Guideline TR-03110 v2.20 part 1 - Advanced Security Mechanisms for Machine Readable Travel Documents – eMRTDS with BAC/PACEv2 and EACv1

[TR03110-2] Technical Guideline TR-03110 v2.20 part 2 - Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token – Protocols for electronic Identification, Authentication and trust Services (eIDAS)

[TR03110-3] Technical Guideline TR-03110 v2.20 part 3 – Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token - Common Specifications

[TR03110-4] Technical Guideline TR-03110 v2.20 part 4 - Advanced Security Mechanisms for Machine Readable Travel Documents and eIDAS Token – Applications and document Profiles

[ICAO 9303], Machine Readable Travel Documents - Part 1: Machine Readable Passport, Specifications for electronically enabled passports with biometric identification capabilities (including supplement), ICAO Doc 9303, 2006

[TR_Physical_Authentication], Technical Report Physical Authentication v1.0, 2014/12/18,

350 Annex B: Other references

EC_Regulation - REGULATION (EU) No 910/2014 OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL of 23 July 2014 on electronic identification and trust services for electronic transactions in the internal market and repealing Directive 1999/93/EC