

# **MyEID PKI JavaCard Applet**

**Reference manual  
Ver. 1.7.7**

## Contents

<b>Introduction .....</b>	<b>3</b>
References.....	3
<b>1.1.1. Abbreviations.....</b>	<b>4</b>
<b>Contents and File Structure.....</b>	<b>5</b>
<b>Command Interface.....</b>	<b>5</b>
3.1 <i>Command lists</i> .....	5
3.2 <i>Command response status bytes</i> .....	7
<b>Regular Commands .....</b>	<b>8</b>
4.1 <i>GET DATA</i> .....	8
4.2 <i>SELECT FILE</i> .....	10
File Security Attributes.....	12
4.3 <i>GET RESPONSE</i> .....	14
4.4 <i>READ BINARY</i> .....	15
4.5 <i>UPDATE BINARY</i> .....	15
4.6 <i>ERASE BINARY</i> .....	16
4.7 <i>VERIFY</i> .....	16
Blocked PIN .....	17
PIN locking .....	17
4.8 <i>CHANGE REFERENCE DATA</i> .....	18
4.9 <i>RESET RETRY COUNTER</i> .....	19
4.10 <i>MANAGE SECURITY ENVIRONMENT: RESTORE</i> .....	20
4.11 <i>MANAGE SECURITY ENVIRONMENT: SET</i> .....	20
4.12 <i>PERFORM SECURITY OPERATION: COMPUTE DIGITAL SIGNATURE</i> .....	22
4.13 <i>PERFORM SECURITY OPERATION: DECIPHER</i> .....	23
<b>Personalisation and Management Commands .....</b>	<b>24</b>
5.1 <i>PUT DATA: INITIALISE APPLET</i> .....	24
5.2 <i>PUT DATA: INITIALISE PIN</i> .....	25
5.3 <i>ACTIVATE APPLET</i> .....	26
5.4 <i>CREATE FILE</i> .....	26
5.5 <i>DELETE FILE</i> .....	28
5.6 <i>GENERATE PUBLIC KEY PAIR</i> .....	29
5.7 <i>PUT DATA: LOAD KEY</i> .....	30

## Introduction

This document describes the MyEID PKI JavaCard applet, context and command interface.

The command set of the applet is based on ISO/IEC 7816-4 and ISO/IEC 7816-8 standards. The application related data is in files according to the PKCS #15 v1.0 specification. The applet implements the FINEID S1 v1.12 specification and it can be configured to include all or any subset of the features specified in FINEID S4-1 or S4-2 specifications. Starting from version 2.2.0 the applet supports both 1024 and 2048 bit RSA keys. From version 3.0.0 (MyEID3) the applet supports keys from 512 bit up to 2048 bit in increments of 64 bits.

The applet is fully compatible with JavaCard 2.1.1 and GlobalPlatform 2.0.1 specifications. The applet can be set as the default applet on the card and in that case it will be automatically selected after the JavaCard has been powered up.

The MyEID version 3 (MyEID3) uses the JavaCard 2.2.1 and GlobalPlatform 2.1.1 platform. It supports RSA keys from 512 up to 2048 bits in 64 bit increments. MyEID3 supports file sizes up to 32767 bytes and 14 different PIN-codes can be created and used. The number of RSA keys is only limited by the available memory and maximum numbers of files (see *PUT DATA: INITIALISE APPLLET*).

## References

The most relevant specifications and standards are:

- ISO/IEC 7816-4
- ISO/IEC 7816-8
- ISO/IEC 7816-9
- JavaCard 2.1.1, MyEID3: 2.2.1
- GlobalPlatform 2.0.1 ' (Open Platform), MyEID3: GlobalPlatform 2.1.1
- FINEID S1 and S4 documentation

**1.1.1. Abbreviations**

AC	Access Condition
AID	Application Identifier
APDU	Application Protocol Data Unit
ASN.1	Abstract Syntax Notation One
CRDO	Control Reference Data Object
CLA	Class Byte
DF	Dedicated File
EF	Elementary File
FCI	File Control Information
FID	File Identifier
LC	Length of APDU command data
MF	Master File
MSE	Manage Security Environment
P1/P2	APDU command parameter 1 and 2
PIN	Personal Identification Number
PSO	Perform Security Operation
RFU	Reserved for Future Use
SE	Security Environment
SW	Status Word

## Contents and File Structure

The applet follows the PKCS #15 specification and implements all or any subset of files and contents of FINEID S4-1, S4-2, or PKCS#15 specifications.

The applet can contain the following objects:

- private RSA keys,
- public RSA keys,
- authentication objects,
- card holder certificates,
- trusted certificates, and
- any data objects

## Command Interface

This chapter describes the commands of the applet with their parameters.

The commands are split into two categories:

- regular commands
- personalisation and management commands

The regular commands are compatible with the FINEID S1 v1.12 specifications that comply with the ISO/IEC 7816-4 and 7816-8 standards.

The personalisation and management commands comply with the ISO/IEC 7816-4 and 7816-9 standards.

### 3.1 Command lists

Regular commands:

**Table 1** - Regular usage related commands of the MyEID applet.

Command	Standard	Functionality
GET DATA	ISO/IEC 7816-4	Retrieves applet version and other information
SELECT FILE	ISO/IEC 7816-4	Selects the applet or a file within the applet.
GET RESPONSE	ISO/IEC 7816-4	Reads card response in T=0 protocol mode.
READ BINARY	ISO/IEC 7816-4	Reads contents from a transparent (binary) file.
UPDATE BINARY	ISO/IEC 7816-4	Updates contents in a transparent (binary) file.
ERASE BINARY	ISO/IEC 7816-4	Erases contents in a transparent (binary) file.

VERIFY	ISO/IEC 7816-4	Verifies reference data presented by the user (PIN) with the reference data stored in the applet.  The current verification status can be queried with this command.
CHANGE REFERENCE DATA	ISO/IEC 7816-8	Changes the current reference data (PIN).
RESET RETRY COUNTER	ISO/IEC 7816-8	Unblocks a blocked reference data (PIN).
MANAGE SECURITY ENVIRONMENT: <b>RESTORE</b>	ISO/IEC 7816-8	Restores a predefined or empty security environment.
MANAGE SECURITY ENVIRONMENT: <b>SET</b>	ISO/IEC 7816-8	Sets security environment parameters that will be used with subsequent PSO command(s).
PERFORM SECURITY OPERATION: <b>COMPUTE DIGITAL SIGNATURE</b>	ISO/IEC 7816-8	Computes a digital signature using a private key. The algorithm and the key are selected with the preceding MSE command.
PERFORM SECURITY OPERATION: <b>DECIPHER</b>	ISO/IEC 7816-8	Decrypts data with a private key. The algorithm and the key are selected with the preceding MSE command.

Personalisation and management commands:

**Table 2** – MyEID applet personalisation and management related commands.

Command	Standard	Functionality
PUT DATA: <b>LOAD AUID</b>	ISO/IEC 7816-4	Stores a unique identifier in the applet.
PUT DATA: <b>INITIALISE APPLET</b>	ISO/IEC 7816-4	Initialises the file system and pre-creates the permanent files (MF and DF 5015 (PKCS#15)).
PUT DATA: <b>INITIALISE PIN</b>	ISO/IEC 7816-4	Initialises the requested PIN reference data and its unblocking reference data (PUK).
ACTIVATE APPLET	ISO/IEC 7816-9	Activates the applet. Sets files to Activated State.
CREATE FILE	ISO/IEC 7816-9	Creates a file in the applet's file system under the current DF.
DELETE FILE	ISO/IEC 7816-9	Deletes the current file.
GENERATE KEY PAIR	ISO/IEC 7816-8	Generates an RSA key pair in the current key EF. *
PUT DATA: <b>LOAD KEY PAIR</b>	ISO/IEC 7816-4	Loads an externally generated RSA key pair to a key EF. *

\* Applet version 2.2.0 onwards also supports 2048 bit keys. Earlier versions support only 1024 bit keys. MyEID3 supports key sizes from 512 to 2048 bit in 64 bit increments.

### 3.2 Command response status bytes

**Table 3** – General status bytes

<b>SW1 / SW2</b>	<b>Functionality</b>
0x9000	Command successful
0x6700	Incorrect data length
0x6981	Incorrect file type
0x6982	Security status not satisfied
0x6984	Invalid data
0x6985	Conditions not satisfied
0x6A86	Incorrect P1 and/or P2 parameters
0x6D00	Unsupported command instruction

## Regular Commands

### 4.1 GET DATA

The GET DATA command retrieves information about the applet and its current state. The type of the returned information depends on parameter P2.

**Table 4** - GET DATA command APDU

Bytes	Value
CLA	00h
INS	CAh
P1	01h
P2	Get Key File information (a key file must be selected): 00h: algorithm identifier, see Table 6. 01h: modulus, see Table 7. 02h: public exponent, see Table 8.  Get other information: A0h: MyEID Applet information, see Table 9. A1h: current DF file list, see Table 10. A2h: same as A1h, but only includes EF's A3h: same as A1h, but only includes DF's
LC	Empty
Data	Empty
LE	Empty or length of requested data

**Table 5** - GET DATA response APDU

Bytes	Value
Data	Requested data, see details in tables below
SW1 – SW2	Status Bytes

The GET DATA response command APDU data holds specific information about the applet and the current DF.

**Table 6** – Algorithm Identifier

Bytes	Length	Value
Algorithm Identifier	2	9200h: RSA CRT
Modulus size	2	XXXXh: length of modulus in bits
Public exponent size	2	XXXXh: length of public exponent in bits

**Table 7** – Modulus

Bytes	Length	Value
Modulus	N	N = length of modulus in bytes

**Table 8** – Public Exponent

Bytes	Length	Value
Public Exponent	N	N = length of public exponent in bytes

**Table 9** – Applet Information

Bytes	Length	Value
Name	5	"MyEID"
Major version	1	XXh: major version number
Minor version	1	XXh: minor version number
Version revision	1	XXh: revision number
Unique identifier	10	Unique identifier, different from card to card
Change counter	2	XXXXh: Incremented after every successful command that changes the contents of the applet.

**Table 10** – Current DF file list

Bytes	Length	Value
First FID	2	XXXXh: FID of the first file in current DF
Second FID	2	XXXXh: FID of the second file in current DF
...	...	...
Nth FID	2	XXXXh: FID of the Nth file in current DF

## 4.2 SELECT FILE

The SELECT FILE command selects the applet itself with the Application Identifier (AID) or a file within the applet. A file can be selected by a single File Identifier (FID) or by relative or absolute path. A path consists of a sequence of FID's.

**Table 11** - SELECT FILE command APDU

Byte	Value
CLA	00h
INS	A4h
P1	00h: select EF, DF or MF by single FID 04h: select applet by AID or a DF by its name 08h: select file by absolute path from MF 09h: select file by relative path current DF
P2	00h: FCI returned in response
LC	00h or length of data
Data	P1 = 00h: EF, DF or MF FID (or empty for MF) P1 = 04h: AID value (for applet or DF name) P1 = 08h: absolute path from MF without the FID for MF P1 = 09h: relative path from the current DF without the FID of current DF
LE	Empty

**Table 12** - SELECT FILE response APDU

Byte	Value
Data	File Control Information (FCI), empty if protocol is T=0, or on error
SW1 – SW2	Status Bytes  0x6A82 – File not found

The SELECT FILE response command APDU data holds specific information about the selected file.

**Table 13** - MF, DF, and Applet File Control Information data

Byte	Length	Value
FCI tag	1	6Fh
Length	1	17h – 2Fh
File Size tag	1	81h

Length	1	02h
File Size	2	XXXXh: available free space for additional files
File Description tag	1	82h
Length	1	01h
File Descriptor	1	38h: DF or MF
File Identifier tag	1	83h
Length	1	02h
File Identifier	2	XXXXh: FID
Security Attributes tag	1	86h
Length	1	03h
Security Attributes	3	XXXXXXh: See Table 16 for details of MF See Table 17 for details of DF
Proprietary Information tag	1	85h: File status byte
Length	1	02h
Proprietary Information	2	0000h: normal file 0002h: permanent file; DF cannot be deleted with DELETE FILE command.
Life Cycle Status tag	1	8Ah
Length	1	01h
Life Cycle Status	1	X1h: creation state X7h: operational state – activated  The high nibble is RFU
DF Name tag	1	84h: Optional tag
Length	1	01h – 10h
DF Name	1-16	

**Table 14** - EF File Control Information data

Byte	Length	Value
FCI tag	1	6Fh
Length	1	17h
File Size tag	1	80h
Length	1	02h
File Size	2	XXXXh: File size of transparent EF. Must be 0400h or 0800h* for a private RSA key EF (=key size in bits, or modulus, of the key EF) MyEID3 accepts values from 0200h up to 0800h in 0040h increments.
File Description tag	1	82h
Length	1	01h

File Descriptor	1	01h: transparent EF 11h: private RSA key EF
File Identifier tag	1	83h
Length	1	02h
File Identifier	2	XXXXh: FID
Security Attributes tag	1	86h
Length	1	03h
Security Attributes	3	XXXXXXh: See Table 35 for details of transparent EF See Table 19 for details of private RSA key EF
Proprietary Information tag	1	85h: File status byte
Length	1	02h
Proprietary Information	2	<p>First status byte:</p> <p>Transparent EF 00h: RFU Private RSA key EF X0h: key not valid X1h: key valid X3h: key valid, key generated on card</p> <p>X = AC to be cleared after RSA operation. The remaining bits are RFU for EF Key, and for the other file types the whole byte is RFU.</p> <p>Second status byte:</p> <p>00h: RFU</p>
Life Cycle Status tag	1	8Ah
Length	1	01h
Life Cycle Status	1	<p>X1h: creation state X7h: operational state – activated</p> <p>The high nibble is RFU</p>

\* 0800h is supported from MyEID applet version 2.2.0 onwards.

## File Security Attributes

The operations that can be performed on a file, such as reading and updating, are controlled by File Security Attributes. Every file has six Security Attributes, coded in four bits each, and totalling three bytes. A given Security Attribute indicates which Access Condition must be fulfilled before the operation in question can be performed. The Access Conditions can be fulfilled by executing the VERIFY command with the correct data.

The exact meaning of the six File Security Attributes depends on the file type, as listed in tables 14 to 17. Each Security Attribute is coded in four bits as follows:

**Table 15** - File Security Attribute values

Value	Meaning
0h	The operation is allowed at all times
1h...Eh	Indicates the reference number of the PIN that must be valid before the operation can be performed
Fh	The operation is forbidden

**Table 16** - MF Security Attributes

Position	Meaning
X00000h	Create DF's
0X0000h	Create EF's
00X000h	Recreate MF (removes all existing files)
000X00h	RFU
0000X0h	RFU
00000Xh	RFU

**Table 17** - DF Security Attributes

Position	Meaning
X00000h	Create DF's
0X0000h	Create EF's
00X000h	Delete File (this file, with complete sub-tree)
000X00h	RFU
0000X0h	RFU
00000Xh	RFU

**Table 18** - Transparent EF Security Attributes

Position	Meaning
X00000h	Read
0X0000h	Update / Erase
00X000h	Delete File (this file)
000X00h	RFU
0000X0h	RFU
00000Xh	RFU

**Table 19** - RSA Key EF Security Attributes

Position	Meaning
X00000h	Use
0X0000h	Put Data
00X000h	Delete File (this file)
000X00h	Generate
0000X0h	RFU
00000Xh	RFU

### 4.3 GET RESPONSE

The GET RESPONSE command returns the response of the APDU when T=0 protocol is used. The response can only be retrieved once, and the GET RESPONSE command must be executed immediately after the command that generated the response.

This command retrieves the response data of the following commands:

- SELECT FILE,
- PSO: COMPUTE DIGITAL SIGNATURE, and
- PSO: DECIPHER

**Table 20** - GET RESPONSE command APDU

Byte	Value
CLA	00h
INS	C0h
P1	00h
P2	00h
LC	Empty
Data	Empty
LE	Data length of the response to retrieve

**Table 21** - GET RESPONSE response APDU

Byte	Value
Data	Previous APDU commands response data. Empty on error
SW1 – SW2	Status Bytes

#### 4.4 READ BINARY

The READ BINARY command reads data from transparent EF's. The file to read must be selected first with the SELECT FILE command.

**Table 22** - READ BINARY command APDU

Byte	Value
CLA	00h
INS	B0h
P1	00-7Fh: MSB of the 15 bit offset to the first byte to read
P2	00-FFh: LSB of the 15 bit offset to the first byte to read
LC	Empty
Data	Empty
LE	Number of bytes to read.

**Table 23** - READ BINARY response APDU

Byte	Value
Data	Bytes read, or empty on error
SW1 – SW2	Status Bytes

#### 4.5 UPDATE BINARY

The UPDATE BINARY command updates data in transparent EF's. The file to update must be selected first with the SELECT FILE command.

**Table 24** - UPDATE BINARY command APDU

Byte	Value
CLA	00h
INS	D6h
P1	00-7Fh: MSB of the 15 bit offset to the first byte to update
P2	00-FFh: LSB of the 15 bit offset to the first byte to update
LC	Length of data to update
Data	Data to be written
LE	Empty

**Table 25** - UPDATE BINARY response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

## 4.6 ERASE BINARY

The ERASE BINARY command erases bytes starting at the requested offset until the end of the transparent EF. The file to erase must be selected first with the SELECT FILE command.

**Table 26** - ERASE BINARY command APDU

Byte	Value
CLA	00h
INS	0Eh
P1	00-7Fh: MSB of the 15 bit offset to the first byte to erase
P2	00-FFh: LSB of the 15 bit offset to the first byte to erase
LC	Empty
Data	Empty
LE	Empty

**Table 27** - ERASE BINARY response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 28** - Response Status Byte values

SW1 – SW2	Description
0x9000	Verification successful and/or AC acquired
0x6985	PIN locked – <i>CONDITIONS NOT SATISFIED</i>
0x6983	Verification failed and no number of retries left PIN blocked.
0x63CX	Verification failed and X number of retries left.

## 4.7 VERIFY

The VERIFY command is used to authenticate the user. The verification data (PIN) is compared internally with the reference data stored in the applet. A successful verification sets the Access Condition (AC) in parameter P2, which enables the execution of commands that are restricted by that AC in the File Security Attributes of the file in question. This command can also be used just to inquire the status of the given Access Condition. This is achieved by executing the command without any reference data.

Verification can fail if the presented PIN is incorrect, or the PIN is blocked or locked.

## Blocked PIN

Each time the PIN is verified with this command, a retry counter connected to this PIN is updated. If the verification was successful, the counter is reset to its original value of 5 (or optionally a value that has been configured when the PIN was created). If the verification fails, the counter will be decremented. If the counter reaches zero, the PIN will be blocked. To unblock it, a Reset Retry Counter command must be executed with the correct PUK (PIN Unblocking Code) and a new PIN. This also resets the retry counter to the original value.

## PIN locking

Besides blocking, it is also possible to configure the PIN in such a way that it can be locked. The locking can be set to occur in two situations: After the creation of the PIN, and after unblocking the PIN with the Reset Retry Counter command. A locked PIN can be unlocked by changing it with the Change Reference Data command. This optional feature can be used to enforce the changing of an initial PIN when user receives his card. This can be desirable if all the cards initially have a constant PIN. A locked PIN will cause the Verify command to fail with the error code SW1SW2 = 0x6985 (*CONDITIONS NOT SATISFIED*).

**Table 29** - VERIFY command APDU

Byte	Value
CLA	00h
INS	20h
P1	00h
P2	PIN reference number 01h to 03h MyEID3: 01h to 0Eh
LC	00h: No data, just query the status of the PIN 08h: PIN reference data present in data field
Data	Empty or PIN reference data (verification data) padded to 8 bytes. Byte value 00h or FFh can be used for padding.
LE	Empty

**Table 30** - VERIFY response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 31** - Response Status Byte values

SW1 – SW2	Description
0x9000	Verification successful and/or AC acquired
0x6985	PIN locked – <i>CONDITIONS NOT SATISFIED</i>
0x6983	Verification failed and no number of retries left PIN blocked.
0x63CX	Verification failed and X number of retries left.

## 4.8 CHANGE REFERENCE DATA

The CHANGE REFERENCE DATA command replaces the current internal reference data with a new value. The current reference data is first validated with the verification data. Successful validation also removes optional reference data locking.

**Table 32** - CHANGE REFERENCE DATA command APDU

Byte	Value
CLA	00h
INS	24h
P1	00h: exchange reference data
P2	PIN reference number 01h to 03h MyEID3: 01h to 0Eh
LC	10h: Length of data field
Data	Verification reference data padded to 8 bytes, followed by the new reference data padded to 8 bytes. Byte value 00h or FFh can be used for padding.
LE	Empty

**Table 33** - CHANGE REFERENCE DATA response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes: 0x9000 – Command successful 0x6983 – Verification failed and/or no number of retries left PIN blocked. 0x63CX – Verification failed and/or X number of retries left.

## 4.9 RESET RETRY COUNTER

The RESET RETRY COUNTER command unblocks a PIN which has been blocked after too many unsuccessful verification trials. Successful unblocking requires the correct unblocking reference data (PUK).

Successful execution of this command sets the reference data into locked state if the relocking after reset retry counter option is used. The relocking flag is optionally set with the PUT DATA: INITIALISE PIN command.

**Table 34** - RESET RETRY COUNTER command APDU

Byte	Value
CLA	00h
INS	2Ch
P1	00h: reset retry counter and set reference data
P2	PIN reference number 01h to 03h MyEID3: 01h to 0Eh
LC	00h: query status of unblocking reference data 10h: reset retry counter
Data	Empty or unblocking reference data padded to 8 bytes, followed by the new reference data padded to 8 bytes. Byte value 00h or FFh can be used for padding.
LE	Empty

**Table 35** - RESET RETRY COUNTER response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 36** - Response Status Byte values

SW1 – SW2	Description
0x9000	Command successful
0x6985	PIN locked – <i>CONDITIONS NOT SATISFIED</i>
0x6983	Verification failed and no number of retries, PUK blocked.
0x63CX	Verification failed and X number of retries left.

#### 4.10 MANAGE SECURITY ENVIRONMENT: RESTORE

The MANAGE SECURITY ENVIRONMENT: RESTORE command restores an empty or predefined Security Environment (SE).

**Table 37** - MANAGE SECURITY ENVIRONMENT: RESTORE command APDU

Byte	Value
CLA	00h
INS	22h
P1	F3h: Restore SE
P2	00h: SE reference number (00h denotes an empty SE)
LC	Empty
Data	Empty
LE	Empty

**Table 38** - MANAGE SECURITY ENVIRONMENT: RESTORE response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

#### 4.11 MANAGE SECURITY ENVIRONMENT: SET

The MANAGE SECURITY ENVIRONMENT: SET command sets attributes in the current Security Environment (SE).

**Table 39** - MANAGE SECURITY ENVIRONMENT: SET command APDU

Byte	Value
CLA	00h
INS	22h
P1	41h: computation or decipherment
P2	B6h: attributes of DST in data field B8h: attributes of CT in data field
LC	Empty or length of Data field
Data	Empty or concatenation of Control Reference Data Objects (CRDO)
LE	Empty

**Table 40** - MANAGE SECURITY ENVIRONMENT: SET response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 41** - Control Reference Data Objects (CRDO)

Tag	Meaning
80h	XXh: Algorithm Reference (see Table 42 for details)
81h	XXXXh: File Reference; File Identifier (FID) <ul style="list-style-type: none"> <li>- only FIDs shall be used</li> <li>- identifies RSA EF to be used in PSO commands</li> </ul> value according to PKCS#15: private key objects PKCS15Path.path
84h	00h: Key Reference (references the private key in a key file) MyEID supports only one key in each key file

**Table 42** - Values for Algorithm Reference

Algorithm Reference	Meaning
0Xh	No hash algorithm
1Xh	SHA-1 hash algorithm
2Xh	RFU
X0h	Raw RSA operation * No input or output data formatting, padding, or hash encapsulation is applied.  PSO: COMPUTE DIGITAL SIGNATURE: <ol style="list-style-type: none"> <li>1. Input data must be equal to the RSA key modulus length.</li> <li>2. PKCS#1 RSASP1 signature primitive is applied (private key operation).</li> </ol> PSO: DECIPHER <ol style="list-style-type: none"> <li>1. PKCS#1 RSADP decryption primitive is applied (private key operation).</li> </ol>
X1h	RFU

X2h	<p>RSASSA-PKCS1-v1-5 signature scheme According to PKCS#1 v2.0 with RSA algorithm, compatible with PKCS#15 v1.5</p> <p>PSO: COMPUTE DIGITAL SIGNATURE:</p> <ul style="list-style-type: none"> <li>- The data (hash code) is encapsulated into a DigestInfo ASN.1 structure according to the selected hash algorithm. If no algorithm is selected (algorithm reference = 02h) then no encapsulation is done by the applet.</li> <li>- DigestInfo is padded to RSA key modulus length according to PKCS#1 v1.5, block type 01h. Size of the DigestInfo must not exceed 40% of the RSA key modulus length.</li> <li>- PKCS#1 RSASP1 signature primitive is applied (private key operation).</li> </ul> <p>RSAES-PKCS1-v1-5 encryption scheme According to PKCS#1 v2.0 with RSA algorithm, compatible with PKCS#15 v1.5</p> <p>PSO: DECIPHER:</p> <ul style="list-style-type: none"> <li>- PKCS#1 RSADP decryption primitive is applied (private key operation).</li> </ul> <p>PKCS#1 v1.5 padding is removed</p>
-----	---

\* Raw RSA operation is not supported on 2048 bit keys when computing a digital signature.

#### 4.12 PERFORM SECURITY OPERATION: COMPUTE DIGITAL SIGNATURE

The PSO: COMPUTE DIGITAL SIGNATURE command computes a digital signature. The Secure Environment (SE) attributes Algorithm and File Reference must be previously set with MSE: SET command.

**Table 43** - PSO: COMPUTE DIGITAL SIGNATURE command APDU

Byte	Value
CLA	00h
INS	2Ah
P1	9Eh: return digital signature data
P2	9Ah: data field contains data to be signed
LC	Length of data field
Data	<p>SE Algorithm Reference = 00h: *</p> <ul style="list-style-type: none"> <li>- Raw data, data length must be equal to the RSA key modulus length.</li> </ul> <p>SE Algorithm Reference = 02h:</p> <ul style="list-style-type: none"> <li>- DigestInfo data, data is padded by the applet to match the length of the RSA key modulus.</li> </ul> <p>SE Algorithm Reference = 12h:</p> <ul style="list-style-type: none"> <li>- SHA-1 Hash data, the hash is encapsulated into a DigestInfo structure and then padded to the full length of the RSA key modulus.</li> </ul>
LE	Empty or length of response data

\* 00h algorithm reference is not supported on 2048 bit keys.

**Table 44** - PSO: COMPUTE DIGITAL SIGNATURE response APDU

Byte	Value
Data	Digital signature
SW1 – SW2	Status Bytes

### 4.13 PERFORM SECURITY OPERATION: DECIPHER

The PSO: DECIPHER command decrypts an encrypted data. The Secure Environment (SE) attributes Algorithm and File Reference must be set previously with MSE: SET command.

The cryptogram must be presented in two parts when a 2048 bit RSA key is used.

**Table 45** - PSO: DECIPHER command APDU

Byte	Value
CLA	00h
INS	2Ah
P1	80h: return decrypted data
P2	86h: data field contains padding indicator concatenated with the data to be decrypted
LC	81h: Length of data field
Data	Data for smaller than 2048 bit keys: 00h (pad indicator)    cryptogram  Data for 2048 bit keys: 81h (pad and first half indicator)    first half of cryptogram 82h (pad and second half indicator)    second half of cryptogram
LE	Empty or length of response data

**Table 46** - PSO: DECIPHER response APDU

Byte	Value
Data	SE Algorithm Reference = 00h: - Decrypted data is returned including formatting.  SE Algorithm Reference = 02h: - Decrypted data, PKCS#1 v1.5 padding is removed by the applet.
SW1 – SW2	Status Bytes

## Personalisation and Management Commands

### 5.1 PUT DATA: INITIALISE APPLLET

The PUT DATA: INITIALISE APPLLET command initialises the applet's file system. The first time this command is called after the applet has been installed; it also allocates the requested memory space for the file system.

After a successful execution, the applet will be in the personalisation state, and all files will be in the Created State. In the personalisation state all Access Conditions are disabled, i.e. the applet does not check them against the File Security Attributes in the files. In other words, the applet behaves as if all the Security Attributes were set to ALWAYS (00h).

**Table 47** – PUT DATA: INITIALISE APPLLET command APDU

Byte	Value
CLA	00h
INS	DAh
P1	01h
P2	E0h
LC	08h
Data	File system initialisation parameters, see Table 49 for details.
LE	Empty

**Table 48** – PUT DATA: INITIALISE APPLLET response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 49** – File system initialisation parameters

Bytes	Length	Value
File system size		XXXXh
MyEID3: Maximum number of files	2	MyEID3: 0080h to 0200h (128 to 512 files), values outside the range will be changed to the closest allowed value by the applet before execution.
MF ACL	3	See SELECT FILE command for details.
DF 5015 ACL	3	See SELECT FILE command for details.

## 5.2 PUT DATA: INITIALISE PIN

The PUT DATA: INITIALISE PIN command initialises the requested PIN and its unblocking reference data (PUK).

**Table 50** – PUT DATA: INITIALISE PIN command APDU

Byte	Value
CLA	00h
INS	DAh
P1	01h
P2	01h: PIN #1 02h: PIN #2 03h: PIN #3  MyEID3: 01h to 0Eh
LC	10h to 13h
Data	PIN initialisation parameters, see Table 52 for details.
LE	Empty

**Table 51** – PUT DATA: INITIALISE PIN response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

**Table 52** – PIN initialisation parameters

Bytes	Length	Value
PIN reference data	8	Padded with 00h or FFh
Unblocking reference data	8	Padded with 00h or FFh
Optional * PIN retry counter	1	Indicates the number of failed PIN verifications before the PIN is blocked. Default value: 05h = 5 trials
Optional * PUK retry counter	1	Indicates the number of failed PUK verifications before the PUK is blocked. Default value: 0Ah = 10 trials
Optional * PIN locking	1	bit0: 0 = normal mode (default), 1 = PIN is locked after initialisation. PIN will be unlocked after successful execution of the Verify command. bit1: 0 = normal mode (default), 1 = PIN will be locked after a Reset Retry Counter command.

\* Optional field

### 5.3 ACTIVATE APPLLET

The ACTIVATE APPLLET command causes the applet to exit the Personalisation State and enter the Use State. All the files are changed from Created Status to Activated Status. All Access Conditions become active and fully functional.

**Table 53** – ACTIVATE APPLLET command APDU

Byte	Value
CLA	00h
INS	44h
P1	04h: applet AID in data field
P2	00h
LC	01h – 10h: Length of AID (data field)
Data	Applet AID
LE	Empty

**Table 54** – ACTIVATE APPLLET response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

### 5.4 CREATE FILE

The CREATE FILE command creates EF's and DF's into the current DF. After a successful creation the created file is set as the current file. If the applet is in the Use State, the applet is also set to Temporary Personalisation State and the created file is set to Created State.

**Table 55** – CREATE FILE command APDU

Byte	Value
CLA	00h
INS	E0h
P1	00h
P2	00h
LC	19h – 31h: Length of data field
Data	File Control Parameters (FCP), see Table 57 for details.
LE	Empty

**Table 56** – CREATE FILE response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes: 0x6A89 – File already exists

The following table shows the mandatory position and length of each parameter; they must not be rearranged and the only optional parameter is the DF name that can be used when creating DF's.

**Table 57** – File Control Parameters (FCP)

Byte	Length	Value
FCP tag	1	62h
Length	1	17h: EF's 17h - 2Fh: DF's
File Size tag	1	80h: transparent EF 81h: DF and key EF
Length	1	02h
File Size	2	XXXXh: File size of transparent EF 0400h or 0800h*: size of modulus for a RSA key EF 0200h to 0800h: MyEID3, size of modulus for a RSA key EF (in 0040h increments) 0000h: Not applicable for DF's
File Description tag	1	82h
Length	1	01h
File Descriptor	1	01h: transparent EF 11h: RSA key EF 38h: DF
File Identifier tag	1	83h
Length	1	02h
File Identifier	2	XXXXh: FID
Security Attributes Tag	1	86h
Length	1	03h
Security Attributes	3	See SELECT FILE command for details
Proprietary Information tag	1	85h: File status byte
Length	1	02h

Proprietary Information	2	First status byte: 00h: RFU for transparent EF's and DF's X0h: for private RSA key EF's  X = AC to clear after RSA operation. The remaining bits are RFU for EF Key, and for other file types the whole byte is RFU.  Second status byte: 00h: RFU
Life Cycle Status tag	1	8Ah
Length	1	01h
Life Cycle Status	1	00h: RFU
DF Name tag	1	84h: Optional tag, can only be used when creating DF's.
Length	1	01h – 10h
DF Name	1-16	

\* 0800h is supported from MyEID applet version 2.2.0 onwards.

## 5.5 DELETE FILE

The DELETE FILE command removes the currently selected file if the required Access Condition is fulfilled. In case that the file to be deleted is a DF, it is deleted along with its full sub-tree, including all child DF's and EF's.

After a successful file removal the file space is de-allocated and can be reused for new files.

**Table 58** – DELETE FILE command APDU

Byte	Value
CLA	00h
INS	E4h
P1	00h
P2	00h
LC	00h
Data	Empty
LE	Empty

**Table 59** – DELETE FILE response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes

## 5.6 GENERATE PUBLIC KEY PAIR

The GENERATE PUBLIC KEY PAIR command generates and stores a new key into an existing key EF. The key length (length of modulus) is given when the key EF is created; see CREATE FILE command for details. Before the command can be executed successfully, the related Access Condition must be fulfilled. The command returns the modulus of the new key.

**Table 60** – GENERATE PUBLIC KEY PAIR command APDU

Byte	Value
CLA	00h
INS	46h
P1	00h
P2	00h
LC	05h or 07h: Length of data field
Data	TLV with value of public exponent, see Table 62 for details.
LE	Empty or length of modulus in bytes

**Table 61** – GENERATE PUBLIC KEY PAIR response APDU

Byte	Value
Data	Modulus
SW1 – SW2	Status Bytes

**Table 62** – Generate public key pair command data parameters

Bytes	Length	Value
Outer sequence Tag	1	30h
Length	1	03h or 05h
Public exponent tag	1	02h
Length	1	01h or 03h
Public Exponent	1 – 3	03h or 010001h <b>MyEID3: supports only 010001h</b>

## 5.7 PUT DATA: LOAD KEY

The PUT DATA: LOAD KEY command stores an externally generated key to the applet. A key EF must be selected first with the SELECT FILE command.

All the key components must be loaded to create a valid key.

All MyEID versions support the loading of private keys in the CRT format. The MyEID3 version also supports the private exponent and modulus format. Only one private RSA key can be loaded into a key file together with a public key.

**Table 63** – PUT DATA command APDU

Byte	Value
CLA	00h
INS	DAh
P1	01h
P2	RSA key components:  80h: modulus, N 81h: public exponent, E 83h: prime 1, P 84h: prime 2, Q 85h: $d \bmod (p - 1)$ , DP1 86h: $d \bmod (q - 1)$ , DQ1 87h: $q-1 \bmod p$ , PQ  88h: first half of 2048 bit modulus 89h: second half of 2048 bit modulus *1  MyEID3 adds support for: 82h: private exponent, D *3  8Ah: first half of 2048 bit key private exponent, D 8Ah: second half of 2048 bit key private exponent, D
LC	Length of Data field  P2 = 80h: <b>80h</b> P2 = 81h: <b>01h</b> or <b>03h</b> P2 = 83h, 84h, 85h, 86h, 87h: <b>40h</b> or <b>80h</b> *2 P2 = 88h, 89h: <b>80h</b>  MyEID3: P2 = 80h, 82h: <b>40h</b> to <b>C1h</b> P2 = 81h: <b>01h</b> or <b>03h</b> P2 = 83h, 84h, 85h, 86h, 87h: <b>20h</b> or <b>C1h</b> P2 = 88h, 89h: <b>81h</b> P2 = 8Ah, 8Bh: 81h

Data	Component
LE	Empty

\*1 The second half of 2048 bit modulus must be preceded by the first half and the order of the commands must not be reversed.

\*2 80h is used for the 2048 bit key components.

\*3 If a private key is loaded as a modulus and a private exponent, then only the public exponent is required. The private exponent must not be loaded with a CRT formatted key, loading it will remove the CRT components.

**Table 64** – PUT DATA: LOAD KEY response APDU

Byte	Value
Data	Empty
SW1 – SW2	Status Bytes