GLOBALPLATFORM
THE STANDARD FOR SMART CARD INFRASTRUCTURE

GlobalPlatform Card Technology

Secure Channel Protocol 03

Card Specification v 2.2 – Amendment D

Version 1.0

Public Release

April 2009

Document Reference: GPC_SPE_014

Table of contents

1. References

Standard / Specification	Description	Ref
GlobalPlatform Card v 2.2 including the latest errata and precisions	Card Specification from GlobalPlatform	[0]
NIST SP 800-57 Part 1 revised	Recommendation for Key Management – Part 1: General (Revised) March, 2007	[1]
FIPS PUB 197	Federal Information Processing Standard 197, Advanced Encryption Standard (AES), November 2001	[2]
NIST SP 800-38A	Recommendation for Block Cipher Modes of Operation: Methods and Techniques, 2001	[3]
NIST SP 800-78-1	Cryptographic Algorithms and Key Sizes for Personal Identity Verification, August 2007	[4]
FIPS PUB 201-1	Personal Identity Verification (PIV) of Federal Employees and Contractors, March 2006	[5]
ISO 9797-1	Information technology – Security Techniques - Message Authentication Codes (MACs) -Part 1: Mechanisms using a block cipher, 1999-12-15	[6]
ISO/IEC 8825-1	Information technology -- ASN.1 encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER)	[7]
GlobalPlatform Confidential Card Content Management – Card Specification v2.2 – Amendment A v1.0 including the latest errata and precisions	Defines a mechanism for an Application Provider to confidentially manage its own application when using a third party communications network.	[8]
NIST SP 800-108	Recommendation for Key Derivation Using Pseudorandom Functions, November 2008	[9]
NIST SP 800-38B	Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication, May 2005	[10]
FIPS PUB 140-2	Security requirements for cryptographic modules	[11]

Table 1-1: References

2. Abbreviations and Notations

Abbreviation	Meaning
2TDEA	Two key Triple DEA
3TDEA	Three key Triple DEA
AES	Advanced Encryption Standard
CBC	Cipher Block Chaining
C-DECRYPTION	Command Decryption
C-MAC	Command MAC (see note)
CMAC	Cipher-based MAC (see note)
DEA	Data Encryption Algorithm
DGI	Data Grouping Identifier
FCI	File Control Information
ICV	Initial Chaining Vector
ISO	International Organization for Standardization
KDF	Key Derivation Function
Key-DEK	Data Encryption Key
Lc	Exact length of command data in a case 3 or case 4 command
Le	Maximum length of data expected in response to a case 2 or case 4 command
LV	Length Value
MAC	Message Authentication Code
PRF	Pseudorandom Function
R-ENCRYPTION	Response Encryption
R-MAC	Response MAC
SCP	Secure Channel Protocol
S-ENC	Secure Channel command and response encryption key
S-MAC	Secure Channel C-MAC session key
S-RMAC	Secure Channel R-MAC session key
TDEA	Triple DEA
TLV	Tag Length Value

Table 2-1: Abbreviations and Notations

Note: *C-MAC is the abbreviation used in Card Specification v2.2 [0] for the MAC appended to command APDUs. This is not to be confused with CMAC, which is the abbreviation for a MAC calculation scheme specified in NIST SP 800-38B [10].*

3. Secure Channel Protocol '03'

3.1. Scope of the document

This document proposes a new secure channel protocol based on AES keys and specifies:

- A new mechanism to generate session keys.
- The schemes to be used with AES for C-MAC, R-MAC, command data field encryption and response data field encryption.
- The format of PUT KEY for AES.

This new protocol is based on existing SCP01 and SCP02 protocols. It supports AES-based cryptography in lieu of TDEA. The protocol protects bidirectional communication between the Host and the card (decryption/MAC verification for incoming commands, encryption/MAC generation on card response).

3.2. Use Cases and Requirements

This document proposes a specification addendum to support the following requirements:

The Secure Channel is used to personalize cards at Issuance and during Post-Issuance. The mode of the Secure Channel Protocol which uses pseudo-random card challenges allows the offline preparation of personalization scripts while the card is not present and the processing of these scripts on the card without an online connection to the entity that prepared the scripts.

When the personalization involves the loading of a cryptographic key, the transport key that secures the transmission must be at least as strong as the key being transmitted.

To assist in the determination of suitable transport keys, the US National Institute of Standards and Technology (NIST) has published a document called “Recommendation for Key Management”. This document, freely available on the NIST website under the reference NIST 800-57 1 [1], is a mandatory standard for federal use in the United States of America, and is also endorsed by many other governments around the world. It is referred to by the more widely known FIPS 201 [5]. NIST 800-57 1 [1] provides the cryptographic strength of key based on its algorithm and its size.

It results that a 2TDEA key can be used as a transport key to encrypt another 2TDEA key, an RSA 1024 key, or an ECC key with f=160-223, but cannot be used to encrypt the following keys:

- 3TDEA Length Keys
- RSA above 1024
- AES-128
- AES-192
- AES-256
- ECC (f=224 and above).

Furthermore, a 3TDEA key used as a transport key can only encrypt another 3TDEA key, an RSA 2048, or an ECC key with f=224-255, but cannot be used to encrypt:

- RSA above 2048
- AES-128
- AES-192
- AES-256

- ECC (f=256 and above).

Since the above types of keys are starting to become available in the latest generation of Java Cards™, it becomes important that GlobalPlatform provides a mechanism by which such key could be loaded into the card.

NIST has also published another standard called “Cryptographic Algorithms and Key Sizes for Personal Identify Verification”. This document, also freely available on the NIST website under the reference NIST SP 800-78 1 [4], is a mandatory standard for the Personal Identity Verification (PIV) cards for all US Federal employees and contractors. It is referred to by the more widely known FIPS 201 [5]. According to this standard, the time period to use RSA 1024 for digital signature expires on the 31st of December, 2008, and the 2TDEA Keys cannot be used for card authentication after the 31st of December 2010.

According to the above standards, an AES key is more suitable for a transport key as:

- An AES-128 key can be used to encrypt 3TDEA Keys, RSA up to 3072, AES-128 and ECC with f up to 383.
- An AES-192 key can be used in addition to encrypt RSA up to 7680, AES-192 and ECC with f=384-511.
- An AES-256 key can be used in addition to encrypt RSA up to 15360, AES-256 and ECC with f=512+.

This should ensure the permanence of a secure channel based on AES from a crypto analysis standpoint for several years.

4. Specification Amendments

4.1. Algorithm

4.1.1. Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is a symmetric cryptographic algorithm that requires the use of the same secret key to encrypt and decrypt data. In its simplest form it uses a 16-byte key to encrypt a 16-byte block of data and the same 16-byte key to decrypt and retrieve the original clear text.

Two other versions of AES exist that involve 24-byte key and 32-byte key respectively. In these versions, the clear text and the cipher text are still 16-byte long. The different “flavors” may be referred to as “AES-128”, “AES-192”, and “AES-256”.

A specification of AES with its different versions may be found in “Advanced Encryption Standard (AES)” FIPS 197 [2].

This complies with the FIPS PUB 140-2 Annex A (*Symmetric Key Encryption – 1*) (see [11]).

4.1.2. Encryption/Decryption

AES in CBC mode is used as specified in NIST 800-38A [3]. The ICV to be used is defined in sections 6.2.6 and 6.2.7.

This complies with the FIPS PUB 140-2 Annex A (*Symmetric Key Encryption – 1*) (see [11]).

4.1.3. MACing

CMAC as specified in NIST SP 800-38B [10] is used for MAC calculations. Note that NIST SP 800-38B [10] also specifies the padding to be applied to the input.

This complies with the FIPS PUB 140-2 Annex A (*Message authentication – 3*) (see [11]).

4.1.4. AES Padding

Unless specified otherwise, padding prior to performing an AES operation across a block of data is achieved in the following manner:

- Append an '80' to the right of the data block;
- If the resultant data block length is a multiple of 16, no further padding is required;
- Append binary zeroes to the right of the data block until the data block length is a multiple of 16.

This padding complies with one of the padding schemes proposed in NIST SP 800-38A [3].

4.1.5. Data derivation scheme

The following data derivation scheme is used to generate keys, pseudo-random card challenges or cryptograms:

Data derivation shall use KDF in counter mode as specified in NIST SP 800-108 [9]. The PRF used in the KDF shall be CMAC as specified in NIST SP 800-38B [10], used with full 16 byte output length.

The “fixed input data” plus iteration counter shall be the concatenation of the following items in the given sequence (note that NIST SP 800-108 [9] allows the reordering of input data fields as long as the order, coding and length of each field is unambiguously defined):

- A 12 byte “label” consisting of 11 bytes with value ‘00’ followed by a one byte derivation constant as defined below.
- A one byte “separation indicator” with value ‘00’.
- A 2 byte integer “L” specifying the length in bits of the derived data (value ‘0040’, ‘0080’, ‘00C0’ or ‘0100’).
- A 1 byte counter “i” as specified in the KDF (which may take the values ‘01’ or ‘02’; value ‘02’ is used when “L” takes the values ‘00C0’ and ‘0100’, i.e. when the PRF of the KDF is to be called twice to generate enough derived data).
- The “context” parameter of the KDF. Its content is further specified in the sections below applying the data derivation scheme.

Definition of the derivation constant:

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	0	0	0	0	0	x	authentication cryptogram generation
0	0	0	0	0	0	0	0	- card cryptogram
0	0	0	0	0	0	0	1	- host cryptogram
0	0	0	0	0	0	1	0	card challenge generation
0	0	0	0	0	1	x	x	key derivation
0	0	0	0	0	1	0	0	- derivation of S-ENC
0	0	0	0	0	1	1	0	- derivation of S-MAC
0	0	0	0	0	1	1	1	- derivation of S-RMAC
all other values								RFU

Table 4-1: Data derivation constants

5. Secure Channel Protocol Usage

5.1. Secure Communication configuration

The following section defines the usage of Secure Channel Protocol '03'.

The 3 levels of security are supported as defined in appendix D.1.1 of Card Specification v2.2 [0]*:*

- Mutual authentication
- Integrity and data origin authentication
- Confidentiality

In SCP03 the "i" parameter is formed as a bit map on one byte as follows:

b8	b7	b6	b5	b4	b3	b2	b1	Description
				X	X	X	X	RFU (set to 0)
			0					Random card challenge
			1					Pseudo-random card challenge
	0	0						No R-MAC/R-ENCRYPTION support
	0	1						R-MAC support / no R-ENCRYPTION support
	1	1						R-MAC and R-ENCRYPTION support
X								Reserved

Table 5-1: Values of Parameter "i"

Note: *"i" is a sub identifier within an object identifier, and bit b8 is reserved for use in the structure of the object identifier according to ISO/IEC 8825-1 [7].*

5.2. Mutual Authentication

Mutual authentication is achieved through the process of initiating a Secure Channel and provides assurance to both the card and the off-card entity that they are communicating with an authenticated entity. If any step in the mutual authentication process fails, the process shall be restarted, i.e. new challenges and Secure Channel Session keys shall be generated

The process of initiating a Secure Channel is defined in appendix D.1.2 of Card Specification v2.2 [0]. The Mutual authentication flow is described in figure D-1: Explicit Secure Channel initiation Flow of Card Specification v2.2 [0].

5.3. Message Integrity

The C-MAC is generated by applying the NIST CMAC calculation (using S-MAC session key generated during the mutual authentication process) across the header and data field of an APDU command.

The card, on receipt of the message containing a C-MAC, using the same Secure Channel session key, performs the same operation and by comparing its internally generated C-MAC with the C-MAC received from the off-card entity is assured of the integrity of the full command.

If message data confidentiality has also been applied to the message, the C-MAC applies to the message data field after encryption has been performed.

The integrity of the sequence of commands being transmitted to the card is achieved by using the 16 byte C-MAC of a command as part of the input for the computation of the C-MAC of the next command. At any point in time, the last 16 byte C-MAC computed is part of the channel state and is referred to as the “MAC chaining value” further in this document. The first “MAC chaining value” is set to 16 bytes '00' (see section 6.2.3). This chaining (see Figure 6-3) ensures the card that all commands in a sequence have been received.

The integrity of the response is chained to the command sequence integrity by using the “MAC chaining value” as input as well for the computation of the R-MAC on responses (see Figure 6-3).

5.4. Message Data Confidentiality

The message data field is encrypted as specified in section 4.1.2 (using S-ENC Channel session key generated during the mutual authentication process) across the entire data field of the command message to be transmitted to the card, and if required also across the response transmitted from the card, regardless of its contents (clear text data and/or already protected sensitive data).

5.5. API and Security Level

A card implementing SCP03 shall implement the `SecureChannel` interface of the API specified in Card Specification v2.2 [0].

The following shall apply for the Security Level:

The Current Security Level of a communication not included in a Secure Channel Session shall be set to NO_SECURITY_LEVEL.

For Secure Channel Protocol '03', the Current Security Level established in a Secure Channel Session is a bitmap combination of the following values: AUTHENTICATED, C_MAC, R_MAC, C_DECRYPTION and R_ENCRYPTION.

The Current Security Level shall be set as follows:

- NO_SECURITY_LEVEL when a Secure Channel Session is terminated or not yet fully initiated;
- AUTHENTICATED after a successful processing of an EXTERNAL AUTHENTICATE command: AUTHENTICATED shall be cleared once the Secure Channel Session is terminated;
- C_MAC after a successful processing of an EXTERNAL AUTHENTICATE command with P1 indicating C-MAC (P1='x1' or 'x3'): C_MAC shall be cleared once the Secure Channel Session is terminated. Note that C_MAC is always combined with AUTHENTICATED and simultaneously set and cleared;
- C_DECRYPTION after a successful processing of an EXTERNAL AUTHENTICATE command with P1 indicating Command Encryption (P1= 'x3'): C_DECRYPTION shall be cleared once the Secure Channel Session is terminated. Note that C_DECRYPTION is always combined with AUTHENTICATED and C_MAC and simultaneously set and cleared;
- R_MAC after a successful processing of an EXTERNAL AUTHENTICATE command with P1 indicating R-MAC (P1='1x' or '3x'): R_MAC shall be cleared once the Secure Channel Session is terminated. Note that in this case R_MAC is always combined with AUTHENTICATED and simultaneously set and cleared. R_MAC may also be combined with C_MAC or C_DECRYPTION (according to the P1 value of the EXTERNAL AUTHENTICATE command) and simultaneously set and cleared;
- R_ENCRYPTION after a successful processing of an EXTERNAL AUTHENTICATE command with P1 indicating Response Encryption (P1= '3x'): R_ENCRYPTION shall be cleared once the Secure Channel Session is terminated. Note that R_ENCRYPTION is always combined with AUTHENTICATED and R_MAC and simultaneously set and cleared;

- R_MAC and no R_ENCRYPTION after a successful processing of a BEGIN R-MAC SESSION command: R_MAC shall be cleared after a successful processing of an END R-MAC SESSION command. Note that in this case R_MAC is combined with AUTHENTICATED and C_MAC or AUTHENTICATED, C_MAC and C_DECRYPTION depending on the pre-existing Current Security Level of the Secure Channel Session. R_MAC is set and cleared independently of AUTHENTICATED, C_MAC or C_DECRYPTION.
- R_MAC and R_ENCRYPTION after a successful processing of a BEGIN R-MAC SESSION command: R_MAC and R_ENCRYPTION shall be cleared after a successful processing of an END R-MAC SESSION command. Note that in this case R_MAC and R_ENCRYPTION are combined with AUTHENTICATED, C_MAC and C_DECRYPTION. R_MAC and R_ENCRYPTION are set and cleared independently of AUTHENTICATED, C_MAC or C_DECRYPTION.

5.6. Protocol Rules

In accordance with the general rules described in Chapter 10 of Card Specification v2.2 [0], the following protocol rules apply to Secure Channel Protocol '03':

- The successful initiation of a Secure Channel Session shall set the Current Security Level to the security level indicated in the EXTERNAL AUTHENTICATE command: it is at least set to AUTHENTICATED.
- The Current Security Level shall apply to the entire Secure Channel Session unless successfully modified at the request of the Application;
- When the Current Security Level is set to NO_SECURITY_LEVEL:
    - If the Secure Channel Session was aborted during the same Application Session, the incoming command shall be rejected with a security error;
    - Otherwise no security verification of the incoming command shall be performed. The Application processing the command is responsible to apply its own security rules.
- If a Secure Channel Session is active (i.e. Current Security Level at least set to AUTHENTICATED), the security of the incoming command shall be checked according to the Current Security Level regardless of the command secure messaging indicator:
    - When the security of the command does not match (nor exceeds) the Current Security Level, the command shall be rejected with a security error, the Secure Channel Session aborted and the Current Security Level reset to NO_SECURITY_LEVEL;
    - If a security error is found, the command shall be rejected with a security error, the Secure Channel Session aborted and the Current Security Level reset to NO_SECURITY_LEVEL;
    - In all other cases, the Secure Channel Session shall remain active and the Current Security Level unmodified. The Application is responsible for further processing the command.
- If a Secure Channel Session is aborted, it is still considered not terminated;
- The current Secure Channel Session shall be terminated (if aborted or still open) and the Current Security Level reset to NO_SECURITY_LEVEL on either:
    - Attempt to initiate a new Secure Channel Session (new INITIALIZE UPDATE command);
    - Termination of the Application Session (e.g. new Application selection);
    - Termination of the associated logical channel;
    - Termination of the Card Session (card reset or power off);
    - Explicit termination by the Application (e.g. invoking GlobalPlatform API).

6. Cryptographic Keys

6.1. AES Keys

Key	Usage	Length	Remark
Static Secure Channel Encryption Key (Key-ENC)	Generate session key for Decryption/Encryption (AES)	16, 24, 32 bytes	Mandatory
Static Secure Channel Message Authentication Code Key (Key-MAC)	Generate session key for Secure Channel authentication and Secure Channel MAC Verification/Generation (AES)	16, 24, 32 bytes	Mandatory
Data Encryption Key (Key-DEK)	Sensitive Data Decryption (AES)	16, 24, 32 bytes	Mandatory
Session Secure Channel Encryption Key (S-ENC)	Used for data confidentiality	Key-ENC length	Dynamically
Secure Channel Message Authentication Code Key for Command (S-MAC)	Used for data and protocol integrity	Key-MAC length	Dynamically
Secure Channel Message Authentication Code Key for Response (S-RMAC)	User for data and protocol integrity	Key-MAC length	Dynamically and Conditional

Table 6-1: Security Domain Secure Channel Keys

A Security Domain, including the Issuer Security Domain shall have at least one complete key set containing 3 AES keys (the 3 mandatory keys listed in Table 6-1) of a same length.

6.2. Cryptographic Usage

6.2.1. AES Session Keys

AES session keys shall be generated every time a Secure Channel is initiated and are used in the mutual authentication process. These same session keys may be used for subsequent commands if the Current Security Level indicates that secure messaging is required.

Session keys are generated to ensure that a different set of keys is used for each Secure Channel Session.

The session keys are derived from the static Secure Channel keys. The encryption key S-ENC is derived from Key-ENC. The Secure Channel MAC key S-MAC is derived from Key-MAC. Optionally (if the “i” parameter indicates R-MAC support), the Secure Channel R-MAC key S-RMAC is derived from Key-MAC. No AES session keys are generated for key and sensitive data encryption operations. That allows pre-processed data loading and simplifies the personalization process.

Key derivation shall use the data derivation scheme defined in section 4.1.5 with the following settings:

Derived Session Key	Key K_I used in PRF	Derivation Constant (see Table 4-1)
S-ENC	Key-ENC	'04'
S-MAC	Key-MAC	'06'
S-RMAC	Key-MAC	'07'

Table 6-2: AES Key Derivation Elements

The length of the session keys shall be reflected in the parameter "L" (i.e. ‘0080’ for AES-128 keys, ‘00C0’ for AES-192 keys and ‘0100’ for AES-256 keys).

The “context“ parameter shall be set to the concatenation of the host challenge (8 bytes) and the card challenge (8 bytes).

6.2.2. Challenges and Authentication Cryptograms

Both the card and the off-card entity (host) each generate a challenge and an authentication cryptogram. The off-card entity verifies the card cryptogram and the card verifies the host cryptogram. The cryptogram lengths shall be the same as the length of the challenges.

6.2.2.1. Card Challenge

As indicated in the “i” parameter (see Table 5-1), the card challenge shall either be random or pseudo-random.

If the SCP03 for a Security Domain is configured for pseudo-random challenge generation, the card challenge shall be calculated as follows:

- For each SCP03 keyset, the Security Domain shall have one sequence counter of three bytes length. Whenever a keyset is created or the whole keyset is replaced by a single PUT KEY or STORE DATA command, the sequence counter shall be set to zero.
- Whenever a challenge generation is triggered by an INITIALIZE UPDATE command, the sequence counter shall be incremented and the new value shall be used in the calculation described below. When the maximum value is reached, the INITIALIZE UPDATE command shall be rejected with “conditions of use not satisfied”.
- The card challenge (8 bytes) is calculated using the data derivation scheme defined in section 4.1.5 with the static key Key-ENC and the derivation constant set to "card challenge generation" (i.e. ‘02’). The length of the challenge shall be reflected in the parameter "L" (i.e. ‘0040’). The “context“ parameter shall be set to the concatenation of the sequence counter (3 bytes) and the AID of the application invoking the `SecureChannel` interface (5 to 16 bytes).

6.2.2.2. Card Authentication Cryptogram

The card cryptogram (8 bytes) is calculated using the data derivation scheme defined in section 4.1.5 with the session key S-MAC and the derivation constant set to "card authentication cryptogram generation". The length of the cryptogram shall be reflected in the parameter "L" (i.e. ‘0040’).

The “context“ parameter shall be set to the concatenation of the host challenge (8 bytes) and the card challenge (8 bytes).

6.2.2.3. Host Authentication Cryptogram

The host cryptogram (8 bytes) is calculated using the data derivation scheme defined in section 4.1.5 with the session key S-MAC and the derivation constant set to "host authentication cryptogram generation". The length of the cryptogram shall be reflected in the parameter "L" (i.e. ‘0040’).

The “context“ parameter shall be set to the concatenation of the host challenge (8 bytes) and the card challenge (8 bytes).

6.2.3. Message Integrity using Explicit Secure Channel Initiation

SCP03 mandates the use of a MAC on the EXTERNAL AUTHENTICATE command.

For the EXTERNAL AUTHENTICATE command MAC verification, the "MAC chaining value" is set to 16 bytes '00'.

Once the cryptograms are successfully verified, the full 16 byte C-MAC of the previous command becomes the "MAC chaining value" for the subsequent C-MAC verification / R-MAC generation.

6.2.4. APDU Command C-MAC Generation and Verification

A C-MAC is generated by an off-card entity: it uses the S-MAC key and is applied across the MAC chaining value concatenated with the full APDU command being transmitted to the card including the header (5 bytes) and the data field in the command message. (It does not include Le.)

Modification of the APDU command header and padding is required prior to the MAC operation being performed.

The Secure channel shall support a MAC of 8 bytes length (even if the AES block length is 16 bytes). Hence the 8 most significant bytes are considered.

The rules for APDU command header modification are as follows:

- The length of the command message (Lc) shall be incremented by 8 to indicate the inclusion of the C-MAC in the data field of the command message.
- The class byte shall be modified for the generation or verification of the C-MAC: The logical channel number shall be set to zero, bit 4 shall be set to 0 and bit 3 shall be set to 1 to indicate GlobalPlatform proprietary secure messaging. If the Secure Channel Session is occurring on a Supplementary Logical Channel, the class byte shall be modified after the C-MAC generation to indicate the logical channel number. If logical channel number 4 to 19 is used, the GlobalPlatform proprietary secure messaging is indicated by setting bit 6 to 1 – see tables 11-11 and 11-12 of Card Specification v2.2 [0]. Conversely the logical channel number card is discarded and, if required, the secure messaging indication is adjusted for the verification. The logical channel number is not part of the integrity protection by the channel because is it established independently and outside of the scope of the Secure Channel establishment.

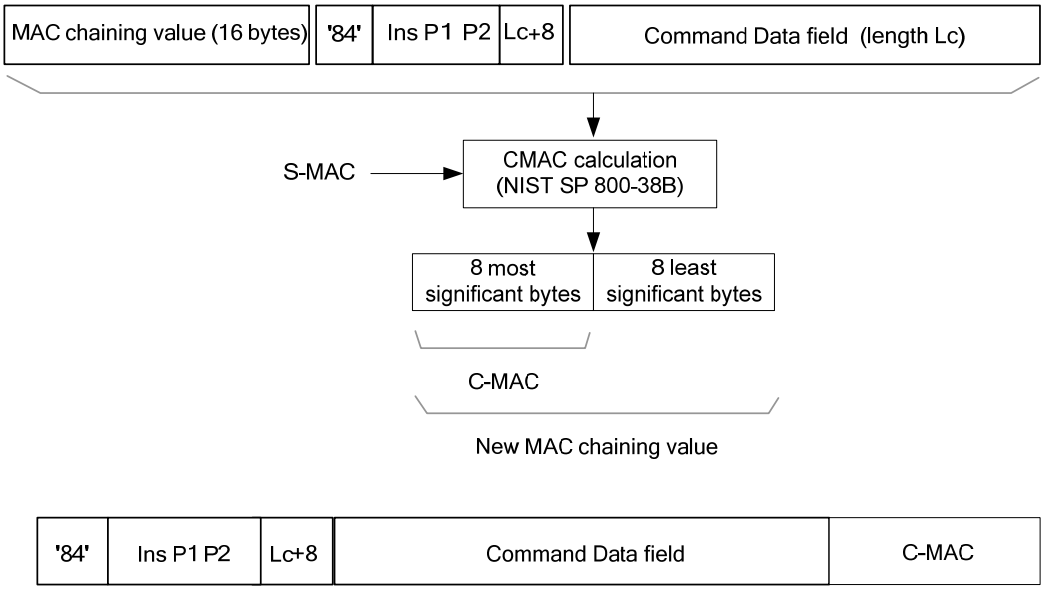

Figure 6-1: APDU C-MAC Generation

6.2.5. APDU Response R-MAC Generation and Verification

No R-MAC shall be generated and no protection shall be applied to a response when status bytes SW1 and SW2 indicate a system error: in this case only status bytes shall be returned in the response.

The EXTERNAL AUTHENTICATE command/response doesn't return R-MAC.

The R-MAC is made of the first 8 bytes of the CMAC computed on the message made of the MAC chaining value, the response data field (if present) and the status bytes. The R-MAC calculation uses the S-RMAC key.

The R-MAC calculation is illustrated in Figure 6-2.

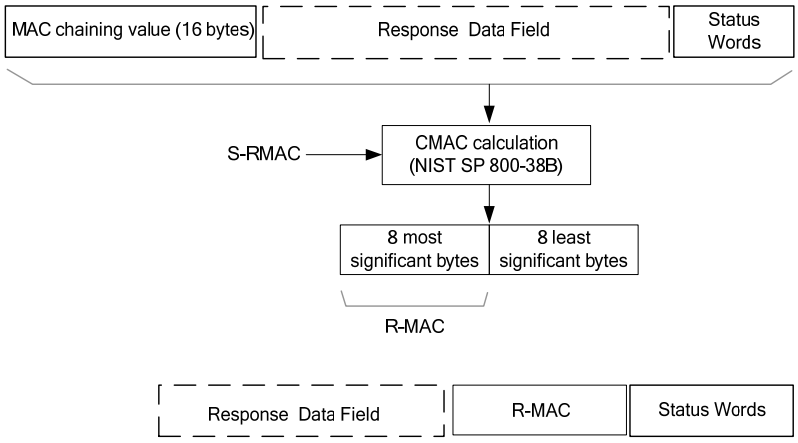

Figure 6-2: APDU R-MAC Generation

The off-card entity shall perform the same CMAC calculation on the response and use the same R-MAC session key employed by the card in order to verify the R-MAC.

The computed R-MAC becomes part of the response message.

Figure 6-3 illustrates the combined MAC chaining for command and responses.

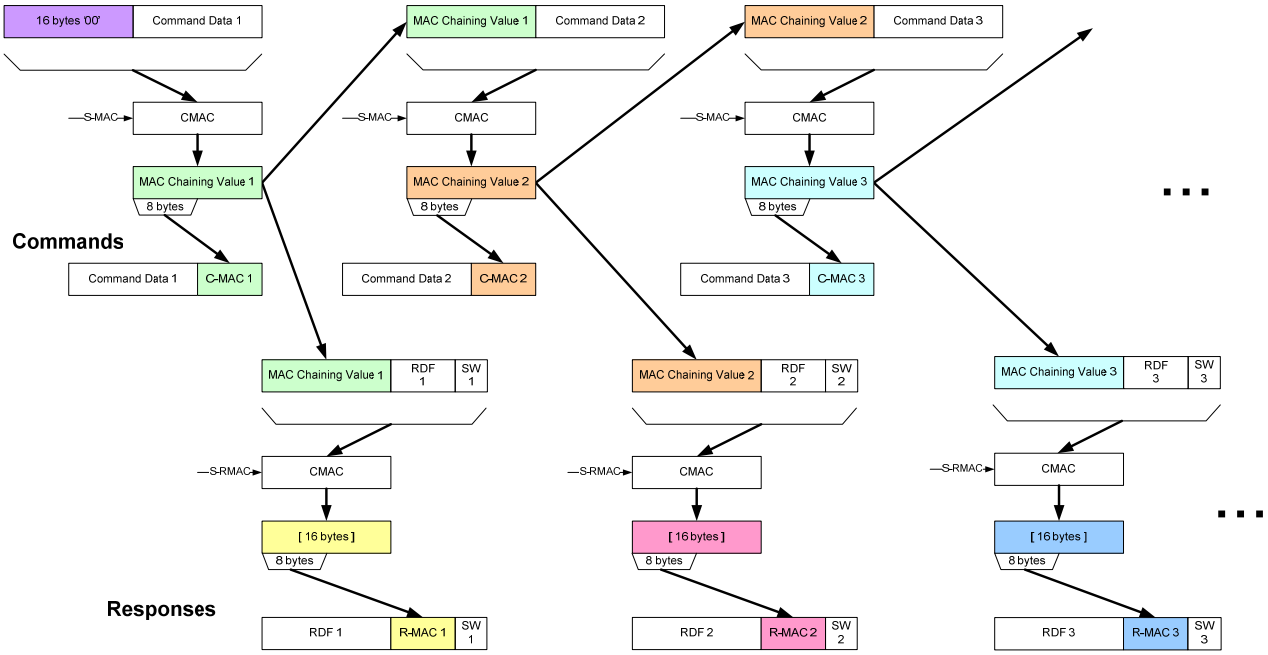

Figure 6-3: MAC Chaining

Via the MAC changing value, consecutive commands are chained to each other, protecting their sequence.

Responses are linked to the respective command via the MAC changing value. As R-MAC uses a different session key than C-MAC, the same MAC chaining value can be used for the response and the next command. The scheme is adapted to the features of SCP03, where

- R-MAC is optional, and
- R-MAC may be switched on and off during a secure channel session (see BEGIN/END R-MAC SESSION commands).

6.2.6. APDU Command C-MAC and C-DECRYPTION Generation and Verification

This section applies when both command confidentiality (C-DECRYPTION) and integrity (C-MAC) are required.

Depending on the security level defined in the initiation of the Secure Channel, all subsequent APDU commands within the Secure Channel may require secure messaging and such as use of a C-MAC (integrity) and encryption (confidentiality).

No encryption shall be applied to a command where there is no command data field: in this case the message shall be protected as defined in section 6.2.4 APDU Command C-MAC Generation and Verification. Otherwise the Off-Card Entity performs the process detailed hereafter.

The Off-Card Entity first encrypts the Command Data field and then computes the C-MAC on the command with the ciphered data field as described in section 6.2.4 APDU Command C-MAC Generation and Verification.

The message encryption and decryption is generated by the Off-Card Entity as defined below:

The command message encryption and decryption uses the Secure Channel encryption (S-ENC) session key and the AES encryption in CBC Mode. Prior to encrypting the data, the data shall be padded as defined in section 4.1.4. This padding becomes part of the data field.

The final Lc value (Lcc) is the sum of:

initial Lc + length of the padding + length of C-MAC

The ICV shall be calculated as follows:

- A counter shall be incremented for each command (i.e. for each pair of command and response APDU) within a secure channel session.
- The starting value shall be 1 for the first command after a successful EXTERNAL AUTHENTICATE.
- The binary counter value shall be left padded with zeroes to form a full block.
- This block shall be encrypted with S-ENC; the result shall be used as ICV.

This scheme fulfils the requirement in NIST 800-38A [3] for unpredictable ICVs if using CBC mode.

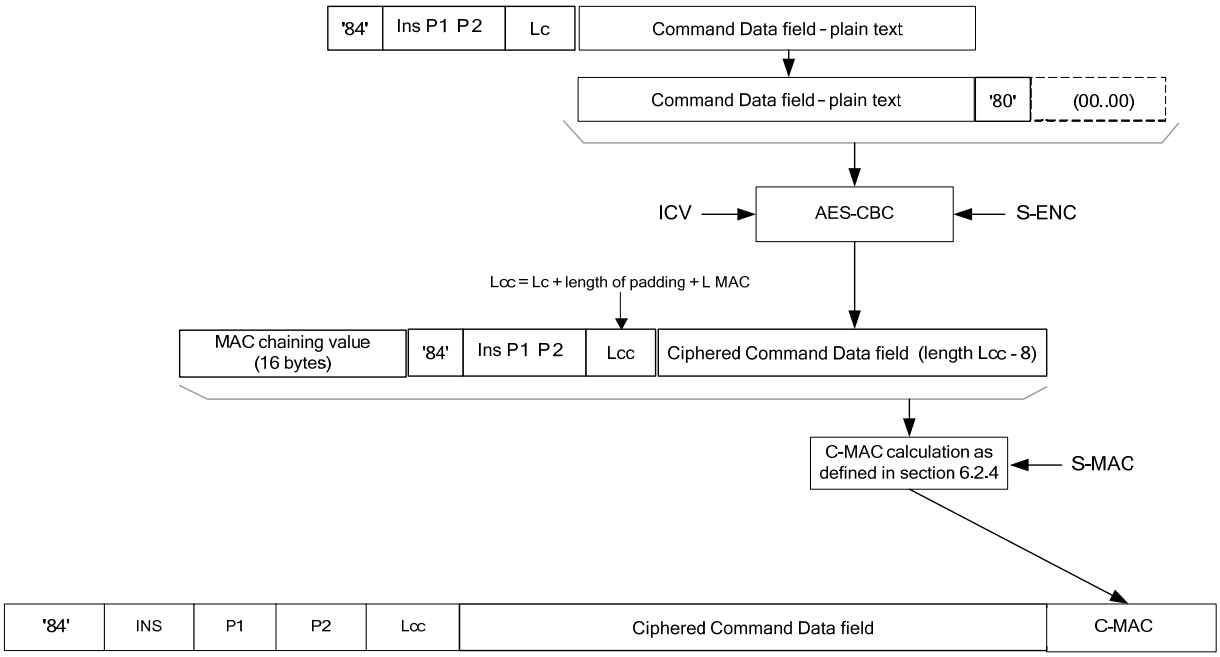

Figure 6-4: APDU Command Data Field Encryption

6.2.7. APDU Response R-MAC and R-ENCRYPTION Generation and Verification

This section applies when both response confidentiality (R-ENCRYPTION) and integrity (R-MAC) are required.

Depending on the security level defined in the initiation of the Secure Channel, all subsequent APDU responses within the Secure Channel may require secure messaging and such as use of a R-MAC (integrity) and encryption (confidentiality).

No encryption shall be applied to a response where there is no response data field: in this case the message shall be protected as defined in section 6.2.5 APDU Response R-MAC Generation and Verification. Otherwise the Card performs the process detailed hereafter.

The Card first encrypts the Response Data field and then computes the R-MAC on the response with the ciphered data field as described in section 6.2.5 APDU Response R-MAC Generation and Verification.

The message encryption is generated by the Card as defined below:

The response message encryption and decryption uses the Secure Channel encryption (S-ENC) session key and the AES encryption in CBC Mode. Prior to encrypting the data, the data shall be padded as defined in section 4.1.4. This padding becomes part of the data field.

The ICV shall be calculated as follows:

- The padded counter block from the generation the ICV for command encryption shall also be used to generate the ICV for response encryption, however, with one additional intermediate step: Before encryption, the most significant byte of this block shall be set to '80'.
- This block shall be encrypted with S-ENC; the result shall be used as ICV.

This scheme fulfils the requirement in NIST 800-38A [3] for unpredictable ICVs if using CBC mode. The modification in the most significant byte guarantees that the ICVs for R-ENCRYPTION are different to those used for C-DECRYPTION.

The final response APDU shall be the concatenation of the ciphered data, the R-MAC and the Status Word.

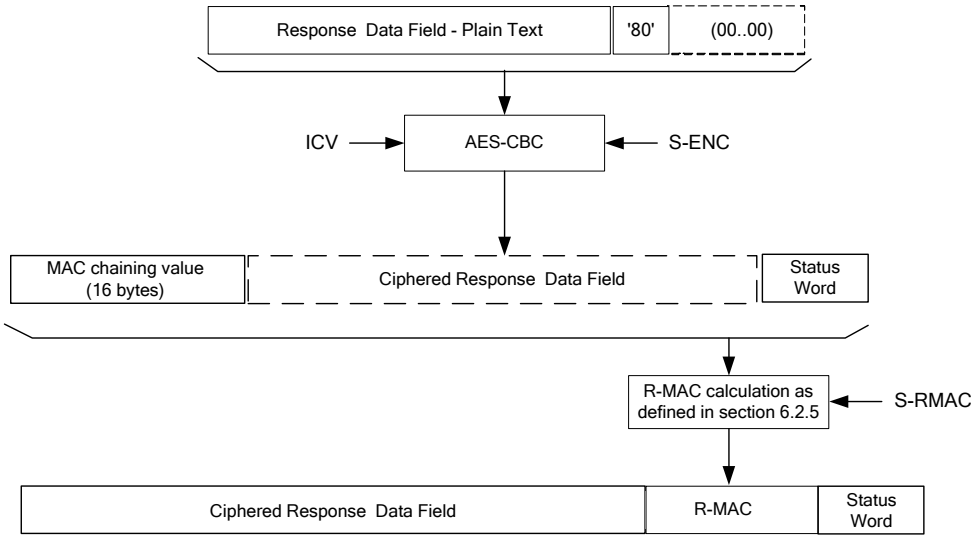

Figure 6-5: APDU Response Data Field Encryption

6.2.8. Key Sensitive Data Encryption Decryption

Key data encryption is used when transmitting key sensitive data to the card and is over and beyond the security level required for the Secure Channel. For instance all AES keys transmitted to a card should be encrypted.

The Data encryption process uses the static data encryption key (Key-DEK) and the encryption method as described in section 4.1.2 Encryption/Decryption.

If the sensitive data to be encrypted are AES keys (16 or 32 byte long), for instance for a Put Key command, then no padding is required for the data field prior to encryption as data block are multiple of 16 byte long. For the 24-byte AES keys, padding of 8 arbitrary bytes shall be appended prior to encryption. For the encryption of other keys see section 7.2. For other sensitive data, padding is application specific and is out of the scope of this document.

The AES CBC encryption with ICV set to zero is performed across the key sensitive data and the result of each encryption becomes part of the encrypted key data. This encrypted key data becomes part of the "clear text" data field in the command message.

The on-card decryption of key data is the exact opposite of the above operation.

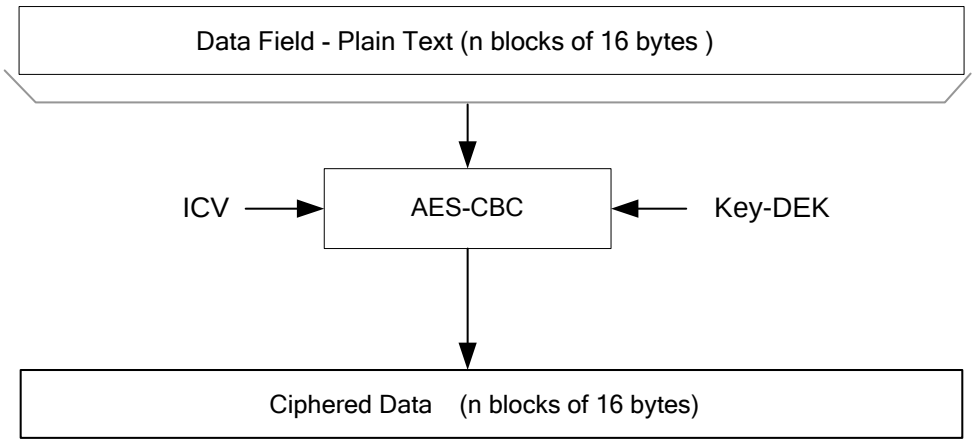

Figure 6-6: Sensitive Data Encryption

7. Commands

The following table presents the commands involved in Secure Channel Initiation and R-MAC Session Management.

Command	Secure Channel Initiation
INITIALIZE UPDATE	✓
EXTERNAL AUTHENTICATE	✓
BEGIN R-MAC SESSION	
END R-MAC SESSION	

Table 7-1: SCP03 Command Support

Ticks (✓) denote that support of the command is mandatory.

Blank cells denote that the support of the command is optional.

7.1. Secure Channel Commands

7.1.1. INITIALIZE UPDATE Command

See *Appendix D.4.1* of Card Specification v2.2 [0] for the structure of the INITIALIZE UPDATE command.

If any of the mandatory keys listed in section 6.1 is missing in the targeted key set version, the INITIALIZE UPDATE command shall fail with error condition "Referenced data not found" as defined in [0].

7.1.1.1. Data Field Returned in the Response Message

The data field of the response message shall contain the concatenation without delimiters of the following data elements:

Name	Length	Presence
Key diversification data	10 bytes	Mandatory
Key information	3 bytes	Mandatory
Card challenge	8 bytes	Mandatory
Card cryptogram	8 bytes	Mandatory
Sequence Counter	3 bytes	Conditional

Table 7-2: INITIALIZE UPDATE Response Message

The key diversification data is data typically used by a backend system to derive the card static keys.

The key information includes the Key Version Number, the Secure Channel Protocol identifier, here '03', and the Secure Channel Protocol “i” parameter used in initiating the Secure Channel Session.

The card challenge is an internally generated random or pseudo random number.

The card cryptogram is an authentication cryptogram.

Sequence Counter is only present when SCP03 is configured for pseudo-random challenge generation.

7.1.2. EXTERNAL AUTHENTICATE Command

Except for the Reference control parameter P1 the GlobalPlatform External Authenticate is compliant with Appendix *D.4.2* of Card Specification v2.2 [0] for the structure of the EXTERNAL AUTHENTICATE command.

7.1.2.1. Reference Control Parameter P1 - Security Level

The reference control parameter P1 defines the level of security for all secure messaging commands following this EXTERNAL AUTHENTICATE command and within the Secure Channel Session.

b8	B7	B6	b5	b4	b3	b2	B1	Description
0	0	1	1	0	0	1	1	C-DECRYPTION, R-ENCRYPTION, C-MAC and R-MAC
0	0	0	1	0	0	1	1	C-DECRYPTION, C-MAC and R-MAC
0	0	0	1	0	0	0	1	C-MAC and R-MAC
0	0	0	0	0	0	1	1	C-DECRYPTION and C-MAC.
0	0	0	0	0	0	0	1	C-MAC
0	0	0	0	0	0	0	0	No secure messaging expected.

Table 7-3: EXTERNAL AUTHENTICATE Reference Control Parameter P1

7.1.3. BEGIN R-MAC SESSION Command

7.1.3.1. Definition and Scope

The BEGIN R-MAC SESSION command is used to initiate additional response security. The BEGIN R-MAC SESSION command may only be issued to the card within a secure channel. It may only be used to increase the security of the responses and only if command messages use at least the same security level.

7.1.3.2. Command Message

The BEGIN R-MAC SESSION command message is coded according to the following table:

Code	Value	Meaning
CLA	'80' - '87', 'C0' - 'CF' or 'E0' - 'EF'	Please refer to section 11.1.4 of Card Specification v2.2 [0]
INS	'7A'	BEGIN R-MAC SESSION
P1	'xx'	Reference control parameter P1
P2	'01'	Reference control parameter P2
Lc	'XX''	Length of data field, if any
Data	'xx xx…'	BEGIN R-MAC SESSION data and C-MAC, if needed
Le		Not present

Table 7-4: BEGIN R-MAC SESSION Command Message

7.1.3.3. Reference Control Parameter P1

The reference control parameter P1 defines the level of security for all subsequent APDU response messages following this BEGIN R-MAC SESSION command (it does not apply to this command).

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	1	1	0	0	0	0	R-ENCRYPTION and R-MAC
0	0	0	1	0	0	0	0	R-MAC

Table 7-5: BEGIN R-MAC SESSION Reference Control Parameter P1

When P1 is set to '10' each APDU response message during the R-MAC session includes an R-MAC. This setting may only be used if the secure channel session does not use R-MAC.

When P1 is set to '30' each APDU response message during the R-MAC session uses R-MAC and R-ENCRYPTION. This setting may only be used if the secure channel session does not use R-ENCRYPTION.

7.1.3.4. Reference Control Parameter P2

The reference control parameter P2 defines the beginning of the session for APDU response message integrity.

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	0	0	0	0	0	1	Begin R-MAC session

Table 7-6: BEGIN R-MAC SESSION Reference Control Parameter P2

7.1.3.5. Data Field Sent in the Command Message

The data field of the BEGIN R-MAC SESSION contains an LV coded ‘data’ element and optionally a C-MAC. The card does not interpret the ‘data’. However since it is included in R-MAC calculation, this gives the off-card entity the possibility to include a challenge in the R-MAC.

7.1.3.6. Data Field Returned in the Response Message

The data field of the response message is not present.

7.1.3.7. Processing State Returned in the Response Message

A successful execution of the command shall be indicated by status bytes '90' '00'.

The card will respond with a '6985' status word without closing the secure channel in the following cases:

- If R-ENCRYPTION is specified in a previous EXTERNAL AUTHENTICATE command;
- If P1 is set to '10' and R-MAC is specified in a previous EXTERNAL AUTHENTICATE command;
- If C-MAC was not specified in a previous EXTERNAL AUTHENTICATE command;
- If P1 is set to '30' and C-DECRYPTION was not specified in a previous EXTERNAL AUTHENTICATE command.

This command may return a general error condition as listed in section 11.1.3 General Error Condition of Card Specification v2.2 [0].

7.1.4. END R-MAC SESSION Command

7.1.4.1. Definition and Scope

The END R-MAC SESSION command is used to terminate the additional response security that was initiated by the preceding BEGIN R-MAC SESSION. The Secure Channel session returns to its original security settings. The END R-MAC SESSION command may be issued to the card at any time during an R-MAC session. In addition to the explicit “END RMAC SESSION” command, the RMAC session should be terminated if the secure channel is closed or the card is reset.

7.1.4.2. Command Message

The END R-MAC SESSION command message is coded according to the following table:

Code	Value	Meaning
CLA	'80' - '87', 'C0' - 'CF' or 'E0' - 'EF'	Please refer to section 11.1.4 of Card Specification v2.2 [0]
INS	'78'	END R-MAC SESSION
P1	'00'	Reference control parameter P1
P2	'03'	Reference control parameter P2
Lc	'xx'	Length of data field, if any
Data	'xx xx…'	C-MAC, if needed
Le	'00'	

Table 7-7: END R-MAC SESSION Command Message

7.1.4.3. Reference Control Parameter P1

Reference control parameter P1 shall always be set to '00'.

7.1.4.4. Reference Control Parameter P2

The reference control parameter P2 is coded according to the following table:

b8	b7	b6	B5	b4	b3	b2	b1	Description
0	0	0	0	0	0	1	1	End R-MAC session & return R-MAC

Table 7-8: END R-MAC SESSION Reference Control Parameter P2

7.1.4.5. Data Field Sent in the Command Message

The data field of the command message may optionally contain a C-MAC.

7.1.4.6. Data Field Returned in the Response Message

The data field of the response message contains the R-MAC of the current R-MAC session.

7.1.4.7. Processing State Returned in the Response Message

A successful execution of the command shall be indicated by status bytes '90' '00'.

This command may return a general error condition as listed in section 11.1.3 General Error Condition of Card Specification v2.2 [0].

7.2. PUT KEY Command (AES Key-DEK)

This section applies if the Key-DEK used to decrypt the new key is an AES key.

AES keys must be encrypted by an AES Key-DEK with the same or higher strength.

The P1 and P2 parameters are formatted according to GP CS v2.2 [0].

7.2.1. Data Field Sent in the Command Message

The general key data field for keys or key components encrypted by an AES Key-DEK takes is specified in Table 7-9, which provides a precision to GP CS v2.2 [0] table 11-68.

If the length of the key or key component is not an integer multiple of the block length of 16 bytes, padding of arbitrary bytes shall be appended prior to encryption to fill the last block. Ciphering shall be done as specified in section 6.2.8.

Name		Value	Length
Key type		see GP CS v2.2 [0] table 11-68	1 byte
Length of key or key component data		coding see GP CS v2.2 [0] table 11-68	1-3 bytes
Key or key component data	Length of the key or key component	'01' – '80', or '81 80' – '81 FF', or '82 01 00' – '82 FF FF'	1-3 bytes
	Ciphered key or key component	'xxxx…'	N *16 bytes
Length of key check value		see GP CS v2.2 [0] table 11-68	1 byte
Key check value		'xxxx…'	K bytes

Table 7-9: Key Data Field (AES Key-DEC) - Basic

If an AES key is encrypted by an AES Key-DEK, the key data field takes the format according to Table 7-10.

Name		Value	Length
Key type		'88'	1 byte
Length of the AES key data		'11' or '21'	1 byte
AES KEY DATA	Length of the AES key	'10' or '18' or '20'	1 byte
	Ciphered AES key	Ciphered AES key	16, 32 bytes
Key check value length		'03'	1 byte
Key check value		Computed according to 7.2.2	3 bytes

Table 7-10: AES Key Data Field - Basic

The same precisions to the key or key component data shall apply for the extended key data field.

7.2.2. Key check value for AES Key

The key check value is calculated and checked by AES encrypting one block of 16 bytes with value '01' and taking the three highest order bytes.

7.3. STORE DATA (AES Key-DEK)

This command shall be coded according to GP CS v2.2 Amd. A [8] section 4.10 with the following precisions:

- The (static) Key-DEK shall be used for sensitive data encryption/decryption.
- If the length of the sensitive data is not an integer multiple of the block length of 16 bytes, padding of arbitrary bytes shall be appended prior to encryption to fill the last block.
- The key check value for AES keys shall be calculated as specified in 7.2.2.

8. Tables of Figures and Tables