Specifications for the Smart-Card Operating System for Transport Applications (SCOSTA)

Version 1.2b
March 15, 2002

National Informatics Centre
Ministry of Communication and
Information Technology
Government of India

Ministry of Road Transport and
Highways
Government of India

Indian Institute of Technology Kanpur

History

1.	August 13, 2001	Internal Release of the First Draft 1.0 R 1
2.	September 1, 2001	Internal Release of the Second Draft 1.0 R 2
3.	December 31, 2001	Public Release Version 1.0 as SCOSTA Specs
4.	January 8, 2002	Public Release Version 1.1 as SCOSTA Specs
5.	February 1, 2002	Public Release Version 1.2 (Part II and III) removed.
6.	February 15, 2002	Public Release Version 1.2a
7.	March 15, 2002	Public Release Version 1.2b (minor updates)

Smart Card Operating System for Transport Applications (SCOSTA)
Part I:
Operating System Interface

1 Introduction

This document specifies the operating system interface for the multi-purpose Smart Cards intended to be used in the context of the applications in India. It covers the interface details of the smart card operating system. However the implementation of the operating system, processor for the smart card and other such details are not covered in this document and are out of its scope.

The SCOSTA describes the minimum support for the application using the Smart Cards. Operating systems with extra features not listed here will be SCOSTA compliant only if they support all the features listed in this document. A SCOSTA compliant operating system will also be compliant to the ISO7816-4, -8 and -9 standards.

The SCOSTA is compatible to the following ISO standards for the Smart Cards:

ISO/IEC 7816-3:1997, *Electronic signals and transmission protocols*
ISO/IEC 7816-4:1995, *Interindustry commands for interchange*
ISO/IEC 7816-4:1995/Amd 1:1997, *Impact of secure messaging on the structures of APDU messages*
ISO/IEC 7816-5:1994, *Numbering system and registration procedure for application identifiers*
ISO/IEC 7816-5:1994/Amd 1:1996
ISO/IEC 7816-6:1996, *Interindustry data elements*
ISO/IEC 7816-6:1996/Cor 1:1998
ISO/IEC 7816-6:1996/Amd 1:2000, *IC manufacturer registration*
ISO/IEC 7816-8:1999, *Security related interindustry commands*
ISO/IEC 7816-9:2000, *Additional interindustry commands and security attributes*

2 Symbols and Terminology

The terminology used in this document is the same as the one used in the ISO/IEC documents and the usual references are marked in this document to the terms, commands in the appropriate ISO/IEC documents. In addition, the following is used.

EF*i* Represents an EF with short EF identifier of *i*. The valid values for the *i* are 1 to 30.

3 Basic Data Structures

SCOSTA will support the following two categories of the files.
Dedicated Files (DF) and Elementary Files (EF).

The structure of the file system will be that of a tree with depth restrictions, if any, not less than 4 (including the MF and EF). Thus it will be possible to have at least one MF, one DF under the MF, another DF under this DF and EFs under the last DF. At each node of the tree, there can be several children nodes, either of type DF or of type EF. The root of the tree is the Master File (of type DF). At any node it should be possible to create at least 30 children nodes (both DF and EF put together). The restrictions on the tree structure may because of the limited size of the memory that may cause a CREATE FILE command to fail.

The files will support at least the fixed size declared at the time of CREATE FILE command using the FCP.

The files under the MF will be global for all applications. The files pertaining to each application shall be stored under a DF node in the file system tree. An application may have sub-applications each represented by an individual DF under the parent DF. In such a case, the application may organize the files as global within the application and local for each sub-application.

3.1 Elementary Files

The elementary files (EF) will be used to keep the data pertaining to an application. An EF can be one of the following types.

- Transparent EF (as defined in ISO/IEC 7816-4 section 5.1.3)
- Linear EF with fixed size records (as defined in ISO/IEC 7816-4 section 5.1.3)
- Linear EF with variable size records (as defined in ISO/IEC 7816-4 section 5.1.3)
- Cyclic EF with fixed size records (as defined in ISO/IEC 7816-4 section 5.1.3).

The OS will use some EFs internally as well. A few pre-determined and fixed short EF identifiers will be used to identify such EFs. There is however no restriction on the 16-bit identifiers for the same. In particular the OS uses short EF identifiers 1 and 2 internally. Section 6.2 of this document describes their use.

3.2 File Referencing Methods

Each file will have one 16-bit file identifier (with unique identifier for all EFs and DFs under a single DF). The EFs can also be referenced using another short EF identifier (5-bits) given at the time of CREATE FILE command. The short EF-identifiers need not be unique. No assumptions shall be made by the SCOSTA about the relation between the 16-bit file identifier and 5-bit short file

identifier for the EFs. While selecting a file using short EF identifier, the results will be unpredictable if two or more files exist with the same short EF identifier.

Each DF in the file system will also have the DF name that shall be unique among all DFs in the file system. Thus it will be possible to refer to any DF with its name irrespective to its location in the file system.

It will be mandatory for each file to have the 16-bit file id. In addition, all the DFs will have a unique name (1 to 16 bytes long).

Files can be referenced using the following four methods as described in the ISO/IEC 7816-4 section 5.1.2.

- Referencing by the file identifier. The values 3F00, 3FFF and FFFF are used for special purposes as given in ISO/IEC 7816-4 standard.
- Referencing by path.
- Referencing by short EF identifier.
- Referencing by DF name.

3.3 Data Referencing Methods

A SCOSTA compliant operating system shall implement at least the following data referencing methods as defined in ISO/IEC 7816-4 section 5.1.4.

- Record referencing
  - Referencing by one byte record number
  - Referencing by one byte record identifier (only for SIMPLE-TLV DOs)

  It is implied in this document that the maximum size of a record shall not exceed 255 bytes.
- Data Unit referencing

  The data unit referencing shall be available for the transparent EF. The operating systems may implement data unit referencing mechanism for the record structure files as well primarily intended to debug the applications. However in such a case, the data returned may include the structural information (meta-data) as well. SCOSTA compliant applications will make no assumptions about the structure of the meta-data.

  The cards shall support variable data unit size (among these support for 8-bits wide data units is mandatory for all SCOSTA compliant cards) as defined in the table number 86 of ISO/IEC 7816-4.

3.4 File control information.

SCOSTA compliant OS will support the FCP templates at least to be returned to the application during the SELECT FILE command execution. In case SELECT FILE command demands other templates, meaningful default information should be supplied if the other templates are not supported.

4 Security architecture

SCOSTA compliant OS will support at least the following security architecture as defined in ISO/IEC 7816-4, ISO/IEC 7816-8 and ISO/IEC 7816-9.

4.1 Security status

The SCOSTA compliant OS will support the following three security statuses.

- Global security status (related to the MF authentication processes)
- File-specific security status (related to the DF authentication processes)
- Command specific security status (related to the command authentication processes)

4.2 Security attributes

It will be possible to attach security attributes to the files using the FCP (as given in ISO/IEC 7816-9 section 5 table 1). Both compact form as well as the expanded form of defining security attributes must be supported.

The SCOSTA compliant OS shall support the referencing of security attributes for the files (using the FCI mechanism). Other methods of referencing the security attributes (as described in ISO/IEC 7816-9 section 8) are optional for SCOSTA compliant operating system.

Similarly SCOSTA compliant OS shall support the access mode bytes (AM bytes) as given in ISO/IEC 7816-9 tables 6 and 7. Other AM bytes are optional for SCOSTA compliant OS. No proprietary coding shall be used in AM/SC bytes.

4.3 Security Environments

The security environments (SEs) shall be stored either in an EF or in the FCP of the DF with a tag 7B. In case the SEs are stored in an EF, the id of the EF shall be notified with tag 8D in the FCP of the corresponding DF. The default SE for an application will be all empty.

4.4 Security mechanisms

The SCOSTA compliant OS shall support at least the following security mechanisms as described in the section 5.2.3 of ISO/IEC 7816-4.

- Entity authentication with password (VERIFY command)
- Entity authentication with key (EXTERNAL AUTHENTICATE, INTERNAL AUTHENTICATE and MUTUAL AUTHENTICATE commands)
- Data authentication (computation/verification of cryptographic checksum). Triple DES (3DES) algorithm in CBC mode will be used for Data authentication, data encipherment and decipherment.
- Data encipherment

- Data decipherment

Triple DES (3DES) algorithm in CBC mode will be the default algorithm (i.e. if the algorithm reference is not given or given as default in the appropriate commands/DOs) used for Data authentication, data encipherment and decipherment.

4.5 Access rule references

The access rule (in expanded or in compact format) for a file can be stored in the file's FCP. These rules shall specify under what security conditions certain operations on that file are permitted. The access rules for the DOs can be stored in the FCI of the corresponding DF. The access rule referencing mechanisms shall be according to the ISO/IEC 7816-9.

5 APDU message structure

SCOSTA compliant OS will support at least the normal 1-byte fields for coding L_e and L_c fields in the command APDU (reference table 5 of ISO/IEC 7816-4). The OS will support all the four structures of the command APDUs as shown in figure 4 of ISO/IEC 7816-4 document. Similarly the response APDU as described in section 5.3.3 of the ISO/IEC 7816-4 will be supported.

5.1 Coding conventions for command headers

The command header has four bytes (CLA, INS, P1 and P2).

5.1.1 Class byte (CLA)

The CLA byte coding as given in table 1 will be applicable to the SCOSTA compliant OS.

Table 1: Coding for the CLA byte in SCOSTA compliant OS

CLA Byte								Meaning
b8	b7	b6	b5	b4	b3	b2	b1	
0	0	0	0	X	0	0	0	Secure Messaging format (table 9 of ISO/IEC 7816-4)
1	0	0	0	X	0	0	0	
1	0	0	1	X	0	0	0	
1	0	1	0	X	0	0	0	
1	0	1	1	X	0	0	0	As defined in ISO/IEC 7816-4, -8 and -9 standards
1	1	0	0	X	0	0	0	
Others								RFU for SCOSTA

The b4 bit of the CLA byte will be 0 if no secure messaging is used and will be 1 for the secure messaging as per ISO/IEC 7816-4. In this version of the SCOSTA, the only CLA byte to be used is 00 (i.e. secure messaging is not required).

5.1.2 Instruction byte (INS)

Instruction bytes will be coded as defined in the ISO/IEC 7816-4, -8 and -9 in the appropriate places. The instructions that shall be implemented as mandatory instructions in SCOSTA compliant OS are described in this document.

5.1.3 Parameters to the instructions (P1 and P2 bytes)

P1 and P2 bytes will be coded as described for the individual instructions in ISO/IEC 7816-4, -8 and -9 standards. The further interpretation of these bytes and restrictions if any are given in this document at appropriate places.

5.2 Coding conventions for data fields, status bytes (SW1 and SW2)

These bytes are coded as described in the ISO/IEC 7816-4, -8 and -9 standards and suitably interpreted in this document at appropriate places.

6 SCOSTA supported commands

SCOSTA compliant cards have two kinds of storage – volatile and non-volatile. All instructions that update the non-volatile information will execute atomically in the SCOSTA compliant cards. The effect of all such commands will be either due to the complete execution or due to no execution. The cards may use the subsequent power up to finish the last command. The commands supported by the SCOSTA are categorized in the following categories. All of these commands are appropriately referred to the ISO/IEC documents.

- File related commands
- Security related commands
- Other commands

6.1 File related commands in SCOSTA

SCOSTA compliant OS will support the following commands as mandatory commands.

- READ BINARY
- READ RECORDS
- WRITE BINARY
- WRITE RECORD
- UPDATE BINARY
- UPDATE RECORD
- APPEND RECORD
- ERASE BINARY
- CREATE FILE
- DELETE FILE
- SELECT FILE
- ACTIVATE FILE
- DEACTIVATE FILE

- TERMINATE DF
- TERMINATE EF
- TERMINATE CARD USAGE

6.1.1 READ BINARY command

Ref: ISO/IEC 7816-4 section 6.1

6.1.2 READ RECORDS command

Ref: ISO/IEC 7816-4 section 6.5

6.1.3 WRITE BINARY command

Ref: ISO/IEC 7816-4 section 6.2

6.1.4 WRITE RECORD command

Ref: ISO/IEC 7816-4 section 6.6

6.1.5 UPDATE BINARY command

Ref: ISO/IEC 7816-4 section 6.3

6.1.6 UPDATE RECORD command

Ref: ISO/IEC 7816-4 section 6.8

6.1.7 APPEND RECORD command

Ref: ISO/IEC 7816-4 section 6.7

6.1.8 ERASE BINARY command

Ref: ISO/IEC 7816-4 section 6.4
Explanatory notes: An EF is either created with the CREATE FILE command or is available in the card initially. The maximum size of the EF is fixed in the beginning. The EF will have the initial state of data as ‘invalid’. Data added in the EF makes its state as ‘valid’. ERASE BINARY command is used to delete the contents previously added using WRITE BINARY command. Such a command for the EF with a record structure is not needed as the record structures themselves may have a status byte (valid/invalid) and the applications can manipulate them by UPDATE RECORD command.

Depending upon the attributes of the file, ERASE BINARY command will initialize the file data to all 1’s (for write-and files) or to all 0’s (for write-or files). For write-once files, the flags shall be maintained at the level of each data-unit and it will be possible to erase data in units of one data-unit. Upon ERASE BINARY, the flags will be set to appropriate values.

The UPDATE commands will only modify the data and not the flags. Thus it will be possible to UPDATE data and later WRITE into a write-once file.

6.1.9 CREATE FILE command

Ref: ISO/IEC 7816-9 section 9.2

The CREATE FILE command in SCOSTA complaint OS will require the file control parameters as given in table 2 (ref: Table 1 of ISO7816-9).

Table 2: File control parameters for SCOSTA files

Tag	Value	Applicable to
80	Size of the file in bytes	Transparent EF
81	Size of the file in bytes including meta-data. This option should be implemented in the OS with the interpretation left to the OS. However applications are discouraged to use this in favor of tag 82.	All files
82	Depending upon the length 1. File descriptor byte-FDB 2. FDB + data coding byte-DCB 5. FDB + DCB + MRL (2 bytes)+ #records (1 byte) 6. FDB + DCB + MRL (2 bytes)+ #records (2 byte) Note: Tag 82 with length 3 and 4 is defined in ISO/IEC 7816-4 but is not used with these values of the length in the SCOSTA. Tag 82 with length 1 and 2 is defined in ISO/IEC 7816-4 for any file. In case of the SCOSTA, these can be used only for the transparent EF and for the DF.	 Transp. EF or DF Transp. EF or DF EFs with records EFs with records
83	File identifier (Mandatory field)	Any file
84	DF Name	DFs
88	Short EF identifier If this tag is not present, least five bits of the File identifier used be used as short EF identifier. If this tag specifies no value (length 0), the corresponding EF will have not short EF identifier.	EFs
8A	Life Cycle Status Integer (LCSI)	Any file
8C	Security attributes in compact form	Any file
AB	Security attributes in expanded form If the security attributes are not defined in expanded or in compact form, default security attributes of ‘no security’ will be used. Thus the information in a file with no security attribute can be read and written by an application.	Any file
7B	SEs as defined in ISO/IEC 7816-8 (section 4.3 of this document). SEs may be contained in an EF in which case tag 8D is to be used.	DFs
8D	File id of the EF containing SE templates. The SEs can be specified using 7B tag also.	DFs

The SCOSTA-compliant OS will have to support P1 and P2 bytes of the command to be all 0s. Other values of the P1 and P2 are optional. In that case, a default value of the FCP will be assumed.

6.1.10 DELETE FILE command

Ref: ISO/IEC 7816-9 section 9.3
Explanatory notes: The DELETE FILE command can be used to delete an application/sub-application in a multi-application smart card or delete an EF within a single application. For deleting a file, the following two security conditions are applicable.

(i) "DELETE CHILD" of the parent DF, and,
(ii) "DELETE FILE" of the EF.

6.1.11 SELECT FILE command

Ref: ISO/IEC 7816-4 section 6.11
Explanatory notes: It should be possible for the SELECT FILE command to return FCP of the selected file depending upon the value in parameter P2. If Le field is empty, no value will be returned by the SELECT FILE command.

6.1.12 ACTIVATE FILE command

Ref: ISO/IEC 7816-9 section 9.5
Explanatory notes: Similar to the SELECT FILE command, it will be possible for the application to get the FCP of the file depending upon the value in the parameter P2. The LCSI byte in the returned FCP will represent the status of the file after the execution of the command.

6.1.13 DEACTIVATE FILE command

Ref: ISO/IEC 7816-9 section 9.4
Explanatory notes: Similar to the SELECT FILE command, it will be possible for the application to get the FCP of the file depending upon the value in the parameter P2. The LCSI byte in the returned FCP will represent the status of the file after the execution of the command.

6.1.14 TERMINATE DF command

Ref: ISO/IEC 7816-9 section 9.6
Explanatory notes: Similar to the SELECT FILE command, it will be possible for the application to get the FCP of the file depending upon the value in the parameter P2. The LCSI byte in the returned FCP will represent the status of the file after the execution of the command.

6.1.15 TERMINATE EF command

Ref: ISO/IEC 7816-9 section 9.7

Explanatory notes: Similar to the SELECT FILE command, it will be possible for the application to get the FCP of the file depending upon the value in the parameter P2. The LCSI byte in the returned FCP will represent the status of the file after the execution of the command.

6.1.16 TERMINATE CARD USAGE command

Ref: ISO/IEC 7816-9 section 9.8

6.2 Security related commands in SCOSTA

The following commands will be supported by the SCOSTA compliant operating systems.

- VERIFY
- INTERNAL AUTHENTICATE
- EXTERNAL AUTHENTICATE
- MUTUAL AUTHENTICATE
- GET CHALLENGE
- MANAGE SECURITY ENVIRONMENT
- PERFORM SECURITY OPERATION
- ENABLE VERIFICATION REQUIREMENT
- DISABLE VERIFICATION REQUIREMENT
- CHANGE REFERENCE DATA
- RESET RETRY COUNTER

6.2.1 VERIFY command

Ref: ISO/IEC 7816-4 section 6.12

VERIFY command in SCOSTA-compliant OS will be used to verify the password with the reference data (read password) stored in the card. The reference data number as provided in P2 should be interpreted as given in table 3.

Table 3: P2 byte for the VERIFY command

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	No information is given
0	0	0	X	X	X	X	X	Reference data number in Global reference data bank
1	0	0	X	X	X	X	X	Reference data number in local reference data bank
All other values								RFU

The reference data for the global data bank is stored in an EF immediately under the MF. This EF is identified using a short EF identifier of 1 (EF1). Similarly, the local reference data for an application is stored in an EF immediately under a DF for that application. The short EF identifier of 1 (EF1) is used to identify this EF. The OS may fix any 16-bit identifiers for such files. If the appropriate EF1 does not exist, the VERIFY command will return a failure condition (SW1=6A and SW2=88, for “reference data not found”).

The changes in the EF1 may or may not be permitted depending upon the security attributes for that file. However this file will be referenced upon for validating the passwords.

The EF1 will be a variable record file (up to a maximum of 32 records) with the following structure.

Pin identifier	1 Byte
Retry counter	4 Bits (see below)
Max retry count	4 Bits (see below)
Pin	Variable length

Records in the EF1 will have one byte containing the Retry counter and Max retry count. The bits b8:b4 of this byte will provide the Retry counter while bits b4:b1 will provide the Max retry count. Bits b8 and b4 will be the MSB of their respective fields. A value F for the Max Retry count with non-zero Retry counter shall mean that there is no limit on the retries.

The Pin identifier will be coded as follows.

b8	b7	b6	b5	b4	b3	b2	B1
V	RFU			Ref Data Number			

V bit represents if the corresponding entry is valid (1) or not (0). The VERIFY command for the invalid data will return a failure (SW1=69, SW2=84, “Reference data invalidated”) in the status bytes. VERIFY command will return a failure if the EF1 file is not found.

V bit is manipulated using the ENABLE VERIFICATION REQUIREMENT and DISABLE VERIFICATION REQUIREMENT commands.

A record in EF1 with Pin identifier of 80 provides the reference data for the DF or MF itself and will be used to verify when the parameter P2 is given as 00H or 80H. A verification requirement as specified in the security attributes of the file will result in verified status if the corresponding reference data is invalid.

If Max retry count is a value other than F, then upon a successful verification, Retry counter for that reference data is set to the Max retry count. Also upon unsuccessful verification, Retry counter decrements by 1 and when it reaches a value of 0, subsequent VERIFY commands return a failure without any verification.

If Max retry count is F (but the Retry counter is non-zero) then no limit will be set on the number of retries. In such a case, the Retry counter will not be changed after a successful or unsuccessful verification.

If the Retry counter is zero, then VERIFY command returns a failure without any verification irrespective of the value of the Max Retry Count.

If the value of P2 is specified as 0 in the command, the implicit verification will take place with the reference data record with PIN identifier byte = 80H stored in the EF1 under MF. All other conditions such as EF1 not present or the record being invalid will have the usual effect on the error conditions. If the data field contains no byte (Lc=0), counters will not be adjusted and SW1-SW2 ='63CX' or '9000' will be returned depending upon the valid bit being 1 or 0 respectively.

Table 4: P2 byte for INTERNAL AUTHENTICATE, EXTERNAL AUTHENTICATE and MUTUAL AUTHENTICATE commands

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
0	0	0	0	0	0	0	0	No information is given
0	0	0	X	X	X	X	X	Number of the secret in Global secret bank
1	0	0	X	X	X	X	X	Number of the secret in local secret bank
All other values								RFU

6.2.2 INTERNAL AUTHENTICATE command

Ref: ISO/IEC 7816-4 section 6.13

The INTERNAL AUTHENTICATE commands require a key reference in P2 as per Table 4. The secrets are stored in EF2 immediately under the MF or a DF. The secrets stored in EF2 immediately under the MF are the global secrets. EF2 is a variable record structured file with the following structure.

Key identifier	1 Byte
Key Type	1 Byte
Key Specific Information	Variable length
Algorithm Reference	1 Byte
Key	Variable length

The key identifier is coded as follows.

b8	b7	b6	b5	b4	b3	b2	b1
V	RFU		Secret Number				

V bit is used to denote if the corresponding secret is valid or not. (0: invalid, 1: valid). The secret number (by which the key is referred to in various security related operations) will be unique for all keys. Thus there can be only up to 32 keys in EF2. No two keys will have the same secret number even if they are used for two different purposes.

The key type field provides the operations for which the key can be used. The value is coded as follows.

b8	b7	b6	b5	b4	b3	b2	B1
CC	DS	Enc-Sym	Enc-Asym	Hash	RFU	Int Auth	Ext Auth

If the CC bit is set to 1, the key can be used for computation of the cryptographic checksum.

If the DS bit is set to 1, the key can be used for computation of digital signature. Since in the current version of the SCOSTA, public key cryptography is not supported, this field should be treated as RFU.

If the Enc-Sym bit is set to 1, the key can be used for symmetric encryption and decryption.

If the Enc-Asym bit is set to 1, the key can be used for asymmetric encryption and decryption. Since the public key cryptography is not supported in this version, this field should be treated as RFU.

If the Hash bit is set to 1, the key can be used for the hashing operation.

If the Int Auth bit is set to 1, the key can be used for the internal authentication.

If the Ext Auth bit is set to 1, the key can be used for the external authentication.

The type specific information is of variable length and is defined as per the key type field. The value is made available for each bit set to 1 in the key type field. The values are provided in the order of the bits in the key type field. Thus if the CC bit is set to 1, the type specific information will first contain the information regarding the usage of the key for the CC. The following type specification information is used.

Operation	**Information**
CC	None (0 bytes)
DS	None (0 bytes)
Enc-Sym	Usage Counter (2 Bytes)
Enc-Asym	Usage Counter (2 Bytes)
Hash	None (0 bytes)
Int Auth	Usage Counter (2 Bytes)

Ext Auth	Retry Counter (4 bits) and Max Retry Count (4 bits)

The Usage counter for the Enc-Sym, Enc-Asym and Int Auth is a monotonically decreasing counter. The counter if set to FFFF, means that the counter is not used (and therefore is not changed by the usage of the key). Values other than FFFF refer to the number of times that the key can be used by the INTERNAL AUTHENTICATE command. The key can be used only if the Usage Counter is non-zero. Upon each usage (whether successful or unsuccessful), one is subtracted from the counter (only if the counter is non FFFF). The initial value of the counter is set at the time writing the record in the EF2. The value can be changed by UPDATE RECORD command if its execution is permitted by the security conditions.

The Retry Counter and Max Retry Count are coded as per the coding given for the EF1. Bits b8:b5 provide the Retry Counter value while bits b4:b1 provide the Max Retry Count. These values are used only upon the use of the EXTERNAL AUTHENTICATE command. If the value of the Retry Counter is 0, the EXTERNAL AUTHENTICATE command results in a failure. If the Retry Counter is other than 0, it is decremented by 1 (only if the Max Retry Count is not F) upon each unsuccessful authentication.

The algorithm reference codes the algorithm for which the key usage is valid. A value of 00 for the algorithm reference implies that the key is valid for all the algorithms available in the card.

The changes in the EF2 may or may not be permitted depending upon the security attributes for that file. However this file will be referenced for validating the keys internally by the operating system.

The key 0 in the EF2 immediately under the MF will be used when no data is information is given (i.e. P2 is 00H).

An invalid key will result in a failure for the INTERNAL AUTHENTICATE command.

When algorithm reference given by P1 in the command is 0, 3DES algorithm will be used. If P1 is not 0, the specified algorithm will be used (which may also be 3DES). If the key is not valid for the algorithm or the operation, SW1-SW2=6985 (“Condition of use not satisfied”) will be returned. Data field will contain the challenge appropriate to the algorithm. For 3DES, the challenge will be at least 8 bytes (and in multiple of 8 bytes). The returned response from the card will contain the data generated by the encryption of the challenge using 3DES in CBC mode. In case, the Le is less than the size of the encrypted data, lower order Le bytes shall be returned.

6.2.3 EXTERNAL AUTHENTICATE command

Ref: ISO/IEC 7816-4 section 6.14

The EXTERNAL AUTHENTICATE command will refer to the EF2 as described in section 6.2.2. When P1 is given as 0, the 3DES algorithm will be used.

If the corresponding EF2 is not found, SW1-SW2=6A88 (“REFERENCE DATA NOT FOUND”) error condition will be returned. If the body of the command is empty, SW1-SW2=63CX (where X is the value of further allowed retries) will be returned if the record corresponding to the referenced key is valid. SW1-SW2=9000 will be returned if the record is invalid meaning that the verification is not required. In this case, no counters will be modified. If the body of the command is non-empty then it will contain response to the previously issued challenge. In this case, if the referenced key is not found or the corresponding record is invalid, SW1-SW2=6984 (“REFERENCE DATA INVALIDATED”) error condition will be returned. If the key is not valid for external authentication, SW1-SW2=6985 (condition of use not valid) will be returned.

6.2.4 MUTUAL AUTHENTICATE command

Ref: ISO/IEC 7816-8 section 14

MUTUAL AUTHENTICATE command will refer to the EF2 as described in section 6.2.2. In addition, the key reference will be valid only if the key type indicates that the key can be used for Internal Authentication as well as for External Authentication.

The body of the command will contain response followed by the challenge. The length of the response will be known due to the algorithm reference. In 3DES, response will be 8 bytes and the challenge will be another 8 bytes.

6.2.5 GET CHALLENGE command

Ref: ISO/IEC 7816-4 section 6.15

The GET CHALLENGE command will return a random number that may be derived from the session counter (but not the session counter itself). Only for the very next command issued by the interface device, this challenge can be used implicitly. The challenge will not be valid for the subsequent commands.

6.2.6 MANAGE SECURITY ENVIRONMENT command

Ref: ISO/IEC 7816-8 section 10
Explanatory notes: The MSE command will operate on the SEs stored in the file or as DOs in the DF. If the space to store the modified SE is not available an error condition SW1-SW2=6600 will be returned. MSE: SET command can be used only to change the values that are already in the SE already. For example, data item to derive the key can be given only if the SE already has a DO with tag 94 in the appropriate CRT. In particular, if the tag 94 has a length 0, then the previously obtained challenge by the GET CHALLENGE command will be used for the data to derive the key.

6.2.7 PERFORM SECURITY OPERATION command

Ref: ISO/IEC 7816-8 section 11
Explanatory Notes: All the security operations relevant for the symmetric key cryptography will be applicable in the SCOSTA. Therefore, computation of the cryptographic checksum, calculation of a hash code (hash code computation will be optional in this version of SCOSTA), verification of the cryptographic checksum, encipherment, and decipherment will be required in the SCOSTA compliant cards. These commands can be performed only if the security status satisfies the security attributes needed for the operation. The following table explains the P1 and P2 parameters specific to the PSO command. Further in this version, command chaining is not necessarily implemented. Hence reference to command chaining in the associated documents is not relevant.

P1	P2	Meaning	Remarks
8E	80	Compute Cryptographic Checksum	Cryptographic Checksum will be at least 4 bytes. The length of the returned value will depend upon the Le field.
00	A2	Verify Cryptographic Checksum	Data field will contain plain text (tag 80) and crypto checksum (tag 8E). The CRT, for the operation will be known from the current SE. Algorithm reference if not known will refer to 3DES. Key reference will be present in the SE in CCT template. Initial check block will be taken as chain block using tag 86 (L=0) in CCT.
86	80	Encipher	P1=82 and 84 not mandatory. Padding indicator byte will be 01 or 02 only. (Ref: Table 23 in ISO/IEC 7816-4)
80	86	Decipher	P2=82 and 84 not mandatory.

6.2.8 ENABLE VERIFICATION REQUIREMENT command

Ref: ISO/IEC 7816-8 section 12.3
Explanatory notes: The enable verification requirement command will update the valid bit in EF1 for the record corresponding to the specified reference data. If P2 is given as 0, then the first record of EF1 will be implied. If originally the record was invalid, the body may contain the reference data that will be replaced in the EF1.

6.2.9 DISABLE VERIFICATION REQUIREMENT command

Ref: ISO/IEC 7816-8 section 12.4
Explanatory notes: This command will update the valid bit in EF1 for the record corresponding to the specified reference data. If P2 is given as 0, the the first record of the EF1 in the currently selected DF will be implied. If originally the record was valid, then the modification of the valid bit will be possible in the following cases.

(i) If the reference data is specified in the body then upon the satisfactory verification. The security status is updated accordingly.
(ii) If the reference data is not specified in the body then upon an earlier satisfactory VERIFY command with reference to the same record as given in the security status of the card.

6.2.10 CHANGE REFERENCE DATA command

Ref: ISO/IEC 7816-8 section 12.2
Explanatory notes: In order to change reference data, the command may choose an option of providing older reference data. The reference data will be updated under the following conditions.

(i) If the old reference data is specified in the body and its verification with the stored reference data is satisfied. The security status is updated accordingly.
(ii) If the reference data is not specified in the body then upon an earlier satisfactory VERIFY command with reference to the same record as given in the security status of the card.

6.2.11 RESET RETRY COUNTER command

Ref: ISO/IEC 7816-8 section 12.5
Explanatory notes: The command will modify a record in the EF1 subject to the interface device proving the knowledge of the reference data (either through VERIFY command or through reference data provided in the command). If the resetting code is provided, the retry counter is set to minimum of the resetting code and Max retry count. If the resetting code in not provided, the retry counter is set to the Max retry count. The resetting code will be one byte.

6.3 Other Commands

The following commands will also be supported in the SCOSTA.

- GET DATA command
- PUT DATA command
- GET RESPONSE command

6.3.1 GET DATA command

Ref: ISO/IEC 7816-4 section 6.9

6.3.2 PUT DATA command

Ref: ISO/IEC 7816-4 section 6.10

6.3.3 GET RESPONSE command

Ref: ISO/IEC 8916-4 section 7.1

7 Data Objects

In SCOSTA compliant OS, certain data objects may be made available. These will be in compliance to the ISO/IEC 7816-6. For example, some of the relevant DOs may be used to represent the Card's Sequence Number, Primary Account Number, and Cardholder's Name etc. The data objects shall be accessed using GET DATA and PUT DATA commands. DOs will be attached to a DF. GET DATA/PUT DATA commands will operate on the DOs attached to the selected DF.

All SCOSTA compliant cards will have a DO with tag 46 (hex) for pre-issuing data. This DO will be available in the life time of the card and will be used to return the chip serial number of the chip embedded in the card and as provided by the chip manufacturer. If the embedded chip does not support the way of providing a unique chip serial number, 32-bit binary value containing all 0s will be returned. Otherwise the unique chip serial number will be returned by GET DATA command. PUT DATA command for DO 46 will not modify the value. The DO 46 will be available at least in the MF and will have variable length. The applications can use the GET DATA command with Le=0 to find the length of the chip serial number. In response to GET DATA command with Le=0 the exact length will be returned to the application in the status bytes (SW1=6C, SW2=exact length).