EMV®
Card Personalisation Specification

Version 2.0

August 2021

Legal Notice

The EMV® Specifications are provided “AS IS” without warranties of any kind, and EMVCo neither assumes nor accepts any liability for any errors or omissions contained in these Specifications. EMVCO DISCLAIMS ALL REPRESENTATIONS AND WARRANTIES, EXPRESS OR IMPLIED, INCLUDING WITHOUT LIMITATION IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT, AS TO THESE SPECIFICATIONS.

EMVCo makes no representations or warranties with respect to intellectual property rights of any third parties in or in relation to the Specifications. EMVCo undertakes no responsibility to determine whether any implementation of the EMV® Specifications may violate, infringe, or otherwise exercise the patent, copyright, trademark, trade secret, know-how, or other intellectual property rights of third parties, and thus any person who implements any part of the EMV® Specifications should consult an intellectual property attorney before any such implementation.

Without limiting the foregoing, the Specifications may provide for the use of public key encryption and other technology, which may be the subject matter of patents in several countries. Any party seeking to implement these Specifications is solely responsible for determining whether its activities require a license to any such technology, including for patents on public key encryption technology. EMVCo shall not be liable under any theory for any party’s infringement of any intellectual property rights in connection with the EMV® Specifications.

Revision Log

The following table lists the version history for the *EMV Card Personalisation Specification*. EMVCo Specification Bulletins provide the detailed updates made with each specification release.

Version	Release Date	Associated Specification Bulletins
1.0	October 2003	EMVCo Specification Update: Bulletin no. 23: First edition October 2003: Corrections and Clarifications to EMV Card Personalisation Specification
1.0	February 2005	EMVCo Specification Update: Bulletin no. 40: First edition February 2005: Additional Corrections and Clarifications to EMV Card Personalisation Specification
1.1	July 2007	None
2.0	August 2021	None (refer to section 1.5)

Contents

Figures

Tables

* This page intentionally left blank *

1 Introduction

1.1 Purpose

Card personalisation is one of the major cost components in the production of EMV cards. This specification standardises the EMV card personalisation process with the objective of reducing the cost of personalisation.

In today's environment, there are numerous methods of personalising EMV cards and many vendors providing the systems to personalise these cards. Each time a native card is developed, or a new application released, issuers and personalisation vendors are obliged to expend significant time and money to develop the corresponding personalisation process. In addition, these cards are typically personalised using proprietary commands, often making it difficult for card issuers to source cards from alternative suppliers or bureaus.

This specification standardises EMV card personalisation leading to faster, more efficient and more economical solutions. It offers benefits which include lower set up costs, faster time to market, greater choice of supplier (card and personalisation bureau) and an enhanced ability to switch suppliers.

1.2 Audience

This document is intended for stakeholders involved in personalisation processes.

In particular, there are three intended audiences for this document:

1. **Designers of EMV applications**
   This audience will use this document as one of the inputs to their design process. The areas that are impacted by this document are:
   - Design of the file and data structure for the EMV application on the IC card.
   - Design and processing of the personalisation commands.
2. **Designers of Personalisation Device systems**
   This audience will use this document as a specification for part of the design for their processing, in particular the input and output interfaces.
3. **Designers of Data Preparation systems**
   This audience will use this document as a specification for part of the design for their processing, in particular the output interface.

1.3 Scope

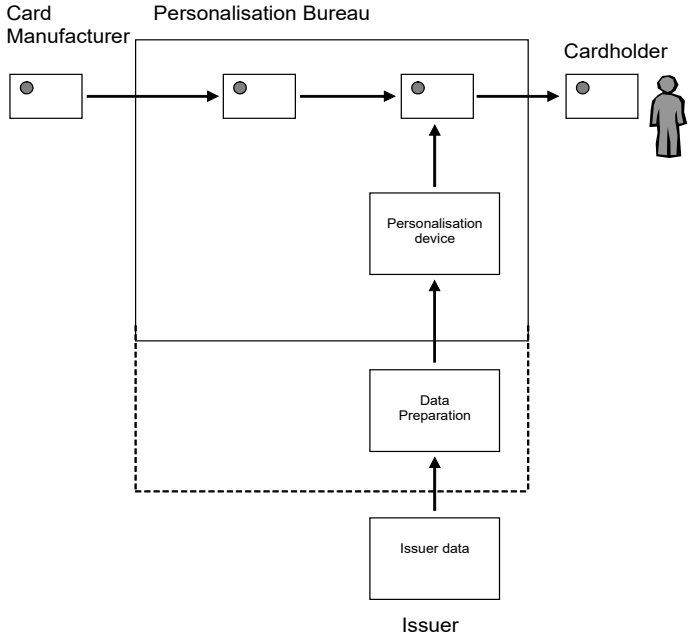
In this specification, card personalisation means the use of data personalisation commands that are sent to a card that already contains the basic EMV application. This is sometimes referred to as "on-card" personalisation. The specification does not cover cards where an application load file is personalised before being loaded onto the card.

In terms of the lifecycle of the card, card personalisation is assumed to take place after pre-personalisation (see Definitions) and prior to card issuance. However non-EMV applications may well use the same personalisation process as defined in this specification. Other card personalisation activities – embossing, magnetic stripe encoding and the personalisation of non-EMV IC applications – are not covered.

The interface between the card issuer and the data preparation system is not covered.

1.4 Involved Entities

The terms used in the diagram are described in the Definitions section below.



1.5 Changes in Version 2.0

The use of AES as an alternative block cipher to DES is introduced. The Card Personalisation Specification now supports use of both SCP02 (DES based) and SCP03 (AES based) for the secure channel terminating on the card.

The use of both protocols in the context of EMV personalisation is described here in this document. The Secure Channel Protocol (SCP) '02' branch as described in this document is compliant with Appendix E of the GlobalPlatform Card Specification. The Secure Channel Protocol (SCP) '03' branch is as defined in Amendment D of the GlobalPlatform Card Specification (including the S8/S16 data structure variants).

1.6 Indirect & Direct Methods of Establishing & Using Secure Personalisation Channels

Two methods are included in this specification – the Indirect Method and the Direct Method.

1.6.1 Indirect Method

This method is defined in all versions of the EMV Card Personalisation Specification. The rationale is that the Data Preparation System should not need to have knowledge of the ICC data used to establish the secure channel for a particular target card/application. However, it is necessary for the Data Preparation System to be aware of whether the cards being targeted support SCP02 or SCP03 and act accordingly e.g. in data formatting to reflect block size differences between DES and AES - as well as of course to use AES cryptographic constructions. Data Preparation systems that have been targeting SCP02 cards will need to make the corresponding changes in order to target SCP03 cards. The required changes have been kept to a minimum. The method assumes two security zones, one between the Data Preparation System and the Personalisation Device, and a second zone between the Personalisation Device and the ICC.

The role of the Personalisation Device is:

- To receive the DGI data and Personalisation Device Instructions (PDI) from the Data Preparation System
- To establish the secure channel to the card
- To decrypt/re-encrypt the sensitive data (application keys & PIN), and
- To create and send personalisation APDU commands to the card.

An example of the process of Indirect Method is shown in Annex B.

1.6.2 Direct Method

This method is in versions (1.1) and higher of the EMV Card Personalisation Specification. The rationale is to reduce the time taken by the Personalisation Device by pre-computing APDU commands in the Data Preparation System. The method assumes a single security zone between the Data Preparation System and the ICC. The Personalisation Device does not need to create APDU commands; it simply passes on the commands received from the Data Preparation System.

However, the Data Preparation System needs to carry out additional preparation work. If the Secure Channel Key Set in the card (K_{ENC}, K_{MAC}, K_{DEC}) is not known, a pre-personalisation process is needed to load a derived key set known by the Data Preparation System on to the ICC. The ICC needs to produce a pseudo-random number, which the Data Preparation System can predict. The Data Preparation System also needs to be able to predict the Secure Channel Sequence Counter. The Data Preparation System must also know if the cards being targeted support SCP02 or SCP03 and for SCP03 whether they support the S8 or S16 variant of SCP03. The Data Preparation System uses this information to pre-compute the personalisation APDU commands for the ICC application.

Note: In terms of the interface between the ICC and the Personalisation Device, the formats of the personalisation messages are the same for either the Indirect Method or the Direct Method. The main difference between the two methods is the allocation of processing between the Personalisation Device and the Data Preparation System.

1.7 Normative Specification References

The following specifications contain provisions that are referenced in this specification. The latest version including all published amendments shall apply unless a publication date is explicitly stated.

Table 1-1 Normative Specification References

Reference	Publication Name
EMV Version 4.3	Integrated Circuit Card Specifications for Payment Systems Book 1 – Application Independent ICC to Terminal Interface Requirements Integrated Circuit Card Specifications for Payment Systems Book 2 – Security and Key Management Integrated Circuit Card Specifications for Payment Systems Book 3 – Application Specification
EMVCo Specification Update: Bulletin no. 23 First edition	EMVCo Specification Update: Bulletin no. 23: First edition October 2003: Corrections and Clarifications to EMV Card Personalisation Specification
EMVCo Specification Update: Bulletin no. 40 First edition	EMVCo Specification Update: Bulletin no. 40: First edition February 2005: Additional Corrections and Clarifications to EMV Card Personalisation Specification
GlobalPlatform Card Specification	GlobalPlatform Technology, Card Specification V2.3.1
GlobalPlatform Messaging Specification	GlobalPlatform Messaging Specification V1.0
GlobalPlatform Systems Profiles Specification	GlobalPlatform Systems Profiles Specification - V1.1 (available from the GlobalPlatform Secretariat)
GlobalPlatform Secure Channel Protocol '03' [SCP03]	GlobalPlatform Technology, Secure Channel Protocol '03', Card Specification v2.3 – Amendment D, Version 1.2, Public Release, April 2020, Document Reference: GPC_SPE_014

1.8 Normative Industry Standards

The following standards contain provisions that are referenced in this specification. The latest version including all published amendments shall apply unless a publication date is explicitly stated.

Table 1-2 Industry Standards

Reference	Publication Name
ISO/IEC 7816-3	Identification cards - Integrated circuit(s) cards with contacts - Part 3: Electronic signals and transmission protocols
ISO/IEC 7816-4	Identification cards - Integrated circuit(s) cards with contacts - Part 4, Inter-industry commands for interchange
ISO/IEC 7816-5	Identification cards - Integrated circuit(s) cards with contacts - Part 5, Numbering system and registration procedure for application identifiers
ISO/IEC 7816-6	Identification cards - Integrated circuit(s) cards with contacts - Part 6, Inter-industry data elements
ISO 9564-1	Banking – Personal Identification Number (PIN) – Part 1- Basic principles and requirements for online PIN handling in ATM and POS systems
ISO/IEC 9797-1	Information Technology – Security Techniques – Message Authentication Codes – Part 1: Mechanisms using a block cipher
ISO/IEC 10116	Information Technology – Modes of Operation of an n-bit block cipher algorithm
ISO/IEC 11770-6	Information technology – Security techniques – Key management — Part 6: Key derivation
ISO/IEC 18033-3	Information Technology – Security techniques – Encryption Algorithms – Part 3: Block Ciphers
NIST 800-38B	NIST Special Publication 800-38B – Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication
NIST 800-108	NIST Special Publication 800-108 – Recommendation for Key Derivation Using Pseudorandom Functions
NIST FIPS 197	NIST Federal Information Processing Standards Publication 197 – Announcing the Advanced Encryption Standard (AES)

1.9 Definitions

The following terms are used in this specification:

Table 1-3 Definitions

Term	Definition
Application	An application resident in an EMV card.
Application Command	For this document specifically, an APDU command acceptable to an application after the application has been selected.
Card	An IC payment card as defined by a payment system.
Card Personalisation	The personalisation of application data within a card, using personalisation commands.
Data Preparation	The process of preparing and formatting data, ready for sending to a personalisation device.
Personalisation	The personalisation of application data to enable a card to be used by a cardholder.
Personalisation Command	A command sent to a selected EMV application in order to personalise application data.
Personalisation Device	A device that accepts data from a data preparation system and sends personalisation commands to a card.
Pre-personalisation	The initialisation of card data prior to personalisation.
Processing Step	The action to be performed by the personalisation device.

1.10 Abbreviations

The abbreviations listed in Table 1.4 are used in this specification. Additional abbreviations can be found at the end of this specification in section 7.

Table 1-4 Abbreviations

Abbreviation	Description
AES	Advanced Encryption Standard
AID	Application Identifier
ANSI	American National Standards Institute
ASCII	American Standard Code for Information Interchange

Abbreviation	Description
ATR	Answer-to-Reset
BER	Basic Encoding Rules
BIN	Bank Identification Number
CBC	Cipher Block Chaining
CDA	Combined DDA/Application Cryptogram Generation Authentication
CLA	Class Byte
C-MAC	Command Message Authentication Code, a service protecting the integrity of a command APDU by using MACs
CMAC	(Block) Cipher based Message Authentication Code
CPLC	Card Production Life Cycle
CPS	Card Personalisation Specification
CRN	Card and Application Management System (CAMS) Reference Number
CRT	Chinese Remainder Theorem
CSN	Chip Serial Number
DDA	Dynamic Data Authentication
DEK	Data Encryption Key
DES	Data Encryption Standard
DF	Dedicated File
DGI	Data Grouping Identifier
ECB	Electronic Code Book
EF	Elementary File
FCI	File Control Information
FCP	File Control Parameter
HSM	Hardware Security Module
IC	Integrated Circuit
ICC	Integrated Circuit Card
ICV	Initial Chaining Vector

Abbreviation	Description
ID	Identifier
IEC	International Electrotechnical Commission
IIN	Issuer Identification Number
INS	Instruction Byte
ISO	International Organization for Standardization
IV	Initialisation Vector
KDF	Key Derivation Function
KEK	Key Encryption Key
KMC	Master Key for Personalisation Session Keys (DES or AES)
LSB	Least Significant Byte
M/O	Mandatory or Optional
MAC	Message Authentication Code, a value or a function generating such values, for example ANSI Retail MAC, CMAC
MIC	Module Identifier Code
MSB	Most Significant Byte
PAN	Application Primary Account Number
PDI	Personalisation Device Instructions
PEK	PIN Encryption Key
PIN	Personal Identification Number
PK	Public Key
PRF	Pseudo Random Function
PS	Processing Step
RFU	Reserved for Future Use (values to be ignored)
R-MAC	Response Message Authentication Code, a service protecting the integrity of a response APDU by using MACs
RSA	Rivest, Shamir and Adleman (Cryptographic Algorithm)
SDA	Static Data Authentication
SFI	Short File Identifier

Abbreviation	Description
SKU	Personalisation Session Key
TK	Transport Key
TLV	Tag, Length, Value
var.	Variable

1.11 GlobalPlatform Secure Channel Protocol Status

The following table provides the Protocol Version Number status by GlobalPlatform Secure Channel Protocol Version Number underpinning the CPS.

Table 1-5 Protocol Version Numbers

Protocol Version Number	Status
SCP02	Legacy
SCP03	Active

1.12 Supporting Documentation

There is no further supporting documentation referenced.

1.13 Terminology and Conventions

The following words are used often in this specification and have a specific meaning:

Shall or Must

Defines a product or system capability which is mandatory.

May

Defines a product or system capability which is optional or a statement which is informative only and is out of scope for this specification.

Should

Defines a product or system capability which is recommended.

The following notations apply:

Hexadecimal Notation

Values expressed in hexadecimal form are enclosed in single quotes (e.g., '_'). For example, 27509 decimal is expressed in hexadecimal as '6B75'.

Letters used to express constant hexadecimal values are always upper case ('A' - 'F'). Where lower case is used, the letters have a different meaning explained in the text.

Binary Notation

Values expressed in binary form are followed by a lower case “b”. For example, '08' hexadecimal is expressed in binary as 00001000b (most significant bit first).

Length Fields

Length fields are “big-endian” encoded. For example, if a two-byte length field has a hexadecimal value of '13F' (319 in decimal), it is encoded as '013F'.

Operators and Functions

$\wedge$	Logical AND.
$\vee$	Logical OR.
:=	Assignment (of a value to a variable).
( ) or [ ]	Ordered set (of data elements).
$B_1 \mid\mid B_2$	Concatenation of bytes B_1 (the most significant byte) and B_2 (the least significant byte).
$[B_1 \mid\mid B_2]$	Value of the concatenation of bytes B_1 and B_2.
AES( )[ ]	The data in the square brackets is encrypted using AES encryption and the key in the normal brackets.

$AES^{-1}( )[ ]$	The data in the square brackets is decrypted using AES decryption and the key in the normal brackets.
DES( )[ ]	The data in the square brackets is encrypted using DES encryption and the key in the normal brackets.
$DES^{-1}( )[ ]$	The data in the square brackets is decrypted using DES decryption and the key in the normal brackets.
DES3( )[ ]	The data in the square brackets is encrypted using triple DES encryption and the key in the normal brackets. Triple DES consists of encrypting an 8-byte plaintext block X to an 8-byte ciphertext block Y using a double length (16-byte) secret key $K = (K_L \| K_R)$ where K_L and K_R are DES keys. This is done as follows: $Y := DES3(K)[X] := DES(K_L)[DES^{-1}(K_R)[DES(K_L)[X]]]$
$DES3^{-1}( )[ ]$	The data in the square brackets is decrypted using triple DES decryption and the key in the normal brackets. Triple DES consists of decrypting an 8-byte ciphertext block Y to an 8-byte plaintext block X using a double length (16-byte) secret key $K = (K_L \| K_R)$ where K_L and K_R are DES keys. This is done as follows: $X := DES3^{-1}(K)[Y] := DES^{-1}(K_L)[DES(K_R)[DES^{-1}(K_L)[Y]]]$
XOR	Exclusive OR

Requirement Numbering

Requirements are highlighted by being indented and numbered with a four-digit reference namely, section, subsection and requirement number. All requirements in this specification are therefore uniquely numbered with the number appearing next to each requirement. This convention is adopted to allow test specifications to be conveniently developed.

A requirement can have different numbers in different versions of the specifications. Hence, all references to a requirement must include the version of the document as well as the requirement's number.

1.14 Constraints

The Specification or any implementation of the Specification is not intended to replace or interfere with any international, regional, national or local laws and regulations; those governing requirements supersede any industry standards.

2 Card Personalisation Data Processing

2.1 Overview of the Process

Within a personalisation bureau environment the processing of Personalisation Device Instructions (PDI) and IC card personalisation data processing requires the following three functional steps:

1. Data preparation
2. Personalisation device set-up and processing
3. IC card application processing.

Each of these steps, together with the two interfaces (1 to 2, 2 to 3), is briefly described below and discussed in detail in subsequent chapters.

An overview diagram of the complete EMV Card Personalisation process Indirect Method appears in Annex B.

Data Preparation

Data preparation is the process that creates the data that is to be placed in an IC card application during card personalisation. Some of the data created may be the same across all cards in a batch; other data may vary by card. Some data, such as keys, may be secret and may need to be encrypted at all times during the personalisation process.

Data preparation may be a single process, or it may require interaction between multiple systems.

Much of the definition of data preparation is application specific. This document focuses on the data preparation processes that are commonly used for EMV cards and a description of these is given in section 3.

Data Preparation-Personalisation Device Interface

The output of the data preparation process is a file of personalisation data, which is passed to the personalisation device. The format of the file records is shown in Table 3-8.

The data preparation system must protect the completed personalisation data file for integrity and authenticity (e.g.by use of a MAC or signed hash). An example of implementation in XML format is provided in the GlobalPlatform Messaging Specification section 5.

Personalisation Device

The personalisation device is the terminal that acts on Processing Step and Personalisation Device Instruction data to control how personalisation data is selected and then sent to the IC card application. The Processing Step indicates the format of the personalisation data to be sent to the IC card application. Depending on the Processing Step for IC card personalisation processes this device must have access to a security module (HSM) to establish and operate a secure channel between the personalisation device on the one hand and the application on an IC card on the other. If the Processing Step indicates the pre-computed APDU commands as personalisation data then the commands to establish the secure channel are part of the pre-computed APDU commands, in this case therefore the personalisation device is not required to explicitly establish the secure channel with the IC card application. The secure channel services consist of MAC verification/ generation e.g. on commands sent to the application, and decryption and re-encryption of secret data e.g. PIN values. Personalisation device processing is described in section 4.

Personalisation Device-ICC Interface

The personalisation device sends a series of personalisation commands to the ICC. The personalisation command flow is shown in Figure 6.

The IC Card Application

The IC card application receives the personalisation data from the personalisation device and stores it in its assigned location, for use when the EMV card application becomes operational.

Section 5.1 describes the processing requirements for an EMV card application that must be performed prior to the start of personalisation. The actual processing of the EMV card prior to personalisation (pre-personalisation) is outside the scope of this specification. However, it is assumed that the EMV card application will have secure channel keys established for personalisation prior to the start of the personalisation process.

2.2 The Infrastructure of Card Personalisation

The personalisation process described in this document is designed to facilitate the personalisation of the EMV application on IC cards. It creates a personalisation infrastructure that allows for upgrades to EMV applications without requiring a change to the personalisation device processing, and one that can also be extended to other applications in a generic way.

The personalisation infrastructure consists of:

- Standard security between the personalisation device and the IC card. This is summarised in section 2.3.
- Standard commands for sending personalisation data to the IC card application. These are summarised in section 2.4.
- A standard record format for the personalisation data sent to the personalisation device. This is summarised in section 2.5.

2.3 Secure Messaging

At the beginning of processing by the personalisation device, a secure channel is established between the personalisation device and the IC card EMV application. Depending on the personalisation method (see section 3), the secure channel is established either by the personalisation device for the Indirect Method or through the pre-computed APDU commands for the Direct Method. The commands used to establish this secure channel are the INITIALIZE UPDATE command (see section 4.3.2) and the EXTERNAL AUTHENTICATE command (see section 4.3.3). These commands apply to both SCP02 and SCP03. When a secure channel is established (SCP02 or SCP03), authentication or privacy security services can be invoked to protect command data sent to the IC card application from the personalisation device according to the security level set in the EXTERNAL AUTHENTICATE command.

The privacy service is referenced as C-DECRYPTION and the authentication service is referenced as C-MAC.

Additionally, in SCP03 only, the EXTERNAL AUTHENTICATE command can invoke privacy and authentication security services on data returned from the card application, these services are referenced as R-ENCRYPTION and R-MAC respectively.

Two derived keys, K_{ENC} and K_{MAC}, on the IC card are used during the establishment of the secure channel. K_{ENC} is used for SCP03 to generate card challenges and for SCP02 to generate a session key SKU_{ENC} which is in turn used to create and validate authentication cryptograms. K_{MAC} is used to generate a session key SKU_{MAC} which is in turn used for SCP02 to generate card challenges, for SCP03 SKU_{MAC} is used to create and validate authentication cryptograms, and for both SCP02 and SCP03 it is used to compute the MAC of the EXTERNAL AUTHENTICATE command. Both of these keys (K_{ENC} and K_{MAC}) are derived from the same master key, the KMC. When the secure channel is to be established explicitly by the personalisation device the IC card provides the personalisation device with the identifiers of the KMC and the diversification data to be used to create the derived keys. The identification of the KMC is described in section 4.1.1. The creation of derived keys is described in section 5.1.

Once a secure channel is established, personalisation data can be sent to the IC card application. Based on the security level set in the EXTERNAL AUTHENTICATE command, the SKU_{ENC} may be used to encrypt the command data field, and the SKU_{MAC} to produce the Command Message Authentication Code (C-MAC). In Direct Method the APDU commands are pre-computed according to the requested security level by the data preparation system rather than by the personalisation device. For SCP03, if required by the security level, K_{MAC} is used during the establishment of the secure channel to generate a session key SKU_{RMAC} which in turn is used to produce the Response Message Authentication Code (R-MAC) in card responses.

2.4 The STORE DATA Command

The STORE DATA command is used to send personalisation data to the card application; it is described in detail in section 4.3.4.

In order to reduce personalisation time, the data preparation process organises the personalisation data to be sent to an EMV card application by the personalisation device into data groupings. A Data Grouping Identifier (DGI) identifies each data grouping. The IC card application uses the DGI to determine how the data grouping is to be processed after it is received from the personalisation device. Much of the data for an application is organised into records within the application when the application is designed. Where this is the case, the easiest way to create data groupings for an application is to make each record in the application a data grouping. The principles of data grouping are described in section 3.2. In the Indirect Method the personalisation devices parse the input record and create a STORE DATA command for each data grouping or group of data groupings (see section 3.4.5) in the input record.

Some data groupings will contain data that must be kept confidential during transmission from the personalisation device to the card application. The STORE DATA command has access to a data encryption service. In the Indirect Method, data encryption using an additional derived key (K_{DEK}) available on the IC card is used. The IC card provides the personalisation device with the identifiers of the KMC and the diversification data to be used by the personalisation device to re-derive the IC card K_{DEK} key value. The K_{DEK} is derived from the same master key (KMC) as the K_{ENC} and K_{MAC} as described in section 5.1. The K_{DEK} is used either directly for encryption in SCP03 or in SCP02 to derive an encryption session key SKU_{DEK} - see section 6.3 for the derivation mechanism. Encryption is performed as described in section 6.5.

In the Direct Method the data grouping or group of data groupings are wrapped into the pre-computed STORE DATA commands and the data groupings containing secret data have been encrypted under the IC card's K_{DEK} (SCP03) or SKU_{DEK} (SCP02) by the data preparation system.

It is possible to apply the K_{DEK} encryption mechanism and additionally apply the secure messaging security services described in section 2.3.

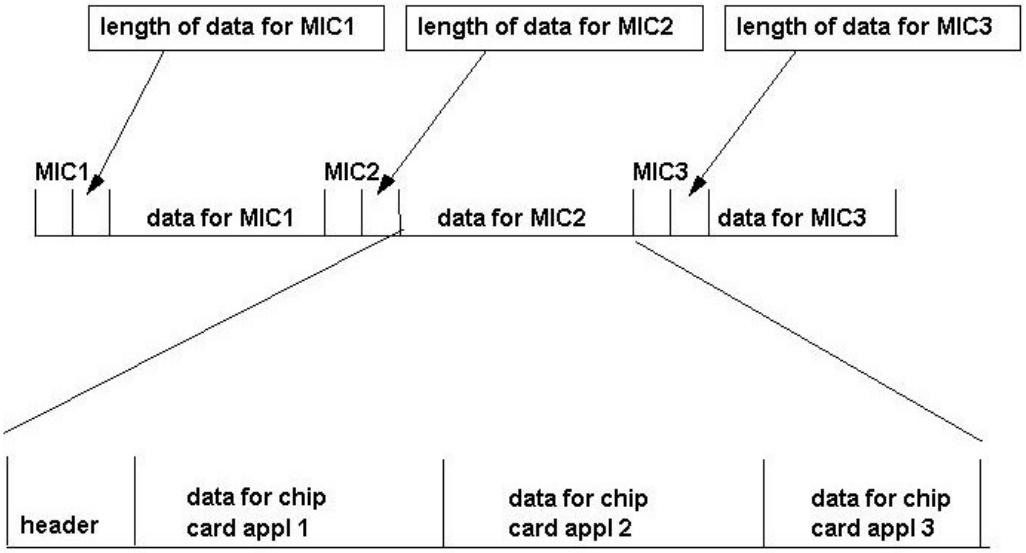
2.5 The Common Personalisation Record Format

The common personalisation approach requires a common personalisation record format. This record format is described in section 3.6. This format has been developed to support the personalisation of one or more applications on a single IC card.

The overall card personalisation process normally consists of a series of processing modules that perform personalisation tasks (e.g. embossing and magnetic stripe encoding). Each processing module uses data from the input record for a card to perform its task for that card. In the format defined in this document, the data for a processing module is identified by a Module Identifier Code (MIC). Each MIC is followed by the data to be processed for that processing module. Many processing modules also require a length field that specifies the length of the data for that processing module. The input for the personalisation process for non IC card application data is defined in documentation provided by the personaliser, however, the basic structure of the most commonly used personalisation record format allows all types of personalisation data to be included in the same file.

There will be a MIC that identifies data to be placed on an IC card. The exact MIC used for personalisation data must be established between the data preparation processing system(s) and the personalisation device processing system(s). In Figure 1, which shows the organisation of personalisation data for the IC card module, MIC2 is used to represent the IC card personalisation data. MIC1 and MIC3 indicate non-ICC personalisation data.

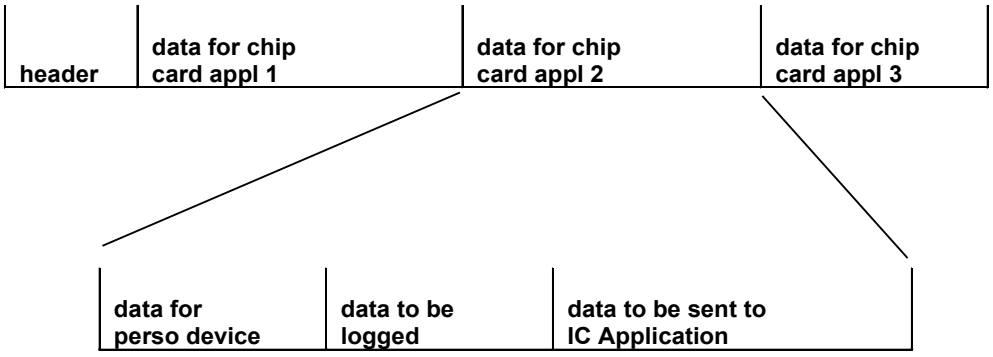
Figure 1 Overview of IC Card Personalisation Data Format



The header portion of this record contains information that is common to all IC card applications to be personalised. Header information is described in Table 3-8.

An overview of the format of the data for each application is shown in Figure 2.

Figure 2 Overview of Personalisation Data for an IC Card Application



- **data for personalisation device**

  The first part of the data for an IC card application is data that provides processing information to the personalisation device. This data consists of the AID for the application, if required the identifier of the Transport Key (TK) used to encrypt secret data in the record for the application (see section 7.37), processing steps and if required the processing instructions for the personalisation device. The creation of personalisation device instructions is described in section 3.4.

- **data to be logged**

  The second part of the data for an IC card application is the data to be logged for that application. However, the personalisation device is not aware of what data is contained in the data groupings. Therefore, data that must be logged must be sent to the personalisation device in a manner that indicates that this is data that must be logged. The data preparation process for the application must perform this task and instruct the personalisation device. The data to be logged may be a duplicate copy of some of the data to be sent to the IC card application. See section 3.5 for additional information.

- **data to be sent to the IC application**

  The data to be sent to the IC card application is in the third part of the data for an IC card application. Depending on the Processing Step this data may consist of one or more data groupings with a Data Grouping Identifier (DGI) and a length field for Processing Step '0F' in the Indirect Method. See section 3.2 for additional information on the format and creation of this data. For Processing Step '0B' in the Direct Method this data consists of one or more pre-computed APDU commands pre-pended by an indicator of the APDU command case and the length of the APDU command. See section 4.2 for additional information on the format and creation of this data. Note that the information provided in section 3.2 applies to each individual data grouping within a pre-computed STORE DATA command.

A complete description of the format for the personalisation data for an IC card application is given in section 3.6.

2.6 Cryptography

This specification relies on cryptography to deliver authenticity and confidentiality security services.

2.6.1 SCP02 Authenticity and Confidentiality

For SCP02 the delivery of these services is based on the use of the DES block cipher as composed to always use a double length key in a manner commonly known as triple DES (DES3).

For authenticity services two different DES MAC constructions are used by this specification. The ANSI Retail MAC is used for C-MAC secure messaging purposes and the MACing of the personalisation data file exchanged between the Data Preparation function and the Personalisation Device in the Indirect Method for SCP02. The ANSI Retail MAC is also used to generate the card challenge.

The full triple DES CBC MAC is used for the generation/validation of authentication cryptograms.

Triple DES CBC mode of operation is used in session key derivation.

For confidentiality services, both DES3 ECB and CBC modes of operation are used by this specification.

2.6.2 SCP03 Authenticity and Confidentiality

For SCP03 personalisation, the delivery of these security services is based on the use of the AES block cipher. Unlike [SCP03] this specification only allows the use of AES-128 and AES-256. Although AES-192 is not supported by this personalisation specification, this does not prevent the personalisation of AES-192 keys into EMV applications. It does mean that to maintain “weakest link” security at 192-bit strength that AES-256 must be used for the personalisation of these applications, not AES-128.

For authenticity services the AES based CMAC construction is always used for MAC calculations [NIST 800-38B]. Additionally, the AES based CMAC construction is always used as the underlying Pseudo Random Function (PRF) for Key Derivation Function (KDF) processes [NIST 800-108] that generate authentication challenges/cryptograms or derive keys as required in SCP03.

AES is used in CBC mode of operation to provide encryption services.

[SCP03] (April 2020) introduces the use of 16-byte length challenges and 16-byte length MACs (denoted S16) into the secure channel set-up and processing, 8-byte challenges and 8-byte MAC lengths (denoted S8) are categorised as legacy. However, 8-byte lengths have not been deprecated and so do not need to be sunset. The choice of 8-byte or 16-byte lengths is left to the implementor. In any transition from SCP02 to SCP03, preserving 8-byte lengths reduces effort and provides ample security. In text where there is an SCP03 branch – 8-byte challenges/MACs vs 16-byte challenges/MACs - are flagged as S8 or S16 respectively.

3 Data Preparation

The data preparation process creates application data that is used to personalise the IC card application and depending on the Processing Step the Personalisation Device Instructions (PDI) data that are used to direct the personalisation process. Whatever is produced by the data preparation process must be transported securely to the personalisation device itself (unless it is created in an HSM attached to the personalisation device). Any secret data created by the data preparation process remotely from the personalisation device must therefore be encrypted before onward transmission and all data files generated remotely from the personalisation device must be protected by a Message Authentication Code (MAC) before onward transmission.

For the Indirect Method where the personalisation device computes the APDU commands the data preparation process consists of the following five steps:

1. Creating personalisation data.
2. Combining personalisation data into data groupings.
3. Creating personalisation instructions.
4. Creating data to be logged for the application.
5. Creating the input file to the personalisation device.

For the Direct Method where the data preparation system computes the APDU commands the data preparation process consists of the following five steps:

1. Creating personalisation data.
2. Combining personalisation data into data groupings.
3. Pre-computing APDU commands.
4. Creating data to be logged for the application.
5. Creating the input file to the personalisation device.

3.1 Creating Personalisation Data

The EMV CPS compliant application developer must determine what data is to be placed in the application at personalisation time, and must determine the source of the data. In some cases a single data preparation process will create all of the data. In other cases, the data will come from multiple sources. The data falls into three categories:

1. Issuer Master Keys and Data
2. Application Keys and Certificates
3. Application Data

3.1.1 Issuer Master Keys and Data

EMV personalisation cannot take place unless the card issuer creates master keys and other specific data. The master keys are used in two ways, firstly to support secure transmission of personalisation data and secondly to create application-level data for personalisation of an EMV application. Some of the data may be used to manage the personalisation process and some will be placed on the card during personalisation.

Other processes may also use one or more of the master keys used by the personalisation process. In this circumstance, a method of importing or exporting master keys to allow appropriate data sharing between processes will be required.

Prior to the personalisation process the following must be placed on the card:

- the identifier of the personalisation master key (KMC_{ID});
- key version number;
- KEYDATA;
- the corresponding relevant keys.

KMC_{ID} and key version number are used to access the issuer personalisation master key (KMC) in order to derive the card unique static keys using diversification data (KEYDATA).

The 6-byte KMC_{ID} (e.g. IIN right justified and left padded with 1111b per quartet) concatenated with the 4-byte CSN (least significant bytes) form the key diversification data that must be placed in tag 'CF'. This same data must be used to form the response to the INITIALIZE UPDATE command.

Table 3-1 Data Content for tag 'CF'

Data Element	Description	Length	Format
KEYDATA	Key diversification data: - KMC_{ID} (6 bytes) - CSN (4 bytes)[1]	10	Binary

3.1.2 EMV Application Keys and Certificates

For EMV applications supporting SDA, DDA, or CDA an issuer RSA key pair must be generated and certified by the Payment System Certification Authority. For EMV applications supporting XDA an issuer ECC key pair must be generated and certified by the Payment System Certification Authority.

Application level symmetric DES/AES secret keys for transaction certificate generation etc. must be created during data preparation. In most cases such keys are derived from appropriate issuer master keys.

If DDA or CDA is supported in the card, then a card level EMV application RSA key pair must be generated and certified by the issuer. A second application RSA key pair may be required for PIN decipherment. If XDA is supported in the card, then a card level EMV application ECC key pair must be generated and certified by the issuer. A second application ECC key pair may be required for ECC based Offline Data Encipherment. Even if the same ECC key is used for XDA and Offline Data Encipherment two certificates will be required. The issuer certificate(s) corresponding to the issuer private key used to certify the application public key(s) must also be stored in the application.

[1] If the CSN does not ensure the uniqueness of KEYDATA across different batches of cards other unique data (e.g. 2 rightmost bytes of IC serial number and 2 bytes of IC batch identifier) should be used instead.

3.1.3 Application Data

Application data will fall into two categories. It may be common across all IC cards for an issuer (e.g. the identifier of the issuer) or it may be unique to the IC card (e.g. the PAN of a debit/credit application).

3.2 Creation of Data Groupings

After the personalisation data for the IC card application has been created, it must be placed into the correct data grouping.

The design of the data groupings plays a key part of the personalisation process. The application designer will have created default file and record definitions. An optional file structure creation mechanism is defined in Annex A.5.

In general, a data grouping should be a record for the application. As all of the data in a grouping will be sent to the IC card application in the minimum number of commands, having a data grouping based on a record will reduce the processing required by the IC card application.

The following requirements and guidelines shall be used when creating data groupings:

1. All data elements to be sent to the IC card during the personalisation process must be placed in a data grouping.
2. DES keys must be placed in a data grouping that consists of only DES keys. More than one data grouping of DES keys may be created. AES keys must be placed in a data grouping consisting only of AES keys. More than one data grouping of AES keys may be created.
3. For ease of IC card management, most, if not all, of the content of each data grouping should be identical to the content of a record for the application.
4. When assigning identifiers for data groupings, use of a combination of the SFI and the record number is recommended. For the DGIs with the first byte equal to '01' through '1E', the first byte indicates the SFI in which the data is to be stored and the second byte indicates the record number within the SFI.
5. The first two or three data bytes of any DGI for the EMV application in the range '01xx' through '0Axx' will consist of a record template tag '70' and the length of the remaining record data.
6. There are benefits from grouping all fixed length data elements together and placing all variable length data elements into another data grouping.
7. Application designers do not usually place data elements that are only used by internal IC card application processing in records. However, these data elements must be placed in one or more DGIs. It is recommended that these DGIs contain only data elements used exclusively for internal IC card application processing.
8. If an application contains information in its FCI (the response to the SELECT command) that is dependent on the personalisation data, one or more data groupings must be created for the FCI data.
9. Data grouping identifiers (DGI) '9F60' through '9F6F' are reserved for sending payment systems' proprietary data, e.g. card production life cycle (CPLC) data to the IC card (rather than the application) and must not be used as a DGI by an EMV CPS compliant application.

10. Data grouping identifiers (DGIs) '7FF0' through '7FFE' are reserved for application-independent personalisation processing and must not be used as a DGI by an EMV CPS compliant application.
11. Data grouping identifiers (DGIs) '8F01', '7F01' and '00CF' are reserved respectively for the secure channel derived keys, the key related data and the key diversification data and must not be used as a DGI by an EMV CPS compliant application for other purposes.
12. Data grouping identifier (DGI) '0062' is reserved for file structure creation and must not be used as a DGI by an EMV CPS compliant application for other purposes.
13. The number of data groupings should be minimised to improve performance during the personalisation process.
14. If the data grouping is to be encrypted and it contains data that is not a symmetric block cipher key or a PIN block, it must always be padded. Padding is accomplished by appending an '80', followed by 0-7/0-15 bytes of '00' depending on the block size of the cipher (DES/AES). The length of the data part of the data grouping must be a multiple of 8 bytes/16 bytes for DES/AES respectively. If the data is one or more symmetric keys and a multiple of 16 bytes in length no padding is required. If AES is used and the symmetric key or keys to be protected is a multiple of 8 bytes but not 16 bytes then 8 bytes of random padding shall be used to form a sequence of 16-byte blocks. If the data is an 8-byte PIN block and the block cipher is DES no padding is required. If AES is used, then an 8-byte PIN block is padded with 8 bytes of random data to form a 16-byte block. N.B. this is not to be confused with the RANDOM field mechanism for blinding low entropy data.
15. The Key Check Value for any DES key will be computed by encrypting 8 bytes of '00' using ECB 3DES with the key concerned and for an AES key by encrypting 16 bytes of '01' using ECB AES with the key concerned. The Key Check Value is defined in section 7.15.
16. It is recommended that the data preparation system generate the Reference PIN Block according to ISO 9564-1 format 1. The card application must reformat the PIN Block if necessary, discarding random padding as necessary.
17. The data within a DGI with a leftmost byte of the identifier in the range of '80' to '8F' is always encrypted. Nevertheless, encrypted data may also be included in a DGI outside this range.

The DGI must be coded on two bytes in binary format, followed by a length indicator coded as follows:

- On 1 byte in binary format if the length of data is from '00' to 'FE' (0 to 254 bytes).
- On 3 bytes with the first byte set to 'FF' followed by 2 bytes in binary format from '0000' to 'FFFE' (0 to 65 534), e.g. 'FF01AF' indicates a length of 431 bytes.

3.3 Completion of Personalisation

If specific data needs to be sent to the IC card application at the end of the personalisation process, this can be indicated to the Personalisation Device by using a special Data Grouping Identifier ('7FFF') that will be included in the last STORE DATA command. The contents of this DGI are shown in Table 3-2.

Table 3-2 Data Content for DGI '7FFF'

Data Element	Description	Length	Format
DGI	'7FFF'	2	Binary
Length	Length of data	1 or 3	Binary
	Data	var.	var.

The DGI '7FFF' is not mandatory but can be used as the command body of a final STORE DATA command issued in the personalisation of the EMV application.

Independently of the DGIs (including '7FFF'), the personalisation device shall set b8 of the P1 parameter in the last STORE DATA command to 1b, indicating the completion of personalisation for the application.

3.3.1 Multiple Transport Key Capability

It is also possible to encrypt secret data per application under different Transport Keys while the data is being transferred from the data preparation system to the personalisation device. See Table 3-9 for an explanation of this process. The usual reason for having multiple TKs would be to encrypt PINs under a PEK (PIN Encryption Key) and encrypt other secret data under a different Transport Key. This use of multiple TKs allows separation between true KEKs (Key Encryption Keys), PEKs and DEKs (Data Encryption Keys).

3.4 Processing Steps and Personalisation Device Instructions

A Processing Step (PS) describes what action is to be performed by the personalisation device. For EMV cards the only action to be performed is personalisation based on DGIs and is indifferent to whether the personalisation APDU commands are computed by the personalisation device (i.e. DGIs within the ICC data field) or by the data preparation system (i.e. pre-computed APDU commands within the ICC data field).

For Indirect Method as the APDU commands are to be computed by the personalisation device the value of the ACT field (see Table 3-11) within the PS data element (see Table 3-8 section 3a.2) for this step must be set to '0F'.

For Direct Method as the personalisation is performed using the pre-computed APDU commands by the Data Preparation System the value of the ACT field must be set to '0B'.

The personalisation device instructions are special instructions that are created during the data preparation process and are used by the personalisation device during the personalisation process. The personalisation device instructions are not sent to the card.

At this point no personalisation device instructions are defined for the Processing Step '0B'.

The personalisation device instructions for Processing Step '0F' are:

Order – The order in which data groupings must be sent to the IC card application is specified in this instruction.

Version Control – As IC card applications are modified and different versions of their specifications are created, the required data groupings for the application may change. Data groupings that vary by application version, and which may be rejected by the IC card application without causing the personalisation process to terminate with an error, are specified in this instruction.

Encryption – Data groupings that must be decrypted and re-encrypted by the personalisation device are specified in this instruction.

Random Number – If a random number (RANDOM) is required for any blinding (not padding) data of this application, the random number in the RANDOM field must be used.

Group – Any set of data groupings that must be sent to the IC card application within one STORE DATA command is specified with this instruction.

Security Level – The expected security level of the secure channel to be established between the personalisation device and the IC card application.

Update CPLC – This is used if the personalisation device is required to generate the DGI '9F66' with the appropriate CPLC data and send it to the card within one STORE DATA command.

The personalisation device instructions for personalisation must be placed in a PDI field within the Processing Step.

3.4.1 Order that Data must be sent to the IC Card

In general, the data for an IC card application should be organised into data groupings in a manner that allows the data groupings to be sent to the IC card application in any order. However, there may be applications where it is not possible to organise data in this manner. To inform the Personalisation Device of these situations, the ORDER field in the PDI field (see Table 3-12) must be created. The contents of the ORDER field are shown in Table 3-3. If a DGI is part of a group of DGIs (See section 3.4.5) then only the first DGI listed for that group should be included in this ORDER field. Multiple data grouping orders may be created. It is also possible to indicate to the personalisation device the order in which the DGI or a group of DGIs is to be sent to the IC application. A DGI can only occur once within the value fields of all ORDER statements taken together.

If the orders of STORE DATA command are set to '00' for Order #1 and Order #n then there is no requirement that Order #1 must be sent to the IC card before Order #n.

The DGI entries must be concatenated together without separators. For example, if DGIs '0101' and '0019' were the DGIs in Order # 1, they would be coded '01010019'. When a DGI is part of a set of GROUPed DGIs, for example DGI '0129' is part of GROUP '02080129' and should be sent to the IC application after DGI '0101' then the Order #n should be '01010208'.

Table 3-3 Data Content for the Field ORDER

Data Element	Description	Length	Format
Order #1	'01'	1	Binary
Order of STORE DATA command for Order #1	'00' order does not matter '01' must be placed in the first STORE DATA command	1	Binary

Data Element	Description	Length	Format
	'02' must be placed in the second STORE DATA command 'nn' must be placed in the 'nn'th STORE DATA command 'FF' must be placed in the last STORE DATA command		
Length	Length of Order#1	1	Binary
Order of DGIs	List of DGIs in Order #1	var.	Binary
…			
Order #n	n	1	Binary
Order of STORE DATA command for Order #n	'00' order does not matter '01' must be placed in the first STORE DATA command '02' must be placed in the second STORE DATA command 'nn' must be placed in the 'nn'th STORE DATA command 'FF' must be placed in the last STORE DATA command	1	Binary
Length	Length of Order #n	1	Binary
Order of DGIs	List of DGIs in Order #n	var.	Binary

3.4.2 Support for Migration to New Versions

The data preparation process does not necessarily know the version of the application it is creating data for. Given that applications will reject all unknown DGIs with error responses, migrating application versions will require synchronisation between the data preparation process and the personalisation process. This is undesirable. In order to ease migration between new versions, one approach would be to produce personalisation data that is a superset of the personalisation data for all possible application versions. By itself this does not completely ease migration problems, it is necessary to inform the personalisation device of such a transition process so that it can detect allowed rejections. The version control field (VERCNTL) in the IC Card Application Data (see Table 3-12) is the mechanism to signal to the personalisation device what are acceptable rejections. The contents of the VERCNTL field are shown in Table 3-4. If a DGI is listed in VERCNTL field and the response from the IC card application is '6A88' (considered a rejection of a data grouping), the personalisation process must not be aborted.

Table 3-4 Data Contents for the Version Control Field VERCNTL

Data Element	Description	Length	Format
List of DGIs	List of data grouping identifiers that may be rejected without error.	var.	Binary

The DGI entries must be concatenated together without separators. For example, if DGIs '0101' and '0029' may be rejected without error, they would be coded '01010029'.

3.4.3 Encrypted Data Groupings

To inform the personalisation device of data groupings that must be encrypted, the ENC field in the IC Card Application Data (see Table 3-12) must be created. All of the data, not just the secret data, in the data grouping is encrypted. The DGI and the length field for the data grouping are not encrypted. The contents of the ENC field are shown in Table 3-5.

The specifications for the encryption process appropriate to a given data grouping for the IC card application must match the equivalent encryption process, either in the data preparation process or in a local process associated with the personalisation device.

Normally the encryption process of the card application mirrors that of the data preparation process. Since if the encryption process for the EMV application is in is ECB mode (if the $FORMAT_{TK}$ '00' is used or $FORMAT_{TK}$ '01' is used with CMODE '11'), the encryption of the prepared data must also be done in ECB mode. In this case the personalisation device decrypts the DGI in ECB mode using the Transport Key before re-encrypting it in ECB mode under the card secure channel session key SKU_{DEK}.

On the other hand, if the card application supports CBC mode for personalisation (if the $FORMAT_{TK}$ '01' is used with CMODE '10') re-encryption at SCP03 based systems is performed in CBC mode after decryption using the Transport Key. In this instance if AES CBC encryption/decryption is to be used it will be as described in ISO/IEC 10116 with SV of 16 bytes of '00' and using key K_{DEK}. In the ISO/IEC 10116 guidance associated with CBC mode it is recommended but not mandated that the first plaintext block should be set to be a unique identifier, however this is not possible in SCP03 but no security weakness has been identified. Personalisers may wish to super-encrypt using the C-DECRYPTION option.

Table 3-5 Data Content for the Field ENC

Data Element	Description	Length	Format
DGI #1	The identifier of the first data grouping to be encrypted	2	Binary
Encryption type	SCP02 DES: Type of encryption needed '11' – to be encrypted using SKU_{DEK} in ECB mode	1	Binary
	SCP03 AES: Type of encryption needed '10' – to be encrypted using K_{DEK} in CBC mode		
DGI #n	The identifier of the n^{th} data grouping to be encrypted	2	Binary
	SCP02 DES: Type of encryption needed '11' – to be encrypted using SKU_{DEK} in ECB mode		Binary

Data Element	Description	Length	Format
Encryption type	SCP03 AES: Type of encryption needed '10' – to be encrypted using K_{DEK} in CBC mode	1	

3.4.4 PIN Block Format and Random Numbers

Between the data preparation system and the personalisation system the simple encryption of secret data is sometimes insufficient for protecting the exchanged data. If the encrypted data is not sufficiently diverse (e.g. a PIN-block having only PIN-digits as variations), a random number can be used to XOR the secret data before encryption. To inform the personalisation device that the random number is to be used for processing of this application, the RANDOM field in the IC Card Application Data (see Table 3-8) is the mechanism that must be used.

The use of this random number mechanism must be indicated by the setting of $TYPE_{TK}$. In particular the $TYPE_{TK}$ in Table 3-10 takes a value of “TK using RANDOM field and XOR operation” (b7=1, b6=0) as described in Table 7-1. In this case the secret data must be XORed with the random number prior to encryption and after decryption.

If the RANDOM field is shorter than the data to be XORed, the data to be XORed must be divided into blocks the length of the RANDOM field. If the final block is shorter than the RANDOM field, the leftmost bytes of the RANDOM field must be used to XOR the short block.

The random number to use is contained in the RANDOM field as shown in Table 3-8.

If ISO 9564-1 format 1 is used to transport the PIN block, the use of the above random number method to diversify the PIN block during transportation is unnecessary. In this case, if the PIN block format stored within the card application is different from ISO 9564-1 format 1, the IC application must reformat the PIN block (e.g. to the format defined in the EMV VERIFY command).

If AES is used, then as described in section 3.2 an 8-byte PIN block is padded with 8 bytes of random data to form a 16-byte block. In this case the use of the above random number method to diversify the PIN block during transportation is unnecessary.

3.4.5 Grouping of DGIs

DGIs may be grouped using the GROUP mechanism described below. Personalisation devices must not group DGIs within a single STORE DATA command without a GROUP field within the Processing Device Instruction specifying the DGIs to be applied jointly. Where a STORE DATA command is to process multiple DGIs, DGIs within a group shall be concatenated in the same order indicated in the GROUP field.

The DGI entries must be concatenated without separators. For example, if DGIs '0208' and '0029' were the Grouped DGIs in GROUP # 1, they would be coded '02080029'. The Personalisation Device must set the STORE DATA command with DGI '0208' followed by DGI '0029' (including identifier, length and content of each DGI).

Table 3-6 Data Content for the Field GROUP

Data Element	Description	Length	Format
Group #1	'01'	1	Binary

Data Element	Description	Length	Format
Length	Length of Group #1	1	Binary
List and order of DGIs for Group #1	List of DGIs in Group #1	var.	Binary
…			
Group #n	n	1	Binary
Length	Length of Group #n	1	Binary
List and order of DGIs for Group #n	List of DGIs in Group #n	var.	Binary

3.4.6 Security Level Indicator of Secure Channel Sessions

While not required, it is possible for a data preparation system to define a different security level of secure channel for different DGIs in the personalisation process of an application. This could be achieved by duplicating the SECLEV field within the PDI as many times as it is required by the Issuer. The following requirements exist:

- When only one security level is required for the entire personalisation session of an application only one SECLEV field shall be present in the PDI.
- A value of 'FF' in the SECLEV indicates that two or more security levels of secure channel are required for the personalisation of the application.
- If several security levels are specified, a DGI shall appear only once in the DGI list associated to security levels.
- When several security levels are specified, each security level requires a new secure channel initiation.

Performance objectives may be achieved by regrouping and ordering DGIs according to their security level requirements.

The LASTSCS field indicates whether the personalisation is complete and the personalisation device shall set b8 of P1 of the last STORE DATA command to 1b or not. If the value of the LASTSCS field is set to '01' it indicates that the personalisation of the application is not complete yet and further processing is required. Note that further processing may be in a subsequent Processing Step or in another personalisation session that is out of scope of this document.

Table 3-7 Data Content for the Field SECLEV

Data Element	Description	Length	Format
SECLEV	- Security level for the only secure channel of the personalisation session (See Table 4-7 for the value of this field) - 'FF' = more than one security level for secure messaging of the personalisation session defined (see next field)	1	Binary

Data Element	Description	Length	Format
NBSECLEV	Number of different security levels defined for the personalisation session	0 or 1	Binary
SECLEV #1	Security level for secure channel assigned to the first set of DGIs. See Table 4-7 for the value of this field	0 or 1	Binary
$NBDGI_{SECLEV\#1}$	Number of DGIs assigned to the SECLEV #1	0 or 2	Binary
$DGI_{SECLEV\#1}$	List of DGIs assigned to the security level SECLEV #1. For grouped DGIs only the first DGI of the group shall be listed	0 or var.	Binary
SECLEV #2	Security level for secure channel assigned to the second set of DGIs. See Table 4-7 for the value of this field	0 or 1	Binary
$NBDGI_{SECLEV\#2}$	Number of DGIs assigned to the SECLEV #2	0 or 2	Binary
$DGI_{SECLEV\#2}$	List of DGIs assigned to the security level SECLEV #2. For grouped DGIs only the first DGI of the group shall be listed	0 or var.	Binary
…			
SECLEV #n	Security level for secure channel assigned to the remaining DGIs (those that are not assigned to a previous security level). See Table 4-7 for the value of this field.	0 or 1	Binary
LASTSCS	'00' = Last STORE DATA command shall have b8 of P1 set to 1b '01' = Last STORE DATA command shall have b8 of P1 set to 0b	0 or 1	Binary

3.5 Creation of Personalisation Log Data

Issuers may need data from the personalisation process to be logged. However, the personalisation device is not aware of what data is contained in the data groupings. Therefore, data that must be logged must be sent to the personalisation device in a manner that indicates that this is data that must be logged. The personalisation device must handle the logging of data that the personalisation device is aware of, such as personalisation date. To inform the personalisation device of application data that must be logged, the LOGDATA field in the IC Card Application Data (see Table 3-8) must be created. The data contained in this field usually duplicates some of the data in the data groupings to be sent to the IC card application. This is the actual data to be logged; it is not a list of DGIs.

One example of the possible contents of LOGDATA would be all of the personalisation data for the application. In that case, the contents of LOGDATA and ICCDATA (see Table 3-8) would be the same. Another example would be to only include an identifier of the card/cardholder for the application in LOGDATA. For a credit/debit application, this could be just the PAN.

3.6 Data Preparation-Personalisation Device Interface Format

This section describes the format for sending EMV card application data to the personalisation device.

The record format allows card management data for personalisation as well as functions other than personalisation to be included in the file. These functions are referred to in this document as processing steps. Examples of other types of card management data are load or install applications.

Table 3-8 provides the details of the format for the EMV card application data.

Sections 1 and 2 of the data format table provide the details for the data that is common to all IC card applications. Section 3 provides the details of the data for the first IC card application (the EMV application); there must be as many section 3s as the value of $COUNT_{AID}$. They must be presented in the same order as in the list of AIDs in section 2 of Table 3-8.

The data for an application has four parts. These are identified as a, b, c and d in section 3 of Table 3-8.

The first part (a) is processing data for the personalisation device for the IC card application. This part of the data is subdivided into parts “a1” and “a2”.

The second part (b) is the data for the IC card application to be logged by the personalisation device.

The third part (c) is the data to be sent to the IC card application.

The fourth part (d) is the MAC related information for the IC card application data.

Table 3-8 IC Card Application Data sent to the Personalisation Device

Section	Data Element	Description	Length	Format
1	MIC (Module Identifier Code)	An identifier that specifies that this is data for an IC card application(s). This field matches current personalisation standards for identifying the type of data that follows. The length and values of this field are established when the personalisation device is configured and is fixed format and length thereafter.	1 - 7	ASCII
	LCCA	Length of the IC card application data - includes all of sections 2 and 3 below. This is coded as ASCII-represented decimal digits, padded at the most significant end by zeros (ASCII “30”).	7	ASCII

Section	Data Element	Description	Length	Format
2	VNL	Version number of this layout - shall be "02.2"[2]	4	ASCII
	L_{DATA}	Length from L_{HDR} to the end of the last application's MAC_{INP} inclusive	2	Binary
	L_{HDR}	Length of the header data for IC card applications. The length of the data from the byte immediately following this length field up to and including the AID for application #n	2	Binary
	L_{CRN}	Length of the CRN, this field may be zero meaning no CRN present	1	Binary
	CRN	Shall be unique per record - see GlobalPlatform Messaging Specification	var.	Binary
	$STATUS_{COLL}$	Collator (see GlobalPlatform Messaging Specification) Status: "00" – Not Applicable "01" – request for collation "02" – request for collation and collation data "03" – collation data only "04" – collation complete	2	ASCII
	$NUMBER_{PID}$	Number of Profile Identifiers ID_{VC} see GlobalPlatform Messaging Specification, '00' means no Profile ID present	1	Binary
	L_{IDVC1}	Length of ID_{VC1}	1	Binary
	ID_{VC1}	Identifier(s) of the first card profile (see GlobalPlatform Systems Profiles Specification)	var.	Binary
	…			
	L_{IDVCn}	Length of ID_{VCn}	1	Binary
	ID_{VCn}	Identifier(s) of the n^{th} card profile	var.	Binary
	$COUNT_{AID}$	Number of Application IDs	1	Binary
	L_{AID1}	Length of the AID for application #1	1	Binary

[2] A personalisation system implementing VNL 02.2 shall accept a MIC with VNL = 02.1.

Section	Data Element	Description	Length	Format
	AID_1	Application ID#1 in its correct relative position	var.	Binary
	…			
	L_{AIDn}	Length of the AID for application #n	1	Binary
	AID_n	Application ID n in its correct relative position	var.	Binary
3	L_{APPL}	Length of the data for IC card application #1. The length of the data from the byte immediately following this length field to and including the MAC for application #1	2	Binary
3a.1	L_{PDD1}	Length of the non-script driven personalisation device data for application #1. The length of the data from the byte immediately following this length field up to but not including the field L_{PDD2} for application #2	1	Binary
	L_{AID}	Length of the AID for application #1	1	Binary
	AID	AID for application #1	5 to 16	Binary
	L_{TK}	Length of the two TK fields that follow. Set to '0D' if $FORMAT_{TK}$ = '00'	1	Binary
	$FORMAT_{TK}$	The format of the TK fields that follow. See Table 3-9	1	Binary
	TKDATA	See Table 3-9 and Table 3-10	var.	Binary
3a.2	L_{PDD2}	Length of the second part of the personalisation device data for application #1. The length of the data from the byte immediately following this length field up to but not including the length field for LOGDATA for application #1	2	Binary
	$L_{IDOWNER}$	Length of ID_{OWNER} field	1	Binary
	ID_{OWNER}	The identifier of the owner of the specifications for this application	var.	Binary
	L_{PS}	Length of processing steps (PS)	2	Binary
	PS	Processing steps for application #1. See Table 3-11	var.	Binary
3b	$L_{LOGDATA}$	Length of LOGDATA – if there is no data to be logged, this field must be '0000'	2	Binary

Section	Data Element	Description	Length	Format
	LOGDATA	See section 3.5 for a description of this field.	var.	Binary
3c	$L_{ICCDATA}$	Length of the data for application #1 to be sent to the IC card. The length of the data from the byte immediately following this length field to but not including the length field for MAC data for application #1	2	Binary
	ICC Data	The data for application #1 to be sent to the IC card (see Figure 3)	var.	Binary
3d	$L_{MACDATA}$	Length of the MACkey plus the length of the MAC_{INP} for application #1, e.g. if a 16-byte key and a 4-byte MAC, 20 bytes decimal in total coded as '14'. If this field is '00' the data has not been MACed (if generated remotely from the personalisation device the data for the IC card application must be protected by a MAC)	1	Binary
	MACkey	Key used to compute the MAC for the data for application #1. For $FORMAT_{TK}$ = '00' this key is encrypted under the TK listed in the TKDATA field. For all other values of $FORMAT_{TK}$, this key is encrypted under the TK marked for MACkey encryption. If the TK is a DES key then MACkey is a DES key, If the TK is an AES key then MACkey is an AES key	0 (No MAC) 16 (DES) 16 or 32 (AES)	Binary
	MAC_{INP}	MAC of the data for application #1. The MAC is computed using the fields starting with the length of IC card application #1 (L_{APPL}) up to but not including the key used to compute the MAC (MACkey field) for the data in application #1; the MAC is computed with encrypted data still encrypted	0 (No MAC) 4 or 8 (DES) 8 or 16 (AES)	Binary

Table 3-9 $FORMAT_{TK}$ Codes and Associated Data

L_{TK}	$FORMAT_{TK}$	Description of TKDATA
'0D'	'00'	The identifier of the Transport Key (TK) used to encrypt secret data for the application. This field has two parts, the first 4 bytes are the Issuer ID (the issuer BIN/IIN right justified and left padded, if necessary, with 1111b per quartet), the last 8 bytes are the version identifier of the TK. Note that Byte 5 of the 13-byte field is undefined.
'00' to 'FF'	'01'	See Table 3-10

Table 3-10 Layout of TKDATA for FORMAT$_{TK}$ '01'

Field	Description	Length	Format
For entry #1			
ID$_{TK}$ #1	The identifier of the Transport Key (TK) used to encrypt secret data for the application. This field has two parts, the first 4 bytes are the Issuer ID (the issuer BIN/IIN right justified and left padded, if necessary, with 1111b per quartet), the last 8 bytes are the version identifier of the TK	12	Binary
TYPE$_{TK}$ #1	See Table 7-1	1	See Table 7-1
CMODE for TK #1	Block cipher mode of operation associated with the TK to encrypt the secret data. '11' indicates ECB '10' indicates CBC	1	Binary
Number of DGIs for TK #1	The number of DGIs in the following list.	1	Binary
DGIs for TK #1	A list of the DGIs encrypted under this TK. There are no delimiters in this string. If a TK encrypts DGIs '9101' and '9104', this list would be coded '91019104'	var.	Binary
…			
For entry #n			
ID$_{TK}$ #n	The identifier of the Transport Key (TK) used to encrypt secret data for the application. This field has two parts, the first 4 bytes are the Issuer ID (the issuer BIN/IIN right justified and left padded, if necessary, with 1111b per quartet), the last 8 bytes are the version identifier of the TK	12	Binary
TYPE$_{TK}$ #n	See Table 7-1	1	See Table 7-1
CMODE for TK #n	Block cipher mode of operation associated with the TK to encrypt the secret data. '11' indicates ECB '10' indicates CBC	1	Binary
Number of DGIs for TK #n	The number of DGIs in the following list.	1	Binary

Field	Description	Length	Format
DGIs for TK #n	A list of the DGIs encrypted under this TK. There are no delimiters in this string. If a TK encrypts DGIs '9101' and '9104', this list would be coded '91019104'	var.	Binary

Table 3-11 Layout of Processing Steps Field

Field	Description	Length	Format
L_{S1}	Length of the information for Processing Step #1	1	Binary
ACT_{S1}	Action to be performed in Processing Step #1. The action depends on the ICC Data format: '0B': Pre-computed APDU commands in the ICC Data field within the TAG_{S1} data field '0F': DGI in the ICC Data field within the TAG_{S1} data field	1	Binary
REQ_{S1}	Processing Step #1 mandatory ('01') or optional ('00'). For example, if the action is “initialise” and the application is already initialised, a setting of mandatory says re-initialise, a setting of optional says do not re-initialise	1	Binary
TAG_{S1}	TAG of the data within the ICC data field where the Processing Step #1 must apply. 'EE' for Pre-computed APDU commands 'EF' for DGIs as personalisation data	1	Binary
L_{PDI}	Length of the Processing Device Instructions	2	Binary
PDI_{S1}	Processing Device Instructions for Processing Step #1. Currently only defined for the personalisation step. See Table 3-12.	var.	Binary
$L_{POINTER}$	Length of the POINTER	2	Binary
$POINTER_{S1}$	RFU	var.	Binary
…			
L_{Sn}	Length of the information for Processing Step #n	1	Binary
ACT_{Sn}	Action to be performed in Processing Step #n. The action depends on the ICC Data format: '0B': Pre-computed APDU commands in the ICC Data field within the TAG_{Sn} data field '0F': DGI in the ICC Data field within the TAG_{Sn} data field	1	Binary

Field	Description	Length	Format
REQ_{Sn}	Processing Step #n mandatory ('01') or optional ('00'). For example, if the action is “initialise” and the application is already initialised, a setting of mandatory means re-initialise, a setting of optional means do not re-initialise	1	Binary
TAG_{Sn}	Tag of the data within the ICC data field where the Processing Step #n must apply. 'EE' for Pre-computed APDU commands 'EF' for DGIs as personalisation data	1	Binary
L_{PDI}	Length of the Processing Device Instructions	2	Binary
PDI_{Sn}	Processing Device Instructions for Processing Step #n. Currently only defined for the personalisation step. See Table 3-12	var.	Binary
$L_{POINTER}$	Length of the POINTER	2	Binary
$POINTER_{Sn}$	RFU	var.	Binary

The PDI Field

The data in the personalisation device instruction (PDI) field may be part of each Processing Step field. It also contains instructions for the Processing Step. The contents of the field for the personalisation Processing Step '0F' are shown in Table 3-12.

Table 3-12 Personalisation Device Instructions for the Personalisation Processing Step '0F'

Field	Description	Length	Format
L_{ORDER}	Length of ORDER – if no order is required for data groupings, this field is '0000'	2	Binary
ORDER	See section 3.4.1 for a description of this field	var.	Binary
$L_{VERCNTL}$	Length of VERCNTL – if there is no version control required, this field is '0000'	2	Binary
VERCNTL	See section 3.4.2 for a description of this field	var.	Binary
L_{ENC}	Length of ENC – if there is no data to be encrypted, this field is '0000'	2	Binary
ENC	See section 3.4.3 for a description of this field	var.	Binary
L_{RANDOM}	Length of RANDOM –if there is no random number, this field is '0000'	2	Binary
RANDOM	See section 3.4.4 and Table 7-1 for a description of this field	var.	Binary

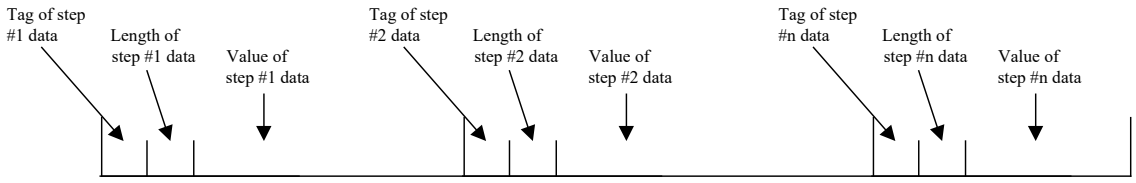
Field	Description	Length	Format
L_{GROUP}	Length of GROUP – if there is no grouped DGIs, this field is '0000'	2	Binary
GROUP	See section 3.4.5 for a description of this field	var.	Binary
SECLEV	Security level for secure messaging. See section 3.4.6 for a description of this field	1 or var.	Binary
$UPDATE_{CPLC}$	'00' means no update – If other values, see requirement for specific applications	1	Binary

There is no personalisation device instruction (PDI) for the personalisation Processing Step '0B'.

The ICC Data Field

The data in the ICC Data field has been “tagged” to allow it to contain data for other processing steps and personalisation data. The format is shown in Figure 3.

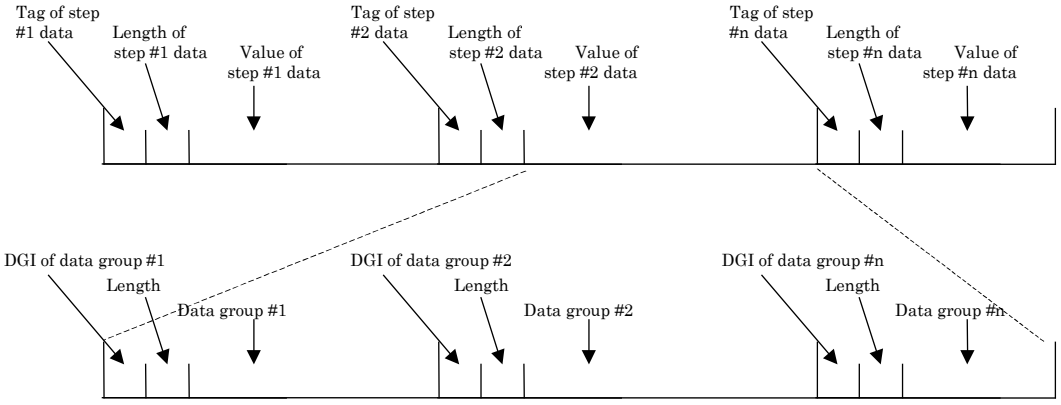
Figure 3 Layout of ICC Data Portion of Record (Section 3c of Table 3-8)



The ICC Data Field for the Processing Step '0F'

If personalisation using DGIs within the ICC Data field is step #2 of the processing steps, Figure 4 shows how the value portion for step #2 is expanded into the data layout for personalisation data. The value of the field TAG for this format is 'EF' so personalisation data within ICC data field for step #2 will be identified by tag 'EF'. The length field for this tag is BER-TLV encoded. The data for this tag is the data groupings for the IC application.

Figure 4 Formatting of Personalisation Data using DGIs within ICC Data Portion of Record

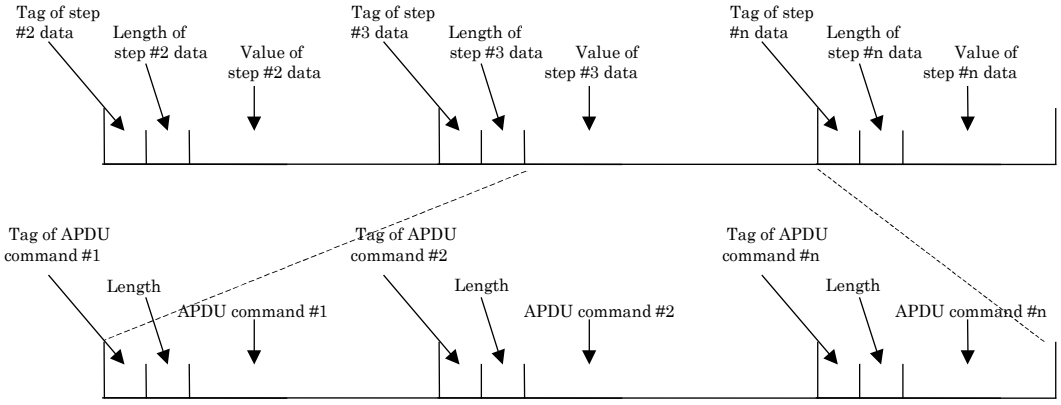


If the data to be sent to the IC card is to be encrypted based on the ENC field (see section 3.4.3), only the value portion of the data grouping is encrypted. The DGI and the length field are not encrypted.

The ICC Data Field for the Processing Step '0B'

If personalisation using pre-computed APDU command is step #3 of the processing steps, Figure 5 shows how the value portion for step #3 is expanded into the data layout within the ICC data field. When this portion contains pre-computed APDU commands the tag value of step #3 shall be 'EE'. The data preparation system incorporates a pre-computed APDU command as the value of a BER–TLV structure as shown in Figure 5. The length of a tag itself is one byte.

Figure 5 Formatting of pre-computed APDU commands within ICC Data Portion of Record



4 Personalisation Device-ICC Interface

4.1 Processing Step '0F'

For the Processing Step '0F' the personalisation device activates the IC card (Reset) and the IC card responds with Answer To Reset (ATR). At this point protocol selection and a warm reset may be used to allow more efficient communications to speed up personalisation.

The SELECT command is used, with the AID retrieved from the input data, to select the application to be personalised[3].

The INITIALIZE UPDATE command provides the IC card with a personalisation device random number (host challenge) to be used as part of authentication cryptogram creation. The IC card responds with:

- A card-maintained sequence counter (for each secure channel key set) to be used as part of the session key derivation.
- A card challenge to be used as part of authentication cryptogram creation.
- A card cryptogram that the personalisation device will use to authenticate the IC card.
- An identifier and a version number of the KMC and the diversification data for the K_{ENC}, K_{MAC} and K_{DEK}.
- The Secure Channel Protocol identifier.

The personalisation device authenticates itself to the IC card application via the EXTERNAL AUTHENTICATE command by providing a host cryptogram for the IC card application to authenticate. The level of security to be used in subsequent commands is also specified in the EXTERNAL AUTHENTICATE command.

The application is personalised using one or more STORE DATA commands. Personalisation is completed with a last STORE DATA command that usually concludes the personalisation process for the application.

If there are more applications to be personalised, then the next AID is retrieved from the input data and the process described above is repeated with a new SELECT command.

After personalisation is complete for all applications, subject to payment system rules, the personalisation data for the card production life cycle (CPLC) data may be placed on the IC card.

[3] Alternatively, on GlobalPlatform cards the Security Domain may be selected to create the Secure Channel and personalisation completed using the INSTALL [for personalization] and STORE DATA commands.

4.1.1 Key Management

4.1.1.1 The personalisation device and the entity that loads the K_{ENC}, the K_{MAC} and the K_{DEK} to the IC card application prior to personalisation, share the KMC. Personalisation device processing must be able to identify the KMC in the personalisation device key storage by an issuer identifier and a version number. The identifiers of the KMC for use with a specific IC card will be retrieved from the IC card itself (see Table 4-3).

4.1.1.2 The TK must be used without modification. The KMC must be used to create card static derived keys which must, as applicable, in turn be used to create session keys for communicating with the IC card application (see section 6.3).

4.1.2 Processing Flow

The following requirements exist for the Processing Step '0F':

4.1.2.1 The personalisation device must read an IC Card Application Data record (see Table 3-8) for each IC card to be personalised and reset the IC card.

4.1.2.2 For each IC card application to be personalised, the TK identified in the input record (ID_{TK}) and its associated algorithm type (DES or AES) must be located in the key storage.

4.1.2.3 If present, data grouping '7FFF' must be retrieved from the input record and must be used as the data to be sent to the IC card application with the last STORE DATA command. The ORDER within PDI field may override this requirement.

4.1.2.4 The AID of each application to be personalised must be retrieved from the data record for the IC card (see Table 3-8).

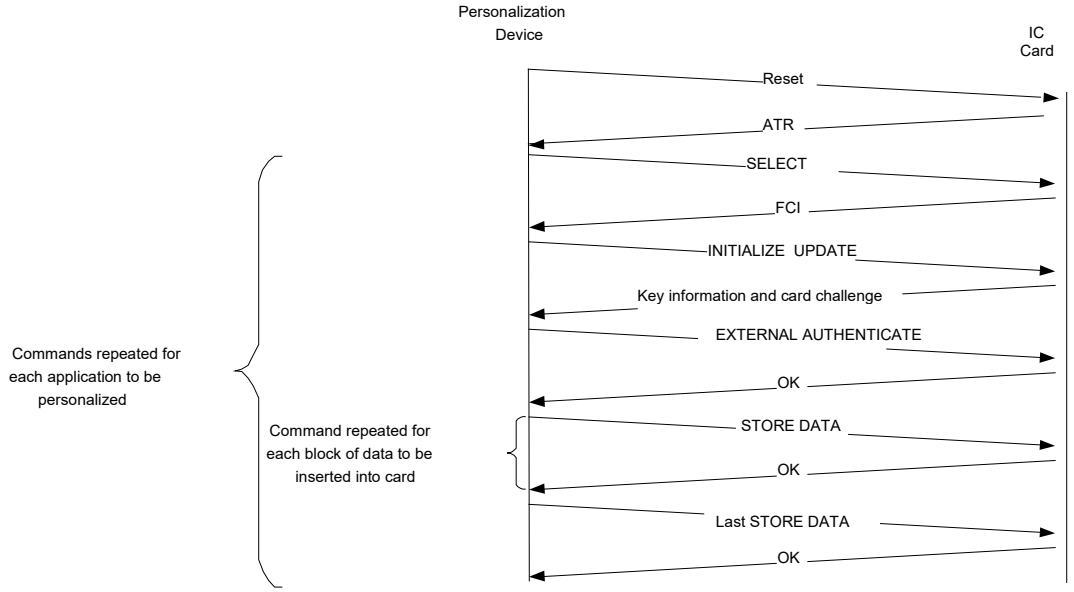
4.1.2.5 A series of STORE DATA commands are sent to the IC card application, followed by a final STORE DATA command (see also LASTSCS in Table 3-7).

4.1.2.6 The card, the application and the personalisation device must be compatible with the interface requirements defined in EMV Version 4.3 Book 1.

4.1.2.7 The personalisation device must use the information contained within the Answer to Reset (for example negotiable mode) to determine the card capabilities as a basis for choosing a mode of operation to minimise personalisation time.

Figure 6 summarises the flow of commands and responses that must occur between the personalisation device and the IC card.

Figure 6 Personalisation Command Flow



4.2 Processing Step '0B'

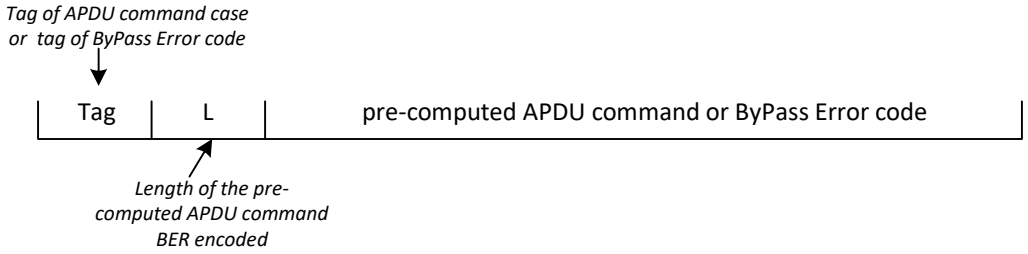
The objective of this feature in the data preparation process is to enable pre-computation of APDUs for direct interpretation by the card. Thus, application keys and PINs do not require to be decrypted and re-encrypted by the personalisation device.

The Processing Step with the indicator '0B' as value for the field ACT is used with pre-computed APDU commands. The value of the field TAG for this format is 'EE'. There are no explicit Personalisation Device Instructions and therefore the PDI field is empty.

For the Processing Step '0B' the personalisation device activates the IC card (Reset) and the IC card responds with Answer To Reset (ATR).

For each pre-computed APDU command the parsing information for APDU case (see ISO/IEC 7816-3) related instructions and the length (BER encoded) of the pre-computed APDU are pre-pended to the pre-computed APDU command. The data preparation system incorporates a pre-computed APDU command as the value of a BER–TLV structure as shown in Figure 7. The length of a tag itself is one byte.

Figure 7 Pre-computed APDU command placed in BER–TLV structure



4.2.1.1 The following tag values shall be supported:

- **Tag '86' - Push command:** The card command or the command response shall not be returned to the issuer.
- **Tag 'C1' - Pull command:** The command response shall be returned to the issuer.
- **Tag 'C7' - Dialogue command:** The card command and the command response shall be returned to the issuer.
- **Tag 'C9' - Error Bypass flag:** The error bypass flag see requirement 4.2.1.5.

The different types of pre-computed APDU commands are coded as concatenations of TLV elements. The tags used for the different commands and the error bypass flag are coded in a single byte and have all primitive encoding. Each of three commands can have an error bypass flag. Note that LOGDATA field could be used should any command response data be returned to the issuer. Other examples of implementation for return data are provided in the GlobalPlatform Messaging Specification section 19.

4.2.1.2 If the value field of a tag is not present, the personalisation device shall interpret this as a request to perform a card reset.

4.2.1.3 If the pre-computed APDU command in the value field of the tag is present and its length is shorter than 4 bytes or longer than 261 bytes, then it implies a syntax error.

4.2.1.4 In some cases, card-reader specific protocols may impose upper limits on the length of commands. If the command cannot be sent due to this reason, a length limit error shall be generated.

4.2.1.5 Requirements for coding of error bypass flag – tag 'C9':

- The value field of the error bypass flag is a single byte equal to '01'. If the value is different, a syntax error shall be generated.
- A command has an error bypass flag if the error bypass flag TLV element precedes the pre-computed APDU command TLV element. An error bypass flag that is not followed by a command shall generate a syntax error.

4.2.1.6 A command with an error bypass flag shall not result in termination of the personalisation process in case of APDU command processing error.

4.2.2 Key Management

If the card secure channel keys are not known by the data preparation system, it is required that the data preparation system generates a new secure channel key set for the preparation of its pre-computed APDU commands. The data preparation system should place the new secure channel keys according to section 5.2. Prior to Processing Step '0B' the personalisation system shall switch the card secure channel derived keys by the ones used in the generation of pre-computed APDU commands. Once the card secure channel keys are updated the pre-computed APDU commands can be sent to the card. Note that the updating of secure channel keys is performed using a Processing Step '0F' which would precede any Processing Step '0B'.

4.2.3 Processing Flow

Independently of the prior steps to Processing Step '0B', a consistent sequence of pre-computed APDU commands will have been generated. The processing flow of messages across the interface is shown in Figure 6.

4.3 Commands

4.3.1 SELECT Command

The SELECT command is used to select each IC card application to be personalised. Application selection is described in EMV Version 4.3 Book 1.

The SELECT command will be issued once for each IC card application to be personalised.

4.3.1.1 The SELECT command must be coded according to EMV Version 4.3 Book 1. There are no application-dependent status conditions associated with this command.

4.3.1.2 The AID in the SELECT command for the Processing Step '0F' is obtained from section 3a.1 of the input data record for the application (see Table 3-8).

4.3.1.3 The response to the SELECT command is not used to direct processing by the personalisation device. The data in the FCI (the response to the SELECT command) may be added or changed during the personalisation process.

4.3.2 INITIALIZE UPDATE Command

The INITIALIZE UPDATE command is the first command issued to the IC card after the personalisation device selects the application. INITIALIZE UPDATE is used to establish the Secure Channel Session to be used during personalisation. The data to perform mutual authentication is exchanged. The identifier and version number for the KMC and the diversification data to be used to derive the K_{ENC}, the K_{MAC} and the K_{DEK} for the application are also returned.

The INITIALIZE UPDATE command will be issued once for each secure channel initiation. It shall be issued at least once for each IC card application to be personalised.

4.3.2.1 Information relating to a Secure Channel Session shall be destroyed and the Secure Channel Session terminated for any one of the following reasons:

- Loss of power or card reset.
- If the command immediately following this INITIALIZE UPDATE command is not an EXTERNAL AUTHENTICATE command.
- If an Application other than the Application that initiated the setting up of a Secure Channel is selected, the current Secure Channel shall either be already terminated or shall be terminated at this point.
- Secure Channel failure. These errors are caused by:
  - The inability to verify the host cryptogram or the C-MAC of the EXTERNAL AUTHENTICATE command.
  - The inability to verify a MAC (if the security level requires MAC) of any command received within the Secure Channel Session.
  - The padding resulting from command data field decryption (if the security level requires MAC and encryption) is not correct.
  - A message that did not have the required level of security.
- At any point within a current Secure Channel Session, the INITIALIZE UPDATE command can be issued to the card in order to initiate a new Secure Channel Session.

4.3.2.2 The INITIALIZE UPDATE command must be coded as shown in Table 4-1. Host challenge (R_{TERM}) is generated by the personalisation device. Table 4-2 shows the response to a successful INITIALIZE UPDATE command. Table 4-4 lists status conditions that may occur, in addition to the general ones listed in ISO/IEC 7816-3.

Table 4-1 INITIALIZE UPDATE Command Coding

Field	Content	Length
CLA	'80'	1
INS	'50'	1
P1	'00' – '7F'[4] if only one of SCP02 or SCP03 is supported. If both are supported then either '20' – '2F' (SCP02) or '30' – '3F' (SCP03) as appropriate to which protocol is being initialised (see section 4.3.2.3)	1
P2	'00'	1
Lc	'08' (SCP02 & S8 variant of SCP03) or	1

[4] '7F' is retained for backward compatibility with legacy implementations but '6F' is now the upper limit for GlobalPlatform alignment.

Field	Content	Length
	'10' (S16 variant of SCP03)	
Host challenge (R_{TERM})	Random number used in host and card cryptogram generation	8 (SCP02 & S8 variant of SCP03) or 16 (S16 variant of SCP03)
Le	'00'	1

Table 4-2 Response to INITIALIZE UPDATE Command

Field		Length
KEYDATA (See Table 4-3)		10
Version number of the master key (KMC)		1
Identifier for Secure Channel Protocol (ALGSCP = '02')		1
Sequence Counter		2
Card challenge (R_{CARD})		6
Card cryptogram		8
SW1 SW2		2
SCP02 above **OR** **SCP03 below**		
KEYDATA (See Table 4-3)		10
Key information	Version number of the master key (KMC) 1 byte	3
	Identifier for Secure Channel Protocol (ALGSCP = '03') 1 byte	
	"i" parameter for this SCP session 1 byte If X=0 below then 8-byte challenges, cryptograms, and MACs are used for the secure channel - S8 variant of SCP03: If X=1 below then 16-byte challenges, cryptograms, and MACs are used for the secure channel – S16 variant of SCP03:	

Field		Length
	='0X' if random challenge and no R-MAC support ='1X' if pseudo-random challenge and no R-MAC/R-ENCRYPTION support ='2X' if random challenge and R-MAC support but no R_ENCRYPTION support ='3X' if pseudo-random challenge and R-MAC support but no R_ENCRYPTION support ='6X' if random challenge and R-MAC and R-ENCRYPTION support ='7X' if pseudo-random challenge and R-MAC and R-ENCRYPTION support	
Card challenge (R_{CARD})		8 S8 or 16 S16
Card cryptogram		8 S8 or 16 S16
Sequence Counter - only present when SCP03 is configured for pseudo-random challenge generation		0 or 3
SW1 SW2		2

Table 4-3 Initial Contents of KEYDATA

Field	Length	Format
Identifier of the KMC (e.g. IIN right justified and left padded with 1111b per quartet)	6	BCD
Chip Serial Number (CSN)	4	Binary

Table 4-4 Status Conditions for INITIALIZE UPDATE Command

SW1	SW2	Meaning
'69'	'82'	Security status not satisfied
'6A'	'88'	Referenced data not found

4.3.2.3 The Key Version Number defines the key set Version Number to be used to initiate the Secure Channel Session. If this value is zero, the default key set for the selected application shall be used. This feature must be implemented but its use is optional dependent on the personalisation context (e.g. if the entity personalising the EMV application is different from the entity personalising the PSE application).

4.3.2.4 If the value indicates a Key Version Number that is not present or indicates a Key Version Number that is incomplete (i.e. the Key Version Number does not contain Key Identifiers 1, 2 and 3), a response of SW1 SW2 '6A88' shall be returned.

4.3.2.5 KEYDATA is a data element available to each IC card application. Its initial value should be coded as shown in Table 4-3.

4.3.2.6 The identifier of the KMC is part of the response data to the INITIALIZE UPDATE command and it facilitates locating the issuer's KMC.

4.3.2.7 The first 6 bytes of KEYDATA returned from the INITIALIZE UPDATE command are used to identify the master key for secure messaging (KMC). For SCP02 the six least significant bytes of KEYDATA are used as key diversification data. For SCP03 all of the bytes of KEYDATA are used as key diversification data. The personalisation device must use the KMC and KEYDATA to generate the K_{ENC}, the K_{MAC} and the K_{DEK} for this IC card, as defined in section 5.1. These keys must have been placed in the IC card prior to the start of the personalisation process.

4.3.2.8 Subsequent processing must use the identifier for Secure Channel Protocol (ALGSCP) in the response to the INITIALIZE UPDATE command to determine how to create MACs and when to create session keys.

The session keys must be calculated during processing of the INITIALIZE UPDATE response using data from the IC card. The session keys must be calculated as described in section 6.3.

A pseudo-random card challenge generation algorithm in the card provides the data preparation system with the ability to pre-compute the card challenge as opposed to being random. This allows an off-card entity, with knowledge of the relevant Secure Channel secret keys and the ability to track the Secure Channel Sequence Counter, to pre-compute an authentication sequence without first communicating with the card.

4.3.2.9 The card challenge is a pseudo-random number that shall be unique to a Secure Channel Session.

A pseudo-random card challenge[5] shall be generated either according to section E.4.2.3 of the GlobalPlatform Card Specification as follows for SCP02 based personalisation or according to SCP03 as reprised further below:

If SCP02 based then the following applies:

- The AID of the currently selected Application is padded according to DES padding (see bullet 15 of section 3.2);
- A MAC is calculated across the padded data using single DES plus final triple DES, the SKU_{MAC} session key and an ICV of binary zeroes;
- The six leftmost bytes of the resultant MAC constitute the card challenge.

[5] Implementation of the pseudo-random card challenge as defined in this requirement is mandatory unless issuer policy requires a random number generation in which case the card may implement a switch so at the initialisation of the card it would be possible to choose a random or pseudo-random card challenge. Note that it is not possible to generate the pre-computed APDU command at the data preparation system for a card implementing a random as card challenge.

- If the received host challenge is identical to the concatenation of the Secure Channel Sequence Counter of the identified Key Version Number and the card challenge (6 bytes generated as described previously), a response of SW1 SW2 '6982' shall be returned.

Otherwise with SCP03 AES based personalisation the following applies:

The sequence counter (3 bytes) shall be incremented. The card challenge (8 bytes for S8 variant of SCP03, 16 bytes for S16 variant of SCP03) is then calculated using the data derivation scheme defined in section 4.1.5 of [SCP03]. In brief, challenge and cryptogram derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]). The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher used with 8-byte output length for S8, and 16-byte output length for S16. In this instance the card challenge is calculated with the static key K_{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' for S8, '0080' for S16), the counter byte is set to '01'. The “context” parameter shall be set to the concatenation of the sequence counter (3 bytes) and the AID of the currently selected application (5 to 16 bytes). This data train, to which the KDF will be applied, is illustrated in the table below.

IC Card Key (SCP03)	Card Challenge Derivation Data (SCP03 S8)
K_{ENC}	'00000000000000000000000200' || '004001' || Sequence Counter || AID

IC Card Key (SCP03)	Card Challenge Derivation Data (SCP03 S16)
K_{ENC}	'0000000000000000000000200' || '008001' || Sequence Counter || AID

4.3.2.10 The card generates its card cryptogram which is returned in the response to the INITIALIZE UPDATE command and subsequently the personalisation device must verify this card cryptogram by generating a duplicate cryptogram and comparing it to the value returned in the response.

For SCP02 the card cryptogram is a DES based MAC created as described in section 6.4.1 using key SKU_{ENC} and the data to be MACed is as shown in the table below.

IC Card Key (SCP02)	Card Cryptogram Derivation Data (SCP02)
SKU_{ENC}	R_{TERM} (8 bytes) || Sequence Counter (2 bytes) || R_{CARD} (6 bytes)

If the card cryptogram does not verify correctly, personalisation processing must be terminated and a log of the process written, as described in section 4.5.

For SCP03 the card cryptogram (8 bytes for S8 variant of SCP03, 16 bytes for S16 variant of SCP03) is calculated using the data derivation scheme defined in section 4.1.5 of [SCP03] with the session key SKU_{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' for S8, '0080' for S16) and the one-byte counter set to '01'. The “context” parameter shall be set to the concatenation of the host challenge (8 bytes for S8, 16 bytes for S16) and the card challenge (8 bytes for S8, 16 bytes for S16) as depicted below.

IC Card Key (SCP03)	Card Cryptogram Derivation Data (SCP03 S8)
SKU_{MAC}	'00000000000000000000000000' || '004001' || Host Challenge || Card Challenge

IC Card Key (SCP03)	Card Cryptogram Derivation Data (SCP03 S16)
SKU_{MAC}	'00000000000000000000000000' || '008001' || Host Challenge || Card Challenge

If the card cryptogram does not verify correctly, personalisation processing must be terminated and a log of the process written, as described in section 4.5.

For pre-computing the INITIALIZE UPDATE command during the data preparation process, the data preparation system needs to generate or to have access to the derived K_{ENC}, the K_{MAC} and the K_{DEK} for each given card to be personalised.

4.3.3 EXTERNAL AUTHENTICATE Command

The EXTERNAL AUTHENTICATE command follows the INITIALIZE UPDATE command and is used to authenticate the personalisation device to the IC card application.

The EXTERNAL AUTHENTICATE command will be issued once for each secure channel initiation. It shall be issued at least once for each application to be personalised.

4.3.3.1 The EXTERNAL AUTHENTICATE command must be coded as shown in Table 4-5. The host cryptogram is calculated by the personalisation device. The response to the EXTERNAL AUTHENTICATE command consists only of SW1 SW2. Table 4-6 lists status conditions that may occur, in addition to the general ones listed in ISO/IEC 7816-3.

Table 4-5 EXTERNAL AUTHENTICATE Command Coding

Field	Content	Length
CLA	'84'	1
INS	'82'	1
P1	Security Level - see Table 4-7	1
P2	'00'	1
Lc	'10' (SCP02 & S8 variant of SCP03) or '20' (S16 variant of SCP03)	1
	Host cryptogram	8 (SCP02 & S8 variant of SCP03) or 16 (S16 variant of SCP03)
	C-MAC	8 (SCP02 & S8 variant of SCP03)

Field	Content	Length
		or 16 (S16 variant of SCP03)

Table 4-6 Status Conditions for EXTERNAL AUTHENTICATE Command

SW1	SW2	Meaning
'69'	'82'	MAC failed verification
'69'	'85'	Conditions of use not satisfied
'63'	'00'	Authentication of host cryptogram failed
'68'	'00'	Functions in CLA not supported
'6A'	'86'	Incorrect parameters P1-P2
'6E'	'00'	Class not supported

4.3.3.2 If an EXTERNAL AUTHENTICATE command is received and is not preceded immediately by an INITIALIZE UPDATE command that has been successfully processed, the SW1 SW2 '6985' shall be returned by the card.

4.3.3.3 If the secure messaging bit of the class byte (bit 3) is not set, a response of SW1 SW2 '6E00' shall be returned.

4.3.3.4 The security level is determined by the SECLEV field within the PDI.

4.3.3.5 If the security level requested by P1 is not supported by the card, a response of SW1 SW2 '6A86' shall be returned. In particular, for SCP03 the security level requested by P1 must correspond to the availability of support for R-MAC and R-ENCRYPTION as indicated by the "i" parameter returned within Key information in the INITIALIZE UPDATE response.

4.3.3.6 Table 4-7 applies to all commands following the EXTERNAL AUTHENTICATE command. The security level does not apply to the EXTERNAL AUTHENTICATE command.

Table 4-7 Security Level (P1)

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	0	0	0	0	1	1	Command Encryption (C-DECRYPTION) and MAC (C-MAC)
0	0	0	0	0	0	0	1	Command MAC (C-MAC)
0	0	0	1	0	0	1	1	Command Encryption (C-DECRYPTION) and MAC (C-MAC), and Response MAC (R-MAC)
0	0	0	1	0	0	0	1	Command MAC (C-MAC) and Response MAC (R-MAC)

b8	b7	b6	b5	b4	b3	b2	b1	Description
0	0	1	1	0	0	1	1	Command Encryption (C-DECRYPTION) and MAC (C-MAC), and Response Encryption (R-ENCRYPTION) and MAC (R-MAC)[6]
0	0	0	0	0	0	0	0	No security

No security – All subsequent commands received by the IC card application will not include any security, i.e. no C-MAC and no encryption of the entire command data string by the SKU_{ENC}.

Command MAC – All subsequent commands received by the IC card application must contain a C-MAC.

Command Encryption and MAC - All subsequent commands received by the IC card application will include a C-MAC and the command data field will be encrypted by the SKU_{ENC}.

Command MAC and Response MAC – All subsequent commands received by the IC card application must contain a C-MAC. All responses sent by the IC card application will include an R-MAC.

Command Encryption and MAC, and Response MAC - All subsequent commands received by the IC card application will include a C-MAC and the command data field will be encrypted by the SKU_{ENC}. All responses sent by the IC card application will include an R-MAC.

Command Encryption and MAC, and Response Encryption and MAC - All subsequent commands received by the IC card application will include a C-MAC and the command data field will be encrypted by the SKU_{ENC}. All responses sent by the IC card application will be encrypted and include an R-MAC.

Note that the encryption specified in the EXTERNAL AUTHENTICATE command does not impact the security specified by the value of P1 in the STORE DATA command (see section 4.3.4). If both the EXTERNAL AUTHENTICATE command and the STORE DATA command specify encryption, the data is encrypted twice.

4.3.3.7 For SCP02 the host cryptogram must be created by generating a MAC as described in section 6.4.1 using SKU_{ENC}. The data to be MACed is as shown in the table below:

IC Card Key (SCP02)	Host Cryptogram Derivation Data (SCP02)
SKU_{ENC}	Sequence Counter (2 bytes) || R_{CARD} (6 bytes) || R_{TERM} (8 bytes).

The IC card must verify the host cryptogram by generating a duplicate cryptogram and comparing it to the value received in the command data field.

[6] Although this security level is a valid option the STORE DATA command does not return a response so there is no encryption/decryption of the response required and consequently the encryption/decryption process is not defined in this specification. For details of the use of this security level for the encryption of the responses to other commands that are beyond the scope of this specification see [SCP03].

For SCP03 the host cryptogram (8 bytes for S8 variant of SCP03 or 16 bytes for S16 variant of SCP03) is calculated using the data derivation scheme defined in section 4.1.5 of [SCP03] with the session key SKU_{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' for S8, '0080' for S16) and the one-byte counter set to '01'. The “context” parameter shall be set to the concatenation of the host challenge (8/16 bytes) and the card challenge (8/16 bytes) as depicted below.

IC Card Key (SCP03)	Host Cryptogram Derivation Data (SCP03 S8)
SKU_{MAC}	'000000000000000000000000100' || '004001' || Host Challenge || Card Challenge

IC Card Key (SCP03)	Host Cryptogram Derivation Data (SCP03 S16)
SKU_{MAC}	'000000000000000000000000100' || '008001' || Host Challenge || Card Challenge

4.3.3.8 The C-MAC must be calculated by the personalisation device and verified by the IC card as described in section 6.4.2.

4.3.3.9 If the EXTERNAL AUTHENTICATE command is successful, for SCP02 the sequence counter shall be incremented by 1. Irrespective of which secure channel protocol is employed, processing must continue with one or more STORE DATA commands.

4.3.3.10 There is no secure messaging applied to the response to the EXTERNAL AUTHENTICATE command.

4.3.4 STORE DATA Command

The STORE DATA command is used to personalise the EMV applications. There will be one STORE DATA command for each data grouping in the record from the data preparation process, although not necessarily one data grouping per single STORE DATA command – multiple DGIs may be sent in one STORE DATA command, as directed by a GROUP Personalisation Device Instruction. This version of the STORE DATA command requires a secure channel to be established. A security level of '00' may be specified. The DGIs and data groupings used by the STORE DATA command are dependent on the data in the input record. Field ENC in the input record lists the identifiers of data groupings that must be encrypted independently of the secure channel encryption mechanism.

4.3.4.1 The CLA byte of the STORE DATA command must be consistent with the security level of secure messaging set in EXTERNAL AUTHENTICATE command (CLA = '80' if security level = '00', otherwise CLA = '84').

4.3.4.2 The version of the STORE DATA command used to personalise the IC card applications must be coded as shown in Table 4-8. One or more triplets (DGI, Length, Data body) may be sent. The response to the STORE DATA consists usually of SW1 SW2. Table 4-10 lists the status conditions that may occur in addition to those specified in ISO/IEC 7816-3.

Table 4-8 STORE DATA Command Coding for Application Personalisation Data

Field	Content	Length
CLA	'80' or '84'	1
INS	'E2'	1
P1	See Table 4-9	1
P2	Sequence Number	1
Lc	Length of command data	1 or 3
DGI_1	Identifier of first data to be stored	2
$Length_1$	Length of first data grouping	1 or 3
$Data_1$	First data to be stored	var.
…		
DGI_n	Identifier of n'th data to be stored	2
$Length_n$	Length of n'th data grouping	1 or 3
$Data_n$	n'th data to be stored	var.
C-MAC	Present if CLA = '84'	0 or 8 (SCP02 & S8 variant of SCP03) or 0 or 16 (S16 variant of SCP03)
Le	Not Present or '00'. Present if security level indicates R-MAC.	0 or 1

Table 4-9 Coding of P1 in STORE DATA Command

b8	b7	b6	b5-b2	b1	Meaning
x					Last STORE DATA command indicator

1					Last STORE DATA command
0					Not the last STORE DATA command
	x	x			Encryption indicators:
	1	1			All DGIs encrypted under SKU_{DEK}
	0	0			No DGI is encrypted[7]
	0	1			Application dependent[8]
	1	0			RFU
			xxxx		RFU[9]
				x	Response expected
				0	No response expected (no R-MAC)
				1	Response expected (R-MAC)

Table 4-10 Status Conditions for STORE DATA Command

SW1	SW2	Meaning
'6A'	'88'	Referenced data not found (Unrecognised DGI)
'6A'	'86'	Incorrect parameters P1-P2
'6A'	'80'	Incorrect parameters in the data field

[7] The IC application may ignore or override this value to protect itself against intrusion. For example, the application must reject a clear text DGI if it expects to receive it always encrypted.

[8] For example, this bit can be set to indicate that some but not all of the DGI data is encrypted.

[9] RFU values are ignored by the IC application.

4.3.4.3 If the card implements extended APDU command data lengths as in ISO/IEC 7816-4 (indicated in the third software function of the card capabilities of the Answer To Reset historical bytes), then the DGI or grouped DGIs (if indicated by the GROUP instruction), must be sent to the IC application in one STORE DATA command.

4.3.4.4 If the card does NOT implement extended APDU command data length and the final length of an intended STORE DATA command data field (data grouping(s) and possible MAC) would be more that 255 bytes, then the DGI or grouped DGIs must be sent to the card in multiple STORE DATA commands as follows:

- The first APDU contains a STORE DATA command according to Table 4-8, truncated at the maximum allowable length (Lc equals 255 bytes including possible MAC).
- The subsequent APDU contains a STORE DATA command with any remaining data as command data, possibly again truncated at the maximum allowable length (Lc less than or equal to 255 bytes including possible MAC).

4.3.4.5 The application is responsible for managing any DGI data that spans more than one STORE DATA command. For the first STORE DATA command, the application will use the length of the last (or only) DGI, and the actual length of the DGI data, to determine whether a subsequent STORE DATA command is expected. For subsequent STORE DATA commands, the application concatenates the segments of data until the total length of the data equals the DGI length in the first STORE DATA command. Any inconsistency between the total length of the data segments in the STORE DATA commands, and the DGI length, shall cause an SW1 SW2 of '6A80' to be returned by the card.

4.3.4.6 The personalisation device must set P2 to '00' for the first STORE DATA command sent to the IC card application. The personalisation device must then increment the value of P2 by 1 for each subsequent STORE DATA command[10].

4.3.4.7 If the $FORMAT_{TK}$ field in the input record is set to '00' and a DGI is listed in the ENC field in the input record, the TK identified in the input record must be used to decrypt the data created by the data preparation process. After the input data is decrypted as described in section 4.3.4.8, it must be re-encrypted prior to sending it to the IC card as described in section 4.3.4.12. Only the data portion of the data grouping is encrypted. The DGI and length field are not encrypted.

4.3.4.8 If the $FORMAT_{TK}$ field in the input record is set to '00' and the encryption type for the DGI listed in field ENC is '11', decryption of data by the personalisation device must be done in ECB mode. This mode is described in section 6.6.1.

If the $FORMAT_{TK}$ field in the input record is set to '00' and the encryption type for the DGI listed in field ENC is '10', decryption of data by the personalisation device must be done in CBC mode. This mode is described in section 6.6.2.

4.3.4.9 If the $FORMAT_{TK}$ field in the input record is set to '01' and a DGI is listed in the “DGIs for TK” field (as shown in Table 3-10) in the input record, the TK identified in the input record must be used to decrypt the data created by the data preparation process as described in section 4.3.4.10. If the same DGI is listed in the ENC field in the input record, after the input data is decrypted, it must be re-encrypted prior to sending it to the IC card as described in section 4.3.4.12. Only the data portion of the data grouping is encrypted. The DGI and length field are not encrypted.

[10] The ICC application may ignore the value of P2.

4.3.4.10 If the $FORMAT_{TK}$ field in the input record is set to '01' and the CMODE field (as shown in Table 3-10) is set to '11', decryption of data by the personalisation device must be done in ECB mode. This mode is described in section 6.6.1.

If the $FORMAT_{TK}$ field in the input record is set to '01' and the CMODE field (as shown in Table 3-10) is set to '10', decryption of data by the personalisation device must be done in CBC mode. This mode is described in section 6.6.2.

4.3.4.11 If a DGI is listed in the ENC field in the input record, after the input data is decrypted, it must be re-encrypted prior to sending it to the IC card as described in section 4.3.4.12. Only the data portion of the data grouping is encrypted. The DGI and length field are not encrypted.

4.3.4.12 If the encryption type for the DGI listed in field ENC is '11', the personalisation device uses the session key SKU_{DEK} to encrypt the data. Encryption must be done in ECB mode described in 6.5.1. If the encryption type for the DGI listed in field ENC is '10', the personalisation device uses the key K_{DEK} to encrypt the data. In this instance encryption must be done in CBC mode. This mode is described in section 6.5.2.

4.3.4.13 If all DGIs within STORE DATA command are encrypted the personalisation device must set b7 of P1 to 1b, and b6 of P1 to 1b. Based on this setting of P1, the IC card must use session key SKU_{DEK} and ECB mode (as described in section 6.6.1) for SCP02 or key K_{DEK} and CBC mode (as described in section 6.6.2) for SCP03 to decrypt each individual DGI data.

If some DGIs within STORE DATA command are encrypted the personalisation device must set b7 of P1 to 0b, and b6 of P1 to 1b. Based on this setting of P1 the IC application must have the knowledge of which DGIs are encrypted. The IC card must use session key SKU_{DEK} and ECB mode (as described in section 6.6.1) for SCP02 or key K_{DEK} and CBC mode (as described in section 6.6.2) for SCP03 to decrypt each individual encrypted DGI data.

4.3.4.14 If the security level in the EXTERNAL AUTHENTICATE command specified command MACing, the C-MAC must be calculated by the personalisation device using SKU_{MAC} and verified by the IC card as described in section 6.4.2. The padding is not sent to the IC card.

4.3.4.15 For SCP02 if the security level in the EXTERNAL AUTHENTICATE command specified MACing and encryption, the C-MAC must be calculated by the personalisation device using SKU_{MAC} and the command data field must be encrypted as described in section 6.5.3 using SKU_{ENC}. Note that the padding added to the data field to ensure that encryption takes place over a data size which is a multiple of 8 bytes becomes part of the data field and must be reflected in the Lc. The IC card must decrypt the command data field as described in section 6.6.3 and verify the C-MAC as described in section 6.4.2.

For SCP03 if the security level in the EXTERNAL AUTHENTICATE command specified MACing and encryption, the command data field must be encrypted as described in section 6.5.3 using $SKU_{ENC.}$ Note that the padding added to the data field to ensure that encryption takes place over a data size which is a multiple of 16 bytes becomes part of the data field and must be reflected in the Lc. The C-MAC must be calculated by the personalisation device using SKU_{MAC}. The IC card must verify the C-MAC as described in section 6.4.2 and decrypt the command data field as described in section 6.6.3.

4.3.4.16 If the SW1 SW2 that is returned by the IC card is '6A88', the input data record for the IC card application must be checked for the presence of the field VERCNTL. If the Data Grouping Identifier (DGI) that was rejected by the IC card is listed in field VERCNTL, normal processing must continue and the MAC chaining value (SCP02 C-MAC or full SCP03 CMAC precursor result) must be saved to be used in the computation of the next C-MAC i.e. it is prepended to the next command data body, and it is this modified command data body that is MACed. If the data grouping DGI is not listed in field VERCNTL, personalisation processing of the IC card application must be ended. The IC card must be rejected and an error log entry written as described in section 4.5.

4.3.4.17 For SCP03 if the security level in the EXTERNAL AUTHENTICATE command specified response MACing, the R-MAC must be calculated by the IC card using SKU_{RMAC} and verified by the personalisation device as described in section 6.4.3.

4.3.5 Last STORE DATA Command

4.3.5.1 Last STORE DATA command is indicated to the IC card application by the setting of b8 of P1 to 1b by the personalisation device. If the conditions to transition the status of the application to “PERSONALIZED” are not satisfied as a result of missing DGIs or missing data element(s), the SW1 SW2 '6A86' may be returned by the application. Other status conditions may be used and are specific to the application.

4.3.5.2 Data grouping '7FFF', if present, must be retrieved from the input record and must be used as the data to be sent to the IC card application with the last STORE DATA command. The ORDER field within PDI may override this rule.

4.3.5.3 If the IC card application being personalised uses DGI '7FFF', there may be additional status conditions that are specific to the application.

4.3.5.4 If required by the application and after successful completion of the personalisation, the acceptance of the STORE DATA command may be disabled by the application.

4.4 Command Responses

All responses to commands, whether successfully processed or not, include two status bytes, SW1 and SW2. SW1 and SW2 are one byte long each as defined in ISO/IEC 7816-3.

Beside specific behaviour defined for each command in section 4.1.2 when a personalisation device receives an SW1 SW2 code different from '9000', '6A88', '61xx' or '67xx'; it shall abort the personalisation process for the application if recovery is not possible.

4.4.1.1 For SCP03, if required by the security level, responses from the IC card requiring a MAC include an 8-byte R-MAC for the S8 variant of SCP03 or a 16-byte R-MAC for the S16 variant of SCP03 which in the Indirect Method shall be verified by the personalisation device.

4.4.1.2 To verify an R-MAC, the personalisation device must generate a duplicate R-MAC and compare it with the R-MAC included in the response data. The R-MAC must be calculated as described in section 6.4.3.

4.4.1.3 If the R-MAC does not verify correctly, personalisation processing must be terminated and a log of the process written, as described in section 4.4.

4.5 Personalisation Log Creation

At the end of the personalisation process for an IC card application, a log of the personalisation process for that IC card application must be created. At the end of the entire process an audit file containing the logs of the personalisation processes for all IC card applications must be created. The format of the data in the log for each IC card is shown in Table 4-11. An example of the complete structure of the log file in XML format is provided in the GlobalPlatform Messaging Specification section 19.

Table 4-11 Contents of Personalisation Log

Field	Content	Length	Format
SEQNO	Sequence number of card personalised	3	Binary
DTHR	Date and time of personalisation - YYMMDDHHMMSS	6	BCD
ID_{TERM}	Identifier of the personalisation device	4	Binary
$LKMC_{ID}$	Length of KMC_{ID}	1	Binary
KMC_{ID}	Identifier of the personalisation master key	var.	Binary
L_{CRN}	Length of CRN	1	Binary
CRN	CRN	var.	Binary
CSN	Chip Serial Number (IC Serial Number from KEYDATA returned by application #1)	4	Binary
L_{AID}	Length of the AID of the 1st application	1	Binary
AID	AID of the 1st application personalised	5-16	Binary
VER_{KEY}	Version Number of the master update key (KMC) returned in response to INITIALIZE UPDATE command for first application	1	Binary
SW1 SW2	The status condition returned from the last step of the personalisation process for the first application	2	Binary
STATUS	'00' personalisation successful	1	Binary

Field	Content	Length	Format
$L_{LOGDATA}$	Length of the field LOGDATA for the first application	2	Binary
LOGDATA	Data from data preparation to be logged for the first application	var.	Binary
…			
L_{AID}	Length of the AID for the n^{th} application	1	Binary
AID	AID of the n^{th} application personalised	5-16	Binary
VER_{KEY}	Version Number of the master update key (KMC) returned in response to INITIALIZE UPDATE command for n^{th} application	1	Binary
SW1 SW2	The status condition returned from the last step of the personalisation process for the n^{th} application	2	Binary
STATUS	'00' personalisation successful	1	Binary
$L_{LOGDATA}$	Length of the field LOGDATA for the n^{th} application	2	Binary
LOGDATA	Data from data preparation to be logged for the n^{th} application	var.	Binary

4.5.1.1 The personalisation log must include the identifier of the personalisation bureau in the header record.

5 IC Card Personalisation Processing

IC card personalisation processing is preceded by a preparation or pre-personalisation process. This preparation is described here in order to establish the data that is assumed to be present in the ICC prior to personalisation.

5.1 Preparation for Personalisation (Pre-Personalisation)

Prior to personalisation the ICC must be enabled/activated, the basic EMV application loaded, and the file and data structure established. In addition certain data must be placed onto the IC card. In some cases this data applies to the entire card (e.g. KMC_{ID}). In some cases, this data only applies to a single application (e.g. the AID).

5.1.1.1 Unless the card is capable of dynamically building files and records and initialising them to binary zeros, files must have been built with space allocated for the data described in the specifications for the IC card application and the space must be initialised to binary zeros.

5.1.1.2 Each application must be selectable by its AID.

5.1.1.3 If the File Control Information (FCI) for the application is not to be personalised, it must be created prior to personalisation.

5.1.1.4 KEYDATA must be set as shown in Table 4-3. KEYDATA is composed of KMC_{ID} and Chip Serial Number (CSN). KMC_{ID} is the identifier of the master personalisation key to be supplied by the card issuer or the personaliser. The length of KMC_{ID} is 6 bytes. The CSN is rightmost 4 bytes of the physical identifier of the card.

5.1.1.5 The version number of the personalisation master key (KMC) used to generate the initial personalisation keys (the K_{ENC}, the K_{MAC} and the K_{DEK}) for each application must be on the IC card.

5.1.1.6 A derived key (K_{ENC}) must be generated for each IC card and placed into the application, it has a role both in authentication and encryption.

As regards authentication K_{ENC} is used via a derived session key SKU_{ENC} in SCP02 and used directly in SCP03:

- in SCP02 K_{ENC} is used to derive a session key SKU_{ENC} that in its turn is used to generate the card cryptogram and to verify the host cryptogram.
- in SCP03 K_{ENC} is used to generate the card challenge

Both protocols use a session key SKU_{ENC} derived from K_{ENC} for decryption, in particular to decrypt the STORE DATA command data field in CBC mode if the security level of secure messaging requires the command data field to be encrypted.

The K_{ENC} must be either a 16-byte (112 bits plus parity) DES key (SCP02), or an AES key of 16 or 32 bytes length for use with SCP03.

For DES as the block cipher (SCP02) the K_{ENC} will be derived in the following way: K_{ENC} is the 16 bytes derived by concatenating the ECB mode output of applying DES3(KMC) to each of the 8-byte derivation data blocks as shown in the table below.

DES Derived Static Key (SCP02)	IC Card Derivation Key	Diversification Data (SCP02)
K_{ENC} 16-byte key	KMC	DES3(KMC)[Six least significant bytes of the KEYDATA || 'F0' || '01'] || DES3(KMC)[Six least significant bytes of the KEYDATA || '0F' || '01']

For SCP03, key derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]) for one or two iterations. The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher used with full 16-byte output length and KMC as the key. The “fixed input data” plus iteration counter shall be the concatenation of the following items in the given sequence presented in the table below. When two iterations are used the outputs are concatenated. This process is aligned with the key derivation method described in section 4.1.5 of [SCP03].

AES Derived Static Key (SCP03)	IC Card Derivation Key	Diversification Data (SCP03)
K_{ENC} 16-byte key	KMC	'00000000000000000000000400' || '008001' || KEYDATA

AES Derived Static Key (SCP03)	IC Card Derivation Key	Diversification Data (SCP03)
K_{ENC} 32-byte key	KMC	'00000000000000000000000400' || '010001' || KEYDATA and '00000000000000000000000400' || '010002' || KEYDATA

5.1.1.7 A derived key (K_{MAC}) must be generated for each IC card and placed into the card. This key is used to derive the session key that is used for SCP03 to generate the card cryptogram and verify the host cryptogram. For SCP02 and SCP03 this session key is used to verify the C-MAC for the EXTERNAL AUTHENTICATE command and also to verify the C-MAC for the STORE DATA command(s) if the security level of secure messaging requires a MAC of the command data.

The K_{MAC} must be either a 16-byte (112 bits plus parity) DES key (SCP02), or an AES key of 16 or 32 bytes length for use with SCP03.

For DES as the block cipher (SCP02) the K_{MAC} will be derived in the following way: K_{MAC} is the 16 bytes derived by concatenating the ECB mode output of applying DES3(KMC) to each of the 8-byte derivation data blocks as shown in the table below.

DES Derived Static Key (SCP02)	IC Card Derivation Key	Derivation Data (SCP02)
K_{MAC} 16-byte key	KMC	DES3(KMC)[Six least significant bytes of the KEYDATA || 'F0' || '02'] || DES3(KMC)[Six least significant bytes of the KEYDATA || '0F' || '02']

For SCP03, key derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]) for one or two iterations. The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher used with full 16-byte output length and KMC as the key. The “fixed input data” plus iteration counter shall be the concatenation of the following items in the given sequence presented in the table below. When two iterations are used the outputs are concatenated. This process is aligned with the key derivation method described in section 4.1.5 of [SCP03].

AES Derived Static Key (SCP03)	IC Card Derivation Key	Derivation Data (SCP03)
K_{MAC} 16-byte key	KMC	'00000000000000000000000600' || '008001' || KEYDATA

AES Derived Static Key (SCP03)	IC Card Derivation Key	Derivation Data (SCP03)
K_{MAC} 32-byte key	KMC	'00000000000000000000000600' || '010001' || KEYDATA and '00000000000000000000000600' || '010002' || KEYDATA

5.1.1.8 A derived key (K_{DEK}) must be generated for each IC card and placed into the card. For SCP02 this key is used to derive a session key that is used to decrypt in ECB mode secret data received in the STORE DATA command. For SCP03 this key is used directly to decrypt in CBC mode the secret data.

The K_{DEK} must be either a 16-byte (112 bits plus parity) DES key (SCP02), or an AES key of 16 or 32 bytes length for use with SCP03.

For DES as the block cipher (SCP02) the K_{DEK} will be derived in the following way: K_{DEK} is the 16 bytes derived by concatenating the ECB mode output of applying DES3(KMC) to each of the 8-byte derivation data blocks as shown in the table below.

DES Derived Static Key (SCP02)	IC Card Derivation Key	Derivation Data (SCP02)
K_{DEK} 16-byte key	KMC	DES3(KMC)[Six least significant bytes of the KEYDATA || 'F0' || '03'] || DES3(KMC)[Six least significant bytes of the KEYDATA || '0F' || '03']

For SCP03, key derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]) for one or two iterations. The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher used with full 16-byte output length and KMC as the key. The “fixed input data” plus iteration counter shall be the concatenation of the following items in the given sequence presented in the table below. When two iterations are used the outputs are concatenated. This process is aligned with the key derivation method described in section 4.1.5 of [SCP03].

AES Derived Static Key (SCP03)	IC Card Derivation Key	Derivation Data (SCP03)
K_{DEK} 16-byte key	KMC	'00000000000000000000000500' || '008001' || KEYDATA

AES Derived Static Key (SCP03)	IC Card Derivation Key	Derivation Data (SCP03)
K_{DEK} 32-byte key	KMC	'00000000000000000000000500' || '010001' || KEYDATA and '00000000000000000000000500' || '010002' || KEYDATA

5.1.1.9 For each Secure Channel key set the sequence counter to be returned in the response to the INITIALIZE UPDATE command must be initialised to '0000' for SCP02 or '000000' for SCP03.

5.2 Load / Update of Secure Channel Key Set

To load or update the secure channel keys the new key set should be placed in the DGI '8F01' and encrypted under a TK shared between the data preparation system (or the entity that generates these keys) and the personalisation system. The key related data should be placed in the DGI '7F01' and the KEYDATA in the DGI '00CF'. If the data preparation system (or the entity that generates these keys) has no knowledge of the Chip Serial Number (CSN) this entity can generate a Card Image Number (CIN) unique across different batches of cards which will replace the CSN portion of the KEYDATA to be placed in DGI '00CF' (see Annex A.4 for formatting these DGIs).

5.2.1.1 Each key set requires three static keys K_{ENC}, K_{MAC} and K_{DEK} and a sequence counter that must be initialised to '0000' for SCP02 or '000000' for SCP03.

5.3 File Structure for Records

During personalisation, the application receives a series of STORE DATA commands corresponding to the record values then stores the record values. Depending on the card platform a file structure may need to be created or the application may simulate the files and records. In either case the application must have mutable persistent memory (e.g. EEPROM) available to store such records, using one of the following methods:

- The pre-allocation of the memory and file structure
- The allocation of the memory and file structure during personalisation

Annex A.5 provides an optional mechanism to submit the file structure creation instructions within a DGI to the application using STORE DATA command.

5.4 Personalisation Requirements

5.4.1 IC Card Requirements

5.4.1.1 The application to be personalised must be on a card conforming to EMV Version 4.3 Book 1 Part II and must use the Application Selection process specified in EMV Version 4.3 Book 1 Part III.

5.4.2 Command Support

5.4.2.1 Each IC card application that supports this specification must support the personalisation commands described in section 4.1.2:

- SELECT
- INITIALIZE UPDATE
- EXTERNAL AUTHENTICATE
- STORE DATA

5.4.2.2 Each IC card must maintain a sequence counter for each secure channel key set version that can be returned in the response to the INITIALIZE UPDATE command. For SCP02 this counter must be incremented by 1 after each successful EXTERNAL AUTHENTICATE command. For SCP03 this counter must be incremented by 1 before each card challenge generation. Every time a key set version is updated with a new set of keys its sequence counter must be initialised to '0000' for SCP02 or '000000' for SCP03.

5.4.2.3 If the IC card application does not recognise the DGI in the STORE DATA command, it must respond with an SW1 SW2 of '6A88'.

5.4.3 Secure Messaging

Secure messaging shall be required by all applications for the following personalisation commands:

- EXTERNAL AUTHENTICATE command (see section 4.3.3)
- STORE DATA command if indicated by the security level of the EXTERNAL AUTHENTICATE command (see section 4.3.4)

5.4.3.1 Commands requiring a MAC shall include a C-MAC (8 bytes for SCP02 & S8 variant of SCP03, or 16 bytes for S16 variant of SCP03) that must be verified by the IC card prior to accepting the command. If the C-MAC fails to verify successfully, the IC card must reject the command with SW1 SW2 = '6982' and the secure channel session is terminated.

5.4.3.2 To verify a C-MAC, the IC card must generate a duplicate C-MAC and compare it with the C-MAC included in the command data. The C-MAC must be calculated as described in section 6.4.2.

5.4.3.3 If a command C-MAC is verified, the IC card application must prepare a MAC chaining value based on the C-MAC value of that command:

- for SCP02 the MAC chaining value is the verified 8-byte C-MAC value, or
- for SCP03 the MAC chaining value is the entirety of the 16-byte CMAC precursor result for verifying the 8 bytes (for S8 variant of SCP03) or 16 bytes (for S16 variant of SCP03) received as the C-MAC value

The MAC chaining value is to be concatenated to the beginning of the next APDU for the computation of the next C-MAC. The SCP02 C-MAC value or full SCP03 CMAC precursor result must be retained for a command even if the response to the command is not '9000'.

5.4.3.4 The IC card application must be able to decrypt data as specified in section 6.6.

Either of SCP02 or SCP03 may be used as the underlying secure channel protocol. As regards SCP02, the secure channel described in this document is compliant with Secure Channel Protocol '02' - implementation option '55'[11] as defined in Appendix E of the GlobalPlatform Card Specification. If SCP03, then the secure channel described in this document is compliant with the Secure Channel Protocol (SCP) '03' branch as defined in Amendment D of the GlobalPlatform Card Specification implementation options '10', '11', '30' and '31'[11] with the exception that AES-192 is not supported.

5.4.3.5 For SCP03, if required by the security level, responses from the IC card requiring a MAC shall include an 8-byte R-MAC (for S8 variant of SCP03) or a 16-byte R-MAC (for S16 variant of SCP03). The R-MAC must be calculated as described in section 6.4.3.

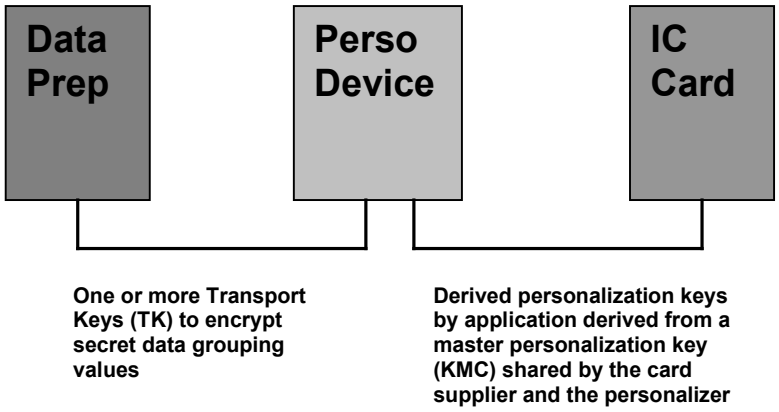
[11] Implementation of pseudo-random generation as defined in the section 4.3.2.9 unless issuer policy requires a random number generation as opposed to pseudo-random number generation in which case the secure channel shall be compliant with Secure Channel Protocol '02' – implementation option '15' as defined in Appendix E of the GlobalPlatform Card Specification and for Secure Channel Protocol '03' – implementation options '00', '01', '20', and '21' as defined in section 5.1 of [SCP03].

6 Cryptography for Personalisation

6.1 Two Key Zones

Two key zones exist in the personalisation process in the Indirect Method. There is a key zone between the data preparation processes and the personalisation device. There is also a key zone between the personalisation device and the IC card. These two key zones are shown in Figure 8.

Figure 8 Personalisation Two Key Zones

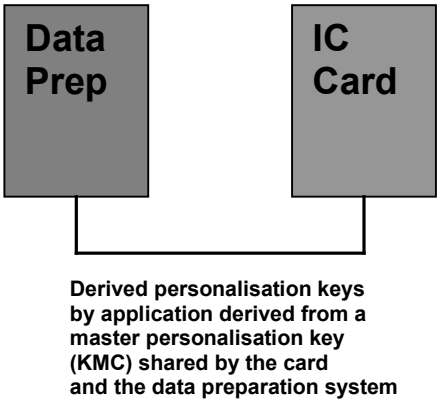


The purpose of having two key zones is to enable the data preparation process to be independent of the IC card type, as far as possible.

6.2 One Key Zone

One key zone exists in the personalisation process in the Direct Method between the data preparation processes and the IC card. This key zone is shown in Figure 9.

Figure 9 Personalisation One Key Zone



6.3 Session Keys

DES/AES session keys are generated every time a secure channel is initiated. These session keys may be used for subsequent commands if secure messaging is required. Up to three session keys may be generated, namely SKU_{ENC}, SKU_{MAC}, for SCP02 SKU_{DEK}, and for SCP03 SKU_{RMAC}.

6.3.1.1 All encryption, decryption and MACing in commands that are sent to the IC card must be performed using session keys (SKU_{ENC}, SKU_{MAC}, and SKU_{DEK}) with the exception that for SCP03 K_{DEK} is used directly rather than a session key.

6.3.1.2 Session keys must be calculated using the triple DES/AES algorithms presented in section 6.7 and the base keys K_{ENC}, K_{MAC}, and for SCP02 K_{DEK} to produce SKU_{ENC}, SKU_{MAC}, and for SCP02 SKU_{DEK} respectively. For SCP03, if required by the security level, K_{MAC} is used to produce SKU_{RMAC}.

For SCP02 DES session keys must be calculated in CBC mode as defined in ISO/IEC 10116 with a Starting Variable equal to '00 00 00 00 00 00 00 00' and the data in Table 6-1. Padding is not added prior to encryption. The 16 bytes of derivation data, when encrypted, will result in a 16-byte double length DES key.

For SCP03 AES based, session key derivation shall use the KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]) for one or two iterations. The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], 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 presented in the table below. When two iterations are used the outputs are concatenated.

Table 6-1 Derivation Data for Session Keys, DES or AES context

DES Session Key	IC Card Key	Derivation Data DES Keys
SKU_{ENC} 16-byte key	K_{ENC}	'0182' || Sequence Counter || '00000000000000000000000000'
SKU_{MAC} 16-byte key	K_{MAC}	'0101' || Sequence Counter || '00000000000000000000000000'

DES Session Key	IC Card Key	Derivation Data DES Keys
SKU_{DEK} 16-byte key	K_{DEK}	'0181' || Sequence Counter || '00000000000000000000000000'

AES Session Key	IC Card Key	Derivation Data for AES-128 Keys
SKU_{ENC} 16-byte key	K_{ENC}	'00000000000000000000000400' || '008001' || Host Challenge || Card Challenge
SKU_{MAC} 16-byte key	K_{MAC}	'00000000000000000000000600' || '008001' || Host Challenge || Card Challenge
SKU_{RMAC} 16-byte key	K_{MAC}	'00000000000000000000000700' || '008001' || Host Challenge || Card Challenge

AES Session Key	IC Card Key	Derivation Data for AES-256 Keys
SKU_{ENC} 32-byte key	K_{ENC}	'00000000000000000000000400' || '010001' || Host Challenge || Card Challenge and '00000000000000000000000400' || '010002' || Host Challenge || Card Challenge
SKU_{MAC} 32-byte key	K_{MAC}	'00000000000000000000000600' || '010001' || Host Challenge || Card Challenge and '00000000000000000000000600' || '010002' || Host Challenge || Card Challenge
SKU_{RMAC} 32-byte key	K_{MAC}	'00000000000000000000000700' || '010001' || Host Challenge || Card Challenge and '00000000000000000000000700' || '010002' || Host Challenge || Card Challenge

The session keys must be calculated for each IC card secure channel key set used during processing of the INITIALIZE UPDATE command using a sequence counter of the key set version provided by the IC card. See section 4.3.2 for specifications for the INITIALIZE UPDATE command.

These session keys are used for all cryptography for personalising the IC card application until the completion of the last STORE DATA command. See section 4.3.5 for specifications for the last STORE DATA command.

6.4 MACs and KDF

The personalisation process creates and uses outputs from MAC functions and KDF functions for a variety of purposes. For SCP02 based operation DES based MAC functions are used to generate card challenges, host cryptograms and card cryptograms. For SCP03 instead of re-using an unmodified MAC function for generating host or card authentication cryptograms and the card challenge, a KDF based on CMAC using AES is used instead.

1. During the SCP02 IC personalisation process (INITIALIZE UPDATE command) the personalisation device sends a host challenge to the IC card and the IC card returns a card challenge and a MAC (the card cryptogram). The mutual authentication process is completed (EXTERNAL AUTHENTICATE command) when the host submits a MAC (host cryptogram) to the card which is corroborated by the card. The IC card and the personalisation device authenticate each other using these cryptograms. The process of creating the MACs listed in this clause is described in section 6.4.1.
2. During the SCP03 IC personalisation process (INITIALIZE UPDATE command and EXTERNAL AUTHENTICATE command) the IC card returns a challenge (card challenge) and a cryptogram (the card cryptogram) and the personalisation device sends the host cryptogram to the IC card. The IC card and the personalisation device authenticate each other using these cryptograms. For SCP03, card challenge and card/host cryptogram derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]). The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher and K_{ENC} as the key for card challenges and the session key SKU_{MAC} as the key for card/host cryptograms. (Secure Channel Protocol '03' – Public Release v1.1.2 section 4.1.5 [SCP03])
3. The EXTERNAL AUTHENTICATE command requires a C-MAC to be sent from the personalisation device to the IC card. Based on the security level set in the EXTERNAL AUTHENTICATE command, each STORE DATA command may require a C-MAC. The C-MACs ensure the integrity of the personalisation data. Because each C-MAC incorporates the MAC chaining value (for SCP02 the C-MAC or for SCP03 the entire precursor CMAC result) from the previous command (including that from the EXTERNAL AUTHENTICATE command) as the first block to compute the next MAC (there is no preceding MAC value to be used as the first block for computing the C-MAC of the EXTERNAL AUTHENTICATE command so a block of all zeros is used), they also ensure that all the personalisation data is sent to the IC card and in the correct order. The process of creating the MACs listed in this clause is described in section 6.4.2.
4. For SCP03 and based on the security level set in the EXTERNAL AUTHENTICATE command, each response to the STORE DATA command may require an R-MAC. The R-MACs ensure the integrity of the card responses. Because each R-MAC incorporates the MAC chaining value (the entire precursor CMAC result of the C-MAC) from the command as the first block to compute the MAC they also ensure that the responses from the IC card are sent in the correct order. The process of creating the MACs listed in this clause is described in section 6.4.3.
5. All data sent from the data preparation process to the personalisation device must be MACed to ensure the integrity of the personalisation data file. The process of creating the MACs listed in this clause is described in section 6.4.4.

6.4.1 MACs for Authentication Cryptograms

6.4.1.1 DES based SCP02:

Input to the MAC is first padded to the right with '80'. The result is padded to the right with up to 7 bytes of '00' to make the result a multiple of 8 bytes long. This is defined in ISO/IEC 9797-1, as padding method 2.

The full triple DES MAC is as defined in ISO/IEC 9797-1 as MAC Algorithm 1 with output transformation 1, without truncation, and with triple DES taking the place of the block cipher.

The entire 8 bytes of the final output block are used as the MAC created for authentication cryptograms (card cryptogram and host cryptogram).

6.4.1.2 AES based SCP03:

The CMAC construction as described in NIST “Special Publication 800-38B - Recommendation for Block Cipher Modes of Operation: The CMAC Mode for Authentication” [NIST 800-38B], using AES as the underlying block cipher with $C_0=0^{128}$ is used as the PRF in the KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]).

For the S8 variant of SCP03 the leftmost 8 bytes of the CMAC result, or for the S16 variant of SCP03 the entire 16 bytes of the CMAC result, are used as the MAC created for authentication cryptograms (card cryptogram and host cryptogram).

6.4.1.3 Verification of a cryptogram must be performed by computing a MAC based on the same parameters (and key) and then comparing the result with the cryptogram received.

6.4.2 C-MAC for Secure Messaging

Secure messaging is required for the following personalisation command:

- EXTERNAL AUTHENTICATE (see section 4.3.3)

Based on the security level set in the EXTERNAL AUTHENTICATE command, secure messaging may also be required for the following personalisation command:

- STORE DATA (see section 4.3.4)

6.4.2.1 Commands using secure messaging must include an 8-byte C-MAC for SCP02 and S8 variant of SCP03, or a 16-byte C-MAC for S16 variant of SCP03 created by the personalisation device and verified by the IC card prior to accepting the command. If the command C-MAC fails to verify successfully, the IC card must reject the command with SW1 SW2 = '6982' and close the secure channel.

Note: To avoid confusion with the MAC function defined in 6.4.1, C-MAC is used as the name of the function, which relies on either the ANSI Retail MAC or the CMAC functions to generate a secure message MAC value, and also as the name of the resulting secure message MAC value – the meaning will be clear from the context.

6.4.2.2 The C-MAC must be calculated as follows:

1. Concatenate the command header (CLA INS P1 P2 Lc) with the command data (excluding the C-MAC itself).

   The command header must be modified as follows:

   - The value of Lc in the data to compute the C-MAC must reflect the presence of the C-MAC in the command data, i.e. Lc = Lc + 8 for SCP02 and S8 variant of SCP03, Lc = Lc + 16 for S16 variant of SCP03.

- The class byte shall be modified to indicate that this APDU includes secure messaging. This is achieved by setting bit 3 of the class byte. A STORE DATA command that contains C-MAC will have a class byte of '84'.

If both the STORE DATA command and the security level of the EXTERNAL AUTHENTICATE command specify encryption, the encryption required by the STORE DATA command (using SKU_{DEK} or K_{DEK}) will be done before the C-MAC is computed and the encryption required by the security level of the EXTERNAL AUTHENTICATE command (using SKU_{ENC}) will be done either after the C-MAC is computed for SCP02 or before the C-MAC is computed for SCP03.

The specific rules are:

- Data groupings that are sent to the IC card with a P1 setting in the STORE DATA command indicating that the data is encrypted are encrypted under the SKU_{DEK} for SCP02 or K_{DEK} for SCP03 before the C-MAC is computed.
- If the security level in the EXTERNAL AUTHENTICATE command indicates that both encryption and MACing are used, then for SCP02 the C-MAC must be created on the original command data (includes data encrypted under SKU_{DEK}) then the APDU command data field is encrypted under SKU_{ENC} and for SCP03 the APDU command data field is encrypted under SKU_{ENC} then the C-MAC must be created on the encrypted command data.

2. Prepend the C-MAC computed for the previous command and validated by the card (or 8 bytes of '00' for the EXTERNAL AUTHENTICATE command) for SCP02. Otherwise for SCP03 prepend,
   - 16 bytes of '00' for the EXTERNAL AUTHENTICATE command
   - or for the S8 variant of SCP03 the entirety of the 16-byte CMAC result computed for the previous command where the leftmost 8 bytes were the C-MAC validated by the card
   - or for the S16 variant of SCP03 the entirety of the 16-byte CMAC result computed for the previous command which were the C-MAC validated by the card

   to the left of the data requiring the MAC. If the IC card rejects a STORE DATA command with a DGI that is listed in field VERCNTL (see section 3.4.2), processing must continue. The personalisation device and the IC card must keep any C-MAC for SCP02 that has been validated by the IC card or for SCP03 the entire 16-byte CMAC result where the leftmost 8 bytes (for S8 variant of SCP03) or entire 16 bytes (for S16 variant of SCP03) were the C-MAC validated by the card to use as the first block of data for a subsequent C-MAC generation.
3. For SCP02 apply padding by appending a byte of '80' to the right of the data and if the resultant data block length is a multiple of 8 bytes no further padding is required. Otherwise, append up to 7 bytes of '00' until the length is a multiple of 8 bytes. For SCP03 no padding is applied as this is handled by the CMAC function.
4. An Initialisation Vector (IV) of all zeros is always used for DES based MAC generation and in the case of the AES based CMAC MAC generation with $C_0=0^{128}$ as per the NIST CMAC reference [NIST 800-38B].
5. The C-MAC is computed as defined in section 6.4.2.3, using SKU_{MAC} as the key.

6.4.2.3 There are two possible MAC functions for the process of generating a C-MAC, either the ANSI Retail MAC for use with DES or the CMAC construction for use with AES.

- DES based SCP02:
  - The “ANSI Retail MAC” shown in Figure 10 where the process of generating a C-MAC is performed with single length DES CBC calculations applied to all the blocks successively bar the final block where triple DES is applied according to ISO/IEC 9797-1:2011 as MAC Algorithm 3 with output transformation 3, without truncation, and with DES taking the place of the block cipher.
- AES based S8 variant of SCP03:
  - The CMAC construction as described in [NIST 800-38B], using AES as the underlying block cipher with $C_0=0^{128}$. The C-MAC is the leftmost 8 bytes of the CMAC result.
- AES based S16 variant of SCP03:
  - The CMAC construction as described in [NIST 800-38B], using AES as the underlying block cipher with $C_0=0^{128}$. The C-MAC is the entire 16 bytes of the CMAC result.

Both the personalisation device and the IC card must create the C-MAC. The IC card verifies the C-MAC by comparing the C-MAC it creates to the C-MAC in the command. Both the personalisation device and the IC card must also save the verified C-MAC (or in the case of SCP03 the entire 16-byte CMAC precursor result) to be used as the first block in the next C-MAC creation or verification.

6.4.3 R-MAC for Secure Messaging (SCP03)

No secure messaging is applied to the response to the EXTERNAL AUTHENTICATE command.

Based on the security level set in the EXTERNAL AUTHENTICATE command, secure messaging may be required for the responses to the following personalisation command:

- STORE DATA (see section 4.3.4)

6.4.3.1 No R-MAC shall be generated and applied to a response that includes an error SW1 SW2: in this case only the SW1 SW2 shall be returned in the response. All SW1 SW2 values except '9000' and warning SW1 SW2 values (i.e. '62xx' and '63xx') shall be interpreted as an error SW1 SW2.

6.4.3.2 The R-MAC must be calculated as follows:

1. Concatenate the MAC chaining value (16-byte CMAC precursor result of the C-MAC) to any response data field and the SW1 SW2 status word.
2. Apply the CMAC construction as described in [NIST 800-38B], using AES as the underlying block cipher with $C_0=0^{128}$ and using SKU_{RMAC} as the key. For the S8 variant of SCP03 the R-MAC is the leftmost 8 bytes of the CMAC result, for the S16 variant of SCP03 the R-MAC is the entire 16 bytes of the CMAC result.

   Both the IC card and the personalisation device must create the R-MAC. The personalisation device verifies the R-MAC by comparing the R-MAC it creates to the R-MAC in the IC card response.

6.4.4 MAC for Integrity of the Personalisation Data File

Integrity and authenticity are required for all data sent from the data preparation process to the personaliser to ensure the integrity of the personalisation data file.

6.4.4.1 For each application the input to the MAC is the data within section 3 of Table 3-8 excluding the fields MACkey and MAC_{INP} (L_{APPL} includes the length for fields MACkey and MAC_{INP}).

6.4.4.2 There are two possible MAC functions, the ANSI Retail MAC for use with DES or the CMAC construction for use with AES.

- DES based SCP02:
  - Input to the MAC is first appended (to the right) with '80'. The result is padded to the right with up to 7 bytes of '00' (possibly none) to make the input data a multiple of 8-byte blocks.
  - The process of generating a MAC using MACkey must be performed as single DES plus final triple DES MAC according to ISO/IEC 9797-1:2011 i.e. MAC Algorithm 3 with output transformation 3, without truncation, and with DES taking the place of the block cipher. This is also known as the ANSI Retail MAC and presented in Figure 10 (with no MAC from previous message).
  - The leftmost 4 bytes or entire 8 bytes of the final output block are used as the MAC value created during the data preparation process. The length to be used depends on the capability of the data preparation system. Historically 4 bytes were used but it is now recommended to use 8 bytes.
- AES based SCP03:
  - The CMAC construction as described in NIST “Special Publication 800-38B - The CMAC Mode for Authentication” [NIST 800-38B], using AES as the underlying block cipher with MACkey and $C_0=0^{128}$.
  - The leftmost 8 bytes of the CMAC result or entire 16 bytes of the CMAC result are used as the MAC value created during the data preparation process. The length to be used is the choice of the data preparation system.

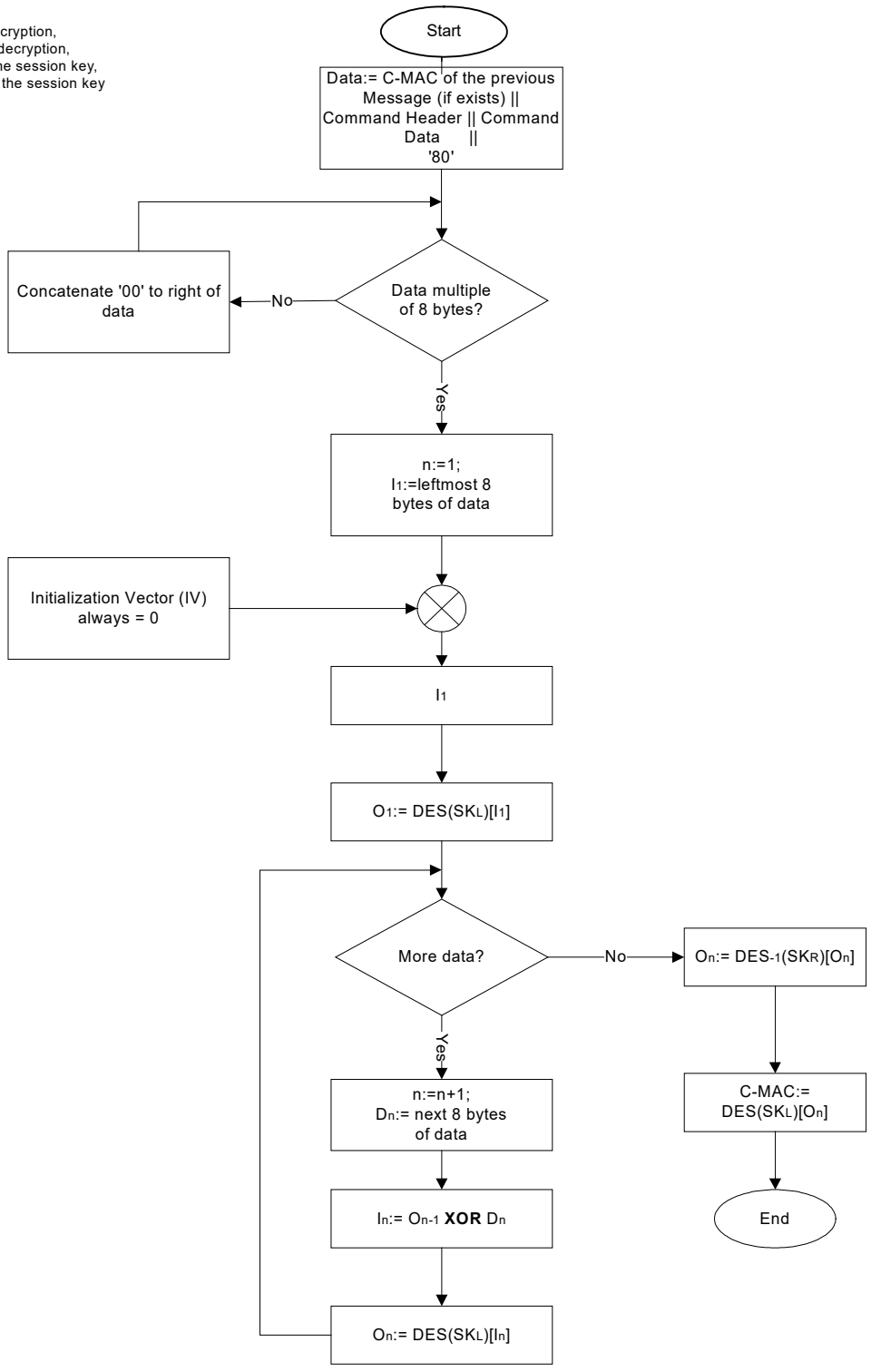
6.4.4.3 Both the data preparation system and the personalisation system must create the MAC. The personalisation system must compare the MAC it creates to the MAC in the field MAC_{INP} within the personalisation data file to verify it. This includes ensuring that the length of the MAC_{INP} field in the personalisation data file matches the length that is expected to be received from the data preparation system (4 or 8 bytes for DES, 8 or 16 bytes for AES).

6.4.4.4 Once the input to an application has been verified, subsequent processing must ensure that the same input is delivered to the application without alteration or addition.

Figure 10 C-MAC and MAC Computation SCP02 Only

Procedure: C-MAC and MAC Computation

Note: In this diagram,
DES indicates single DES encryption,
DES-1 indicates single DES decryption,
SK_L is the leftmost bytes of the session key,
SK_R is the rightmost bytes of the session key



6.5 Encryption

This section describes the encryption of secret data during personalisation.

After personalisation, confidential or secret data may be exchanged between a terminal and an IC card application. For example, a PIN may be changed between a terminal and an IC card during an online transaction. This section does not apply to encryption of secret data after personalisation. Post personalisation encryption is covered in application specific documents.

6.5.1 Secret Data Encryption Using ECB Mode

6.5.1.1 The data preparation function may encrypt DES, AES, RSA and ECC keys and secret data e.g. PIN Block with triple DES or AES in ECB mode using a Transport Key. Although this specification supports the encryption of data under an AES transport key using ECB mode this is not recommended and CBC mode should be used instead as described in section 6.5.2.

6.5.1.2 For SCP02 the personalisation device must encrypt keys and secret data with triple DES in ECB mode using the session key SKU_{DEK}. For SCP03 CBC mode is used as described in section 6.5.2.

6.5.1.3 Triple DES/AES in ECB mode, as defined in ISO/IEC 10116, is used. Note that any required padding will have been added by the data preparation system when the data grouping is created as described in section 3.2.

6.5.2 Secret Data Encryption Using CBC Mode

6.5.2.1 The data preparation function may encrypt DES, AES and RSA keys and secret data e.g. PIN Block, with triple DES or AES in CBC mode using a Transport Key and a Starting Variable produced by encrypting with the Transport Key a binary counter starting at 1b and counting the invocations of CBC mode with a given key.

6.5.2.2 For SCP03 the personalisation device must encrypt keys and secret data with AES in CBC mode using the key K_{DEK} and a Starting Variable of all zeros. For SCP02 ECB mode is used as described in section 6.5.1.

6.5.2.3 Triple DES/AES in CBC mode, as defined in ISO/IEC 10116, is used. Note that any required padding will have been added by the data preparation system when the data grouping is created as described in section 3.2.

6.5.3 Command Data Encryption Using CBC Mode

6.5.3.1 If the security level set in EXTERNAL AUTHENTICATE command requires MAC and encryption, the personalisation device must encrypt the APDU command data field in CBC mode using the session key SKU_{ENC} either after the MAC has been computed for SCP02 or before the MAC has been computed for SCP03. Input to the encryption process is first padded to the right with '80'. The result is padded to the right with up to 7/15 bytes of '00' (possibly none) to make the input data a multiple of 8/16-byte blocks depending on whether the underlying block cipher is DES or AES.

6.5.3.2 Encryption of data must use either triple DES in CBC mode, as defined in ISO/IEC 10116 with a Starting Variable equal to '00 00 00 00 00 00 00 00' or AES in CBC mode, as defined in ISO/IEC 10116 and following the guidance of ISO/IEC 10116 where a Starting Variable is produced by AES encrypting with SKU_{ENC} a 128 bit binary counter starting at 1b and counting the invocations of CBC mode with a given key.

6.6 Decryption

The personalisation device must decrypt secret data encrypted by the data preparation process. This secret data will then be re-encrypted prior to sending to the IC card. The IC card should decrypt the secret data prior to storing it for future use. This section describes the decryption of secret data during personalisation.

6.6.1 Secret Data Decryption Using ECB Mode

6.6.1.1 The personalisation device must use the TK identified in the input record for the decryption of secret data (prior to re-encryption under SKU_{DEK} by the personalisation device) with triple DES or AES in ECB mode. Although this specification supports the encryption of data under an AES transport key using ECB mode this is not recommended and CBC mode should be used instead as described in section 6.6.2.

6.6.1.2 For SCP02 the IC card must use SKU_{DEK} for decryption of encrypted data grouping values with triple DES in ECB mode. Note that for SCP03 ECB mode is not used for this purpose instead CBC mode is used as described in section 6.6.2.

6.6.1.3 Triple DES or AES in ECB mode, as defined in ISO/IEC 10116, is used.

6.6.2 Secret Data Decryption Using CBC Mode

6.6.2.1 The personalisation device must use the TK identified in the input record for the decryption of secret data (prior to re-encryption under K_{DEK} by the personalisation device) with triple DES or AES in CBC mode as defined in ISO/IEC 10116 with a Starting Variable produced by encrypting with the TK a binary counter starting at 1b and counting the invocations of CBC mode with a given key.

6.6.2.2 For SCP03 the IC card must use K_{DEK} for decryption of encrypted data with AES in CBC mode as defined in ISO/IEC 10116 with a Starting Variable of all zeros. For SCP02 ECB mode is used as described in section 6.6.1.

6.6.2.3 If the data grouping contains data that is not a symmetric block cipher key or a PIN block, then the padding must be removed after decryption by the IC card. The IC card determines the length of the padding by searching the decrypted data stream from the rightmost byte. The first '80' found is the start of the padding. If the data enciphered is a symmetric key or keys that is a multiple of 8 bytes but not 16 bytes or is an 8-byte PIN block, then the 8 bytes of random data padding at the rightmost end must be removed.

6.6.3 Command Data Decryption Using CBC Mode

6.6.3.1 The SKU_{ENC} must be used for decryption done by the IC card.

6.6.3.2 Decryption of data must be done either with triple DES in CBC mode, as defined in ISO/IEC 10116 with a Starting Variable equal to '00 00 00 00 00 00 00 00', or AES in CBC mode, as defined in ISO/IEC 10116 and following the guidance of ISO/IEC 10116 where and a Starting Variable is produced by AES encrypting with SKU_{ENC} a 128 bit binary counter starting at 1b, and counting the invocations of CBC mode with a given key.

6.6.3.3 Padding should be removed after decryption by the IC card. The IC card determines the length of the padding by searching the decrypted data stream from the rightmost byte. The first '80' found is the start of the padding.

6.7 Block Cipher Calculations

Triple DES uses a compound operation of DES encryption and decryption. DES and triple DES are defined in ISO/IEC 18033-3. Triple DES, as used in this specification, uses keying option 2 as defined in ISO/IEC 18033-3.

AES is defined in ISO/IEC 18033-3 (also in NIST FIPS 197). AES block cipher modes of operation (ECB/CBC) are defined in ISO/IEC 10116. AES is used for MAC creation (CMAC) and key derivation (as a component of a KDF). This Card Personalisation Specification relies on the CMAC construction described in NIST “Special Publication 800-38B -The CMAC Mode for Authentication”, using AES as the underlying block cipher for MAC constructions. NIST Special Publication 800-108 describes the use of a block cipher such as AES in the derivation of keys and this is used by SCP03 and this specification. In brief, key/challenge/cryptogram derivation processes shall use KDF in counter mode as specified in NIST SP 800-108 ([NIST 800-108]). The PRF used in the KDF shall be CMAC as specified in [NIST 800-38B], using AES as the block cipher.

7 Personalisation Data Elements

The data elements are presented in alphabetical sequence of abbreviated name.

7.1 ACT (Action to be Performed)

Purpose: Identifier for the action to be performed in a Processing Step.

Format: 1 byte binary.

Content: A value of '0F' is used for the personalisation step in conjunction with DGIs. A value of '0B' is used for the personalisation step in conjunction with pre-computed APDU commands. See the appropriate platform reference for other values.

7.2 AID (Application Identifier)

Reference: ISO/IEC 7816-5.

Purpose: Identifier for the application. Used during Application Selection as defined in EMV Version 4.3 Book 1.

Format: 5-16 bytes.

Content: RID || PIX where the RID is the five-byte global registered identifier as specified in ISO/IEC 7816–5 and the PIX (0–11 bytes) is at the discretion of the owner of the RID.

7.3 ALGSCP (Algorithm for Secure Channel Protocol)

Purpose: Identifies the Secure Channel Protocol that is implemented on the IC card.

Format: 1 byte, binary.

Content: '02' – session keys are generated, and MACs are generated and validated as described in SCP02.

'03' – session keys are generated, and MACs are generated and validated as described in SCP03.

7.4 C-MAC

In SCP02 a C-MAC is generated by an off-card entity and applied across the full APDU command being transmitted to the card including the C-MAC from the previous command (if there was one) - prepended to the header and data field of the command. The scope of the MAC calculation does not include Le.

SCP03 is identical except that the entire 16-byte CMAC precursor result used to create the C-MAC of the previous command is prepended rather than the C-MAC value.

Purpose: To protect the integrity of a command individually and within an ordered sequence of commands sent to the card during a secure channel session.

Format: Binary, 8 bytes for SCP02 & S8 variant of SCP03; 16 bytes for S16 variant of SCP03.

Content: var.

7.5 CMODE (Chaining Mode)

Purpose: Identifies the chaining mode to use to decrypt the related DGI

Format: 1 byte, binary.

Content: '11' – ECB or '10' – CBC mode as described in sections 6.6.1 and 6.6.2 respectively.

7.6 CSN (Chip Serial Number)

Purpose: To uniquely identify each chip from a specific chip/card manufacturer.

Format: Binary, variable.

7.7 DTHR (Date and Time)

Purpose: The date and time in YYMMDDHHMMSS format.

Format: BCD, 6 bytes.

7.8 ENC (Encryption Personalisation Instructions)

Purpose: To specify the data groupings that must be encrypted. See section 3.4.3.

Format: Binary, variable.

7.9 ID_{TK} (Identifier of the Transport Key)

Purpose: Identifier of the key used to encrypt secret data sent from the data preparation process to the personalisation device.

Format: Binary, 12 bytes.

Content: See Table 3-8.

7.10 ID_{OWNER} (Identifier of the Application Specification Owner)

Purpose: Used to identify the owner of the specifications for this application.

Format: Binary, variable.

Content: It is recommended that the RID of the owner of the specifications for the application be used in the coding of this field.

7.11 ID_{TERM} (Identifier of the Personalisation Device)

Purpose: Identifies the device used to personalise an IC card.

Format: Binary, 4 bytes.

7.12 K_{ENC} (DES/AES Key)

Purpose: This key is created prior to the start of the personalisation process. In SCP02 it is the DES key used create DES session keys in turn used for authentication cryptograms and confidentiality. In SCP03 it is an AES key used for card challenge generation and the creation of personalisation session keys for confidentiality. The session keys are designated SKU_{ENC}.

Format: Binary, 16 or 32 bytes.

Notes: If DES must be generated with odd parity.

7.13 K_{DEK} (DES/AES Key)

Purpose: This key is created prior to the start of the personalisation process. In SCP02 it is a DES key used to create a personalisation session key (SKU_{DEK}) for Key and PIN Encryption. In SCP03 it is an AES key used directly as is for Key and PIN Encryption.

Format: Binary, 16 or 32 bytes.

Notes: If DES must be generated with odd parity.

7.14 K_{MAC} (DES/AES Key)

Purpose: DES or AES key, created prior to the start of the personalisation process. In SCP02 it is used in card challenge generation and in SCP03 it is used for authentication cryptograms. As appropriate it is used by the personalisation process to create the session key SKU_{MAC} a MAC session key.

Format: Binary, 16 or 32 bytes.

Notes: If DES must be generated with odd parity.

7.15 Key Check Value

Purpose: The data is used to prove that a card/processor has access to a specific DES/AES key value.

Format: Binary, 3 bytes.

Contents: If DES based, the three leftmost bytes of the result of ECB encrypting eight bytes of zeros by the DES key concerned, alternatively if AES based the three leftmost bytes of the result of ECB encrypting 16 bytes of '01'.

7.16 KEYDATA (Diversification Data for Secure Channel Keys)

Purpose: The data used to derive the K_{DEK}, K_{MAC} and the K_{ENC} from the KMC.

Format: Binary, 10 bytes.

Contents: See Table 4-3.

7.17 KMC (DES/AES Master Key for Personalisation Session Keys)

Purpose: DES or AES key used for generating derived keys to create MACs and encrypt and decrypt DES/AES keys and secret data during personalisation (K_{ENC}, K_{MAC} and K_{DEK}).

Format: Binary, 16 or 32 bytes.

Notes: If DES must be generated with odd parity.

7.18 KMC_{ID} (Identifier of the Master Key for Personalisation)

Purpose: Data required by the personalisation device for identification of the Master Key for personalisation. This field is used to determine which KMC is to be used to derive the keys for secure channel. This data must be placed in the IC card prior to personalisation and must be retrievable from the KEYDATA returned from the INITIALIZE UPDATE command.

This field will normally contain the card issuer BIN/IIN (e.g. IIN right justified and left padded with 1111b per quartet). However, if the personalisation device is at a personalisation bureau that uses an identifier for card issuers other than a BIN/IIN, that identifier may be used in this field.

If a card issuer has multiple card suppliers and uses different KMCs for each card supplier, this field may contain an identifier of the card supplier as well as the identifier of the card issuer.

The KMC_{ID} is used as input data to derive the card static keys (K_{ENC}, K_{MAC}, K_{DEK}).

Format: Binary, 6 bytes.

7.19 L (Length of Data)

Purpose: The length of data that follows.

Format: Binary, variable.

Remarks: Length and content depend upon usage.

7.20 LCCA (Length of IC Card Application Data)

Purpose: The length of the data for IC card application(s).

Format: ASCII decimal digits, padded at the most significant end by ASCII zeros ('30'), 7 bytes.

7.21 LOGDATA (Data Logging Personalisation Instructions)

Purpose: To specify the data that must be logged by the personalisation process. See section 3.5.

Format: Binary, variable.

7.22 MAC_{INP} (MAC of All Data for an Application)

Purpose: A MAC created over all of the data for an IC card application as described in section 6.4.4. MAC_{INP} is the 4/8/16 leftmost bytes of the created MAC.

Format: Binary, 4 or 8 bytes for DES, 8 or 16 for AES.

7.23 MACkey (MAC Key)

Purpose: DES or AES key used to create MAC_{INP}. This key should be uniquely created for the data for each application on each IC card.

Format: Binary, 16 or 32 bytes.

7.24 MIC (Module Identifier Code)

Purpose: An identifier that specifies that this is data for IC card application(s). The length and values of this field are established when the personalisation device is configured.

Format: ASCII, variable, 1 to 7 bytes.

7.25 ORDER (Data Grouping Order Personalisation Instructions)

Purpose: To specify the order in which data groupings must be sent to the IC card. See section 3.4.1.

Format: Binary, variable.

7.26 POINTER (Additional Pointer to Personalisation Data or Instructions)

Purpose: A pointer to additional data or instructions for the personalisation process. A separate pointer is available for each Processing Step.

The usage of this field is outside the scope of this specification. It is envisioned that is will be set based on agreements between data preparation systems and personalisation systems.

Format: Binary, variable.

7.27 R-MAC

In SCP03, if required by the security level, an R-MAC is generated by the IC card over the response from the IC card and includes the entire 16-byte CMAC precursor result used to verify the C-MAC from the command prepended to the response data field and SW1 SW2 of the response.

Purpose: To protect the integrity of a response individually and within an ordered sequence of responses sent by the card during a secure channel session.

Format: Binary, 8 bytes for SCP02 & S8 variant of SCP03; 16 bytes for S16 variant of SCP03.

Content: var.

7.28 R_{CARD} (Pseudo-Random Number from the IC Card)

Purpose: A pseudo-random number (see 4.3.2.9) generated by the IC card or the IC card application. Used in the creation of the host and card cryptograms.

Format: Binary, 6 bytes for SCP02, 8 bytes for S8 variant of SCP03, or 16 bytes for S16 variant of SCP03.

7.29 R_{TERM} (Random Number from the Personalisation Device)

Purpose: A random number generated by the personalisation device. Used in the creation of the host and card cryptograms.

Format: Binary, 8 bytes for SCP02, 8 bytes for S8 variant of SCP03, or 16 bytes for S16 variant of SCP03.

7.30 RANDOM (Random Number)

Purpose: May be used during encryption and decryption of secret data encrypted with a TK.

Format: Binary, variable – multiple of 8 bytes recommended.

Content: A random number to be used during personalisation processing.

7.31 REQ (Required or Optional Action)

Purpose: Indicates whether the action to be performed in a Processing Step is required or optional. If a Processing Step is required, it must be done, even if it has been done before. If a Processing Step is optional, it must not be re-done if it has been done before.

Format: 1 byte binary.

Content: '00' is optional, '01' is required.

7.32 SEQNO (Sequence Number)

Purpose: The sequence number of a personalised IC card in a run of IC cards personalised. The first card has sequence number 1, the second, number 2, etc.

Format: Binary, 3 bytes.

7.33 SKU_{ENC} (Personalisation Session Key for Confidentiality and SCP02 Authentication Cryptogram)

Purpose: DES or AES key, created during the personalisation process, used to create SCP02 authentication cryptograms and may be used to encrypt and decrypt secret data in CBC mode for both SCP02 and SCP03.

Format: Binary, 16 or 32 bytes.

Content: Derived as described in section 6.3.

Remarks: Parity convention not required.

7.34 SKU_{DEK} (SCP02 Personalisation Session Key for Key and PIN Encryption)

Purpose: DES key created during the SCP02 personalisation process, used to encrypt and decrypt secret data in ECB mode.

Format: Binary, 16 bytes.

Content: Derived as described in section 6.3.

Remarks: Parity convention not required.

7.35 SKU_{MAC} (Personalisation Session Key for MACing, SCP02 Card Challenge Generation, and SCP03 Authentication Cryptograms)

Purpose: DES or AES key, created during the personalisation process, used for SCP02 card challenge generation, SCP03 authentication cryptogram creation, and when secure MACing is required to create C-MACs.

Format: Binary, 16 or 32 bytes.

Content: Derived as described in section 6.3.

Remarks: Parity convention not required.

7.36 TAG (Identifier of Data for a Processing Step)

Purpose: Identifier for the data in ICC Data in the input record to be used in a Processing Step.

Format: Binary, variable.

Content: A BER TLV encoded tag. A value of 'EF' is used for the personalisation Processing Step '0F'. A value of 'EE' is used for the personalisation Processing Step '0B'. See the appropriate platform reference for other values.

7.37 TK (Transport Key)

Purpose: A DES or AES key used to encrypt other key values for transmission between the data preparation system and the personalisation device.

Format: Binary, 16 or 32 bytes.

Remarks: This key is not related to the K_{DEK} and the SKU_{DEK} used to encrypt secret data sent to an IC card.

7.38 $TYPE_{TK}$ (Indicator of Use(s) of Transport Key)

Purpose: To identify the uses of a Transport Key

Format: Binary, 1 byte.

Content: Provides information on what the TK encrypts and the encryption algorithm used. See Table 7-1.

If a TK is a general purpose TK with an encryption algorithm (b7 and b6) = 01b, then the TK encrypts secret data as described in section 6.5. If a TK is indicated by an encryption method (b7 and b6) of 10b, then the secret data is XORed with the RANDOM number for this application before encryption (by the data preparation system) and after decryption (by the personalisation system).

Table 7-1 Coding of $TYPE_{TK}$

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
x								TK for MACkey encryption
0								TK not used for MACkey encryption
1								TK used for MACkey encryption

b8	b7	b6	b5	b4	b3	b2	b1	Meaning
	x	x						Encryption method
	0	1						General purpose TK
	1	0						TK using RANDOM field and XOR operation
	0	0						RFU
	1	1						RFU
			x	x	x			RFU
						x	x	RFU for Proprietary Implementations

7.39 VERCNTL (Version Control Personalisation Instructions)

Purpose: To specify the data groupings that may be rejected by the IC card without interrupting the personalisation process. See section 3.4.2.

Format: Binary, variable.

7.40 VNL (Version Number of Layout)

Purpose: The version number of the layout of the file from the data preparation process to the personalisation device.

Format: Binary or ASCII, 4 bytes.

Content: ”02.2” (ASCII) is defined for this specification.

Annex A Common EMV Data Groupings

A.1 Introduction

This Annex defines DGIs for use in the personalisation of EMV card applications that are common between payment systems. The first group of common DGIs is linked to the main EMV payment application. The second group of DGIs is linked to the Payment System Environment (PSE) application.

A.2 Common DGIs for EMV Payment Applications

The recommended common data groupings for EMV payment applications are defined below.

Table A-1 Data Grouping Identifiers for Payment Applications

DGI	Data Content	Function	Encrypt	External Access
'8000'	Block cipher (DES/AES) keys – Table A-2	CAM* / Issuer Auth/ Issuer Script	Yes	None
'9000'	Block cipher (DES/AES) Key Check Values – Table A-3		No	None
'8010'	Offline PIN Block - Table A-4	Offline PIN	Yes	PIN CHANGE / UNBLOCK
'801n'**	Duplicate Offline PIN Block - Table A-5	Offline PIN	Yes	Same as above
'9010'	PIN Related Data - Table A-6	PIN Try Limit/Counter	No	GET DATA
'901n'**	Duplicate PIN Related Data - Table A-7	PIN Try Limit/Counter	No	Same as above
'8101'	ICC Private Key (DDA/PIN Encipherment) - Table A-8	DDA/PIN Encipher	Yes	None
'8102'	ICC PIN Encipherment Private Key - Table A-9	PIN Encipher	Yes	None
'8103'	ICC Modulus (DDA/PIN Encipherment) - Table A-10	DDA/PIN Encipher	Yes	None
'8104'	ICC Modulus PIN Encipherment - Table A-11	PIN Encipher	Yes	None

DGI	Data Content	Function	Encrypt	External Access
'8105'	ICC ECC Secret Key - Table A-11b	XDA/Offline Data Encipherment	Yes	None
'8106'	ICC ECC Offline Data Encipherment Secret Key - Table A-11c	Offline Data Encipherment	Yes	None
'8201'	CRT constant DDA/PIN Encipherment - Table A-12 and Table A-12b	DDA/PIN Encipher	Yes	None
'8202'	CRT constant DDA/PIN Encipherment - Table A-12 and Table A-12b	DDA/PIN Encipher	Yes	None
'8203'	CRT constant DDA/PIN Encipherment - Table A-12 and Table A-12b	DDA/PIN Encipher	Yes	None
'8204'	CRT constant DDA/PIN Encipherment - Table A-12 and Table A-12b	DDA/PIN Encipher	Yes	None
'8205'	CRT constant DDA/PIN Encipherment - Table A-12 and Table A-12b	DDA/PIN Encipher	Yes	None
'8301'	CRT constant PIN Encipherment - Table A-13 and Table A-13b	PIN Encipher	Yes	None
'8302'	CRT constant PIN Encipherment - Table A-13 and Table A-13b	PIN Encipher	Yes	None
'8303'	CRT constant PIN Encipherment - Table A-13 and Table A-13b	PIN Encipher	Yes	None
'8304'	CRT constant PIN Encipherment - Table A-13 and Table A-13b	PIN Encipher	Yes	None
'8305'	CRT constant PIN Encipherment - Table A-13 and Table A-13b	PIN Encipher	Yes	None
'9102'	SELECT Response Data - Table A-14	-	No	SELECT
'9104'	GPO Response Data - Table A-15	-	No	GET PROCESSING OPTIONS
'91nn'**	Duplicate GPO Response Data - Table A-16	-	No	GET PROCESSING OPTIONS
'3000'	Application Common Internal data – Table A-17	-	No	Data dependent
'3001'	Application Internal data – Table A-18	-	No	Data dependent

NOTE: * Card Authentication Method

** Indicates store in last record(s) in the file, because it is a duplicate data element

The requirement column titled “Req.”, in the following tables of data elements for each DGI, lists the requirements for each data element:

M (Mandatory) indicates that the data element must be present.

C (Conditional) indicates that the data element is necessary under certain conditions.

O (Optional) indicates that the data element is optional.

Table A-2 Data Content for DGI '8000'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	Unique Derivation Key (UDK) DES/AES Key	16 or 32	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
		Message Authentication (MAC UDK) DES/AES Key	16 or 32	
		Data Encipherment (ENC UDK) DES/AES Key	16 or 32	

Table A-3 Data Content for DGI '9000'

Req.	Tag	Data Element	Length	Encrypt
O	N/A	Key Check Values for card keys UDK, MAC UDK and ENC UDK	3-9	N/A

Table A-4 Data Content for DGI '8010'

Req.	Tag	Data Element	Length	Encrypt
C	—	Reference PIN Block (see section 3.4.4)	8 or 16	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-5 Data Content for DGI '801n'

Req.	Tag	Data Element	Length	Encrypt
C	—	Duplicate Reference PIN Block (see section 3.4.4)	8 or 16	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-6 Data Content for DGI '9010'

Req.	Tag	Data Element	Length	Encrypt
C	—	PIN Try Counter (Tag '9F17')	1	N/A
C	—	PIN Try Limit	1	N/A

Table A-7 Data Content for DGI '901n'

Req.	Tag	Data Element	Length	Encrypt
C	—	Duplicate PIN Try Counter (Tag '9F17')	1	N/A
C	—	Duplicate PIN Try Limit	1	N/A

To support DDA (and also PIN Encipherment using the same key), personalise either DGIs '8101' ICC Private Key and '8103' Modulus, or '8201' to '8205' CRT (Chinese Remainder Theorem) constants.
Additionally, to support a separate key for PIN Encipherment, personalise either DGIs '8102' and '8104', or '8301' to '8305'.

Table A-8 Data Content for DGI '8101'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC Private Key (DDA / PIN Encipherment) Exponent	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-9 Data Content for DGI '8102'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC Private Key (PIN Encipherment) Exponent	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-10 Data Content for DGI '8103'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC Key (DDA / PIN Encipherment) Modulus	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-11 Data Content for DGI '8104'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC Key (PIN Encipherment) Modulus	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-11b Data Content for DGI '8105'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC ECC Secret Key (XDA / Offline Data Encipherment)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-11c Data Content for DGI '8106'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	ICC ECC Offline Data Encipherment Secret Key	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-12 Data Content for DGI '8301' through '8305' for p^{-1} mod q convention

Req.	Tag	Data Element	Length	Encrypt
C	N/A	CRT constant p^{-1} mod q	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant d mod $(p - 1)$	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant d mod $(q - 1)$	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant prime factor p	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant prime factor q	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-12b Data Content for DGI '8201' through '8205' for q^{-1} mod p convention

Req.	Tag	Data Element	Length	Encrypt
C	N/A	CRT constant q^{-1} mod p	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant d mod (q – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant d mod (p – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant prime factor q	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant prime factor p	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-13 Data Content for DGI '8301' through '8305' for p^{-1} mod q convention

Req.	Tag	Data Element	Length	Encrypt
C	N/A	CRT constant p^{-1} mod q	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant d mod (p – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

C	N/A	CRT constant *d* mod (*q* – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant the prime factor *p*	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant the prime factor *q*	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-13b Data Content for DGI '8301' through '8305' for q^{-1} mod p convention

Req.	Tag	Data Element	Length	Encrypt
C	N/A	CRT constant q^{-1} mod *p*	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant *d* mod (*q* – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant *d* mod (*p* – 1)	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
C	N/A	CRT constant the prime factor *q*	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

C	N/A	CRT constant the prime factor *p*	var.	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)

Table A-14 Data Content for DGI '9102'

Req.	Tag	Data Element	Length	Encrypt
M	'A5'	FCI Proprietary Template	var.	
M	'50'	Application Label	1-16	N/A
C	'87'	Application Priority Indicator	1	N/A
C	'9F38'	Processing Option Data Object List (PDOL)	var.	N/A
O	'5F2D'	Language Preference	2-8	N/A
C	'9F11'	Issuer Code Table Index	1	N/A
O	'9F12'	Application Preferred Name	1-16	N/A
O	'BF0C'	FCI Issuer Discretionary Data	var.	N/A

Table A-15 Data Content for DGI '9104'

Req.	Tag	Data Element	Length	Encrypt
C	'82'	Application Interchange Profile (AIP)	2	N/A
C	'94'	Application File Locator (AFL)	var.	N/A

Table A-16 Data Content for DGI '91nn'

Req.	Tag	Data Element	Length	Encrypt
C	'82'	Duplicate Application Interchange Profile (AIP)	2	N/A
C	'94'	Duplicate Application File Locator (AFL)	var.	N/A

Table A-17 Application Common Internal Data Content for DGI '3000'

Req.	Tag	Data Element	Length	Encrypt
C	'9F36'	ATC	2	N/A
C	'9F4F'	Log Format	var.	N/A

Table A-18 Application Internal Data Content for DGI '3001'*

Req.	Tag	Data Element	Length	Encrypt
	Non-common Tags	Application Internal Data Elements		

*If necessary multiple DGI '3001' can be submitted to the application.

A.3 Common DGIs for EMV PSE

The recommended data groupings for the EMV Payment System Environment (PSE) application are defined below.

Table A-19 Data Grouping Identifiers for PSE

DGI	Data Content	Function	Encrypt	External Access
'0101'	First Record Data - Table A-19	-	No	READ RECORD
'01nn'	Subsequent Record Data - Table A-19	-	No	READ RECORD
'9102'	SELECT PSE Response Data - Table A-20	-	No	SELECT

Table A-20 Data Content for DGI '0101' and '01nn'

Req.	Tag	Data Element	Length	Encrypt
M	'70'	Record Template	var.	N/A
M	'61'	Directory Entry Template	var.	N/A
M	'4F'	Dedicated File Name (AID)	5-16	N/A
O	'50'	Application Label	1-16	N/A
O	'9F12'	Application Preferred Name	1-16	N/A
C	'87'	Application Priority Indicator (used when there is more than one payment application on the card)	1	N/A
O	'73'	Directory Discretionary Template	var.	N/A

Table A-21 Data Content for DGI '9102'

Req.	Tag	Data Element	Length	Encrypt
M	'A5'	FCI Proprietary Template	var.	N/A
M	'88'	SFI of the directory elementary file	1	N/A
O	'5F2D'	Language Preference	2-8	N/A
O	'9F11'	Issuer Code Table Index	1	N/A
O	'BF0C'	FCI Issuer Discretionary Data	var.	N/A

A.4 Common DGIs to Load/Update Secure Channel Static Keys

The Data Grouping Identifiers '7F01', '8F01' and '00CF' are recommended for use to load or update the secure channel static keys i.e. K_{ENC}, K_{MAC} and K_{DEK}. The condition of use is that the application allows load or update of the secure channel static keys after initialisation of the card. Key lengths can be 16 bytes DES/AES-128 or 32 bytes for AES-256.

These DGIs are defined to standardise any secure channel static keys update after initialisation of the card.

Table A-22 Data Grouping Identifiers for Secure Channel Static Keys

DGI	Data Content	Function	Encrypt	External Access
'7F01'	Secure Channel block cipher (DES/AES) keys related data – Table A-23	Related data for Load/Update personalisation static keys K_{ENC}, K_{MAC} and K_{DEK}	No	None
'8F01'	Secure Channel block cipher (DES/AES) Keys – Table A-24	Load/Update personalisation static keys K_{ENC}, K_{MAC} and K_{DEK}	Yes	None
'00CF'	Secure Channel block cipher (DES/AES) Key diversification data - Table A-25	Diversification data for Load/Update personalisation static keys K_{ENC}, K_{MAC} and K_{DEK}	No	INITIALIZE UPDATE GET DATA (Tag 'CF')

Table A-23 Data Content for DGI '7F01'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	Current Key Version Number ('00' or '01' to '6F')	1	N/A
		New Key Version Number ('01' to '6F')	1	
		Key type ('80' for DES, '88' for AES)	1	
		Check value for new Key Identifier 1	3	
		Check value for new Key Identifier 2	3	
		Check value for new Key Identifier 3	3	

Table A-24 Data Content for DGI '8F01'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	New Key Identifier 1 (Encryption)	16 or 32	SKU_{DEK} (SCP02) or K_{DEK} (SCP03)
		New Key Identifier 2 (MAC)	16 or 32	
		New Key Identifier 3 (DEK)	16 or 32	

Table A-25 Data Content for DGI '00CF'

Req.	Tag	Data Element	Length	Encrypt
C	N/A	New Key diversification data	10	N/A

A.5 Common DGIs to Create File Structure for EMV Records

This provides a recommended way to create EFs under the EMV application in order to create, update, and read EMV, payment system, and issuer records.

The creation of EFs used as internal files (including files to store PIN and application keys) for EMV application is outside of scope of this document.

The Data Grouping Identifier '0062' is recommended for use to create EFs under the EMV CPS compliant application. One or more DGI '0062' may be used to create EFs.

Table A-26 Data Grouping Identifiers for Secure Channel Static Keys

DGI	Data Content	Function	Encrypt	External Access
'0062'	Concatenation of (one FCP per EF) File Control Parameter – Table A-27	Create EF to store/retrieve EMV, payment system and issuer records	No	READ RECORD STORE DATA UPDATE RECORD

Table A-27 Data Content for DGI '0062'

Req.	Tag	Data Element	Length	Encrypt
M	'62'	FCP	13 or 16	N/A
M	'80'	Number of data bytes in the file, excluding structural information (see ISO/IEC 7816-4 Table 10)	2	N/A
M	'82'	File Descriptor Byte (see ISO/IEC 7816-4 Table 11)	2 or 5	N/A
M		Data Coding Byte (see ISO/IEC 7816-4 Table 118)		
O		Maximum record size on two bytes ('00 01' to '00 FE')		
O		Number of records on one byte ('01' to 'FF')		
M	'88'	Short EF identifier ('01' to '1E')	1	N/A
C	'8C'	Security attribute in compact format – Table A-28	4	N/A

The access rules for the EF are:

- Write (through STORE DATA command) after external authentication at the EMV application level (initiation of a secure channel as defined in this specification)
- Rewrite (through UPDATE RECORD command) after validation of the secure messaging of issuer scripting as defined in EMV Version 4.3 - Book 3
- Read (through READ RECORD command) with no conditions (free access)

If the access rules are implicitly known by the EMV application (at the DF level) then the presence of TLV tagged '8C' is not required or if present it shall be ignored by the EMV application.

If the access rules are to be communicated to the EMV application (at the DF level) then they shall be coded as defined in Table A-28.

Table A-28 Security attribute (tag '8C')

Req.	Data Element	Length	Value	External Access
M	Access mode byte for EFs - Write record - Update record - Read record	1	'07'	
M	Write record allowed after initiation of a secure channel with DF (SE 1)	1	'A1'	STORE DATA
M	Update record allowed after validation of secure messaging of issuer scripting (SE 2)	1	'C2'	UPDATE RECORD
M	Read record allowed with no conditions	1	'00'	READ RECORD

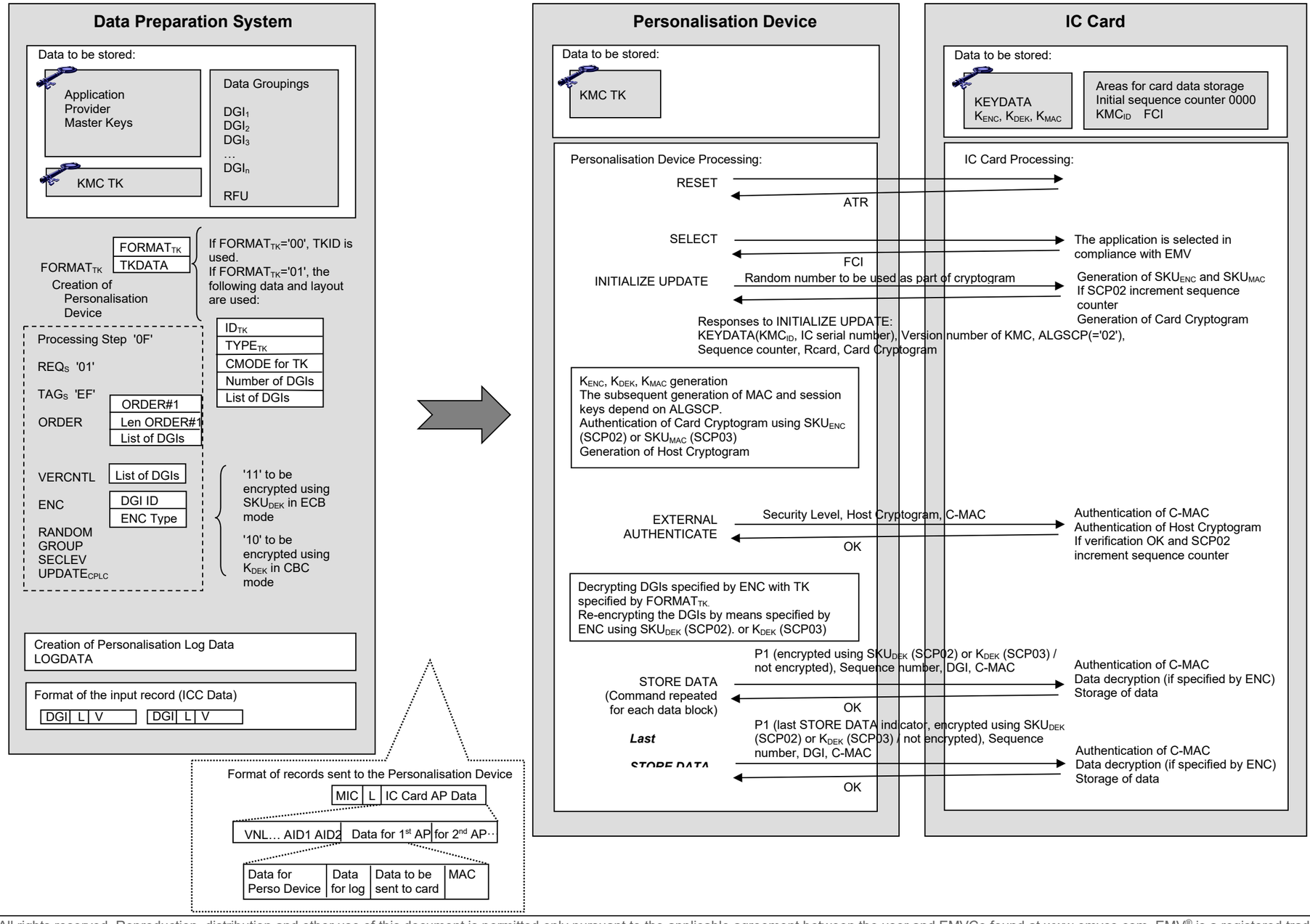
For an explanation on the value of the access mode byte for EFs see ISO/IEC 7816-4 Table 18.

For an explanation on the value of the security condition byte see ISO/IEC 7816-4 Table 30.

The security environment 1 (SE 1) shall indicate access after external authentication at the EMV application level (initiation of a secure channel as defined in this specification) and shall be implicitly known by the EMV application (at the DF level).

The security environment 2 (SE 2) shall indicate access after validation of the secure messaging of issuer scripting as defined in EMV Version 4.3 - Book 3 and shall be implicitly known by the EMV application (at the DF level).

Annex B Overview of EMV Card Personalisation Indirect Method



***** END OF DOCUMENT *****