EMV2000
Integrated Circuit Card Specification for Payment Systems

Book 2 - Security and Key Management

Version 4.0
December, 2000

Table of Contents

1. Scope 5
2. Normative References 6
3. Definitions 7
4. Abbreviations and Notations 11
5. Static Data Authentication 15
5.1 Keys and Certificates 16
5.2 Retrieval of the Certification Authority Public Key 19
5.3 Retrieval of the Issuer Public Key 20
5.4 Verification of the Signed Static Application Data 22
6. Dynamic Data Authentication 24
6.1 Keys and Certificates 27
6.2 Retrieval of the Certification Authority Public Key 30
6.3 Retrieval of the Issuer Public Key 30
6.4 Retrieval of the ICC Public Key 32
6.5 Standard Dynamic Data Authentication 34
6.5.1 Dynamic Signature Generation 34
6.5.2 Dynamic Signature Verification 36
6.6 Combined Dynamic Data Authentication/Application Cryptogram Generation 37
6.6.1 Dynamic Signature Generation 37
6.6.2 Dynamic Signature Verification 39
7. Personal Identification Number Encipherment 42
7.1 Keys and Certificates 42
7.2 PIN Encipherment and Verification 44
8. Application Cryptogram and Issuer Authentication 47
8.1 Application Cryptogram Generation 47
8.1.1 Data Selection 47
8.1.2 Application Cryptogram Algorithm 48
8.2 Issuer Authentication 48
8.3 Key Management 48
9. Secure Messaging 49
9.1 Secure Messaging Format 49
9.2 Secure Messaging for Integrity and Authentication 49
9.2.1 Command Data Field 49
9.2.2 MAC Session Key Derivation 50
9.2.3 MAC Computation 50
9.3 Secure Messaging for Confidentiality 51
9.3.1 Command Data Field 51
9.3.2 Encipherment Session Key Derivation 51
9.3.3 Encipherment/Decipherment 51
9.4 Key Management 52
10. Certification Authority Public Key Management Principles and Policies 53
10.1 Certification Authority Public Key Life Cycle 53
10.1.1 Normal Certification Authority Public Key Life Cycle 53
10.1.2 Certification Authority Public Key Pair Compromise 56
10.2 Key Revocation Principles and Policies by Phase 57

10.2.1 General Principles 57
10.2.2 Planning Phase 58
10.2.3 Generation Phase 59
10.2.4 Distribution Phase 59
10.2.5 Key Usage Phase 60
10.2.6 Detection Phase 61
10.2.7 Assessment Phase 62
10.2.8 Decision Phase 62
10.2.9 Revocation Phase 63
10.3 Sample Timelines 63
10.3.1 Key Introduction 64
10.3.2 Key Withdrawal 65
11. Terminal Security and Key Management Requirements 66
11.1 Security Requirements 66
11.1.1 Tamper-Evident Devices 66
11.1.2 PIN Pads 68
11.2 Key Management Requirements 69
11.2.1 Certification Authority Public Key Introduction 69
11.2.2 Certification Authority Public Key Storage 70
11.2.3 Certification Authority Public Key Usage 71
11.2.4 Certification Authority Public Key Withdrawal 72

Annexes

Annex A - Security Mechanisms 75
A1. Symmetric Mechanisms 75
A1.1 Encipherment 75
A1.2 Message Authentication Code 76
A1.3 Session Key Derivation 77
A1.3.1 Description 78
A1.3.2 Implementation 79
A1.4 Master Key Derivation 81
A2. Asymmetric Mechanisms 82
A2.1 Digital Signature Scheme Giving Message Recovery 82
A2.1.1 Algorithms 82
A2.1.2 Signature Generation 82
A2.1.3 Signature Verification 83
Annex B - Approved Cryptographic Algorithms 85
B1. Symmetric Algorithms 85
B1.1 Data Encryption Standard (DES) 85
B2. Asymmetric Algorithms 85
B2.1 RSA Algorithm 85
B2.1.1 Keys 86
B2.1.4 Key Generation 87
B3. Hashing Algorithms 87
B3.1 Secure Hash Algorithm (SHA-1) 87
Annex C - Informative References 89

Tables

Table 1 - Issuer Public Key Data to be Signed by the Certification Authority 18
Table 2 - Static Application Data to be Signed by the Issuer 19
Table 3 - Data Objects Required for Static Data Authentication 19
Table 4 - Format of the Data Recovered from the Issuer Public Key Certificate 21
Table 5 - Format of the Data Recovered from the Signed Static Application Data 23
Table 6 - Issuer Public Key Data to be Signed by the Certification Authority 28
Table 7 - ICC Public Key Data to be Signed by the Issuer 29
Table 8 - Data Objects Required for Public Key Authentication for Dynamic Authentication 30
Table 9 - Format of the Data Recovered from the Issuer Public Key Certificate 31
Table 10 - Format of the Data Recovered from the ICC Public Key Certificate 33
Table 11 - Dynamic Application Data to be Signed 35
Table 12 - Additional Data Objects Required for Dynamic Signature Generation and Verification 35
Table 13 - Format of the Data Recovered from the Signed Dynamic Application Data 36
Table 14 - Dynamic Application Data to be Signed 38
Table 15 – Contents of the ICC Dynamic Data 38
Table 16 - Additional Data Objects Required for Dynamic Signature Generation and Verification 39
Table 17 - Additional Data Objects Required for Dynamic Signature Generation and Verification 39
Table 18 - Format of the Data Recovered from the Signed Dynamic Application Data 40
Table 19 - ICC PIN Encipherment Public Key Data to be Signed by the Issuer 43
Table 20 - Data Objects Required for the Retrieval of the ICC PIN Encipherment Public Key 44
Table 21 - Data to be Enciphered for PIN Encipherment 45
Table 22 – Recommended Minimum Set of Data Elements for Application Cryptogram Generation 47
Table 23 – Minimum Set of Certification Authority Public Key related Data Elements to be Stored in the Terminal 71
Table 24 - Mandatory Upper Bound for the Size in Bytes of the Moduli 85

Figures

Figure 1 - Diagram of Static Data Authentication 15
Figure 2 - Diagram of Dynamic Data Authentication 25
Figure 3 - Format 1 Command Data Field for Secure Messaging for Integrity and Authentication 50
Figure 4 - Format 2 Command Data Field for Secure Messaging for Integrity and Authentication 50
Figure 5 - Format 1 Enciphered Data Object in a Command Data Field 51
Figure 6 - Format 2 Command Data Field for Secure Messaging for Confidentiality 51
Figure 7 – Certification Authority Public Key Distribution 54
Figure 8 – Issuer Public Key Distribution 55

1. Scope

Book 2 of the *Integrated Circuit Card (ICC) Specification for Payment Systems* describes the minimum security functionality required of integrated circuit cards (ICCs) and terminals to ensure correct operation and interoperability. Additional requirements and recommendations are provided with respect to the on-line communication between ICC and issuer and the management of cryptographic keys at terminal, issuer and payment system level.

More precisely, this Book covers:

- Offline static data authentication.
- Offline dynamic data authentication.
- Offline PIN encipherment.
- Application cryptogram generation and issuer authentication.
- Secure messaging.
- Public key management principles and policies.
- Terminal security and key management requirements.

Furthermore it includes a specification of the security mechanisms and the approved cryptographic algorithms required to implement the security functions specified.

2. Normative References

The following standards contain provisions that are referenced in this specification.

EMV 2000 Version 4.0: December 2000	Integrated Circuit Card Specification for Payment Systems Book 3 - Application Specification
EMV2000 Version 4.0: December 2000	Integrated Circuit Card Specification for Payment Systems Book 4 - Cardholder, Attendant and Acquirer Interface Requirements
FIPS 180-1: 1995	Secure Hash Standard
ISO/IEC 7816-4: 1995	Identification cards – Integrated circuit(s) cards with contacts – Part 4: Inter-industry commands for interchange
ISO 8731-1: 1987	Banking – Approved algorithms for message authentication - Part 1: DEA
ISO DIS 9564-1 (to be published)	Banking – PIN management and security – Part 1: PIN protection principles and techniques
ISO 9564-2: 1991	Banking – PIN management and security – Part 2: Approved algorithms for PIN encipherment
ISO/IEC 9796-2: 1997	Information technology – Security techniques - Digital signature scheme giving message recovery - Part 2: Mechanism using a hash function
ISO/IEC 9797-1: 1999	Information technology – Security techniques – Message Authentication Codes – Part 1: Mechanisms using a block cipher
ISO/IEC 10116: 1997	Information technology - Modes of operation of an n-bit block cipher algorithm
ISO/IEC 10118-3: 1998	Information technology – Security techniques - Hash functions - Part 3: Dedicated hash functions
ISO11568-2: 1994	Banking – Key management (retail) – Part 2: Key management techniques for symmetric ciphers
ISO 13491-1: 1998	Banking – Secure cryptographic devices (retail) – Part 1: Concepts, requirements and evaluation methods

3. Definitions

The following terms are used in this specification.

Accelerated Revocation - A key revocation performed on a date sooner than the published key expiry date.

Application – The application protocol between the card and the terminal and its related set of data.

Asymmetric Cryptographic Technique - A cryptographic technique that uses two related transformations, a public transformation (defined by the public key) and a private transformation (defined by the private key). The two transformations have the property that, given the public transformation, it is computationally infeasible to derive the private transformation.

Authentication - The provision of assurance of the claimed identity of an entity.

Byte - 8 bits.

Card - A payment card as defined by a payment system.

Certificate - The public key and identity of an entity together with some other information, rendered unforgeable by signing with the private key of the certification authority which issued that certificate.

Certificate Revocation - The process of revoking an otherwise valid certificate by the entity that issued that certificate.

Certification Authority - Trusted third party that establishes a proof that links a public key and other relevant information to its owner.

Ciphertext - Enciphered information.

Command - A message sent by the terminal to the ICC that initiates an action and solicits a response from the ICC.

Compromise - The breaching of secrecy or security.

Concatenation - Two elements are concatenated by appending the bytes from the second element to the end of the first. Bytes from each element are represented in the resulting string in the same sequence in which they were presented to the terminal by the ICC, that is, most significant byte first. Within each byte bits are ordered from most significant bit to least significant. A list of elements or objects may be concatenated by concatenating the first pair to form a new element, using that as the first element to concatenate with the next in the list, and so on.

Cryptogram - Result of a cryptographic operation.

Cryptographic Algorithm - An algorithm that transforms data in order to hide or reveal its information content.

Cryptoperiod - Defined period of time during which a specific cryptographic key is authorized for use, or during which time the cryptographic keys for a given system may remain in effect.

Data Integrity - The property that data has not been altered or destroyed in an unauthorised manner.

Decipherment - The reversal of a corresponding encipherment.

Digital Signature - An asymmetric cryptographic transformation of data that allows the recipient of the data to prove the origin and integrity of the data, and protect the sender and the recipient of the data against forgery by third parties, and the sender against forgery by the recipient.

Encipherment - The reversible transformation of data by a cryptographic algorithm to produce ciphertext.

Financial Transaction - The act between a cardholder and a merchant or acquirer that results in the exchange of goods or services against payment.

Hash Function - A function that maps strings of bits to fixed-length strings of bits, satisfying the following two properties:

- It is computationally infeasible to find for a given output an input which maps to this output.
- It is computationally infeasible to find for a given input a second input that maps to the same output.

Additionally, if the hash function is required to be collision-resistant, it must also satisfy the following property:

- It is computationally infeasible to find any two distinct inputs that map to the same output.

Hash Result - The string of bits that is the output of a hash function.

Integrated Circuit(s) - Electronic component(s) designed to perform processing and/or memory functions.

Integrated Circuit(s) Card - A card into which one or more integrated circuits are inserted to perform processing and memory functions.

Interface Device - That part of a terminal into which the ICC is inserted, including such mechanical and electrical devices that may be considered part of it.

Key - A sequence of symbols that controls the operation of a cryptographic transformation.

Key Expiry Date - The date after which a signature made with a particular key is no longer valid. Issuer certificates signed by the key must expire on or before this date. Keys may be removed from terminals after this date has passed.

Key Introduction - The process of generating, distributing, and beginning use of a key pair.

Key Life Cycle - All phases of key management, from planning and generation, through revocation, destruction, and archiving.

Key Replacement - The simultaneous revocation of a key and introduction of a key to replaced the revoked one.

Key Revocation - The key management process of withdrawing a key from service and dealing with the legacy of its use. Key revocation can be as scheduled or accelerated.

Key Revocation Date - The date after which no legitimate cards still in use should contain certificates signed by this key, and therefore the date after which this key can be deleted from terminals. For a planned revocation the Key Revocation Date is the same as the key expiry date.

Key Withdrawal - The process of removing a key from service as part of its revocation.

Logical Compromise - The compromise of a key through application of improved cryptanalytic techniques, increases in computing power, or combination of the two.

Message - A string of bytes sent by the terminal to the card or vice versa, excluding transmission-control characters.

Message Authentication Code - A symmetric cryptographic transformation of data that protects the sender and the recipient of the data against forgery by third parties.

Padding - Appending extra bits to either side of a data string.

Payment System - For the purposes of this specification, Europay International S.A., MasterCard International Incorporated, or Visa International Service Association.

PIN Pad - Arrangement of numeric and command keys to be used for personal identification number (PIN) entry.

Plaintext - Unenciphered information.

Physical Compromise - The compromise of a key resulting from the fact that it has not been securely guarded, or a hardware security module has been stolen or accessed by unauthorised persons.

Planned Revocation - A key revocation performed as scheduled by the published key expiry date.

Potential Compromise - A condition where cryptanalytic techniques and/or computing power has advanced to the point that compromise of a key of a certain length is feasible or even likely.

Private Key - That key of an entity's asymmetric key pair that should only be used by that entity. In the case of a digital signature scheme, the private key defines the signature function.

Public Key - That key of an entity's asymmetric key pair that can be made public. In the case of a digital signature scheme, the public key defines the verification function.

Public Key Certificate - The public key information of an entity signed by the certification authority and thereby rendered unforgeable.

Redundancy - Any information that is known and can be checked.

Response - A message returned by the ICC to the terminal after the processing of a command message received by the ICC.

Secret Key - A key used with symmetric cryptographic techniques and usable only by a set of specified entities.

Symmetric Cryptographic Technique - A cryptographic technique that uses the same secret key for both the originator's and recipient's transformation. Without knowledge of the secret key, it is computationally infeasible to compute either the originator's or the recipient's transformation.

Terminal - The device used in conjunction with the ICC at the point of transaction to perform a financial transaction. It incorporates the interface device and may also include other components and interfaces such as host communications.

4. Abbreviations and Notations

The following abbreviations and notations are used in this specification.

AAC	Application Authentication Cryptogram
AC	Application Cryptogram
AFL	Application File Locator
AID	Application Identifier
APDU	Application Protocol Data Unit
ARC	Authorisation Response Code
ARPC	Authorisation Response Cryptogram
ARQC	Authorisation Request Cryptogram
ATC	Application Transaction Counter
ATM	Automatic Teller Machine
b	Binary
CBC	Cipher Block Chaining
CDOL	Card Risk Management Data Object List
CLA	Class Byte of the Command Message
cn	Compressed Numeric
DDA	Dynamic Data Authentication
DDOL	Dynamic Data Authentication Data Object List
DES	Data Encryption Standard
ECB	Electronic Code Book
FIPS	Federal Information Processing Standard
hex.	Hexadecimal
IC	Integrated Circuit

ICC	Integrated Circuit Card
IEC	International Electrotechnical Commission
IFD	Interface Device
INS	Instruction Byte of Command Message
K_M	Master Key
K_S	Session Key
L_{DD}	Length of the ICC Dynamic Data
MAC	Message Authentication Code
MMYY	Month, Year
N	Numeric
N_{CA}	Length of the Certification Authority Public Key Modulus
N_I	Length of the Issuer Public Key Modulus
N_{IC}	Length of the ICC Public Key Modulus
N_{PE}	Length of the ICC PIN Encipherment Public Key Modulus
P1	Parameter 1
P2	Parameter 2
PAN	Primary Account Number
P_{CA}	Certification Authority Public Key
P_I	Issuer Public Key
P_{IC}	ICC Public Key
PIN	Personal Identification Number
P_{PE}	ICC PIN Encipherment Public Key
RID	Registered Application Provider Identifier
RSA	Rivest, Shamir, Adleman Algorithm
S_{CA}	Certification Authority Private Key

SDA	Static Data Authentication
S_I	Issuer Private Key
S_{IC}	ICC Private Key
SHA	Secure Hash Algorithm
S_{PE}	ICC PIN Encipherment Private Key
TC	Transaction Certificate
TLV	Tag, Length, Value
var.	Variable

The following notations apply:

'0' to '9' and 'A' to 'F'	16 hexadecimal digits
A := B	A is assigned the value of B
A = B	Value of A is equal to the value of B
A ≡ B mod n	Integers A and B are congruent modulo the integer n, that is, there exists an integer d such that $(A - B) = dn$
A mod n	The reduction of the integer A modulo the integer n, that is, the unique integer r, $0 \leq r < n$, for which there exists an integer d such that $A = dn + r$
A / n	The integer division of A by n, that is, the unique integer d for which there exists an integer r, $0 \leq r < n$, such that $A = dn + r$
b-ary representation $(x_0, x_1, \ldots, x_{n-1})$ of X	For a positive integer b, the representation of a nonnegative integer X in the base b: $X = x_0 b^{n-1} + x_1 b^{n-2} + \ldots + a_{n-2} b + a_{n-1}$ for the unique integers $n \bullet 0$ and $0 \bullet a_i < b$, $0 \bullet i < n$

$Y := ALG(K)[X]$	Encipherment of a 64-bit data block X with a 64-bit block cipher as specified in Annex A1 using a secret key K
$X = ALG^{-1}(K)[Y]$	Decipherment of a 64-bit data block Y with a 64-bit block cipher as specified in Annex A1 using a secret key K
$Y := Sign\ (S_K)[X]$	The signing of a data block X with an asymmetric reversible algorithm as specified in Annex A2, using the private key S_K
$X = Recover(P_K)[Y]$	The recovery of the data block X with an asymmetric reversible algorithm as specified in Annex A2, using the public key P_K
$C := (A \mid\mid B)$	The concatenation of an n-bit number A and an m-bit number B, which is defined as $C = 2^m A + B$.
$H := Hash[MSG]$	Hashing of a message MSG of arbitrary length using a 160-bit hash function

The following terminology is used:

proprietary	Not defined in and/or outside the scope of this specification
shall	Denotes a mandatory requirement
should	Denotes a recommendation

5. Static Data Authentication

Static data authentication is performed by the terminal using a digital signature scheme based on public key techniques to confirm the legitimacy of critical ICC-resident static data identified by the AFL and by the optional Static Data Authentication Tag List. This detects unauthorised alteration of data after personalisation.

Static data authentication requires the existence of a certification authority, which is a highly secure cryptographic facility that 'signs' the issuer's public keys. Every terminal conforming to this specification shall contain the appropriate certification authority's public key(s) for every application recognised by the terminal. This specification permits multiple AIDs to share the same 'set' of certification authority public keys. The relationship between the data and the cryptographic keys is shown in Figure 1.

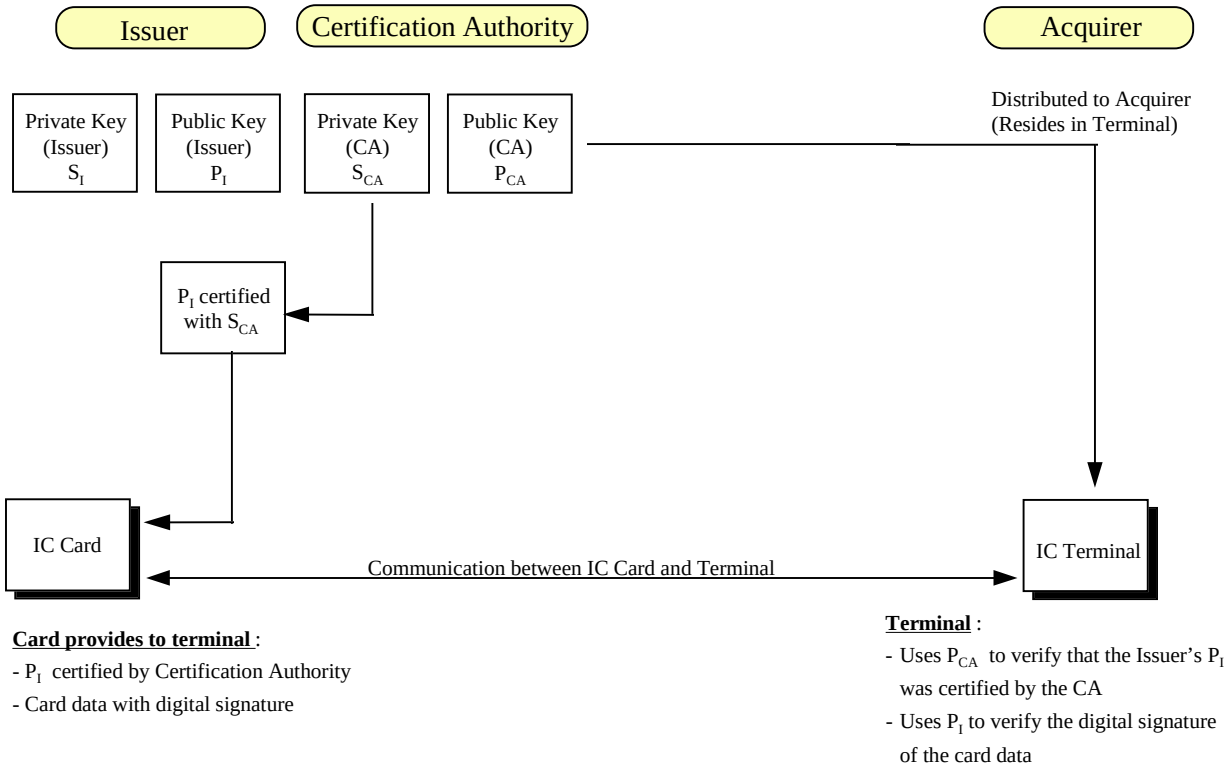


Figure 1 - Diagram of Static Data Authentication

ICCs that support static data authentication shall contain the following data elements:

- Certification Authority Public Key Index: This one-byte data element contains a binary number that indicates which of the application's certification authority

public keys and its associated algorithm that reside in the terminal is to be used with this ICC.

- Issuer Public Key Certificate: This variable-length data element is provided by the appropriate certification authority to the card issuer. When the terminal verifies this data element, it authenticates the Issuer Public Key plus additional data as described in section 5.3.

- Signed Static Application Data: A variable-length data element generated by the issuer using the private key that corresponds to the public key authenticated in the Issuer Public Key Certificate. It is a digital signature covering critical ICC-resident static data elements, as described in section 5.4.

- Issuer Public Key Remainder: A variable length data element. Its presence in the ICC is optional. See section 5.1 for further explanation.

- Issuer Public Key Exponent: A variable length data element provided by the issuer. See section 5.1 for further explanation.

To support static data authentication, each terminal shall be able to store six certification authority public keys per Registered Application Provider Identifier (RID) and shall associate with each such key the key-related information to be used with the key (so that terminals can in the future support multiple algorithms and allow an evolutionary transition from one to another, see section 11.2.2). The terminal shall be able to locate any such key (and the key-related information) given the RID and Certification Authority Public Key Index as provided by the ICC.

Static data authentication shall use a reversible algorithm as specified in Annex A2.1 and Annex B2. Section 5.1 contains an overview of the keys and certificates involved in the static data authentication process, and sections 5.2 to 5.4 specify the three main steps in the process, namely

- Retrieval of the Certification Authority Public Key by the terminal.

- Retrieval of the Issuer Public Key by the terminal.

- Verification of the Signed Static Application Data by the terminal.

5.1 Keys and Certificates

To support static data authentication, an ICC shall contain the Signed Static Application Data, which is signed with the Issuer Private Key. The Issuer Public Key shall be stored on the ICC with a public key certificate.

The bit length of all moduli shall be a multiple of 8, the leftmost bit of its leftmost byte being 1. All lengths are given in bytes.

The signature scheme specified in Annex A2.1 is applied to the data specified in Table 1 using the Certification Authority Private Key S_{CA} in order to obtain the Issuer Public Key Certificate.

The public key pair of the certification authority has a public key modulus of N_{CA} bytes, where $N_{CA} \leq 248$. The Certification Authority Public Key Exponent shall be equal to 3 or $2^{16} + 1$.

The signature scheme specified in Annex A2.1 is applied to the data specified in Table 2 using the Issuer Private Key S_I in order to obtain the Signed Static Application Data.

The public key pair of the issuer has an Issuer Public Key Modulus of N_I bytes, where $N_I \leq N_{CA} \leq 248$. If $N_I > (N_{CA} - 36)$, the Issuer Public Key Modulus is split into two parts, namely one part consisting of the $N_{CA} - 36$ most significant bytes of the modulus (the Leftmost Digits of the Issuer Public Key) and a second part consisting of the remaining $N_I - (N_{CA} - 36)$ least significant bytes of the modulus (the Issuer Public Key Remainder). The Issuer Public Key Exponent shall be equal to 3 or $2^{16} + 1$.

All the information necessary for static data authentication is specified in Table 3 and stored in the ICC. With the exception of the RID, which can be obtained from the AID, this information may be retrieved with the READ RECORD command. If any of this data is missing, static data authentication has failed.

Field Name	Length	Description	Format
Certificate Format	1	Hex. value '02'	b
Issuer Identification Number	4	Leftmost 3-8 digits from the Primary Account Number (PAN) (padded to the right with hex. 'F's)	cn 8
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the certification authority	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Issuer Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the Issuer Public Key[1]	b
Issuer Public Key Length	1	Identifies the length of the Issuer Public Key Modulus in bytes	b
Issuer Public Key Exponent Length	1	Identifies the length of the Issuer Public Key Exponent in bytes	b
Issuer Public Key or Leftmost Digits of the Issuer Public Key	N_{CA} – 36	If $N_I \leq N_{CA} - 36$, this field consists of the full Issuer Public Key padded to the right with $N_{CA} - 36 - N_I$ bytes of value 'BB' If $N_I > N_{CA} - 36$, this field consists of the $N_{CA} - 36$ most significant bytes of the Issuer Public Key[2]	b
Issuer Public Key Remainder	0 or $N_I - N_{CA} + 36$	This field is only present if $N_I > N_{CA} - 36$ and consists of the $N_I - N_{CA} + 36$ least significant bytes of the Issuer Public Key.	b
Issuer Public Key Exponent	1 or 3	Issuer Public Key Exponent equal to 3 or $2^{16} + 1$	b

Table 1 - Issuer Public Key Data to be Signed by the Certification Authority (i.e., input to the hash algorithm)

[1] See Annex B for specific values assigned to approved algorithms.

[2] As can be seen in Annex A2.1, $N_{CA} - 22$ bytes of the data signed are retrieved from the signature. Since the length of the first through the eighth data elements in Table 1 is 14 bytes, there are $N_{CA} - 22 - 14 = N_{CA} - 36$ bytes remaining in the signature to store the Issuer Public Key Modulus.

Field Name	Length	Description	Format
Signed Data Format	1	Hex. Value '03'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Data Authentication Code	2	Issuer-assigned code	b
Pad Pattern	$N_I - 26$	Pad pattern consisting of $N_I - 26$ bytes of value 'BB'[3]	b
Static Data to be Authenticated	var.	Static data to be authenticated as specified in Part II of Book 3 of these specifications (see also below)	-

Table 2 - Static Application Data to be Signed by the Issuer (i.e., input to the hash algorithm)

Input to the authentication process is formed from the records identified by the AFL, followed by the value of the AIP, if identified by the optional Static Data Authentication Tag List (tag '9F4A'). If present, the Static Data Authentication Tag List shall only contain the tag '82' identifying the AIP.

Tag	Length	Value	Format
-	5	Registered Application Provider Identifier (RID)	b
'8F'	1	Certification Authority Public Key Index	b
'90'	N_{CA}	Issuer Public Key Certificate	b
'92'	$N_I - N_{CA} + 36$	Issuer Public Key Remainder, if present	b
'9F32'	1 or 3	Issuer Public Key Exponent	b
'93'	N_I	Signed Static Application Data	b
-	Var.	Static data to be authenticated as specified in Part II of Book 3 of these specifications (see also above)	-

Table 3 - Data Objects Required for Static Data Authentication

5.2 Retrieval of the Certification Authority Public Key

The terminal reads the Certification Authority Public Key Index. Using this index and the RID, the terminal shall identify and retrieve the terminal-stored Certification Authority Public Key Modulus and Exponent and the associated key-related information, and the corresponding algorithm to be used. If the terminal

[3] As can be seen in Annex A2.1, $N_I - 22$ bytes of the data signed are retrieved from the signature. Since the first through the third data elements in Table 2 total 4 bytes, there are $N_I - 22 - 4 = N_I - 26$ bytes left for the data to be stored in the signature.

does not have the key stored associated with this index and RID, static data authentication has failed.

5.3 Retrieval of the Issuer Public Key

1. If the Issuer Public Key Certificate has a length different from the length of the Certification Authority Public Key Modulus obtained in the previous section, static data authentication has failed.

2. In order to obtain the recovered data specified in Table 4, apply the recovery function specified in Annex A2.1 to the Issuer Public Key Certificate using the Certification Authority Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', static data authentication has failed.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. Value '6A'	b
Certificate Format	1	Hex. Value '02'	b
Issuer Identification Number	4	Leftmost 3-8 digits from the PAN (padded to the right with hex. 'F's)	cn 8
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the certification authority	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Issuer Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the Issuer Public Key[1]	b
Issuer Public Key Length	1	Identifies the length of the Issuer Public Key Modulus in bytes	b
Issuer Public Key Exponent Length	1	Identifies the length of the Issuer Public Key Exponent in bytes	b
Issuer Public Key or Leftmost Digits of the Issuer Public Key	N_{CA} –36	If $N_I \leq N_{CA} - 36$, this field consists of the full Issuer Public Key padded to the right with $N_{CA} - 36 - N_I$ bytes of value 'BB' If $N_I > N_{CA} - 36$, this field consists of the $N_{CA} - 36$ most significant bytes of the Issuer Public Key[2]	b
Hash Result	20	Hash of the Issuer Public Key and its related information	b
Recovered Data Trailer	1	Hex. value 'BC'	b

Table 4 - Format of the Data Recovered from the Issuer Public Key Certificate

3. Check the Recovered Data Header. If it is not '6A', static data authentication has failed.

4. Check the Certificate Format. If it is not '02', static data authentication has failed.

5. Concatenate from left to right the second to the tenth data elements in Table 4 (that is, Certificate Format through Issuer Public Key or Leftmost Digits of the Issuer Public Key), followed by the Issuer Public Key Remainder (if present) and finally the Issuer Public Key Exponent.

6. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

7. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, static data authentication has failed.

8. Verify that the Issuer Identification Number matches the leftmost 3-8 PAN digits (allowing for the possible padding of the Issuer Identification Number with hexadecimal 'F's). If not, static data authentication has failed.

9. Verify that the last day of the month specified in the Certificate Expiration Date is equal to or later than today's date. If the Certificate Expiration Date is earlier than today's date, the certificate has expired, in which case static data authentication has failed.

10. Verify that the concatenation of RID, Certification Authority Public Key Index, and Certificate Serial Number is valid. If not, static data authentication has failed[4].

11. If the Issuer Public Key Algorithm Indicator is not recognised, static data authentication has failed.

12. If all the checks above are correct, concatenate the Leftmost Digits of the Issuer Public Key and the Issuer Public Key Remainder (if present) to obtain the Issuer Public Key Modulus, and continue with the next steps for the verification of the Signed Static Application Data.

5.4 Verification of the Signed Static Application Data

1. If the Signed Static Application Data has a length different from the length of the Issuer Public Key Modulus, static data authentication has failed.

2. In order to obtain the Recovered Data specified in Table 5, apply the recovery function specified in Annex A2.1 on the Signed Static Application Data using the Issuer Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', static data authentication has failed.

[4] This step is optional and is to allow the revocation of the Issuer Public Key Certificate against a list that may be kept by the terminal.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. value '6A'	b
Signed Data Format	1	Hex. value '03'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Data Authentication Code	2	Issuer-assigned code	b
Pad Pattern	$N_I - 26$	Pad pattern consisting of $N_I - 26$ bytes of value 'BB'[3]	b
Hash Result	20	Hash of the Static Application Data to be authenticated	b
Recovered Data Trailer	1	Hex. Value 'BC'	b

Table 5 - Format of the Data Recovered from the Signed Static Application Data

3. Check the Recovered Data Header. If it is not '6A', static data authentication has failed.

4. Check the Signed Data Format. If it is not '03', static data authentication has failed.

5. Concatenate from left to right the second to the fifth data elements in Table 5 (that is, Signed Data Format through Pad Pattern), followed by the static data to be authenticated as specified in Part II of Book 3 of these specifications. If the Static Data Authentication Tag List is present and contains tags other than '82', then static data authentication has failed.

6. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

7. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, static data authentication has failed.

If all of the above steps were executed successfully, static data authentication was successful. The Data Authentication Code recovered in Table 5 shall be stored in Tag '9F45'.

6. Dynamic Data Authentication

Dynamic data authentication is performed by the terminal using a digital signature scheme based on public key techniques to authenticate the ICC, and confirm the legitimacy of critical ICC-resident/generated data and data received from the terminal. This precludes the counterfeiting of any such card.

The following two options exist.

- Standard dynamic data authentication executed before card action analysis, where the ICC generates a digital signature on ICC-resident/generated data identified by the ICC Dynamic Data and data received from the terminal identified by the Dynamic Data Authentication Data Object List (DDOL).

- Combined dynamic data authentication and Application Cryptogram generation executed at issuance of the first GENERATE AC command. In the case of a TC or ARQC, the ICC generates a digital signature on ICC-resident/generated data identified by the ICC Dynamic Data, which contains the TC or ARQC, and an Unpredictable Number generated by the terminal and identified by the Card Risk Management Data Object List 1 (CDOL1).

The AIP denotes the options supported by the ICC.

Dynamic data authentication requires the existence of a certification authority, a highly secure cryptographic facility that 'signs' the Issuer's Public Keys. Every terminal conforming to this specification shall contain the appropriate certification authority's public key(s) for every application recognised by the terminal. This specification permits multiple AIDs to share the same 'set' of certification authority public keys. The relationship between the data and the cryptographic keys is shown in Figure 2.

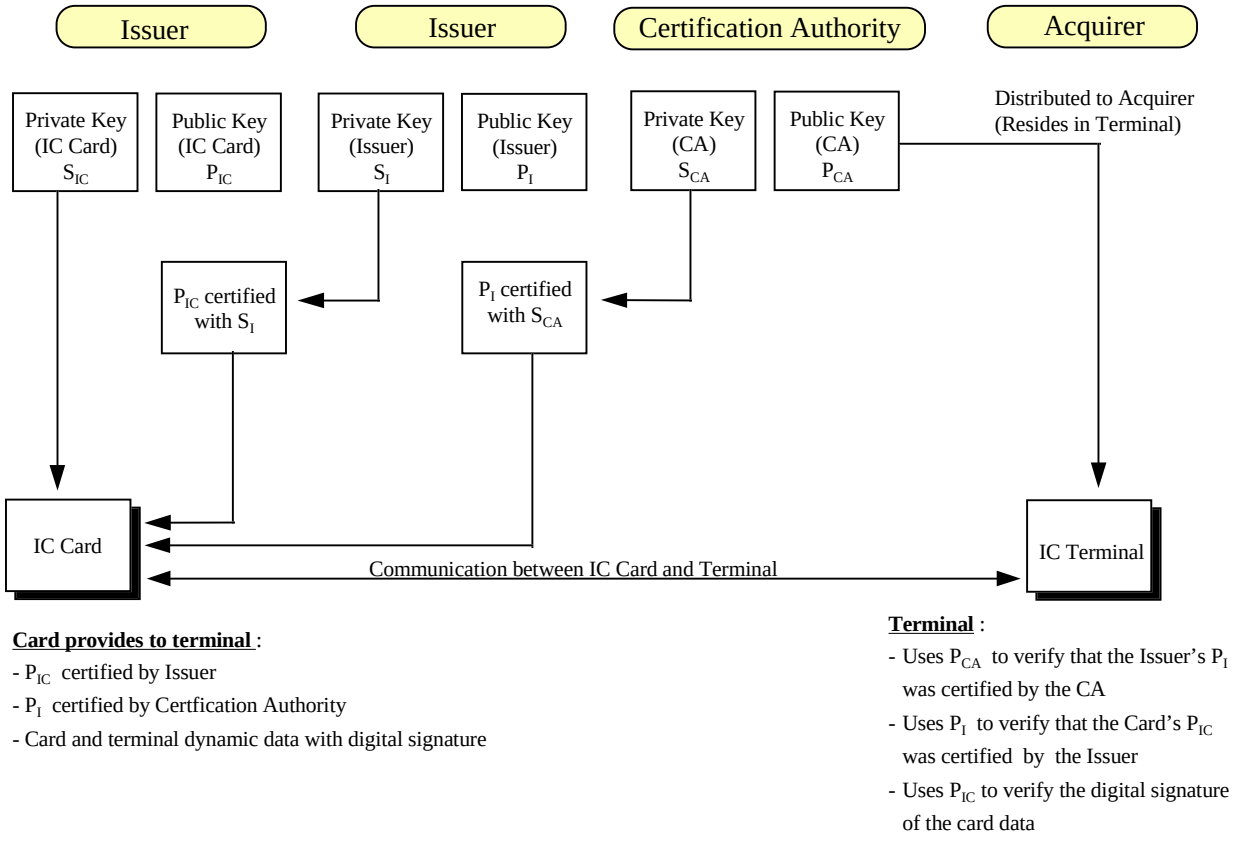


Figure 2 - Diagram of Dynamic Data Authentication

ICCs that support dynamic data authentication shall contain the following data elements:

- Certification Authority Public Key Index: This one-byte data element contains a binary number that indicates which of the application's certification authority public keys and its associated algorithm that reside in the terminal is to be used with this ICC.

- Issuer Public Key Certificate: This variable-length data element is provided by the appropriate certification authority to the card issuer. When the terminal verifies this data element, it authenticates the Issuer Public Key plus additional data as described in section 6.3.

- ICC Public Key Certificate: This variable-length data element is provided by the issuer to the ICC. When the terminal verifies this data element, it authenticates the ICC Public Key plus additional data as described in section 6.4.

- Issuer Public Key Remainder: A variable-length data element. See section 6.1 for further explanation.

- Issuer Public Key Exponent: A variable-length data element provided by the issuer. See section 6.1 for further explanation.

- ICC Public Key Remainder: A variable-length data element. See section 6.1 for further explanation.

- ICC Public Key Exponent: A variable-length data element provided by the issuer. See section 6.1 for further explanation.

- ICC Private Key: An ICC internal variable-length data element used to generate the Signed Dynamic Application Data as described in sections 6.5 and 6.6.

ICCs that support dynamic data authentication shall generate the following data element:

- Signed Dynamic Application Data: A variable-length data element generated by the ICC using the private key that corresponds to the public key authenticated in the ICC Public Key Certificate. It is a digital signature covering critical ICC-resident/generated and terminal data elements, as described in sections 6.5 and 6.6.

To support dynamic data authentication, each terminal shall be able to store six certification authority public keys per Registered Application Provider Identifier (RID) and shall associate with each such key the key-related information to be used with the key (so that terminals can in the future support multiple algorithms and allow an evolutionary transition from one to another, see section 11.2.2). The terminal shall be able to locate any such key (and key-related information) given the RID and Certification Authority Public Key Index as provided by the ICC.

Dynamic data authentication shall use a reversible algorithm as specified in Annex A2.1 and Annex B2. Section 6.1 contains an overview of the keys and certificates involved in the dynamic data authentication process. Sections 6.2 to 6.4 specify the initial steps in the process, namely

- Retrieval of the Certification Authority Public Key by the terminal.

- Retrieval of the Issuer Public Key by the terminal.

- Retrieval of the ICC Public Key by the terminal.

Finally, sections 6.5 and 6.6 specify the dynamic signature generation and verification processes for both options.

6.1 Keys and Certificates

To support dynamic data authentication, an ICC shall own its own unique public key pair consisting of a private signature key and the corresponding public verification key. The ICC Public Key shall be stored on the ICC in a public key certificate.

More precisely, a three-layer public key certification scheme is used. Each ICC Public Key is certified by its issuer, and the certification authority certifies the Issuer Public Key. This implies that, for the verification of an ICC signature, the terminal first needs to verify two certificates in order to retrieve and authenticate the ICC Public Key, which is then employed to verify the ICC's dynamic signature.

The bit length of all moduli shall be a multiple of 8, the leftmost bit of its leftmost byte being 1. All lengths are given in bytes.

The signature scheme specified in Annex A2.1 is applied on the data in Table 6 and on the data in Table 7 using the Certification Authority Private Key S_{CA} and the Issuer Private Key S_I in order to obtain the Issuer Public Key Certificate and ICC Public Key Certificate, respectively.

The public key pair of the certification authority has a Certification Authority Public Key Modulus of N_{CA} bytes, where $N_{CA} \leq 248$. The Certification Authority Public Key Exponent shall be equal to 3 or $2^{16} + 1$.

The public key pair of the issuer has a Public Key Modulus of N_I bytes, where $N_I \leq N_{CA} \leq 248$. If $N_I > (N_{CA} - 36)$, the Issuer Public Key Modulus is divided into two parts, one part consisting of the $N_{CA} - 36$ most significant bytes of the modulus (the Leftmost Digits of the Issuer Public Key) and a second part consisting of the remaining $N_I - (N_{CA} - 36)$ least significant bytes of the modulus (the Issuer Public Key Remainder). The Issuer Public Key Exponent shall be equal to 3 or $2^{16} + 1$.

The public key pair of the ICC has an ICC Public Key Modulus of N_{IC} bytes, where $N_{IC} \leq N_I \leq N_{CA} \leq 248$. If $N_{IC} > (N_I - 42)$, the ICC Public Key Modulus is divided into two parts, one part consisting of the $N_I - 42$ most significant bytes of the modulus (the Leftmost Digits of the ICC Public Key) and a second part consisting of the remaining $N_{IC} - (N_I - 42)$ least significant bytes of the modulus (the ICC Public Key Remainder). The ICC Public Key Exponent shall be equal to 3 or $2^{16} + 1$.

To execute dynamic data authentication, the terminal shall first retrieve and authenticate the ICC Public Key (this process is called ICC Public Key authentication). All the information necessary for ICC Public Key authentication is specified in Table 8 and stored in the ICC. With the exception of the RID, which can be obtained from the AID, this information may be retrieved with the READ RECORD command. If any of this data is missing, dynamic data authentication has failed.

Field Name	Length	Description	Format
Certificate Format	1	Hex. value '02'	b
Issuer Identification Number	4	Leftmost 3-8 digits from the PAN (padded to the right with hex. 'F's)	cn 8
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the certification authority	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Issuer Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the Issuer Public Key [1]	b
Issuer Public Key Length	1	Identifies the length of the Issuer Public Key Modulus in bytes	b
Issuer Public Key Exponent Length	1	Identifies the length of the Issuer Public Key Exponent in bytes	b
Issuer Public Key or Leftmost Digits of the Issuer Public Key	N_{CA} – 36	If $N_I \leq N_{CA} - 36$, this field consists of the full Issuer Public Key padded to the right with $N_{CA} - 36 - N_I$ bytes of value 'BB' If $N_I > N_{CA} - 36$, this field consists of the $N_{CA} - 36$ most significant bytes of the Issuer Public Key [2]	b
Issuer Public Key Remainder	0 or $N_I - N_{CA} + 36$	This field is only present if $N_I > N_{CA} - 36$ and consists of the $N_I - N_{CA} + 36$ least significant bytes of the Issuer Public Key	b
Issuer Public Key Exponent	1 or 3	Issuer Public Key Exponent equal to 3 or $2^{16} + 1$	b

Table 6 - Issuer Public Key Data to be Signed by the Certification Authority (i.e., input to the hash algorithm)

Field Name	Length	Description	Format
Certificate Format	1	Hex. value '04'	b
Application PAN	10	PAN (padded to the right with hex. 'F's)	cn 20
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the issuer	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
ICC Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the ICC Public Key[1]	b
ICC Public Key Length	1	Identifies the length of the ICC Public Key Modulus in bytes	b
ICC Public Key Exponent Length	1	Identifies the length of the ICC Public Key Exponent in bytes	b
ICC Public Key or Leftmost Digits of the ICC Public Key	$N_I - 42$	If $N_{IC} \leq N_I - 42$, this field consists of the full ICC Public Key padded to the right with $N_I - 42 - N_{IC}$ bytes of value 'BB' If $N_{IC} > N_I - 42$, this field consists of the $N_I - 42$ most significant bytes of the ICC Public Key [5]	b
ICC Public Key Remainder	0 or $N_{IC} - N_I + 42$	This field is only present if $N_{IC} > N_I - 42$ and consists of the $N_{IC} - N_I + 42$ least significant bytes of the ICC Public Key	b
ICC Public Key Exponent	1 or 3	ICC Public Key Exponent equal to 3 or $2^{16} + 1$	b
Static Data to be Authenticated	Var.	Static data to be authenticated as specified in Part II of Book 3 of these specifications (see also below)	b

Table 7 - ICC Public Key Data to be Signed by the Issuer (i.e., input to the hash algorithm)

Input to the authentication process is formed from the records identified by the AFL, followed by the value of the AIP, if identified by the optional Static Data Authentication Tag List (tag '9F4A'). If present, the Static Data Authentication Tag List shall only contain the tag '82' identifying the AIP.

[5] As can be seen in Annex A2.1, $N_I - 22$ bytes of the data signed are retrieved from the signature. Since the first through the eighth data elements in Table 7 total 20 bytes, there are $N_I - 22 - 20 = N_I - 42$ bytes left for the data to be stored in the signature.

Tag	Length	Value	Format
-	5	Registered Application Provider Identifier (RID)	b
'8F'	1	Certification Authority Public Key Index	b
'90'	N_{CA}	Issuer Public Key Certificate	b
'92'	$N_I - N_{CA} + 36$	Issuer Public Key Remainder, if present	b
'9F32'	1 or 3	Issuer Public Key Exponent	b
'9F46'	N_I	ICC Public Key Certificate	b
'9F48'	$N_{IC} - N_I + 42$	ICC Public Key Remainder, if present	b
'9F47'	1 or 3	ICC Public Key Exponent	b
-	Var.	Static data to be authenticated as specified in Part II of Book 3 of these specifications (see also above)	-

Table 8 - Data Objects Required for Public Key Authentication for Dynamic Authentication

6.2 Retrieval of the Certification Authority Public Key

The terminal reads the Certification Authority Public Key Index. Using this index and the RID, the terminal can identify and retrieve the terminal-stored Certification Authority Public Key Modulus and Exponent and the associated key-related information, and the corresponding algorithm to be used. If the terminal does not have the key stored associated with this index and RID, dynamic data authentication has failed.

6.3 Retrieval of the Issuer Public Key

1. If the Issuer Public Key Certificate has a length different from the length of the Certification Authority Public Key Modulus obtained in the previous section, dynamic data authentication has failed.

2. In order to obtain the recovered data specified in Table 9, apply the recovery function specified in Annex A2.1 on the Issuer Public Key Certificate using the Certification Authority Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', dynamic data authentication has failed.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. value '6A'	b
Certificate Format	1	Hex. value '02'	b
Issuer Identification Number	4	Leftmost 3-8 digits from the PAN (padded to the right with hex. 'F's)	cn 8
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the certification authority	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
Issuer Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the Issuer Public Key[1]	b
Issuer Public Key Length	1	Identifies the length of the Issuer Public Key Modulus in bytes	b
Issuer Public Key Exponent Length	1	Identifies the length of the Issuer Public Key Exponent in bytes	b
Issuer Public Key or Leftmost Digits of the Issuer Public Key	N_{CA} – 36	If $N_I \leq N_{CA} - 36$, this field consists of the full Issuer Public Key padded to the right with $N_{CA} - 36 - N_I$ bytes of value 'BB' If $N_I > N_{CA} - 36$, this field consists of the $N_{CA} - 36$ most significant bytes of the Issuer Public Key[2]	b
Hash Result	20	Hash of the Issuer Public Key and its related information	b
Recovered Data Trailer	1	Hex. value 'BC'	b

Table 9 - Format of the Data Recovered from the Issuer Public Key Certificate

3. Check the Recovered Data Header. If it is not '6A', dynamic data authentication has failed.

4. Check the Certificate Format. If it is not '02', dynamic data authentication has failed.

5. Concatenate from left to right the second to the tenth data elements in Table 9 (that is, Certificate Format through Issuer Public Key or Leftmost Digits of the Issuer Public Key), followed by the Issuer Public Key Remainder (if present) and finally the Issuer Public Key Exponent.

6. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

7. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, dynamic data authentication has failed.

8. Verify that the Issuer Identification Number matches the leftmost 3-8 PAN digits (allowing for the possible padding of the Issuer Identification Number with hexadecimal 'F's). If not, dynamic data authentication has failed.

9. Verify that the last day of the month specified in the Certificate Expiration Date is equal to or later than today's date. If the Certificate Expiration Date is earlier than today's date, the certificate has expired, in which case dynamic data authentication has failed.

10. Verify that the concatenation of RID, Certification Public Key Index, and Certificate Serial Number is valid. If not, dynamic data authentication has failed[4].

11. If the Issuer Public Key Algorithm Indicator is not recognised, dynamic data authentication has failed.

12. If all the checks above are correct, concatenate the Leftmost Digits of the Issuer Public Key and the Issuer Public Key Remainder (if present) to obtain the Issuer Public Key Modulus, and continue with the next steps for the retrieval of the ICC Public Key.

6.4 Retrieval of the ICC Public Key

1. If the ICC Public Key Certificate has a length different from the length of the Issuer Public Key Modulus obtained in the previous section, dynamic data authentication has failed.

2. In order to obtain the recovered data specified in Table 10, apply the recovery function specified in Annex A2.1 on the ICC Public Key Certificate using the Issuer Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', dynamic data authentication has failed.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. Value '6A'	b
Certificate Format	1	Hex. Value '04'	b
Application PAN	10	PAN (padded to the right with hex. 'F's	cn 20
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the issuer	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
ICC Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the ICC Public Key[1]	b
ICC Public Key Length	1	Identifies the length of the ICC Public Key Modulus in bytes	b
ICC Public Key Exponent Length	1	Identifies the length of the ICC Public Key Exponent in bytes	b
ICC Public Key or Leftmost Digits of the ICC Public Key	$N_I - 42$	If $N_{IC} \leq N_I - 42$, this field consists of the full ICC Public Key padded to the right with $N_I - 42 - N_{IC}$ bytes of value 'BB'[5] If $N_{IC} > N_I - 42$, this field consists of the $N_I - 42$ most significant bytes of the ICC Public Key	b
Hash Result	20	Hash of the ICC Public Key and its related information	b
Recovered Data Trailer	1	Hex. Value 'BC'	b

Table 10 - Format of the Data Recovered from the ICC Public Key Certificate

3. Check the Recovered Data Header. If it is not '6A', dynamic data authentication has failed.

4. Check the Certificate Format. If it is not '04', dynamic data authentication has failed.

5. Concatenate from left to right the second to the tenth data elements in Table 10 (that is, Certificate Format through ICC Public Key or Leftmost Digits of the ICC Public Key), followed by the ICC Public Key Remainder (if present), the ICC Public Key Exponent and finally the static data to be authenticated specified in Part II of Book 3 of these specifications. If the Static Data Authentication Tag

List is present and contains tags other than '82', then dynamic data authentication has failed.

6. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

7. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, dynamic data authentication has failed.

8. Check if the recovered PAN is equal to the Application PAN, read from the ICC. If not, dynamic data authentication has failed.

9. Verify that the last day of the month specified in the Certificate Expiration Date is equal to or later than today's date. If not, dynamic data authentication has failed.

10. If the ICC Public Key Algorithm Indicator is not recognised, dynamic data authentication has failed.

11. If all the checks above are correct, concatenate the Leftmost Digits of the ICC Public Key and the ICC Public Key Remainder (if present) to obtain the ICC Public Key Modulus, and continue with the actual dynamic data authentication described in the two sections below.

6.5 Standard Dynamic Data Authentication

6.5.1 Dynamic Signature Generation

In this section it is assumed that the terminal has successfully retrieved the ICC Public Key as described above. The generation of the dynamic signature takes place in the following steps.

1. The terminal issues an INTERNAL AUTHENTICATE command including the concatenation of the data elements specified by the DDOL according to the rules specified in Part I of Book 3 of these specifications.

   The ICC may contain the DDOL, but there shall be a default DDOL in the terminal, specified by the payment system, for use in case the DDOL is not present in the ICC.

   It is mandatory that the DDOL contains the Unpredictable Number generated by the terminal (tag '9F37', 4 bytes binary).

   If any of the following cases occur, dynamic data authentication has failed.

   - Both the ICC and the terminal do not contain a DDOL.
   - The DDOL in the ICC does not include the Unpredictable Number.

- The ICC does not contain a DDOL and the default DDOL in the terminal does not include the Unpredictable Number.

2. The ICC generates a digital signature as described in Annex A2.1 on the data specified in Table 11 using its ICC Private Key S_{IC} in conjunction with the corresponding algorithm. The result is called the Signed Dynamic Application Data.

Field Name	Length	Description	Format
Signed Data Format	1	Hex. value '05'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result[1]	b
ICC Dynamic Data Length	1	Identifies the length L_{DD} of the ICC dynamic data in bytes	b
ICC Dynamic Data	L_{DD}	Dynamic data generated by and/or stored in the ICC	-
Pad Pattern	$N_{IC} - L_{DD} - 25$	($N_{IC} - L_{DD} - 25$) padding bytes of value 'BB'[6]	b
Terminal Dynamic Data	var.	Concatenation of the data elements specified by the DDOL	-

Table 11 - Dynamic Application Data to be Signed (i.e., input to the hash algorithm)

The length L_{DD} of the ICC Dynamic Data satisfies $0 \le L_{DD} \le N_{IC} - 25$. The 3-9 leftmost bytes of the ICC Dynamic Data shall consist of the 1-byte length of the ICC Dynamic Number, followed by the 2-8 byte value of the ICC Dynamic Number (tag '9F4C', 2-8 bytes binary). The ICC Dynamic Number is a time-variant parameter generated by the ICC (it can for example be an unpredictable number or a counter incremented each time the ICC receives an INTERNAL AUTHENTICATE command).

In addition to those specified in Table 8, the data objects necessary for dynamic data authentication are specified in Table 12.

Tag	Length	Value	Format
'9F4B'	N_{IC}	Signed Dynamic Application Data	b
'9F49'	Var.	DDOL	b

Table 12 - Additional Data Objects Required for Dynamic Signature Generation and Verification

[6] As can be seen in Annex A2.1, $N_I - 22$ bytes of the data signed is recovered from the signature. Since the length of the first three data elements in Table 11 is three bytes, there are $N_I - 22 - 3 = N_I - 25$ bytes remaining for the data to be stored in the signature.

6.5.2 Dynamic Signature Verification

1. If the Signed Dynamic Application Data has a length different from the length of the ICC Public Key Modulus, dynamic data authentication has failed.

2. To obtain the recovered data specified in Table 13, apply the recovery function specified in Annex A2.1 on the Signed Dynamic Application Data using the ICC Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', dynamic data authentication has failed.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. value '6A'	b
Signed Data Format	1	Hex. value '05'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
ICC Dynamic Data Length	1	Identifies the length of the ICC dynamic data in bytes	b
ICC Dynamic Data	L_{DD}	Dynamic data generated by and/or stored in the ICC	-
Pad Pattern	$N_{IC} - L_{DD} - 25$	$(N_{IC} - L_{DD} - 25)$ padding bytes of value 'BB'[6]	b
Hash Result	20	Hash of the Dynamic Application Data and its related information	b
Recovered Data Trailer	1	Hex. value 'BC'	b

Table 13 - Format of the Data Recovered from the Signed Dynamic Application Data

3. Check the Recovered Data Header. If it is not '6A', dynamic data authentication has failed.

4. Check the Signed Data Format. If it is not '05', dynamic data authentication has failed.

5. Concatenate from left to right the second to the sixth data elements in Table 13 (that is, Signed Data Format through Pad Pattern), followed by the data elements specified by the DDOL.

6. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

7. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, dynamic data authentication has failed.

If all the above steps were executed successfully, dynamic data authentication was successful. The ICC Dynamic Number contained in the ICC Dynamic Data recovered in Table 13 shall be stored in Tag '9F4C'.

6.6 Combined Dynamic Data Authentication/Application Cryptogram Generation

6.6.1 Dynamic Signature Generation

In this section it is assumed that

- The terminal has successfully retrieved the ICC Public Key as described above.
- Both the ICC and the terminal support combined dynamic data authentication/AC generation and the ICC will respond as such to the first generate AC command according to Book 3 of these specifications.

The generation of the combined dynamic signature and Application Cryptogram takes place in the following steps.

1. The terminal issues a first GENERATE AC command according to Book 3 of these specifications.
   It is mandatory that the Card Risk Management Data Object List 1 (CDOL1) contains the Unpredictable Number generated by the terminal (tag '9F37', 4 bytes binary). If this is not the case, then combined dynamic data authentication/AC generation will have failed.

2. If the ICC is to respond with a TC or ARQC, the ICC generates the TC or ARQC, and a digital signature as described in Annex A2.1 on the data specified in Table 14 using its ICC Private Key S_{IC} in conjunction with the corresponding algorithm. The result is called the Signed Dynamic Application Data.

Field Name	Length	Description	Format
Signed Data Format	1	Hex. Value '05'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result[1]	b
ICC Dynamic Data Length	1	Identifies the length L_{DD} of the ICC dynamic data in bytes	b
ICC Dynamic Data	L_{DD}	Dynamic data generated by and/or stored in the ICC	-
Pad Pattern	$N_{IC} - L_{DD} - 25$	$(N_{IC} - L_{DD} - 25)$ padding bytes of value 'BB'[7]	b
Unpredictable Number	4	Unpredictable Number generated by the terminal	b

Table 14 - Dynamic Application Data to be Signed (i.e., input to the hash algorithm)

The length L_{DD} of the ICC Dynamic Data satisfies $0 \leq L_{DD} \leq N_{IC} - 25$. The 12-18 leftmost bytes of the ICC Dynamic Data shall consist of the concatenation of the data specified in Table 15.

Length	Value	Format
1	ICC Dynamic Number Length	b
2-8	ICC Dynamic Number	b
1	Cryptogram Information Data	b
8	TC or ARQC	b

Table 15 – Contents of the ICC Dynamic Data

The ICC Dynamic Number is a time-variant parameter generated by the ICC (it can for example be an unpredictable number or a counter incremented each time the ICC receives the first GENERATE AC command during a transaction).

The ICC response to the first GENERATE AC command shall be coded according to format 2 as specified in Part I of Book 3 of these specifications (constructed data object with tag '77') and shall contain the data objects (TLV coded in the response) specified in Table 16.

[7] As can be seen in Annex A2.1, $N_I - 22$ bytes of the data signed is recovered from the signature. Since the length of the first three data elements in Table 11 is three bytes, there are $N_I - 22 - 3 = N_I - 25$ bytes remaining for the data to be stored in the signature.

Tag	Length	Value	Presence
'9F27'	1	Cryptogram Information Data	M
'9F36'	2	Application Transaction Counter	M
'9F4B'	N_{IC}	Signed Dynamic Application Data	M
'9F10'	Var. up to 32	Issuer Application Data	O

Table 16 - Additional Data Objects Required for Dynamic Signature Generation and Verification

3. If the ICC responds with an AAC, the ICC response shall be coded according to either format 1 or format 2 as specified in Part I of Book 3 of these specifications and shall contain the data elements specified in Table 17.

Tag	Length	Value	Presence
'9F27'	1	Cryptogram Information Data	M
'9F36'	2	Application Transaction Counter	M
'9F26'	8	Application Authentication Cryptogram	M
'9F10'	Var. up to 32	Issuer Application Data	O

Table 17 - Additional Data Objects Required for Dynamic Signature Generation and Verification

6.6.2 Dynamic Signature Verification

If the ICC has responded with an AAC, then combined dynamic data authentication/AC generation has failed.

If the ICC has responded with a TC or ARQC, the terminal retrieves from the response the data objects specified in Table 16 and executes the following steps.

1. If the Signed Dynamic Application Data has a length different from the length of the ICC Public Key Modulus, combined dynamic data authentication/AC generation has failed.

2. To obtain the recovered data specified in Table 18, apply the recovery function specified in Annex A2.1 on the Signed Dynamic Application Data using the ICC Public Key in conjunction with the corresponding algorithm. If the Recovered Data Trailer is not equal to 'BC', combined dynamic data authentication/AC generation has failed.

Field Name	Length	Description	Format
Recovered Data Header	1	Hex. Value '6A'	b
Signed Data Format	1	Hex. Value '05'	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
ICC Dynamic Data Length	1	Identifies the length of the ICC dynamic data in bytes	b
ICC Dynamic Data	L_{DD}	Dynamic data generated by and/or stored in the ICC	-
Pad Pattern	$N_{IC} - L_{DD} - 25$	$(N_{IC} - L_{DD} - 25)$ padding bytes of value 'BB'[6]	b
Hash Result	20	Hash of the Dynamic Application Data and its related information	b
Recovered Data Trailer	1	Hex. Value 'BC'	b

Table 18 - Format of the Data Recovered from the Signed Dynamic Application Data

3. Check the Recovered Data Header. If it is not '6A', combined dynamic data authentication/AC generation has failed.

4. Check the Signed Data Format. If it is not '05', combined dynamic data authentication/AC generation has failed.

5. Retrieve from the ICC Dynamic Data the data specified in Table 15.

6. Check that the Cryptogram Information Data retrieved from the ICC Dynamic Data is equal to the Cryptogram Information Data obtained from the response to the first GENERATE AC command. If this is not the case, combined dynamic data authentication/AC generation has failed.

7. Concatenate from left to right the second to the sixth data elements in Table 18 (that is, Signed Data Format through Pad Pattern), followed by the Unpredictable Number.

8. Apply the indicated hash algorithm (derived from the Hash Algorithm Indicator) to the result of the concatenation of the previous step to produce the hash result.

9. Compare the calculated hash result from the previous step with the recovered Hash Result. If they are not the same, dynamic data authentication has failed.

If all the above steps were executed successfully, dynamic data authentication was successful. The ICC Dynamic Number and the ARQC or TC contained in the ICC

Dynamic Data recovered in Table 15 shall be stored in Tag '9F4C' and in Tag '9F26', respectively.

7. Personal Identification Number Encipherment

If supported, Personal Identification Number (PIN) encipherment for offline PIN verification is performed by the terminal using an asymmetric based encipherment mechanism in order to ensure the secure transfer of a PIN from a secure tamper-evident PIN pad to the ICC.

More precisely, the ICC shall own a public key pair associated with PIN encipherment. The public key is then used by the PIN pad or a secure component of the terminal (other than the PIN pad) to encipher the PIN, and the private key is used by the ICC to verify the enciphered PIN.

In the case a secure terminal component other than the PIN pad is used for PIN encipherment, then the transport of the PIN from the PIN pad to the secure component must be secured in accordance with the requirements of section 11.1.

The PIN block used in the data field to be enciphered shall be 8 bytes as shown in Part I of Book 3 of these specifications.

7.1 Keys and Certificates

If offline PIN encipherment is supported, the ICC shall own a unique public key pair consisting of a public encipherment key and the corresponding private decipherment key. This specification allows the following two possibilities.

1. The ICC owns a specific ICC PIN Encipherment Private and Public Key. The ICC PIN Encipherment Public Key shall be stored on the ICC in a public key certificate in exactly the same way as for the ICC Public Key for dynamic data authentication as specified in Section 6.

   The ICC PIN encipherment public key pair has an ICC PIN Encipherment Public Key Modulus of N_{PE} bytes, where $N_{PE} \leq N_I \leq N_{CA} \leq 248$, N_I being the length of the Issuer Public Key Modulus (See section 6.1). If $N_{PE} > (N_I - 42)$, the ICC PIN Encipherment Public Key Modulus is divided into two parts, one part consisting of the $N_I - 42$ most significant bytes of the modulus (the Leftmost Digits of the ICC PIN Encipherment Public Key) and a second part consisting of the remaining $N_{PE} - (N_I - 42)$ least significant bytes of the modulus (the ICC PIN Encipherment Public Key Remainder).

   The ICC PIN Encipherment Public Key Exponent shall be equal to 3 or $2^{16}+1$.

   The ICC PIN Encipherment Public Key Certificate is obtained by applying the digital signature scheme specified in Annex A2.1 on the data in Table 19 using the Issuer Private Key.

Field Name	Length	Description	Format
Certificate Format	1	Hex. Value '04'	b
Application PAN	10	PAN (padded to the right with hex. 'F's)	cn 20
Certificate Expiration Date	2	MMYY after which this certificate is invalid	n 4
Certificate Serial Number	3	Binary number unique to this certificate assigned by the issuer	b
Hash Algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme[1]	b
ICC PIN Encipherment Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the ICC PIN Encipherment Public Key[1]	b
ICC PIN Encipherment Public Key Length	1	Identifies the length of the ICC PIN Encipherment Public Key Modulus in bytes	b
ICC PIN Encipherment Public Key Exponent Length	1	Identifies the length of the ICC PIN Encipherment Public Key Exponent in bytes	b
ICC PIN Encipherment Public Key or Leftmost Digits of the ICC PIN Encipherment Public Key	$N_I - 42$	If $N_{PE} \leq N_I - 42$, this field consists of the full ICC PIN Encipherment Public Key padded to the right with $N_I - 42 - N_{PE}$ bytes of value 'BB' If $N_{PE} > N_I - 42$, this field consists of the $N_I - 42$ most significant bytes of the ICC PIN Encipherment Public Key [8]	b
ICC PIN Encipherment Public Key Remainder	0 or $N_{PE} - N_I + 42$	This field is only present if $N_{PE} > N_I - 42$ and consists of the $N_{PE} - N_I + 42$ least significant bytes of the ICC PIN Encipherment Public Key	b
ICC PIN Encipherment Public Key Exponent	1 or 3	ICC PIN Encipherment Public Key Exponent equal to 3 or $2^{16}+1$	b

Table 19 - ICC PIN Encipherment Public Key Data to be Signed by the Issuer (i.e. input to the hash algorithm)

2. The ICC does not own a specific ICC PIN encipherment public key pair, but owns an ICC public key pair for dynamic data authentication as specified in

[8] As can be seen in Annex A2.1, $N_I - 22$ bytes of the data signed are retrieved from the signature. Since the first through the eighth data elements in Table 7 total 20 bytes, there are $N_I - 22 - 20 = N_I - 42$ bytes left for the data to be stored in the signature.

Section 2.1 of this specification. This key pair can then be used for PIN encipherment. The ICC Public Key is stored on the ICC in a public key certificate as specified in Section 6.

The first step of PIN encipherment shall be the retrieval of the public key to be used by the terminal for the encipherment of the PIN. This process takes place as follows.

1. If the terminal has obtained all the data objects specified in Table 20 from the ICC, then the terminal retrieves the ICC PIN Encipherment Public Key in exactly the same way as it retrieves the ICC Public Key for dynamic data authentication (see Section 6).

2. If the terminal has not obtained all the data objects specified in Table 20, but has obtained all the data objects specified in Table 8, then the terminal retrieves the ICC Public Key as described in Section 6.

3. If the conditions under points 1 and 2 above are not satisfied, then PIN encipherment has failed.

Tag	Length	Value	Format
-	5	Registered Application Provider Identifier (RID)	b
'8F'	1	Certification Authority Public Key Index	b
'90'	N_{CA}	Issuer Public Key Certificate	b
'92'	$N_I - N_{CA} + 36$	Issuer Public Key Remainder, if present	b
'9F32'	1 or 3	Issuer Public Key Exponent	b
'9F2D'	N_I	ICC PIN Encipherment Public Key Certificate	b
'9F2E'	1 or 3	ICC PIN Encipherment Public Key Exponent	b
'9F2F'	$N_{PE} - N_I + 42$	ICC PIN Encipherment Public Key Remainder, if present	b

Table 20 - Data Objects Required for the Retrieval of the ICC PIN Encipherment Public Key

7.2 PIN Encipherment and Verification

The exchange and verification of an enciphered PIN between terminal and ICC takes place in the following steps.

1. The PIN is entered in plaintext format on the PIN pad and a PIN block is constructed as defined in Part I of Book 3 of these specifications.

2. The terminal issues a GET CHALLENGE command to the ICC to obtain an 8-byte unpredictable number from the ICC.

3. The terminal generates a Random Pad Pattern consisting of N – 17 bytes, where N is the length in bytes of the public key to be used for PIN encipherment retrieved as specified in Section 7.1 (hence N = N_{PE} or N = N_{IC}).

4. Using the PIN Encipherment Public Key or the ICC Public Key retrieved as specified in Section 7.1, the terminal applies the RSA Recovery Function specified Annex B2.1.3 to the data specified in Table 21 in order to obtain the Enciphered PIN Data.

Field Name	Length	Description	Format
Data Header	1	Hex. Value '7F'	b
PIN Block	8	PIN in PIN Block	b
ICC Unpredictable Number	8	Unpredictable number obtained from the ICC with the GET CHALLENGE command	b
Random Pad Pattern	N_{IC} – 17	Random Pad Pattern generated by the terminal	b

Table 21 - Data to be Enciphered for PIN Encipherment

5. The terminal issues a VERIFY command including the Enciphered PIN Data obtained in the previous step.

6. With the ICC Private Key, the ICC applies the RSA Signing Function specified in Annex B2.1.2 to the Enciphered PIN Data in order to recover the plain text data specified in Table 16.

7. The ICC verifies whether the ICC Unpredictable Number recovered is equal to the ICC Unpredictable Number generated by the ICC with the GET CHALLENGE command. If this is not the case, PIN verification has failed.

8. The ICC verifies whether the Data Header recovered is equal to '7F'. If this is not the case, PIN verification has failed.

9. The ICC verifies whether the PIN included in the recovered PIN Block corresponds with the PIN stored in the ICC. If this is not the case, PIN verification has failed.

If all the above steps were executed successfully, enciphered PIN verification was successful.

In order for this mechanism to be secure, the steps 3 and 4 must be executed in a secure environment. This can either be

- The tamper-evident PIN pad itself.

- A secure component in the terminal. In this case the transport of the PIN from the PIN pad to the secure component must be secured in accordance with the requirements of section 11.1.

8. Application Cryptogram and Issuer Authentication

The aim of this section is to provide methods for the generation of the Application Cryptograms (TC, ARQC or AAC) generated by the ICC and the Authorisation Response Cryptogram (ARPC) generated by the issuer and verified by the ICC. For more details on the role of these cryptograms in a transaction, see Part II of Book 3 of these specifications.

Note that the methods provided in this specification are not mandatory. Issuers may decide to adopt other methods for these functions.

8.1 Application Cryptogram Generation

8.1.1 Data Selection

An Application Cryptogram consists of a Message Authentication Code generated over data

- Referenced in the ICC's DOLs and transmitted from the terminal to the ICC in the GENERATE AC or other command.
- Accessed internally by the ICC.

The recommended minimum set of data elements to be included in the Application Cryptogram generation is specified in Table 22.

Value	Source
Amount, Authorised (Numeric)	Terminal
Amount Other (Numeric)	Terminal
Terminal Country Code	Terminal
Terminal Verification Results	Terminal
Transaction Currency Code	Terminal
Transaction Date	Terminal
Transaction Type	Terminal
Unpredictable Number	Terminal
Application Interchange Profile	ICC
Application Transaction Counter	ICC

Table 22 – Recommended Minimum Set of Data Elements for Application Cryptogram Generation

8.1.2 Application Cryptogram Algorithm

The method for Application Cryptogram generation takes as input a unique 16-byte ICC Application Cryptogram Master Key MK_{AC} and the data selected as described in section 8.1.1, and computes the 8-byte Application Cryptogram in the following two steps:

1. The first step consists of the derivation with the session key derivation function specified in Annex A1.3 of a 16-byte Application Cryptogram Session Key SK_{AC} from the ICC Application Cryptogram Master Key MK_{AC} and the 2-byte Application Transaction Counter (ATC) of the ICC.

2. The second step consists of the generation of the 8-byte Application Cryptogram by applying the MAC algorithm specified in Annex A1.2 to the data selected and using the 16-byte Application Cryptogram Session Key derived in the previous step.

8.2 Issuer Authentication

The method for the generation of an 8-byte Authorisation Response Cryptogram ARPC consists of applying the Triple-DES algorithm as specified in Annex B1.1 to the 8-byte ARQC generated by the ICC as described in section 8.1, the 2-byte Authorisation Response Code ARC, and using the 16-byte Application Cryptogram Session Key SK_{AC} (see section 8.1) in the following way:

1. Pad the 2-byte ARC with six zero bytes to obtain the 8-byte number

$$X := (ARC \,||\, '00' \,||\, '00' \,||\, '00' \,||\, '00' \,||\, '00' \,||\, '00').$$

2. Compute $Y := ARQC \oplus X$.

3. The 8-byte ARPC is then obtained by

$$ARPC := DES3(SK_{AC})[Y].$$

8.3 Key Management

The mechanisms for Application Cryptogram and Issuer Authentication require the management by the issuer of the unique ICC Application Cryptogram Master Keys. Annex A1.4 specifies a method for the derivation of the ICC Application Cryptogram Master Keys from the Primary Account Number (PAN) and the PAN Sequence Number.

9. Secure Messaging

The objectives of secure messaging are to ensure data confidentiality, data integrity and authentication of the sender. Data integrity and issuer authentication are achieved using a MAC. Data confidentiality is achieved using encipherment of the data field.

9.1 Secure Messaging Format

Secure messaging shall be according to one of the following two formats.

- **Format 1:** Secure messaging format according to ISO/IEC 7816-4, section 5.6, where the data field of the affected command uses BER-TLV encoding and encoding rules of ASN.1/ISO 8825 apply strictly. This is explicitly specified in the lowest significant nibble of the class byte of the command, which is set to 'C'. This also implies that the command header is always integrated in MAC calculation.

- **Format 2:** Secure messaging format where the data field of the affected command does not use BER-TLV encoding for secure messaging, but may use it for other purposes. In this case, the data objects contained in the data field and corresponding lengths of these data objects shall be known by the sender of a command using secure messaging and known by the currently selected application. In compliance with ISO/IEC 7816-4, secure messaging according to Format 2 is explicitly specified in the lowest significant nibble of the class byte of the command, which is set to '4'.

9.2 Secure Messaging for Integrity and Authentication

9.2.1 Command Data Field

9.2.1.1 Format 1

The data field of the command is composed of the following TLV data objects as shown in Figure 3.

- The command data to be signed, if present.

If the command data field is BER-TLV encoded, it shall either not belong to the context-specific class (the tag shall not lie in the range '80' to 'BF') or shall have an odd tag (note that this may be a constructed data object).

If the command data field is not BER-TLV encoded, it shall be encapsulated with the template '81'.

- The second data object is the MAC. Its tag is '8E', and its length shall be in the range of four to eight bytes.

Tag 1	Length 1	Value 1	Tag 2	Length 2	Value 2
T	L	Value (L bytes)	'8E'	'04'-'08'	MAC (4-8 bytes)

Figure 3 - Format 1 Command Data Field for Secure Messaging for Integrity and Authentication

9.2.1.2 Format 2

The data elements (including the MAC) contained in the data field and the corresponding lengths shall be known by the sender of a command using secure messaging and known by the currently selected application. The MAC is not BER-TLV coded and shall always be the last data element in the data field and its length shall be in the range of 4 to 8 bytes (see Figure 4).

Value 1	Value 2
Command data (if present)	MAC (4-8 bytes)

Figure 4 - Format 2 Command Data Field for Secure Messaging for Integrity and Authentication

9.2.2 MAC Session Key Derivation

The first step of the MAC generation for secure messaging for integrity consists of deriving a unique 16-byte MAC Session Key from the ICC's unique 16-byte MAC Master Key and the 2-byte ATC. A method to do this is specified in Annex A1.3.

9.2.3 MAC Computation

The MAC is computed by applying the mechanism described in Annex A1.2 with the MAC Session Key derived as described in section 9.2.2 to the message to be protected.

If secure messaging is according to Format 1, the message to be protected shall be constructed from the header of the command APDU (CLA INS P1 P2) and the command data (if present) according to the rules specified in ISO/IEC 7816-4, Section 5.6.

If secure messaging is according to Format 2, the message to be protected shall be constructed according to the payment scheme proprietary specifications. It shall however always contain the header of the command APDU and the command data (if present).

In all cases, if the MAC used for secure messaging has been specified as having a length less than 8 bytes, the MAC is obtained by taking the leftmost (most significant) bytes from the 8-byte result of the calculation described above.

9.3 Secure Messaging for Confidentiality

9.3.1 Command Data Field

9.3.1.1 Format 1

The format of an enciphered data object in a command data field is shown in Figure 5.

Tag	Length	Value
T	L	Cryptogram (enciphered data)

Figure 5 - Format 1 Enciphered Data Object in a Command Data Field

Depending on the plaintext data to be enciphered, ISO/IEC 7816-4 specifies the tag to be allocated to the resulting cryptogram. An odd tag shall be used if the object is to be integrated in the computation of a MAC; an even tag shall be used otherwise.

9.3.1.2 Format 2

Data encipherment is applied to the full plaintext command data field with the exception of a MAC (see Figure 6).

Value1	Value2
Cryptogram (enciphered data)	MAC (if present)

Figure 6 - Format 2 Command Data Field for Secure Messaging for Confidentiality

9.3.2 Encipherment Session Key Derivation

The first step of the encipherment/decipherment for secure messaging for confidentiality consists of deriving a unique 16-byte Encipherment Session Key from the ICC's unique 16-byte Encipherment Master Key and the 2 byte ATC. A method to do this is specified in Annex A1.3.

9.3.3 Encipherment/Decipherment

Encipherment/decipherment of the plain/enciphered command data field takes place according to the mechanism described in Annex A1.1 with the Encipherment Session Key derived as described in the section 9.3.2.

9.4 Key Management

The secure messaging mechanisms require the management by the issuer of the unique ICC MAC and Encipherment Master Keys. Annex A1.4 specifies a method for the derivation of the ICC MAC and Encipherment Master Keys from the Primary Account Number (PAN) and the PAN Sequence Number.

10. Certification Authority Public Key Management Principles and Policies

This section defines a framework for the principles and policies for a payment system for the management of the Certification Authority Public Keys used for static and dynamic data authentication as specified in this specification.

Principles are concepts identified as the basis for implementing Certification Authority Public Key management. These principles can give rise to policies that may be shared across the payment systems, or policies that are adopted by individual payment systems. Each payment system will develop its own set of procedures to implement these policies.

10.1 Certification Authority Public Key Life Cycle

10.1.1 Normal Certification Authority Public Key Life Cycle

The life cycle of a Certification Authority Public Key in normal circumstances can be divided into the following consecutive phases:

- Planning
- Generation
- Distribution
- Usage
- Revocation (Scheduled).

10.1.1.1 Planning

During the planning phase, the payment system investigates the requirements for the introduction of new Certification Authority Public Key pairs in the near future. These requirements are related to the number of keys required and the parameters of these keys.

An important part of the planning phase is the review of the security of RSA to determine the life expectancy of existing and potential new keys. This review is to lead to the setting of lengths and expiration dates for new keys and the potential modification of the expiration dates of existing keys, and a roll-out schedule of replacement keys.

10.1.1.2 Generation

If the results of the planning phase require the introduction of new Certification Authority Public Key pairs, these will have to be generated by the payment system. More precisely, the payment system certification authority (a physically and logically highly secured infrastructure operated by the payment system) will

generate in a secure way the necessary RSA Certification Authority Private/Public Key pairs for further use.
Subsequent to generation the secrecy of the Certification Authority Private Keys must be maintained, and the integrity of both Certification Authority Public and Private Keys must also be maintained.

10.1.1.3 Distribution

In the key distribution phase, the payment system certification authority will distribute newly generated Certification Authority Public Keys to its member Issuers and Acquirers for the following purposes (see Figure 7):

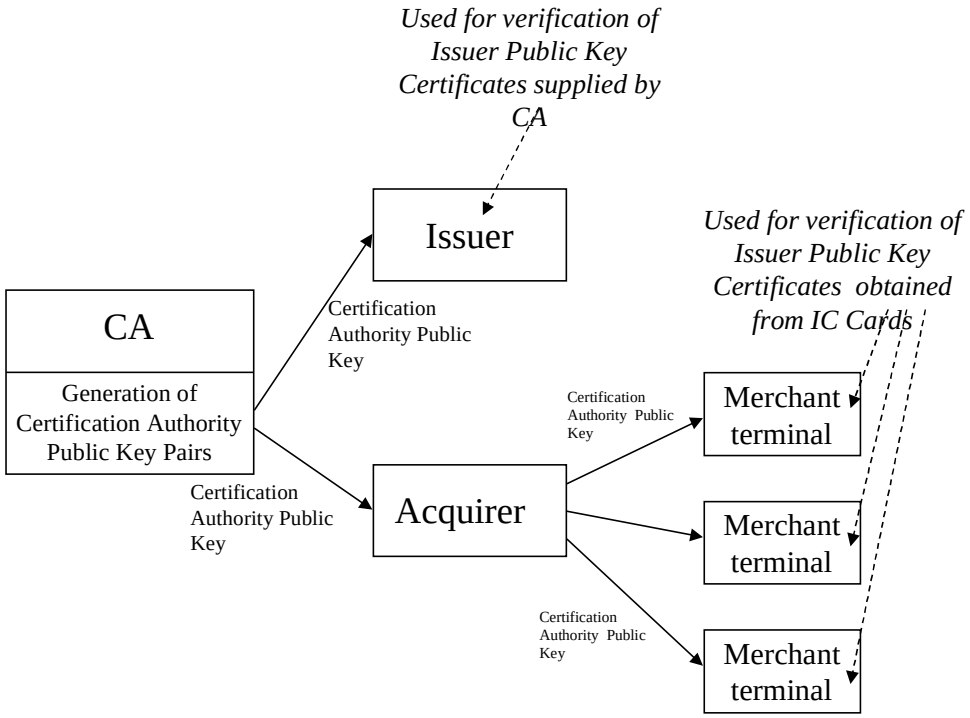


Figure 7 – Certification Authority Public Key Distribution

- To issuers, to verify Issuer Public Key Certificates supplied by the payment system certification authority during the key usage phase (see below).
- To acquirers, for secure loading of the Certification Authority Public Keys in its merchant terminals.

In order to prevent the introduction of fraudulent Certification Authority Public Keys, the interfaces between the payment system certification authority and the issuers and acquirers need to ensure the integrity of the Certification Authority Public Keys distributed.

10.1.1.4 Usage

The Certification Authority Public Key is used in the merchant terminals to perform static or dynamic data authentication as specified in sections 5 and 6 of this specification.

The Certification Authority Private Key is used by the payment system certification authority for the generation of the Issuer Public Key Certificates. More precisely, the following interactions take place (see Figure 8):

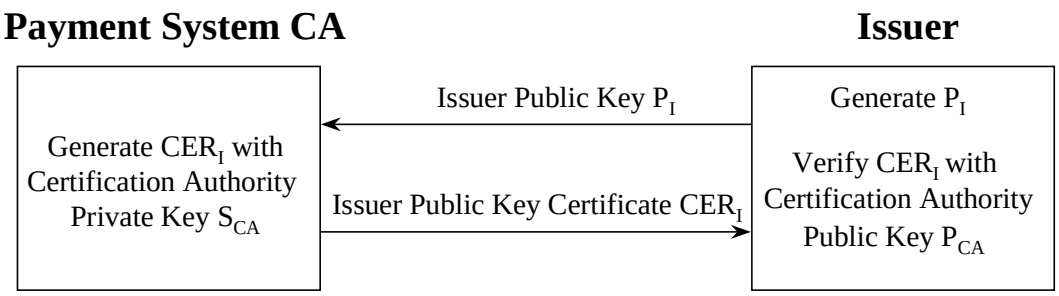


Figure 8 – Issuer Public Key Distribution

- The issuer generates its Issuer Public Key and sends it to the payment system certification authority.
- The payment system certification authority signs the Issuer Public Key with the Certification Authority Private Key to obtain the Issuer Public Key Certificate that is returned to the issuer.
- With the Certification Authority Public Key, the issuer verifies the correctness of the received Issuer Public Key Certificate. If it is correct, the issuer can then include it as part of the personalisation data for its IC Cards.

In order to prevent the introduction of fraudulent Issuer Public Keys, the interfaces between the issuer and the payment system certification authority need to ensure the integrity of the Issuer Public Keys submitted for certification.

10.1.1.5 Revocation (Scheduled)

Once a Certification Authority Public Key pair has reached its planned expiration date set during the planning phase, it has to be removed from service. Practically speaking, this means the following.

- As of that expiration date, Issuer Public Key Certificates produced with the Certification Authority Private Key will no longer be valid. Issuers should therefore ensure that IC Cards personalised with such Issuer Public Key Certificates expire no later than the expiration date of the Certification Authority Public Key pair.
- An appropriate time prior to that expiration date, the payment system certification authority will stop signing Issuer Public Keys with the corresponding Certification Authority Private Key.
- As of that expiration date, acquirers need to remove the Certification Authority Public Keys from service in their terminals within a specific grace period after expiration.

10.1.2 Certification Authority Public Key Pair Compromise

In the event of a Certification Authority Public Key pair compromise, an emergency process needs to be put in place that in the end may lead to the accelerated revocation of the Certification Authority Public Key pair before its planned expiration. In this case, there are additional phases in the key life cycle:

- Detection
- Assessment
- Decision
- Revocation (Accelerated).

These phases are described below.

10.1.2.1 Detection

The compromise of a Certification Authority Public Key pair can be either an actual compromise, for example a confirmed security breach at the payment system certification authority, or a confirmed breaking of the key by cryptanalysis. In addition, compromise may be:

- Suspected: system monitoring or member and cardholder complaint indicates that fraudulent transactions have occurred which could be due to key compromise, but this is not confirmed, or

- Potential: cryptanalytic techniques, for example factorisation, have developed such that with resources available any key of a given length could be compromised, but there is no evidence that this has occurred.

Detection of a key compromise may vary from awareness of an actual physical break-in of the payment system certification authority, through the reporting of fraudulent off-line transactions by the fraud and risk management systems put in place by the payment system and its members, to intelligence on factorisation advances gathered from the cryptographic community.

10.1.2.2 Assessment

The assessment of a (potential) Certification Authority Public Key pair compromise will include technical, risk and fraud, and, most importantly, business impacts for the payment system and its members. The results of the assessment will include the confirmation of the compromise, the determination of possible courses of action against costs and risk of the compromise, and presenting results of the assessment to support a decision.

10.1.2.3 Decision

Based on the results of the assessment phase, the payment system will decide on a course of action that will be taken for a key compromise. In the worst case, this decision will consist of the actual unplanned revocation of a Certification Authority Public Key before its planned expiration date.

10.1.2.4 Revocation (Accelerated)

The decision to revoke a Certification Authority Public Key will lead to the communication to the payment system members of a new expiration date of that key. The process after that is the same as for the planned revocation described in section 10.1.1.5.

10.2 Key Revocation Principles and Policies by Phase

10.2.1 General Principles

- Support of Certification Authority Public Key revocation is a requirement for each payment system's IC Card credit and debit products.
- Payment systems will align policies, procedures, and schedules for Certification Authority Public Key revocation where practical.
- EMVCo, LLC will use a common definition of the phases of the Certification Authority Public Key revocation process and a common terminology in internal and member communications.
- Each payment system operates as a closed system with regard to any legal requirements relative to Certification Authority Public Key pairs.

10.2.2 Planning Phase

10.2.2.1 Phase Definition

The Planning phase involves review and planning of Certification Authority Public Key pairs. Existing keys are reviewed for resistance to attack, and new key planning is undertaken. Length and expiration dates of existing and new keys are reviewed by risk and cryptography experts to confirm that the key life expectancy is considered secure. Lengths of new keys are determined, and a rollout schedule of replacement keys is maintained.

10.2.2.2 EMV Principles

- Key sizes should reflect maximum feasible security consistent with terminal capability and POS operational timing.
- Payment systems should synchronize the expiration date of keys of a particular length where practical. Final decision authority for key revocation rests with each payment system.
- In the event of announcement of an accelerated revocation by a payment system, the payment system may request convening an EMVCo, LLC planning session to address the revocation, the key compromise, and its impacts.

10.2.2.3 Shared Payment System Policies

- EMVCo, LLC will conduct annual review sessions for Certification Authority Public Key pair strength evaluation, using state of the art information and analysis from the fields of computer science, cryptography, and data security. Any payment system may request an emergency meeting for key review at any time.
- EMVCo, LLC will prepare “best information” estimates of relative key strength for existing key lengths based on current evaluation criteria, and will make recommendations for rollout of new key lengths.
- The recommendations of this review process will be circulated to the payment systems, who will use them to set their individual policies. Each payment system will identify areas where payment system differentiation is required.
- Payment systems will use EMVCo, LLC recommendations as a factor in determining policy on number and length of live keys, exponent value, expiry date, and planned revocation schedule. Payment systems will publish these details to members within 90 days of receipt of EMVCo, LLC recommendations.
- Key introduction and revocation will normally be on a planned, scheduled basis, but can be accelerated based on results of key life review.
- All Certification Authority Public Keys will have December 31st as planned expiration date.

- Acquirers have a six month grace period starting from the planned expiration date (until June 30th of the following calendar year) to withdraw an expired key from all terminals.
- All new Certification Authority Public Keys will be distributed prior to December 31st.
- Acquirers have a six month grace period (until June 30th of the following calendar year) to install any new keys in all terminals.
- New Certification Authority Public Key pairs will be valid starting July 1st of that same calendar year.
- In the event of an accelerated revocation, a six-month grace period will similarly be maintained for key withdrawal in all terminals, but the fixed date of December 31st is not applicable.
- Notification to members and timing for any key revocation is the responsibility of each payment system.

10.2.3 Generation Phase

10.2.3.1 Phase Definition

Key generation is the process of a payment system generating a Certification Authority Public Key pair.

10.2.3.2 EMV Principles

- Certification Authority Public Key pairs shall be generated in a secure environment according to accepted industry best practice.
- Within each Registered Application Provider Identifier (RID), the Certification Authority Public Key Index is a unique value pointing to a particular Certification Authority Public Key pair. The value of a Certification Authority Public Key Index for a specific key shall not be changed.

10.2.3.3 Shared Payment System Policies

- None Identified.

10.2.4 Distribution Phase

10.2.4.1 Phase Definition

Key distribution is the process of circulating the public component of a Certification Authority Public Key Pair to get it into the marketplace. Certification Authority Public Keys must ultimately appear in merchant terminals. Certification Authority Private Keys will be used to produce Issuer Public Key Certificates, and are to be kept in the secure environment of the payment system certification authority.

10.2.4.2 EMV Principles

- Key distribution must ensure key integrity and origin authenticity.

10.2.4.3 Shared Payment System Policies

- Payment systems will support distribution of their public keys from the Certification Authority to acquirers and issuers via physical and/or electronic means.
- All new Certification Authority Public Keys will be distributed for receipt by recipients before December 31st.
- Payment systems will include a method allowing a recipient to validate a received public key, regardless of method of transmission.
- Certification Authority Public Keys will be distributed to acquirers with adequate lead time to allow installation in terminals before the corresponding private key is used to sign Issuer Public Keys.
- Certification Authority Public Keys will be distributed to issuers so that they may validate the Issuer Public Key Certificates produced by the certification authority.
- Each payment system certification authority will ensure that it does not distribute more than the maximum number (six) of keys that can be stored per RID in a terminal (see Section 10.2.5).

10.2.5 Key Usage Phase

10.2.5.1 Phase Definition

This phase is concerned with the normal day to day use of the Certification Authority Public Key pairs. Copies of the Certification Authority Public Keys will be used by terminals to perform static or dynamic data authentication (SDA or DDA) and PIN encipherment during transactions with the appropriate payment system branded cards. The Certification Authority Private Keys will be held in the payment system certification authority and used to sign Issuer Public Keys, creating Issuer Public Key Certificates which the issuer will personalize onto its cards.

10.2.5.2 EMV Principles

- Terminals that support SDA and/or DDA shall provide support for six Certification Authority Public Keys per RID for Europay, MasterCard and Visa debit/credit applications based on the *EMV Specifications for Payment Systems,* Version 4.0. Terminals shall support these keys up to 1984 bits (248 bytes) in length, as specified in this specification.
- Terminals shall support the ability to install a Certification Authority Public Key, and the ability to withdraw a key from service as of a given date.
- Terminals shall provide the ability to validate Certification Authority Public Key integrity.
- Payment systems will be responsible for ensuring the security of their Certification Authority Public Key pairs.

10.2.5.3 Shared Payment System Policies

- Payment systems will validate the integrity and origin of Issuer Public Keys prior to issuing a certificate.
- A payment system certification authority will begin using the private component of a Certification Authority Public Key pair no sooner than 6 months after the distribution of that key to acquirers.
- The expiry date of any issued IC Card shall be no later than the expiry date of the Issuer Public Key Certificate on that IC Card, and shall be no later than the published (at the time of card issuance) revocation date of the Certification Authority Public Key pair used to produce the Issuer Public Key Certificate.
- The expiry date of an Issuer Public Key Certificate shall be no later than the published (at the time of certificate issuance) revocation date of the Certification Authority Public Key pair used to produce the Issuer Public Key Certificate.
- The expiry date of an IC Card Public Key Certificate shall be no later than the expiry date of the Issuer Public Key used to produce the IC Card Public Key Certificate.

10.2.6 Detection Phase

10.2.6.1 Phase Definition

The process that enables an entity to recognize that a Certification Authority Public Key pair has been, or is suspected of being compromised. There are multiple types of compromise, including physical and logical, suspected, potential, and confirmed.

10.2.6.2 EMV Principles

- EMVCo, LLC will provide a forum for payment systems to share evaluation of cryptanalytic advances that might lead to potential compromise of the digital signature scheme specified in this specification.
- Monitoring of key integrity and detection of suspected or potential Certification Authority Public Key pair compromise is the responsibility of each payment system.

10.2.6.3 Shared Payment System Policies

- Members shall notify a payment system of conditions or transactions that indicate possible or suspected compromise of a specific Certification Authority Public Key pair from that payment system.

10.2.7 Assessment Phase

NOTE: This phase applies only to accelerated revocations.

10.2.7.1 Phase Definition

If a Certification Authority Public Key compromise is detected or suspected, the owning payment system must assess the impact to business operations. Assessment includes confirming the compromise, determining possible courses of action, evaluating the cost of action against costs and risk of the compromise, and presenting results of the assessment to support a decision.

10.2.7.2 EMV Principles

- Assessment of suspected or potential Certification Authority Public Key pair compromise is the responsibility of each payment system.
- Payment systems will develop assessment policies and procedures that follow generally accepted best practices in risk management.
- There are different levels of compromise requiring different sets of actions depending on the compromise and a business assessment.

10.2.7.3 Shared Payment System Policies

- Payment system assessment will include actual and reputational costs to the payment system and to members. Potential courses of action will include an assessment of member and marketplace impact.

10.2.8 Decision Phase

NOTE: This phase applies only to accelerated revocations.

10.2.8.1 Phase Definition

As a result of the assessment phase, a payment system decides on a course of action that will be taken for a Certification Authority Public Key pair compromise.

10.2.8.2 EMV Principles

- The decision to revoke a specific Certification Authority Public Key Pair is at the sole discretion of the payment system that operates the certification authority for that key.
- Payment systems will develop and publish to their members a set of policies and procedures that detail the decision-making process for accelerated key revocation. These policies will include a method of notification to all affected issuers and acquirers.

10.2.8.3 Shared Payment System Policies

- None identified.

10.2.9 Revocation Phase

10.2.9.1 Phase Definition

The key management process of withdrawing a key from service and dealing with the legacy of its use. Key revocation can be on schedule or accelerated. In the case of Certification Authority Public Key pairs, revocation means that the private key is no longer used to produce Issuer Public Key Certificates and that copies of the public key are withdrawn from service in terminals. Issuer Public Key Certificates signed with the private key are (as of a specific date) no longer valid in circulation on IC Cards.

10.2.9.2 EMV Principles

- Certification Authority Public Key revocation will be according to a previously published schedule unless a payment system has detected an imminent threat to product security. All scheduled revocations will conform to the "revocation window" dates developed by EMVCo, LLC.
- In case of an accelerated revocation, payment systems will take member impact into account, including terminal access, card re-issuance, and increased network traffic. Lead times for member activities shall be the same as during a scheduled revocation.

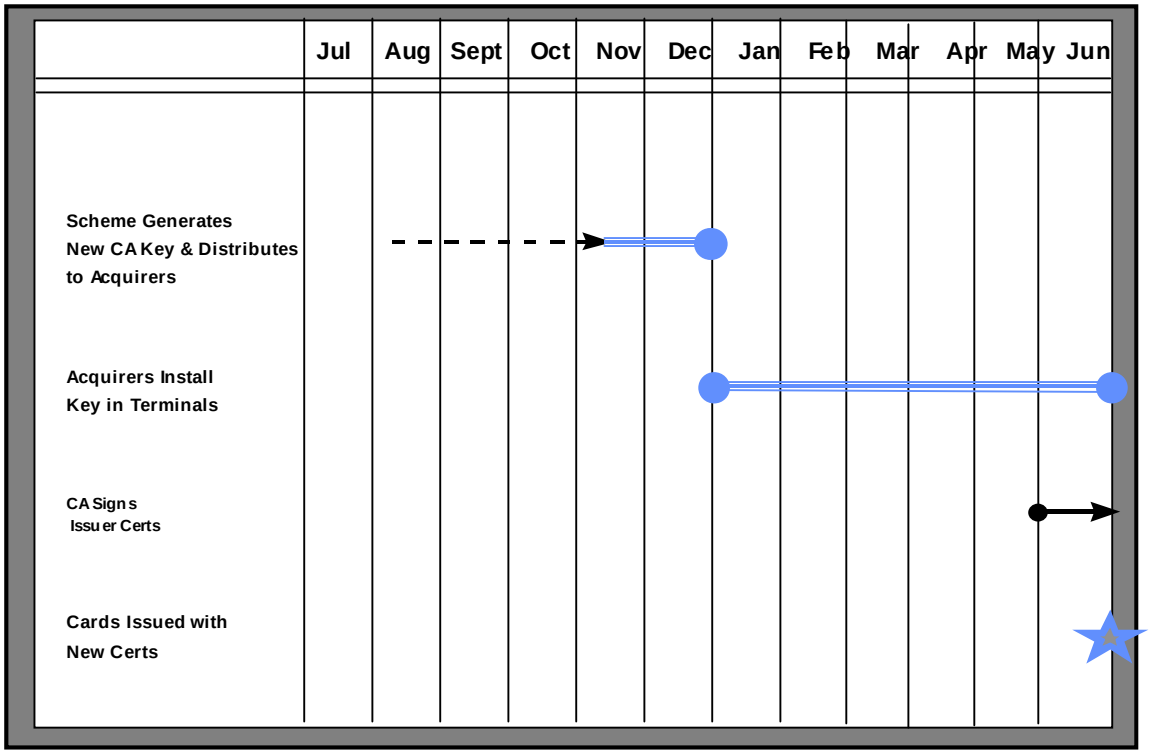
10.2.9.3 Shared Payment System Policies

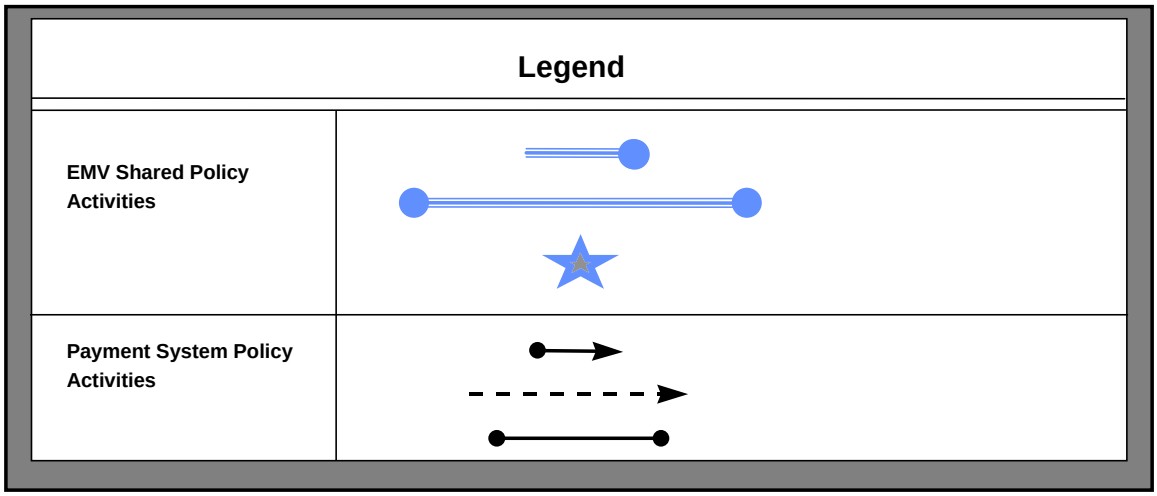
- Revocation policies and procedures will be the same as for scheduled and accelerated revocations, wherever practical.
- All Certification Authority Public Keys will have December 31st as their planned expiration date. Acquirers shall have a six month grace period (until June 30th of the following calendar year) to withdraw the revoked key.
- Revocation of a Certification Authority Public Key pair requires that the public key component is withdrawn from service in all terminals within a six-month timeframe, consistent with payment system rules.
- In the case of an accelerated revocation, the introduction and withdrawal lead times will be the same as for scheduled revocations, however, the revocation date will be determined at the discretion of the payment system.

10.3 Sample Timelines

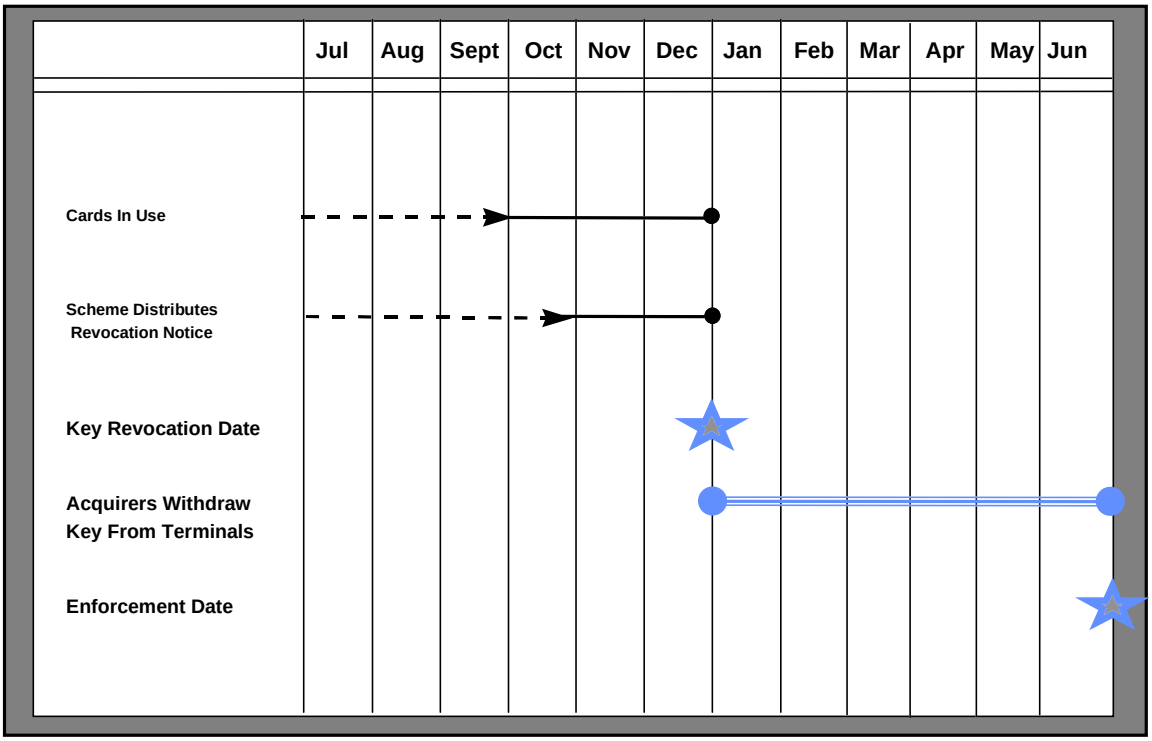
The following diagrams present sample timelines for the revocation and introduction of Certification Authority Public Keys, based on the principles and policies detailed in this document. Each timeline represents a scheduled key introduction or withdrawal. In the case of an accelerated introduction or withdrawal, lead times for tasks would remain the same, but the month of the actual key introduction date and key revocation would be at the discretion of the payment system.

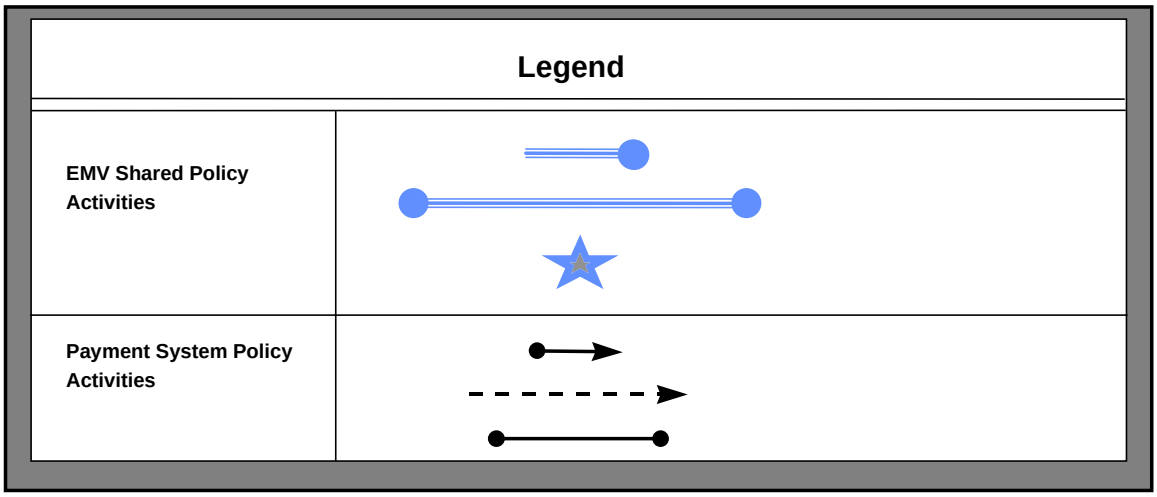
10.3.1 Key Introduction





10.3.2 Key Withdrawal





11. Terminal Security and Key Management Requirements

This section describes the general terminal requirements for handling sensitive data, such as plaintext PINs and cryptographic keys. More specifically, it addresses PIN pad security requirements and key management requirements for Certification Authority Public Keys.

11.1 Security Requirements

11.1.1 Tamper-Evident Devices

A tamper-evident device shall ensure that in its normal operating environment the device or its interface does not disclose or alter any sensitive data that is entering or leaving the device or that is stored or processed in the device. (See ISO 13491 for further requirements for tamper-evident devices.)

When a tamper-evident device is operated in a securely controlled environment, the requirements on device characteristics may be reduced since protection is provided by the controlled environment and the management of the device.

11.1.1.1 Physical Security

A tamper-evident device shall be designed to restrict physical access to internally stored sensitive data and to deter theft, unauthorised use, or unauthorised modification of the equipment. These objectives generally require the incorporation of tamper-resistant, tamper-detection, tamper-indication, or response mechanisms, such as visible or audible alarms.

A tamper-evident device, when not in operation, shall not contain secret cryptographic keys or other sensitive data (e.g. PINs) used by the device for any previous transaction (although it may contain authentication information used solely for the purpose of enhancing the tamper-evidence of the device). It may be penetrated without loss of security, provided that this penetration is detected before the device and the stored cryptographic keys are again placed into operational use. If the device is designed to allow internal access, erasure of sensitive data must be immediately accomplished when the device is tampered with. A tamper-evident device depends on the detection by the user of attacks on its physical security. Therefore, it shall be so designed and have sufficient tamper-evident features so that any tampering shall be obvious to the cardholder or detected by the merchant or acquirer.

The device shall be designed and constructed so that:

- It is not feasible to penetrate the device to make any additions, substitutions, or modifications to the hardware or software of the device; or to determine or

modify any sensitive data and subsequently re-install the device, without requiring specialised skills and equipment not generally available, and without damaging the device so severely that the damage has a high probability of detection.

- Any unauthorised access to or modifications of sensitive data that are input, stored, or processed is achieved only by actual penetration of the device.

- The casing is not commonly available, to deter the manufacture of 'look-alike' counterfeit copies from commonly available components.

- Any failure of any part of the device does not cause the disclosure of secret or sensitive data.

- If the device design requires that parts of the device be physically separate and processing data or cardholder instructions pass between these separate components, there is an equal level of protection among all parts of the device.

- Integration of different device parts into a single tamper-evident housing is the necessary condition for exchanging sensitive data such as plaintext PINs.

11.1.1.2 Logical Security

A tamper-evident device shall be designed that no single function, nor any combination of functions, can result in disclosure of sensitive data, except as explicitly allowed by the security implemented in the terminal. The logical protection shall be sufficient so as to not compromise sensitive data, even when only legitimate functions are used. This requirement can be achieved by internal monitoring of statistics or imposing a minimum time interval between sensitive function calls.

If a terminal can be put into a 'sensitive state', that is, a state that allows functions that are normally not permitted (for example, manual loading of cryptographic keys), such a transition shall require the assistance of two or more trusted parties. If passwords or other plaintext data are used to control transit to a sensitive state, the input of such passwords shall be protected in the same manner as other sensitive data.

To minimise risks resulting from the unauthorised use of sensitive functions, the sensitive state shall be established with limits on the number of function calls (where appropriate), and a time limit. After the first of these limits is reached, the device shall return to normal state.

A tamper-evident device shall automatically clear its internal buffers at the end of a transaction or in a time-out situation.

11.1.2 PIN Pads

A PIN pad shall be a tamper-evident device. It shall support entry of a 4-12 digit PIN. When a display is present on a PIN pad, an indication of the entry of each digit shall be displayed. However, the values of the entered PIN shall not be displayed or disclosed by visible or audible feedback means, in accordance with ISO 9564-1.

When the terminal supports offline PIN verification, the IFD and PIN pad shall either be integrated into a single tamper-evident device or the IFD and PIN pad shall be two separate tamper-evident devices.

- If the IFD and PIN pad are integrated and the offline PIN is to be transmitted to the card in plaintext format, then the PIN pad does not encipher the offline PIN when the plaintext PIN is sent directly from the PIN pad to the IFD.

- If the IFD and PIN pad are integrated and the offline PIN is to be transmitted to the card in plaintext format, but the offline plaintext PIN is not sent directly from the integrated PIN pad to the IFD, then the PIN pad shall encipher the offline PIN according to ISO 9564-1 (or an equivalent payment system approved method) for transmission to the IFD. The IFD will then decipher the offline PIN for transmission in plaintext to the card.

- If the IFD and PIN pad are not integrated and the offline PIN is to be transmitted to the card in plaintext format, then the PIN pad shall encipher the offline PIN according to ISO 9564-1 (or an equivalent payment system approved method) for transmission to the IFD. The IFD will then decipher the offline PIN for transmission in plaintext to the card.

- If the offline PIN is to be transmitted to the card in enciphered format, then the PIN must be enciphered as described in section 7.2. The PIN encipherment process shall take place in either
  - The tamper-evident PIN pad itself.
  - A secure component in the terminal. In this case the PIN pad shall encipher the PIN according to ISO 9564-1 (or an equivalent payment system approved method) for secure transport of the PIN between the PIN pad and the secure component.

If the terminal supports online PIN verification, when the PIN is entered, the PIN shall be protected upon entry by encipherment according to ISO 9564-1, and the terminal shall transmit the PIN according to the payment system's rules.

The prompt for PIN entry messages displayed on the PIN pad shall be generated by the PIN pad.[9] This does not imply that only PIN-related messages may be displayed on the PIN pad, although those messages shall be authorised by the PIN pad prior to display. The PIN pad shall reject any unauthorised message display.

[9] This does not apply to PIN pads operated in a secure environment such as an ATM.

For an attended terminal, the amount entry process shall be separate from the PIN entry process to avoid accidental display of a PIN on the terminal display. In particular, if the amount and PIN are entered on the same key pad, the amount shall be validated by the cardholder before PIN entry is allowed.

The PIN pad shall be designed to provide privacy and confidentiality so that, during normal use, only the cardholder sees the information entered or displayed. The PIN pad shall be installed or replaced so that its immediate surroundings allows sufficient privacy to enable the cardholder to enter a PIN with minimum risk of the PIN being revealed to others.

The PIN pad shall automatically clear its internal buffers when either of the following conditions occur:

- Upon completion of the transaction.
- In a time-out situation, including when an inordinate period of time has elapsed since a PIN character was entered.

11.2 Key Management Requirements

This Section specifies the requirements for the management by the Acquirers of the Certification Authority Public Keys in the terminals. The requirements cover the following phases:

- Introduction of a Certification Authority Public Key in a terminal.
- Storage of a Certification Authority Public Key in a terminal.
- Usage of a Certification Authority Public Key in a terminal.
- Withdrawal of a Certification Authority Public Key from a terminal.

11.2.1 Certification Authority Public Key Introduction

When a payment system has decided that a new Certification Authority Public Key is to be introduced, a process is executed that ensures the distribution of the new key from the payment system to each acquirer. It is then the acquirers responsibility to ensure that the new Certification Authority Public Key and its related data (see Section 11.2.2) is conveyed to its terminals.

The following principles apply to the introduction of a Certification Authority Public Key from an acquirer to its terminals:

- The terminal must be able to verify that it received the Certification Authority Public Key and its related data error-free from the acquirer.

- The terminal must be able to verify that the received Certification Authority Public Key and related data originated from its legitimate acquirer.

- The acquirer must be able to confirm that the new Certification Authority Public Key was indeed introduced correctly in its terminals.

11.2.2 Certification Authority Public Key Storage

Terminals that support static data Authentication and/or dynamic data authentication shall provide support for six Certification Authority Public Keys per Registered Application Provider Identifier (RID) for Europay, MasterCard and Visa debit/credit applications based on the *EMV Specifications for Payment Systems*, Version 4.0.

Each Certification Authority Public Key is uniquely identified by the 5-byte RID that identifies the payment system in question, and the 1-byte Certification Authority Public Key Index, unique per RID and assigned by that payment system to a particular Certification Authority Public Key.

For each Certification Authority Public Key, the minimum set of data elements that has to be available in the terminal is specified in Table 17.

The RID and the Certification Public Key Index together uniquely identify the Certification Authority Public Key and associate it with the proper payment system.

The Certification Authority Public Key Algorithm Indicator identifies the digital signature algorithm to be used with the corresponding Certification Authority Public Key. The only acceptable value at this moment is hexadecimal '01', indicating the usage of the RSA algorithm in the digital signature scheme as specified in the annexes A2.1 and B2.1 of *this specification*. The Hash Algorithm Indicator specifies the hashing algorithm to produce the Hash-Result in the digital signature scheme. The only acceptable value at this moment is hexadecimal '01', indicating the usage of the SHA-1 algorithm.

The Certification Authority Public Key Check Sum is the technique specified in Part II of Book 4 of these specifications, to ensure that a Certification Authority Public Key and its related data is received error-free. The terminal may use this data element to subsequently re-verify the integrity of a Certification Authority Public Key and its related data. Alternately, the terminal may use another technique to ensure the integrity of this data.

The integrity of the stored Certification Authority Public Keys should be verified periodically.

Name	Length	Description	Format
Registered Application Provider Identifier (RID)	5	Identifies the payment system to which the Certification Authority Public Key is associated	b
Certification Authority Public Key Index	1	Identifies the Certification Authority Public Key in conjunction with the RID	b
Certification Authority Hash algorithm Indicator	1	Identifies the hash algorithm used to produce the Hash Result in the digital signature scheme	b
Certification Authority Public Key Algorithm Indicator	1	Identifies the digital signature algorithm to be used with the Certification Authority Public Key	b
Certification Authority Public Key Modulus	Var. (max 248)	Value of the modulus part of the Certification Authority Public Key	b
Certification Authority Public Key Exponent	1 or 3	Value of the exponent part of the Certification Authority Public Key, equal to 3 or $2^{16}+1$	b
Certification Authority Public Key Check Sum[10]	20	A check value calculated on the concatenation of all parts of the certification Authority Public Key (RID, Certification Authority Public Key Index, Certification Authority Public Key Modulus, Certification Authority Public Key Exponent) using SHA-1	b

Table 23 – Minimum Set of Certification Authority Public Key related Data Elements to be Stored in the Terminal

11.2.3 Certification Authority Public Key Usage

The usage of a Certification Authority Public Key during a transaction shall be as specified in this specification.

[10] Only necessary if used to verify the integrity of the Certification Authority Public Key

11.2.4 Certification Authority Public Key Withdrawal

When a payment system has decided to revoke one of its Certification Authority Public Keys, an acquirer must ensure that this Certification Authority Public Key can no longer be used in its terminals for static and dynamic data authentication during transactions as of a certain date.

The following principles apply for the withdrawal by an acquirer of Certification Authority Public Keys from its terminals:

- The terminal must be able to verify that it received the withdrawal notification error-free from the acquirer.
- The terminal must be able to verify that the received withdrawal notification originated from its legitimate acquirer.
- The acquirer must be able to confirm that a specific Certification Authority Public Key was indeed withdrawn correctly from its terminals.

For more details on Certification Authority Public Key revocation and the corresponding timescales involved, see Section 10 of this specification.

Annexes

Annex A - Security Mechanisms

A1. Symmetric Mechanisms

A1.1 Encipherment

Encipherment of data uses a 64-bit block cipher ALG either in Electronic Codebook (ECB) Mode or in Cipher Block Chaining (CBC) mode.

Encipherment of a message MSG of arbitrary length with Encipherment Session Key K_S takes place in the following steps.

1. *Padding and Blocking*

- If the message MSG has a length that is not a multiple of 8 bytes, add a '80' byte to the right of MSG, and then add the smallest number of '00' bytes to the right such that the length of resulting message MSG := (MSG || '80' || '00' || '00' || . . . || '00') is a multiple of 8 bytes.

- If the message MSG has a length that is a multiple of 8 bytes, the following two cases can occur depending on pre-defined rules (see section 9 of this specification).

  - No padding takes place: MSG := MSG.

  - MSG is padded to the right with the 8-byte block

    ('80' || '00' || '00' ||'00' || '00' || '00' || '00' || '00')

  to obtain MSG.

MSG is then divided into 8-byte blocks $X_1, X_2, \ldots, X_k$.

2. *Cryptogram Computation*

ECB Mode

Encipher the blocks $X_1, X_2, \ldots, X_k$ into the 8-byte blocks $Y_1, Y_2, \ldots, Y_k$ with the block cipher algorithm in ECB mode using the Encipherment Session Key K_S. Hence compute for i = 1, 2, . . . , k:

$$Y_i := ALG(K_S)[X_i].$$

CBC Mode

Encipher the blocks $X_1, X_2, \ldots, X_k$ into the 8-byte blocks $Y_1, Y_2, \ldots, Y_k$ with the block cipher algorithm in CBC mode using the Encipherment Session Key K_S. Hence compute for i = 1, 2, . . . , k:

$$Y_i := ALG(K_S)[X_i \oplus Y_{i-1}],$$

with initial value Y_0 := (‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’).

Notation:

$$Y := (Y_1 || Y_2 || \ldots || Y_k) = ENC(K_S)[MSG].$$

Decipherment is as follows.

1. *Cryptogram Decipherment*

ECB Mode

Compute for i = 1, 2, . . . , k:

$$X_i := ALG^{-1}(K_S)[Y_i].$$

CBC Mode

Compute for i = 1, 2, . . . , k:

$$X_i := ALG^{-1}(K_S)[Y_i] \oplus Y_{i-1},$$

with initial value Y_0 := (‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’ || ‘00’).

2. To obtain the original message MSG, concatenate the blocks $X_1, X_2, \ldots, X_k$ and if padding has been used (see above) remove the trailing (‘80’ || ‘00’ || ‘00’ || . . . || ‘00’) byte-string from the last block X_k.

Notation:

$$MSG = DEC(K_S)[Y].$$

A1.2 Message Authentication Code

The computation of an s-byte MAC ($4 \le s \le 8$) is according to ISO/IEC 9797-1 using a 64-bit block cipher ALG in CBC mode. More precisely, the computation of a MAC S over a message MSG consisting of an arbitrary number of bytes with a MAC Session Key K_S takes place in the following steps.

1. *Padding and Blocking*

Pad the message M according to ISO/IEC 7816-4 (which is equivalent to method 2 of ISO/IEC 9797-1), hence add a mandatory ‘80’ byte to the right of MSG, and then add the smallest number of ‘00’ bytes to the right such that the length of

resulting message $\underline{MSG}$:= (MSG || '80' || '00' || '00' || . . . || '00') is a multiple of 8 bytes.

$\underline{MSG}$ is then divided into 8-byte blocks $X_1, X_2, \ldots, X_k$.

2. *MAC Session Key*

The MAC Session Key K_S either consists of only a leftmost key block $K_S = K_{SL}$ or the concatenation of a leftmost and a rightmost key block $K_S = (K_{SL} || K_{SR})$.

3. *Cryptogram Computation*

Process the 8-byte blocks $X_1, X_2, \ldots, X_k$ with the block cipher in CBC mode using the leftmost MAC Session Key block K_{SL}:

$$H_i := ALG(K_{SL})[X_i \oplus H_{i-1}], \text{ for } i = 1, 2, \ldots, k.$$

with initial value H_0 := ('00' || '00' || '00' || '00' || '00' || '00' || '00' || '00').

Compute the 8 byte block H_{k+1} in one of the following two ways.

- According to ISO/IEC 9797-1 Algorithm 1:

$$H_{k+1} := H_k.$$

- According to ISO/IEC 9797-1 Algorithm 3:

$$H_{k+1} := ALG(K_{SL})[ALG^{-1}(K_{SR})[H_k]].$$

The MAC S is then equal to the s most significant bytes of H_{k+1}.

A1.3 Session Key Derivation

Session keys K_S for secure messaging for integrity and confidentiality are derived from unique Master Keys K_M using diversification data R provided by the receiving entity, hence

$$K_S := F(K_M)[R].$$

To prevent replay attacks, the diversification data R should either be unpredictable, or at least different for each session key derivation.

The only requirement for the diversification function F is that the number of possible outputs of the function is sufficiently large and uniformly distributed to prevent an exhaustive key search on the session key.

The remainder of this annex specifies a method for the derivation of session keys for Application Cryptogram generation, issuer authentication and secure messaging (see sections 8 and 9) from a ICC Master Key. The session key derivation method

ensures that the key used to derive the session key is only used a limited number of times.

Note that the session key derivation method provided in this annex is not mandatory. Issuers may decide to adopt another method for this function.

A1.3.1 Description

The session key derivation function takes as input the 16-byte ICC Master Key MK and the 2-byte ATC, and produces as output the 16-byte ICC Session Key SK.

The session key derivation function generates a unique session key for each ICC application transaction. It does this by generating a "tree" of keys. This tree has the ICC Master Key at its base and then numerous levels of intermediate keys above it, each intermediary key being derived from keys beneath it in the tree. On top of the tree are the session keys, one session key per value of the Application Transaction Counter (ATC).

The session key derivation function has two parameters:

- H, the height of the tree, i.e. the number of levels of intermediary keys in the tree excluding the base level;
- b, the branch factor, i.e. the number of "child" keys that a "parent" key (which must be one level lower in the tree) derives.

The number of keys at the i^{th} level is b^i, $0 \le i \le H$.

The number of possible session keys is b^H and this must exceed the maximum value of the ATC which is $2^{16} - 1$.

Let Φ be the function that maps two 16-byte numbers X and Y and an integer j onto a 16-byte number as follows:

$$Z = \Phi(X,Y,j) := (DES3(X)[Y_L \oplus (j \bmod b)] \,||\, DES3(X)[Y_R \oplus (j \bmod b) \oplus \text{'F0'}]),$$

where Y_L and Y_R are two 8-byte numbers and $Y = (Y_L \,||\, Y_R)$.

The reverse function Φ^{-1} of Φ is equal to

$$Y=\Phi^{-1}(X,Z,j) = ((DES3^{-1}(X)[Z_L] \oplus (j \bmod b)) \,||\, (DES3^{-1}(X)[Z_R] \oplus (j \bmod b) \oplus \text{'F0'})),$$

where Z_L and Z_R are two 8-byte numbers and $Z = (Z_L \,||\, Z_R)$.

define $IK_{0,0}$ as the ICC Master Key, hence $IK_{0,0} := MK$. This key is used to derive b intermediate keys at level 1 of the tree. For j = 0, …, b-1:

$$IK_{1,j} := \Phi(MK,IV,j),$$

where IV is a 16-byte initializing value, not necessarily secret.

An intermediate key in a higher level is derived from its parent and grandparent using the function Φ. Specifically the j^{th} key ($0 \le j \le b^i-1$) in level i ($2 \le i \le H$) is derived as

$$IK_{i,j} := \Phi(IK_{i-1,\, j/b},\, IK_{i-2,\, j/b^2}, j).$$

where "/" denotes integer division.

Let

$$X := IK_{H,\, ATC} \oplus IK_{H-2,\, ATC/b^2}.$$

The session key SK is defined to be X, with the exception of the least significant bit of each byte of X which is set to a value that ensures that each of the 16 bytes of SK has an odd number of nonzero bits (this to conform with the odd parity requirements for DES keys). Note that the forcing of the parity bits of DES keys only takes place for the session keys, but not for the intermediate keys.

A1.3.2 Implementation

It is recommended that b be equal to 2 or 4. Thus for a card limited to perform no more than 2^{16} transactions this would imply a tree with 16 and 8 levels respectively.

Below a straightforward implementation of the function is given in pseudo-code. In this implementation (a_0, a_1, . . . , a_{H-1}) denotes the b-ary representation of the ATC at the time of the transaction, hence

$$ATC = a_0 b^{H-1} + a_1 b^{H-2} + \ldots + a_{H-2} b + a_{H-1},$$

and GP and P denote grandparent and parent keys, respectively.

The computation of the session key SK from the ICC Master Key MK for the current value of the ATC takes place as follows.

```
GP=MK;
P=Φ(MK,IV,a0);
for (i=1;i<H-1;i++){
    T=P;
    P=Φ(P,GP,ai);
    GP=T;
}
SK=Φ(P,GP,aH-1)⊕GP;
```

The implementation above uses the MK each time a session key is derived. However an actual ICC implementation should not reuse the MK for each session key derivation, but should calculate the session keys from saved intermediate keys that are changed regularly. This will limit the usage of a specific key in the cryptographic operations. More details are given below.

In the session key derivation function, the derivation of intermediate level keys is reversible, i.e. given knowledge of an intermediate key and its parent it is possible to derive its grandparent. This means that a card which stores an intermediate key IK and its parent can then determine any other key in the tree, including any session key. Below an example is given in pseudo-code of an implementation of the session key derivation function using this property and ensuring that an intermediate key is only used a limited number of times.

In the pseudo-code below, at the beginning of the program P and GP denote the parent and grandparent keys that were used for the computation of the previous session key, and at the end they denote the parent and grandparent keys that were used for the computation of the current session key. For the computation of the first session key (ATC = 0), these values are initialised as

$$GP := IK_{H-2,\,0}, \qquad P := IK_{H-1,\,0}.$$

Let $(a_0, a_1, \ldots, a_{H-1})$ again denote the b-ary representation of the ATC at the time of the transaction, hence

$$ATC = a_0 b^{H-1} + a_1 b^{H-2} + \ldots + a_{H-2} b + a_{H-1},$$

and let $(c_0, c_1, \ldots, c_{H-1})$ denote the b-ary representation of the value ATC_{OLD} of the ATC at the time of the previous session key derivation:

$$ATC_{OLD} = c_0 b^{H-1} + c_1 b^{H-2} + \ldots + c_{H-2} b + c_{H-1}.$$

For ATC = 0, ATC_{OLD} is initialised as $ATC_{OLD} := 0$.

Let PAR(X) denote the function that sets the least significant bit of each byte of a 16-byte number X to odd parity.

The computation of the session key SK for the current value of the ATC takes place as follows.

```
/* determination of the common node for ATC and ATC_OLD */
i=0;
while ((a_i==c_i)&& (i<H-1))
      i++;

/* computation of the new GP and P for the current ATC */
if (i==0) {
   p=Φ(GP,IV,a_0);
   i=1;
else {
   for (j=H-2;j>=i;j--) {
      T=GP;
      GP=Φ^-1(GP,P,c_j);
      P=T;
   }
```

```
}
while (i<H-1) {
      T=P;
      P=Φ(P,GP,a_i);
      GP=T;
      i++;
}
/* computation of the session key */
SK=PAR(Φ(P,GP,a_H-1)⊕GP);
ATC_OLD=ATC;
```

The algorithm above can be made more efficient by storing more than 2 intermediate keys. This however requires more memory.

A1.4 Master Key Derivation

This annex specifies a method for the derivation by the issuer of a 16-byte ICC Master Key used for Application Cryptogram generation, issuer authentication and secure messaging.

Note that this method is not mandatory. Issuers may decide to adopt another method for this function.

The method takes as input the PAN and PAN Sequence Number, a 16-byte Issuer Master Key IMK, and produces the 16-byte ICC Master Key MK in the following way:

1. Concatenate from left to right the Application PAN with the PAN Sequence Number (if the PAN Sequence Number is not present, then it is replaced by a '00' byte). If the result X is less than 16 digits long, pad it to the left with hexadecimal zeros in order to obtain an 8-byte number Y in numeric format. If X is at least 16 digits long, then Y consists of the 16 rightmost digits of X in numeric format.

2. Compute the two 8-byte numbers

$$Z_L := \text{DES3(IMK)[Y]}$$

and

$$Z_R := \text{DES3(IMK)}[Y \oplus (\text{'FF'}||\text{'FF'}||\text{'FF'}||\text{'FF'}||\text{'FF'}||\text{'FF'}||\text{'FF'}||\text{'FF'})],$$

and define

$$Z := (Z_L \,||\, Z_R).$$

The 16-byte ICC Master Key MK is then equal to Z, with the exception of the least ssgnificant bit of each byte of Z which is set to a value that ensures that each of the 16 bytes of MK has an odd number of nonzero bits (this to conform with the odd parity requirements for DES keys).

A2. Asymmetric Mechanisms

A2.1 Digital Signature Scheme Giving Message Recovery

This section describes the special case of the digital signature scheme giving message recovery using a hash function according to ISO/IEC 9796-2, which is used in this specification for both static and dynamic data authentication.

A2.1.1 Algorithms

The digital signature scheme uses the following two types of algorithms.

- A reversible asymmetric algorithm consisting of a signing function $Sign(S_K)[\ ]$ depending on a Private Key S_K and a recovery function $Recover(P_K)[\ ]$ depending on a Public Key P_K. Both functions map N-byte numbers onto N-byte numbers and have the property that

$$Recover(P_K)[Sign(S_K)[X]] = X,$$

for any N-byte number X.

- A hashing algorithm Hash[ ] that maps a message of arbitrary length onto an 20-byte hash code.

A2.1.2 Signature Generation

The computation of a signature S on a message MSG consisting of an arbitrary number L of at least N – 21 bytes takes place in the following way.

1. Compute the 20-byte hash value H := Hash[MSG] of the message M.

2. Split MSG into two parts MSG =(MSG_1 || MSG_2), where MSG_1 consists of the N – 22 leftmost (most significant bytes) of MSG and MSG_2 of the remaining (least significant) L – N + 22 bytes of MSG.

3. Define the byte B := '6A'.

4. Define the byte E := 'BC'.

5. Define the N-byte block X as the concatenation of the blocks B, MSG_1, H and E, hence

$$X := (B \,||\, MSG_1 \,||\, H \,||\, E).$$

6. The digital signature S is then defined as the N-byte number

$$S := Sign(S_K)[X].$$

A2.1.3 Signature Verification

The corresponding signature verification takes place in the following way:

1. Check whether the digital signature S consists of N bytes.
2. Retrieve the N-byte number X from the digital signature S:

$$X = \text{Recover}(P_K)[S].$$

3. Partition X as X= (B || MSG_1 || H || E), where
   - B is one byte long.
   - H is 20 bytes long.
   - E is one byte long.
   - MSG_1 consists of the remaining $N - 22$ bytes.
4. Check whether the byte B is equal to '6A'.
5. Check whether the byte E is equal to 'BC'.
6. Compute MSG = (MSG_1 || MSG_2) and check whether H = Hash[MSG].

If and only if these checks are correct is the message accepted as genuine.

THIS PAGE LEFT INTENTIONALLY BLANK

Annex B - Approved Cryptographic Algorithms

B1. Symmetric Algorithms

B1.1 Data Encryption Standard (DES)

The double-length key triple DES encipherment algorithm (see clause 4.2 of ISO 11568-2) is the approved cryptographic algorithm to be used in the encipherment and MAC mechanisms specified in Annex A1. The algorithm is based on the (single-) DES algorithm standardised in ISO 8731-1.

Triple DES encipherment involves enciphering an 8-byte plaintext block in an 8-byte ciphertext block with a double-length (16-byte) secret key $K = (K_L \,||\, K_R)$ as follows:

$$Y = DES3(K)[X] = DES(K_L)[DES^{-1}(K_R)[DES(K_L)[X]]].$$

Decipherment takes place as follows:

$$X = DES^{-1}(K_L)[DES(K_R)[DES^{-1}(K_L)[Y]]].$$

Single DES is only approved for usage with the version of the MAC mechanism specified in Annex A1 using Algorithm 3 of ISO 9797-1 (triple DES applied to the last block).

B2. Asymmetric Algorithms

B2.1 RSA Algorithm

This reversible algorithm (see informative reference [2]) is the approved algorithm for encipherment and digital signature generation as described in Annex A2.. The only values allowed for the public key exponent are 3 and $2^{16}+1$

The algorithm produces a cryptogram or digital signature whose length equals the size of the modulus used. The mandatory upper bounds for the size of the modulus are specified in Table 18.

Description	Max. Length
Certification Authority Public Key Modulus	248 bytes
Issuer Public Key Modulus	248 bytes
ICC Public Key Modulus	248 bytes
ICC PIN Encipherment Public Key Modulus	248 bytes

Table 24 - Mandatory Upper Bound for the Size in Bytes of the Moduli

Furthermore, the length N_{CA} of the Certification Authority Public Key Modulus, the length N_I of the Issuer Public Key Modulus, the length N_{IC} of the ICC Public Key Modulus and the length N_{PE} of the ICC PIN Encipherment Public Key Modulus shall satisfy $N_{IC} \leq N_I \leq N_{CA}$ and $N_{PE} \leq N_I \leq N_{CA}$.

In the choice of the lengths of the public key moduli, one should take into account the life-time of the keys compared to the expected progress in factoring during that life-time. The ranges (upper and lower bounds) for the key lengths mandated by each of the payment systems are specified in their corresponding proprietary specifications.

The value of the Issuer Public Key Exponent and the ICC Public Key Exponent is determined by the issuer. The Certification Authority, Issuer and ICC Public Key Exponents shall be equal to 3 or $2^{16} + 1$

The Public Key Algorithm Indicator for this digital signature algorithm shall be coded as hexadecimal '01'.

The keys and signing and recovery functions for the RSA algorithm with odd public key exponent are specified below.

B2.1.1 Keys

The private key S_K of the RSA digital signature scheme with an odd public key exponent e consists of two prime numbers p and q such that p – 1 and q – 1 are co-prime to e and a private exponent d such that

$$ed \equiv 1 \bmod (p - 1)(q - 1).$$

The corresponding public key P_K consists of the public key modulus n = pq and the public key exponent e.

B2.1.2 Signing Function

The signing function for RSA with an odd public key exponent is defined as

$$S = Sign(S_K)[X] := X^d \bmod n,\ 0 < X < n,$$

where X is the data to be signed and S the corresponding digital signature.

B2.1.3 Recovery Function

The recovery function for RSA with an odd public key exponent is equal to

$$X = Recover(P_K)[S] := S^e \bmod n.$$

B2.1.4 Key Generation

Payment systems and issuers shall be responsible for the security of their respective RSA public/private key generation processes. Examples of secure key generation methods can be found in reference [1] in Annex C.

B3. Hashing Algorithms

B3.1 Secure Hash Algorithm (SHA-1)

This algorithm is standardised as FIPS 180-1.[11] SHA-1 takes as input messages of arbitrary length and produces a 20-byte hash value.

The Hash Algorithm Indicator for this hashing algorithm shall be coded as hexadecimal '01'.

[11] SHA-1 is also standardised in ISO/IEC 10118-3.

THIS PAGE LEFT INTENTIONALLY BLANK.

Annex C - Informative References

1. A. Bosselaers and B. Preneel (eds.), *Integrity Primitives for Secure Information Systems*, Final Report of the RACE Integrity Primitives Evaluation (RIPE, RACE R1040), LNCS 1007, Springer-Verlag, 1995.

2. R. L. Rivest, A. Shamir, and L. Adleman, 'A method for obtaining digital signatures and public key cryptosystems,' *Communications of the ACM*, vol. 21, 1978, pp. 120-126.