Council Regulation (EEC) No 3821/85 of 20 December 1985 on recording equipment in road transport

[F1[F2ANNEX I B

REQUIREMENTS FOR CONSTRUCTION, TESTING, INSTALLATION AND INSPECTION

Textual Amendments

F1 Inserted by Council Regulation (EC) No 2135/98 of 24 September 1998 amending Regulation (EEC) No 3821/85 on recording equipment in road transport and Directive 88/599/EEC concerning the application of Regulations (EEC) No 3820/85 and (EEC) No 3821/85.

F2 Substituted by Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Text with EEA relevance).

Appendix 11

COMMON SECURITY MECHANISMS

1. GENERALITIES

This appendix specifies the security mechanisms ensuring:

— the mutual authentication between VUs and tachograph cards, including session key agreement,

— the confidentiality, integrity and authentication of data transferred between VUs and tachograph cards,

— the integrity and authentication of data downloaded from VUs to external storage media,

— the integrity and authentication of data downloaded from tachograph cards to external storage media.

1.1. References

The following references are used in this Appendix:

SHA-1	National Institute of Standards and Technology (NIST). FIPS Publication 180-1: Secure Hash Standard. April 1995
PKCS1	RSA Laboratories. PKCS # 1: RSA Encryption Standard. Version 2.0. October 1998
TDES	National Institute of Standards and Technology (NIST). FIPS Publication 46-3: Data Encryption Standard. Draft 1999
TDES-OP	ANSI X9.52, Triple Data Encryption Algorithm Modes of Operation. 1998
ISO/IEC 7816-4	Information Technology — Identification cards — Integrated circuit(s) cards with contacts — Part 4: Interindustry commands for interexchange. First edition: 1995 + Amendment 1: 1997
ISO/IEC 7816-6	Information Technology — Identification cards — Integrated circuit(s) cards with contacts — Part 6: Interindustry data elements. First edition: 1996 + Cor 1: 1998
ISO/IEC 7816-8	Information Technology — Identification cards — Integrated circuit(s) cards with contacts — Part 8: Security related interindustry commands. First edition 1999
ISO/IEC 9796-2	Information Technology — Security techniques — Digital signature schemes giving message recovery — Part 2: Mechanisms using a hash function. First edition: 1997
ISO/IEC 9798-3	Information Technology — Security techniques — Entity authentication mechanisms — Part 3: Entity authentication using a public key algorithm. Second edition 1998
ISO 16844-3	Road vehicles — Tachograph systems — Part 3: Motion sensor interface.

1.2. Notations and abbreviated terms

The following notations and abbreviated terms are used in this Appendix:

(K_a, K_b, K_c)	a key bundle for use by the triple data encryption algorithm
CA	Certification authority
CAR	Certification authority reference
CC	Cryptographic checksum

CG	Cryptogram
CH	Command header
CHA	Certificate holder authorisation
CHR	Certificate holder reference
D()	Decryption with DES
DE	Data element
DO	Data object
d	RSA private key, private exponent
e	RSA public key, public exponent
E()	Encryption with DES
EQT	Equipment
Hash()	hash value, an output of *hash*
Hash	hash function
KID	Key identifier
Km	TDES key. Master Key defined in ISO 16844-3
$\mathrm{Km}_{\mathrm{vu}}$	TDES key inserted in vehicle units
$\mathrm{Km}_{\mathrm{wc}}$	TDES key inserted in workshop cards
m	message representative an integer between 0 and *n*-1
n	RSA keys, modulus
PB	Padding bytes
PI	Padding indicator byte (for use in cryptogram for confidentiality DO)
PV	Plain value
s	signature representative, an integer between 0 and *n*-1
SSC	Send sequence counter
SM	Secure messaging
TCBC	TDEA cipher block chaining mode of operation
TDEA	Triple data encryption algorithm
TLV	Tag length value
VU	Vehicle unit
X.C	the certificate of user X issued by a certification authority
X.CA	a certification authority of user X
$\mathrm{X.CA.PK}_{\mathrm{o}}\mathrm{X.C}$	the operation of unwrapping a certificate to extract a public key. It is an infix operator, whose left operand is the public key of a certification authority, and whose right operand is the certificate issued by that certification authority. The outcome is the public key of the user X whose certificate is the right operand,
X.PK	public key of a user X
X.PK[I]	RSA encipherment of some information I, using the public key of user X
X.SK	RSA private key of a user X
X.SK[I]	RSA encipherment of some information I, using the private key of user X
'xx'	a Hexadecimal value
||	concatenation operator.

2. CRYPTOGRAPHIC SYSTEMS AND ALGORITHMS

2.1. Cryptographic systems

Vehicle units and tachograph cards shall use a classical RSA public-key cryptographic system to provide the following security mechanisms:

— authentication between vehicle units and cards,

— transport of Triple-DES session keys between vehicle units and tachograph cards,

— digital signature of data downloaded from vehicle units or tachograph cards to external media.

Vehicle units and tachograph cards shall use a Triple DES symmetric cryptographic system to provide a mechanism for data integrity during user data exchange between vehicle units and tachograph cards, and to provide, where applicable, confidentiality of data exchange between vehicle units and tachograph cards.

2.2. Cryptographic algorithms

2.2.1. RSA algorithm

The RSA algorithm is fully defined by the following relations:

$$\mathrm{X.SK}[m] = s = m^{d} \bmod n$$

$$\mathrm{X.PK}[s] = m = s^{e} \bmod n$$

A more comprehensive description of the RSA function can be found in reference (PKCS1).

[[F3]Public exponent, e, for RSA calculations is an integer between 3 and n-1 satisfying gcd(e, lcm(p-1, q-1))=1.**]**

Textual Amendments

F3 Substituted by Commission Regulation (EC) No 432/2004 of 5 March 2004 adapting for the eighth time to technical progress Council Regulation (EEC) No 3821/85 of 20 December 1985 on recording equipment in road transport (Text with EEA relevance).

2.2.2. Hash algorithm

The digital signature mechanisms shall use the SHA-1 hash algorithm as defined in reference (SHA-1).

2.2.3. Data encryption algorithm

DES based algorithms shall be used in Cipher Block Chaining mode of operation.

3. KEYS AND CERTIFICATES

3.1. Keys generation and distribution

3.1.1. RSA keys generation and distribution

RSA keys shall be generated through three functional hierarchical levels:

- European level,
- Member State level,
- Equipment level.

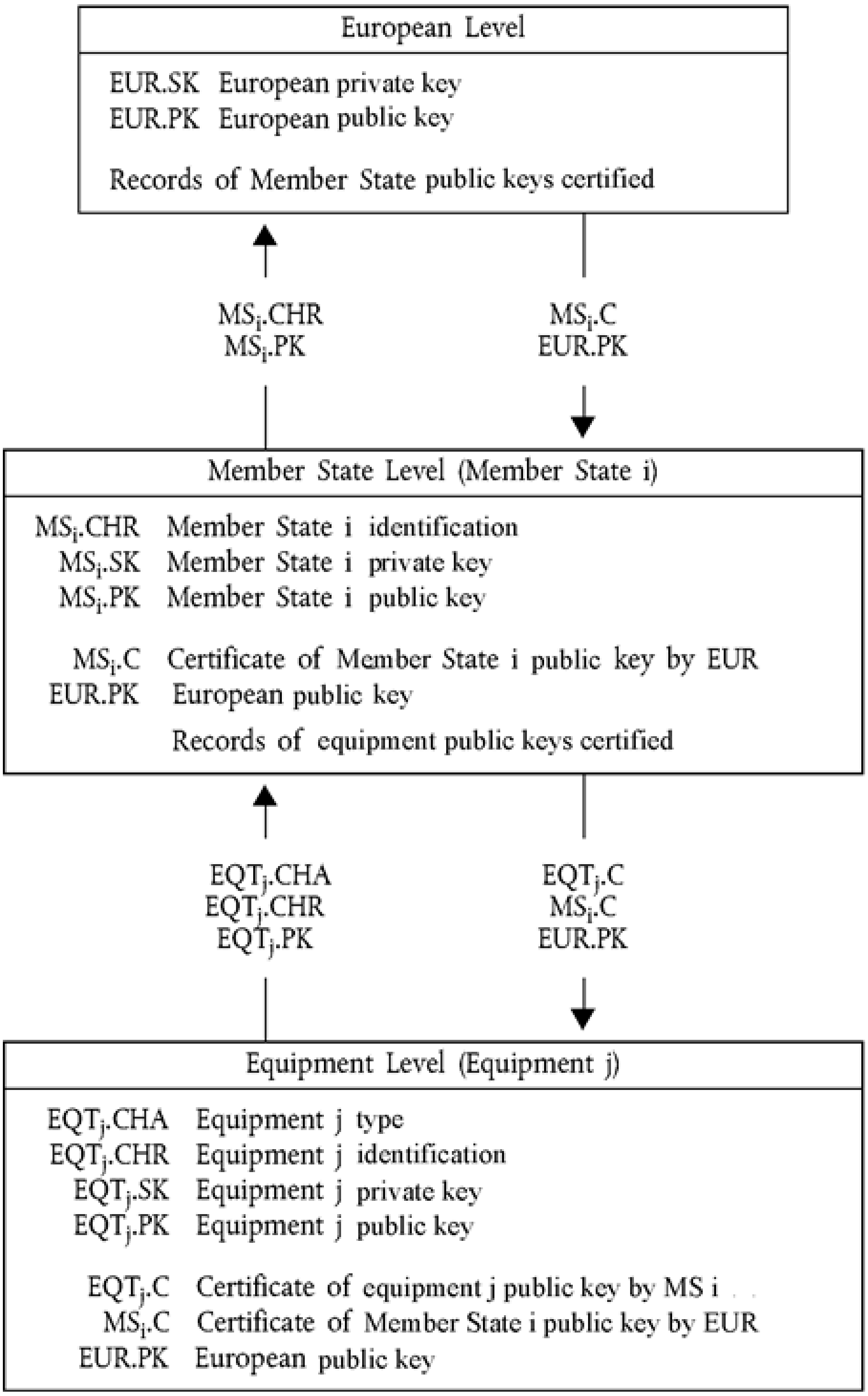
At European level, a single European key pair (EUR.SK and EUR.PK) shall be generated. The European private key shall be used to certify the Member States public keys. Records of all certified keys shall be kept. These tasks shall be handled by a European certification authority, under the authority and responsibility of the European Commission.

At Member State level, a Member State key pair (MS.SK and MS.PK) shall be generated. Member States public keys shall be certified by the European Certification Authority. The Member State private key shall be used to certify public keys to be inserted in equipment (vehicle unit or tachograph card). Records of all certified public keys shall be kept with the identification of the equipment to which it is intended. These tasks shall be handled by a Member State certification authority. A Member State may regularly change its key pair.

At equipment level, one single key pair (EQT.SK and EQT.PK) shall be generated and inserted in each equipment. Equipment public keys shall be certified by a Member State certification authority. These tasks may be handled by equipment manufacturers, equipment personalisers or Member State authorities. This key pair is used for authentication, digital signature and encipherement services

Private keys confidentiality shall be maintained during generation, transport (if any) and storage.

The following picture summarises the data flow of this process:


European Level
EUR.SK European private key
EUR.PK European public key
Records of Member State public keys certified
MS$_i$.CHR
MS$_i$.PK
MS$_i$.C
EUR.PK
Member State Level (Member State i)
MS$_i$.CHR Member State i identification
MS$_i$.SK Member State i private key
MS$_i$.PK Member State i public key
MS$_i$.C Certificate of Member State i public key by EUR
EUR.PK European public key
Records of equipment public keys certified
EQT$_j$.CHA
EQT$_j$.CHR
EQT$_j$.PK
EQT$_j$.C
MS$_i$.C
EUR.PK
Equipment Level (Equipment j)
EQT$_j$.CHA Equipment j type
EQT$_j$.CHR Equipment j identification
EQT$_j$.SK Equipment j private key
EQT$_j$.PK Equipment j public key
EQT$_j$.C Certificate of equipment j public key by MS i
MS$_i$.C Certificate of Member State i public key by EUR
EUR.PK European public key

3.1.2. RSA test keys

For the purpose of equipment testing (including interoperability tests) the European certification authority shall generate a different single European test key pair and at least two Member State test key pairs, the public keys of which shall be certified with the European private test key. Manufacturers shall insert, in equipment undergoing type approval tests, test keys certified by one of these Member State test keys.

3.1.3. Motion sensor keys

The confidentiality of the three TDES keys described below shall be appropriately maintained during generation, transport (if any) and storage.

In order to support recording equipment compliant with ISO 16844, the European certification authority and the Member State certification authorities shall, in addition, ensure the following:

The European certification authority shall generate Km_{VU} and Km_{WC}, two independent and unique Triple DES keys, and generate Km as:

$$\boxed{Km = Km_{VU} \text{ XOR } Km_{WC}}$$

The European certification authority shall forward these keys, under appropriately secured procedures, to Member States certification authorities at their request.

Member States certification authorities shall:

— use Km to encrypt motion sensor data requested by motion sensor manufacturers (data to be encrypted with Km is defined in ISO 16844-3),

— forward Km_{VU} to vehicle unit manufacturers, under appropriately secured procedures, for insertion in vehicle units,

— ensure that Km_{WC} will be inserted in all workshop cards (*SensorInstallationSecData* in *Sensor_Installation_Data* elementary file) during card personalisation.

3.1.4. T-DES session keys generation and distribution

Vehicle units and tachograph cards shall, as a part of the mutual authentication process, generate and exchange necessary data to elaborate a common Triple DES session key. This exchange of data shall be protected for confidentiality through an RSA crypt-mechanism.

This key shall be used for all subsequent cryptographic operations using secure messaging. Its validity shall expire at the end of the session (withdrawal of the card or reset of the card) and/or after 240 use (one use of the key = one command using secure messaging sent to the card and associated response).

3.2. Keys

RSA keys shall have (whatever the level) the following lengths: modulus *n* 1024 bits, public exponent *e* 64 bits maximum, private exponent *d* 1024 bits.

Triple DES keys shall have the form (K_a, K_b, K_a) where K_a and K_b are independent 64 bits long keys. No parity error detecting bits shall be set.

3.3. Certificates

RSA Public key certificates shall be 'non self-descriptive''Card Verifiable' certificates (Ref.: ISO/IEC 7816-8)

3.3.1. Certificates content

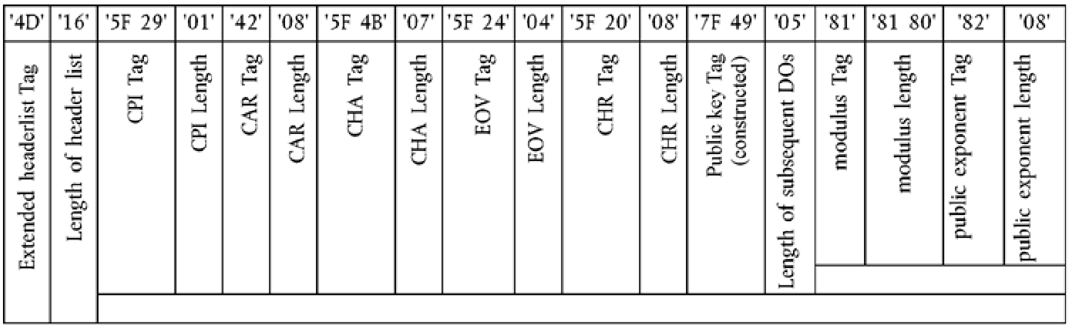
RSA Public key certificates are built with the following data in the following order:

Data	Format	Bytes	Obs
CPI	INTEGER	1	Certificate profile identifier ('01' for this version)
CAR	OCTET STRING	8	Certification authority reference
CHA	OCTET STRING	7	Certificate holder authorisation
EOV	TimeReal	4	Certificate end of validity. Optional, 'FF ' padded if not used
CHR	OCTET STRING	8	Certificate holder reference
n	OCTET STRING	128	Public key (modulus)
e	OCTET STRING	8	Public key (public exponent)
		164	

Notes:

1. The 'Certificate Profile Identifier' (CPI) delineates the exact structure of an authentication certificate. It can be used as an equipment internal identifier of a relevant headerlist which describes the concatenation of Data Elements within the certificate.

The headerlist associated with this certificate content is as follows:



'4D'	'16'	'5F 29'	'01'	'42'	'08'	'5F 4B'	'07'	'5F 24'	'04'	'5F 20'	'08'	'7F 49'	'05'	'81'	'81 80'	'82'	'08'
Extended headerlist Tag	Length of header list	CPI Tag	CPI Length	CAR Tag	CAR Length	CHA Tag	CHA Length	EOV Tag	EOV Length	CHR Tag	CHR Length	Public key Tag (constructed)	Length of subsequent DOs	modulus Tag	modulus length	public exponent Tag	public exponent length

2. The 'Certification Authority Reference' (CAR) has the purpose of identifying the certificate issuing CA, in such a way that the data element can be used at the same time as an authority key identifier to reference the public key of the certification authority (for coding, see Key Identifier below).

3. The ‘Certificate Holder Authorisation’ ((CHA) is used to identify the rights of the certificate holder. It consists of the Tachograph Application ID and of the type of equipment to which the certificate is intended (according to EquipmentType data element, ‘00’ for a Member State).

4. ‘Certificate Holder Reference’ (CHR) has the purpose of identifying uniquely the certificate holder, in such a way that the Data Element can be used at the same time as a Subject Key Identifier to reference the Public Key of the certificate holder.

5. Key Identifiers uniquely identify certificate holder or certification authorities. They are coded as follows:

5.1. Equipment (VU or Card):

Data	**Equipment serial number**	**Date**	**Type**	**Manufacturer**
Length	4 Bytes	2 Bytes	1 Byte	1 Byte
Value	Integer	mm yy BCD coding	Manufacturer specific	Manufacturer code

In the case of a VU, the manufacturer, when requesting certificates, may or may not know the identification of the equipment in which the keys will be inserted.

In the first case, the manufacturer will send the equipment identification with the public key to its Member State authority for certification. The certificate will then contain the equipment identification, and the manufacturer must ensure that keys and certificate are inserted in the intended equipment. The Key identifier has the form shown above.

In the later case, the manufacturer must uniquely identify each certificate request and send this identification with the public key to its Member State authority for certification. The certificate will contain the request identification. The manufacturer must feed back its Member State authority with the assignment of key to equipment (i.e. certificate request identification, equipment identification) after key installation in the equipment. The key identifier has the following form:

Data	**Certificate request serial number**	**Date**	**Type**	**Manufacturer**
Length	4 Bytes	2 Bytes	1 Byte	1 Byte
Value	[F3Integer]	[X1mm yy BCD coding]	′FF′	Manufacturer code

5.2. Certification Authority:

Data	**Authority identification**	**Key serial number**	**Additional info**	**Identifier**
Length	4 Bytes	1 Byte	2 Bytes	1 Byte

Value	1 Byte nation numerical code	Integer	additional coding (CA specific)	'01'
	3 Bytes nation alphanumerical code		'FF FF' if not used	

The key serial number is used to distinguish the different keys of a Member State, in the case the key is changed.

Editorial Information

X1 Substituted by Corrigendum to Commission Regulation (EC) No 1360/2002 of 13 June 2002 adapting for the seventh time to technical progress Council Regulation (EEC) No 3821/85 on recording equipment in road transport (Official Journal of the European Communities L 207 of 5 August 2002).

6. Certificate verifiers shall implicitly know that the public key certified is an RSA key relevant to authentication, digital signature verification and encipherement for confidentiality services (the certificate contains no Object Identifier to specify it).

3.3.2. Certificates issued

The certificate issued is a digital signature with partial recovery of the certificate content in accordance with ISO/IEC 9796-2 [F4except for its Annex A.4], with the 'Certification Authority Reference' appended.

Textual Amendments

F4 Inserted by Commission Regulation (EC) No 432/2004 of 5 March 2004 adapting for the eighth time to technical progress Council Regulation (EEC) No 3821/85 of 20 December 1985 on recording equipment in road transport (Text with EEA relevance).

$$\boxed{X.C = X.CA.SK['6A' \,||\, C_r \,||\, \mathit{Hash}(Cc) \,||\, 'BC'] \,||\, C_n \,||\, X.CAR}$$

With certificate content

$$= Cc = \underset{106\ \text{Bytes}}{C_r} \quad || \quad \underset{58\ \text{Bytes}}{C_n}$$

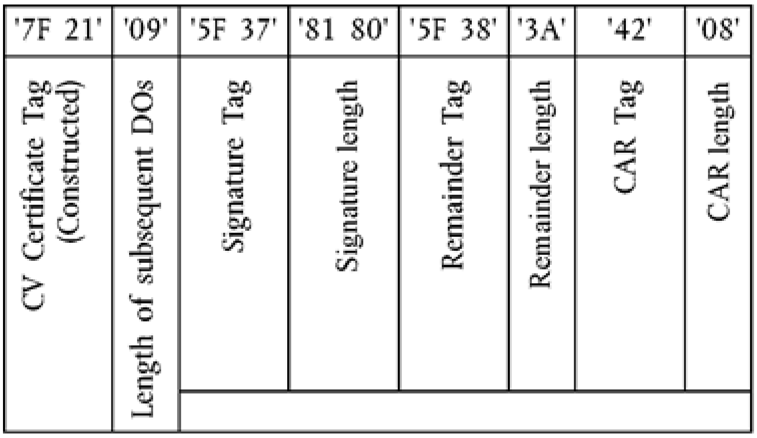
Notes:

1. This certificate is 194 bytes long.

2. CAR, being hidden by the signature, is also appended to the signature, such that the public key of the certification authority may be selected for the verification of the certificate.

3. The certificate verifier shall implicitly know the algorithm used by the certification authority to sign the certificate.

4. The headerlist associated with this issued certificate is as follows:



'7F 21'	'09'	'5F 37'	'81 80'	'5F 38'	'3A'	'42'	'08'
CV Certificate Tag (Constructed)	Length of subsequent DOs	Signature Tag	Signature length	Remainder Tag	Remainder length	CAR Tag	CAR length

3.3.3. Certificate verification and unwrapping

Certificate verification and unwrapping consists in verifying the signature in accordance with ISO/IEC 9796-2, retrieving the certificate content and the public key contained: X.PK = X.CA.PK$_o$X.C, and verifying the validity of the certificate.

It involves the following steps:

verify signature and retrieve content:

— from X.C retrieve Sign, C$_n$′ and CAR′:

— from CAR′ select appropriate Certification Authority Public Key (if not done before through other means)

— open Sign with CA Public Key: Sr′ = X.CA.PK [Sign],

— check Sr′ starts with ′6A′ and ends with ′BC′

— compute Cr′ and H′ from:

— Recover certificate content C′ = C$_r$′ || C$_n$′,

— check *Hash*(C′) = H′

If the checks are OK the certificate is a genuine one, its content is C′.

Verify validity. From C′:

— if applicable, check End of validity date,

Retrieve and store public key, Key Identifier, Certificate Holder Authorisation and Certificate End of Validity from C′:

— X.PK = *n* || *e*

— X.KID = CHR

— X.CHA = CHA

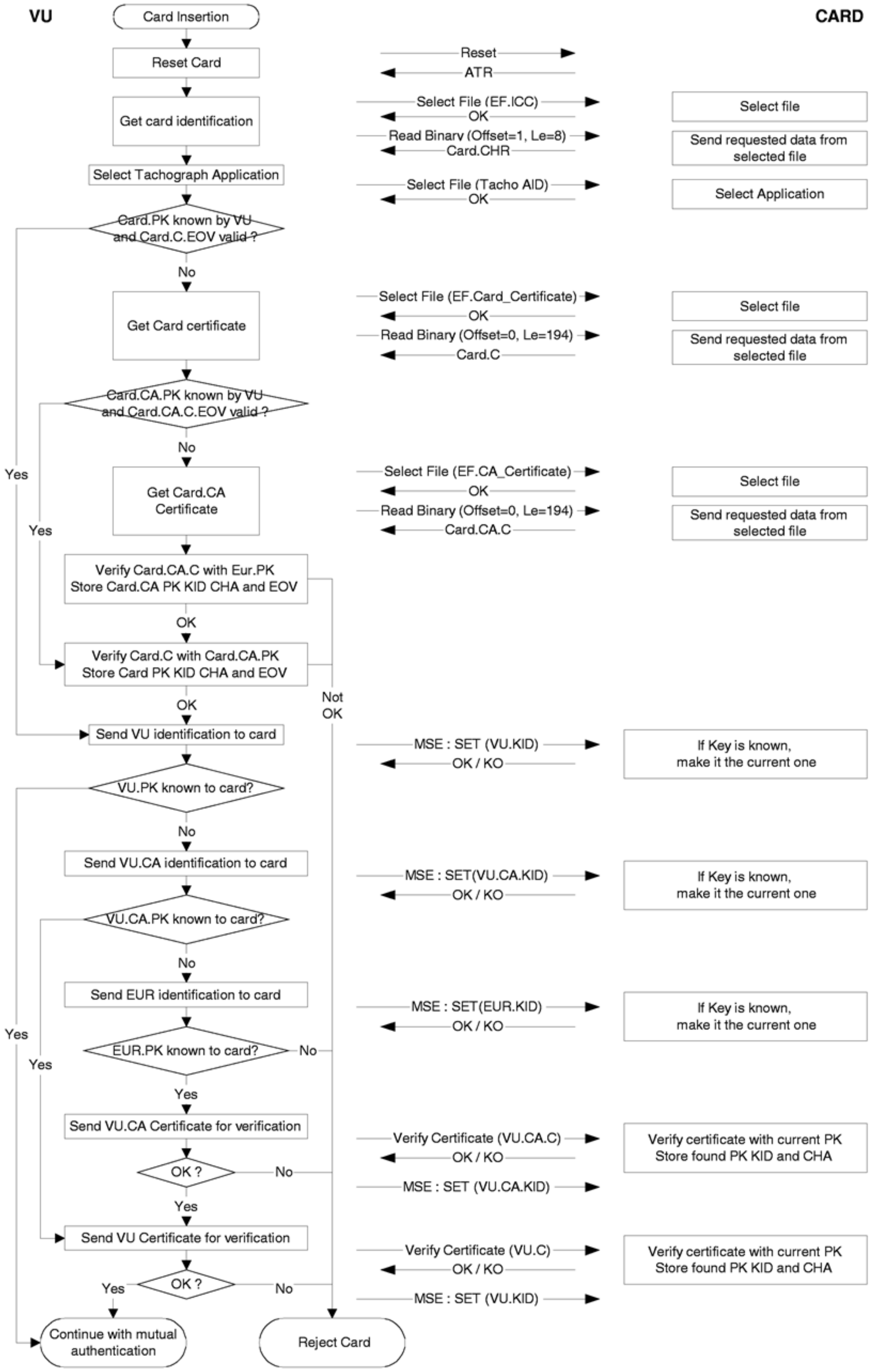
— X.EOV = EOV.

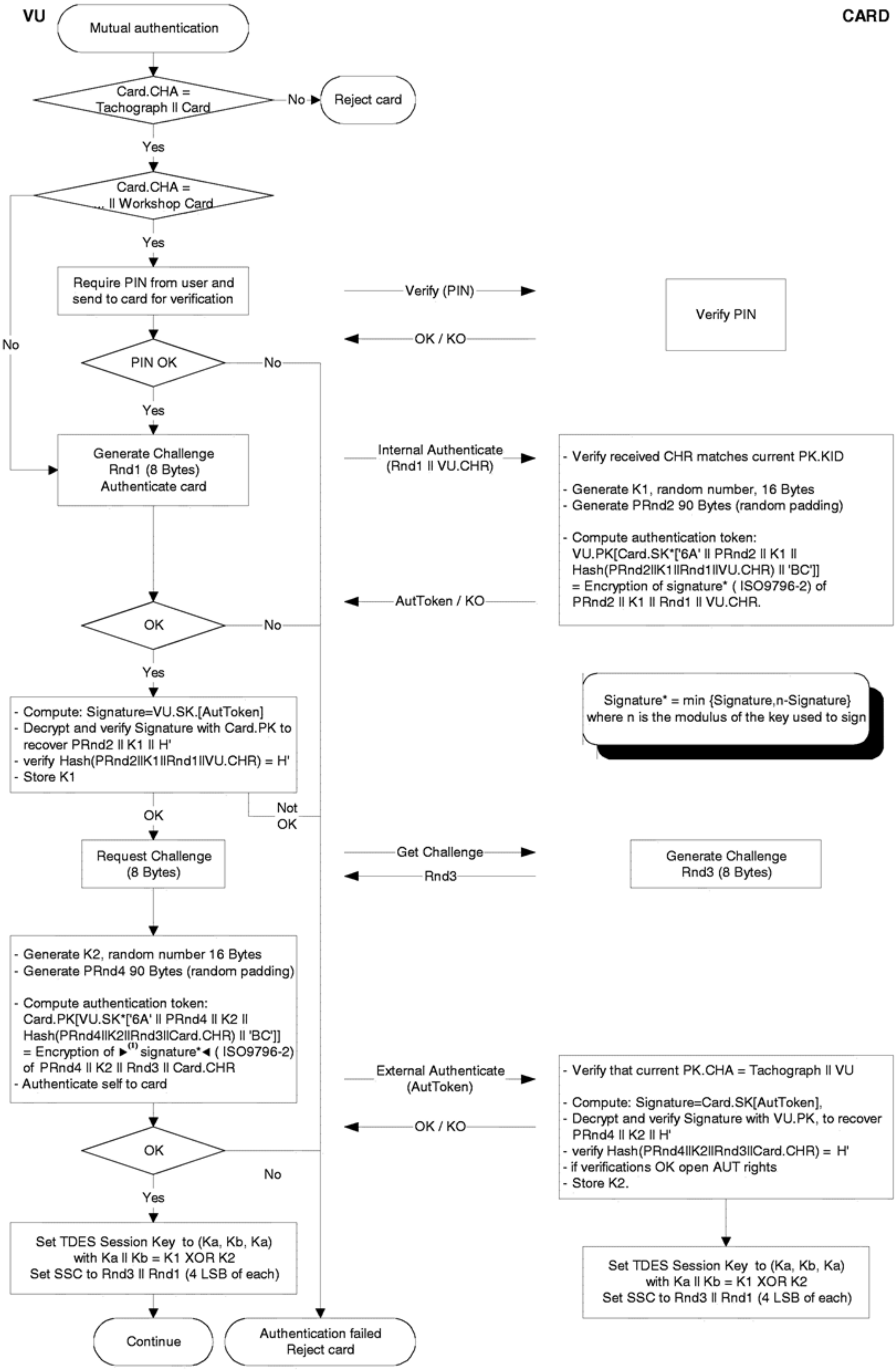
4. MUTUAL AUTHENTICATION MECHANISM

Mutual authentication between cards and VUs is based on the following principle:

> Each party shall demonstrate to the other that it owns a valid key pair, the public key of which has been certified by a Member State certification authority, itself being certified by the European certification authority.
>
> Demonstration is made by signing with the private key a random number sent by the other party, who must recover the random number sent when verifying this signature.
>
> The mechanism is triggered at card insertion by the VU. It starts with the exchange of certificates and unwrapping of public keys, and ends with the setting of a session key.

The following protocol shall be used (arrows indicate commands and data exchanged (see Appendix 2)):


VU
CARD
Card Insertion
Reset Card
Reset
ATR
Get card identification
Select File (EF.ICC)
OK
Select file
Read Binary (Offset=1, Le=8)
Card.CHR
Send requested data from selected file
Select Tachograph Application
Select File (Tacho AID)
OK
Select Application
Card.PK known by VU and Card.C.EOV valid ?
No
Get Card certificate
Select File (EF.Card_Certificate)
OK
Select file
Read Binary (Offset=0, Le=194)
Card.C
Send requested data from selected file
Card.CA.PK known by VU and Card.CA.C.EOV valid ?
No
Yes
Get Card.CA Certificate
Select File (EF.CA_Certificate)
OK
Select file
Read Binary (Offset=0, Le=194)
Card.CA.C
Send requested data from selected file
Yes
Verify Card.CA.C with Eur.PK Store Card.CA PK KID CHA and EOV
OK
Verify Card.C with Card.CA.PK Store Card PK KID CHA and EOV
OK
Not OK
Send VU identification to card
MSE : SET (VU.KID)
OK / KO
If Key is known, make it the current one
VU.PK known to card?
No
Send VU.CA identification to card
MSE : SET(VU.CA.KID)
OK / KO
If Key is known, make it the current one
VU.CA.PK known to card?
No
Send EUR identification to card
MSE : SET(EUR.KID)
OK / KO
If Key is known, make it the current one
Yes
Yes
EUR.PK known to card?
No
Yes
Send VU.CA Certificate for verification
Verify Certificate (VU.CA.C)
OK / KO
Verify certificate with current PK Store found PK KID and CHA
OK ?
No
MSE : SET (VU.CA.KID)
Yes
Send VU Certificate for verification
Verify Certificate (VU.C)
OK / KO
Verify certificate with current PK Store found PK KID and CHA
Yes
OK ?
No
MSE : SET (VU.KID)
Continue with mutual authentication
Reject Card


VU
Mutual authentication
CARD
Card.CHA = Tachograph II Card
No
Reject card
Yes
Card.CHA = ... II Workshop Card
Yes
No
Require PIN from user and send to card for verification
Verify (PIN)
Verify PIN
OK / KO
PIN OK
No
Yes
Generate Challenge Rnd1 (8 Bytes) Authenticate card
Internal Authenticate (Rnd1 II VU.CHR)
- Verify received CHR matches current PK.KID
- Generate K1, random number, 16 Bytes
- Generate PRnd2 90 Bytes (random padding)
- Compute authentication token: VU.PK[Card.SK*['6A' II PRnd2 II K1 II Hash(PRnd2IIK1IIRnd1IIVU.CHR) II 'BC']] = Encryption of signature* (ISO9796-2) of PRnd2 II K1 II Rnd1 II VU.CHR.
AutToken / KO
OK
No
Yes
Signature* = min {Signature,n-Signature} where n is the modulus of the key used to sign
- Compute: Signature=VU.SK.[AutToken]
- Decrypt and verify Signature with Card.PK to recover PRnd2 II K1 II H'
- verify Hash(PRnd2IIK1IIRnd1IIVU.CHR) = H'
- Store K1
OK
Not OK
Request Challenge (8 Bytes)
Get Challenge
Rnd3
Generate Challenge Rnd3 (8 Bytes)
- Generate K2, random number 16 Bytes
- Generate PRnd4 90 Bytes (random padding)
- Compute authentication token: Card.PK[VU.SK*['6A' II PRnd4 II K2 II Hash(PRnd4IIK2IIRnd3IICard.CHR) II 'BC']] = Encryption of ►(1) signature*◄ (ISO9796-2) of PRnd4 II K2 II Rnd3 II Card.CHR
- Authenticate self to card
External Authenticate (AutToken)
- Verify that current PK.CHA = Tachograph II VU
- Compute: Signature=Card.SK[AutToken],
- Decrypt and verify Signature with VU.PK, to recover PRnd4 II K2 II H'
- verify Hash(PRnd4IIK2IIRnd3IICard.CHR) = H'
- if verifications OK open AUT rights
- Store K2.
OK / KO
OK
No
Yes
Set TDES Session Key to (Ka, Kb, Ka) with Ka II Kb = K1 XOR K2 Set SSC to Rnd3 II Rnd1 (4 LSB of each)
Set TDES Session Key to (Ka, Kb, Ka) with Ka II Kb = K1 XOR K2 Set SSC to Rnd3 II Rnd1 (4 LSB of each)
Continue
Authentication failed Reject card

5. VU-CARDS DATA TRANSFER CONFIDENTIALITY, INTEGRITY AND AUTHENTICATION MECHANISMS

5.1. Secure messaging

VU-cards data transfers integrity shall be protected through Secure Messaging in accordance with references (ISO/IEC 7816-4) and (ISO/IEC 7816-8).

When data need to be protected during transfer, a cryptographic checksum data object shall be appended to the data objects sent within the command or the response. The cryptographic checksum shall be verified by the receiver.

The cryptographic checksum of data sent within a command shall integrate the command header, and all data objects sent (= > CLA = ′0C′, and all data objects shall be encapsulated with tags in which b1 = 1).

The response status-information bytes shall be protected by a cryptographic checksum when the response contains no data field.

Cryptographic checksums shall be four bytes long.

The structure of commands and responses when using secure messaging is therefore the following:

The DOs used are a partial set of the Secure Messaging DOs described in ISO/IEC 7816-4:

Tag	Mnemonic	Meaning
′81′	T_{PV}	Plain Value not BER-TLV coded data (to be protected by CC)
′97′	T_{LE}	Value of Le in the unsecured command (to be protected by CC)
′99′	T_{SW}	Status-Info (to be protected by CC)
′8E′	T_{CC}	Cryptographic Checksum
′87′	$T_{PI\ CG}$	Padding Indicator Byte || Cryptogram (Plain Value not coded in BER-TLV)

Given an unsecured command response pair:

Command header	Command body
CLA INS P1 P2	(L_c-field) (Data field) (L_e-field)
four bytes	L bytes, denoted as B_1 to B_L

Response body	Response trailer	
(Data field)	SW1	SW2
L_r data bytes	two bytes	

The corresponding secured command response pair is:

Secured command:

Command header (CH)	**Command body**										
CLA INS P1 P2	(New L_c field)	(New Data field)									(New L_e field)
'OC'	Length of new data field	T_{PV}	L_{PV}	PV	T_{LE}	L_{LE}	L_e	T_{CC}	L_{CC}	CC	'00'
		'81'	L_c	Data field	'97'	'01'	L_e	'8E'	'04'	CC	

Data to be integrated in checksum = CH || PB || T_{PV} || L_{PV} || PV || T_{LE} || L_{LE} L_e || PB

[X1PB = padding bytes (80.. 00) in accordance with ISO-IEC 7816-4 and ISO 9797 method 2]

DOs PV and LE are present only when there is some corresponding data in the unsecured command.

Secured response:

1. Case where response data field is not empty and needs not to be protected for confidentiality:

Response body						**Response trailer**
(New data field)						new SW1 SW2
T_{PV}	L_{PV}	PV	T_{CC}	L_{CC}	CC	
'81'	L_r	Data field	'8E'	'04'	CC	

Data to be integrated in checksum = T_{PV} || L_{PV} || PV || PB

2. Case where response data field is not empty and needs to be protected for confidentiality:

Response body						**Response trailer**
(New data field)						new SW1 SW2
$T_{PI\ CG}$	$L_{PI\ CG}$	PI CG	T_{CC}	L_{CC}	CC	

′87′		PI || CG	′8E′	′04′	CC	

Data to be carried by CG: non BER-TLV coded data and padding bytes.

Data to be integrated in checksum = $T_{PI\ CG}$ || $L_{PI\ CG}$ || PI CG PB

3. Case where response data field is empty:

Response body						**Response trailer**
(New data field)						new SW1 SW2
T_{SW}	L_{SW}	SW	T_{CC}	L_{CC}	CC	
′99′	′02′	New SW1 SW2	′8E′	′04′	CC	

Data to be integrated in checksum = T_{SW} || L_{SW} || SW || PB

5.2. Treatment of secure messaging errors

When the tachograph card recognises an SM error while interpreting a command, then the status bytes must be returned without SM. In accordance with ISO/IEC 7816-4, the following status bytes are defined to indicate SM errors:

′66 88′	:	verification of cryptographic checksum failed,
′69 87′	:	expected SM data objects missing,
′69 88′	:	SM data objects incorrect.

When the tachograph card returns status bytes without SM DOs or with an erroneous SM DO, the session must be aborted by the VU.

5.3. Algorithm to compute cryptographic checksums

Cryptographic checksums are built using a retail MACs in accordance with ANSI X9.19 with DES:

— initial stage: the initial check block y0 is E(Ka, SSC).

— sequential stage: the check blocks y1, …, yn are calculated using Ka.

— final stage: the cryptographic checksum is calculated from the last check block yn as follows: E(Ka, D(Kb, yn)).

where E() means encryption with DES, and D() means decryption with DES.

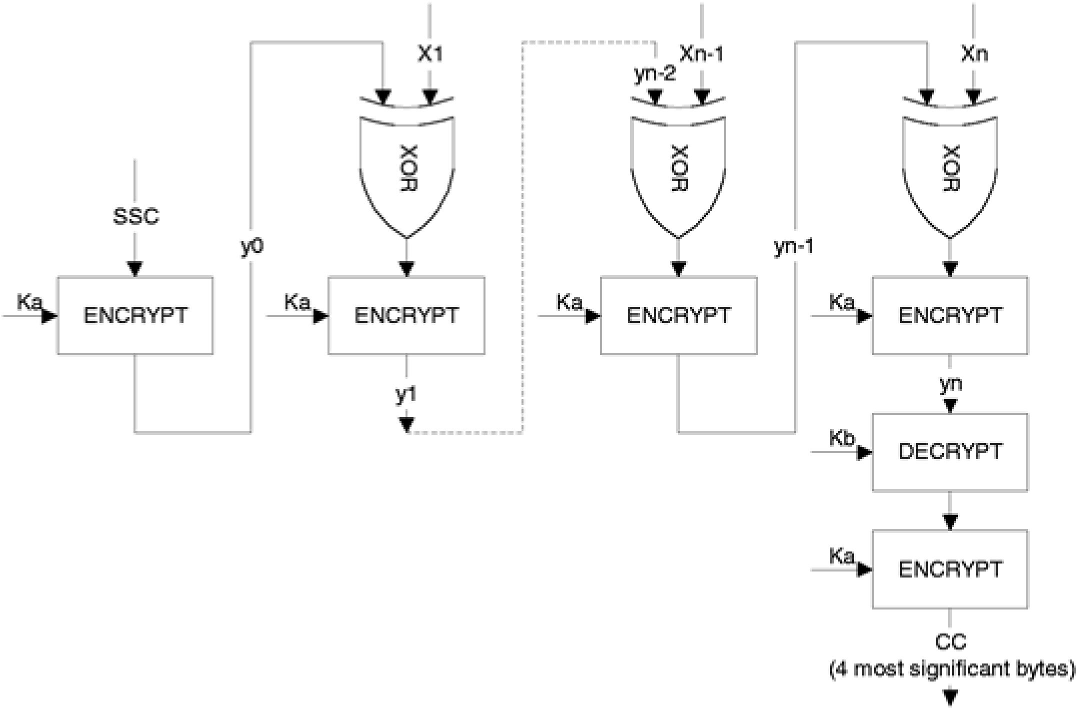
The four most significant bytes of the cryptographic checksum are transferred

The send sequence counter (SSC) shall be initiated during key agreement procedure to:

Initial SSC: Rnd3 (4 least significant bytes) || Rnd1 (4 least significant bytes).

The send sequence counter shall be increased by 1 each time before a MAC is calculated (i.e. the SSC for the first command is Initial SSC + 1, the SSC for the first response is Initial SSC + 2).

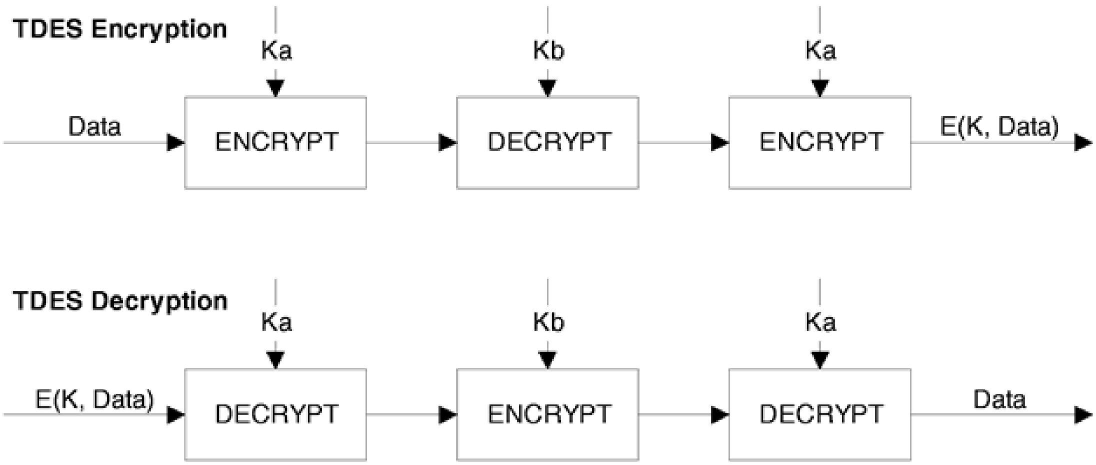
The following figure shows the calculation of the retail MAC:



5.4. Algorithm to compute cryptograms for confidentiality DOs

Cryptograms are computed using TDEA in TCBC mode of operation in accordance with references (TDES) and (TDES-OP) and with the Null vector as Initial Value block.

The following figure shows the application of keys in TDES:



6. DATA DOWNLOAD DIGITAL SIGNATURE MECHANISMS

The intelligent dedicated equipment (IDE) stores data received from an equipment (VU or card) during one download session within one physical data file. This file must contain the certificates

MS_i.C and EQT.C. The file contains digital signatures of data blocks as specified in Appendix 7 Data Downloading Protocols.

Digital signatures of downloaded data shall use a digital signature scheme with appendix such, that downloaded data may be read without any decipherment if desired.

6.1. Signature generation

Data signature generation by the equipment shall follow the signature scheme with appendix defined in reference (PKCS1) with the SHA-1 hash function:

Signature = EQT.SK[′00′ || ′01′ || *PS* || ′00′ || DER(SHA-1(Data))]

PS = Padding string of octets with value ′FF′ such that length is 128.

DER(SHA-1(*M*)) is the encoding of the algorithm ID for the hash function and the hash value into an ASN.1 value of type *DigestInfo* (distinguished encoding rules):

′30′||′21′||′30′||′09′||′06′||′05′||′2B′||′0E′||′03′||′02′||′1A′||′05′||′00′||′04′||′14′|| Hash Value.

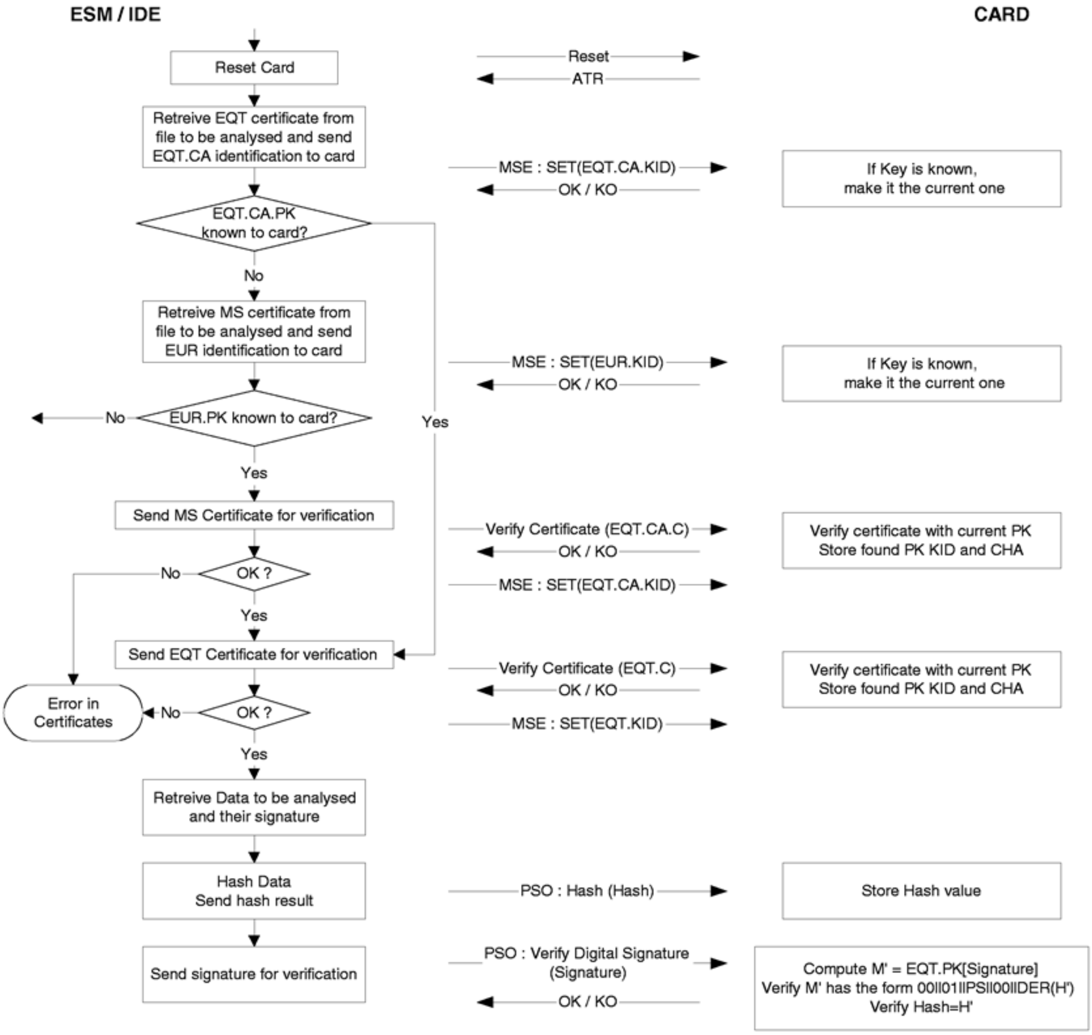
6.2. Signature verification

Data signature verification on downloaded data shall follow the signature scheme with appendix defined in reference (PKCS1) with the SHA-1 hash function.

The European public key EUR.PK needs to be known independently (and trusted) by the verifier.

The following table illustrates the protocol an IDE carrying a Control card can follow to verify the integrity of data downloaded and stored on the ESM (external storage media). The control card is used to perform the decipherement of digital signatures. This function may in this case not be implemented in the IDE.

The equipment that has downloaded and signed the data to be analysed is denoted EQT.


ESM / IDE
CARD
Reset Card
Reset
ATR
Retreive EQT certificate from file to be analysed and send EQT.CA identification to card
MSE : SET(EQT.CA.KID)
OK / KO
If Key is known, make it the current one
EQT.CA.PK known to card?
No
Yes
Retreive MS certificate from file to be analysed and send EUR identification to card
MSE : SET(EUR.KID)
OK / KO
If Key is known, make it the current one
No
EUR.PK known to card?
Yes
Send MS Certificate for verification
Verify Certificate (EQT.CA.C)
OK / KO
Verify certificate with current PK Store found PK KID and CHA
No
OK ?
MSE : SET(EQT.CA.KID)
Yes
Send EQT Certificate for verification
Verify Certificate (EQT.C)
OK / KO
Verify certificate with current PK Store found PK KID and CHA
Error in Certificates
No
OK ?
MSE : SET(EQT.KID)
Yes
Retreive Data to be analysed and their signature
Hash Data Send hash result
PSO : Hash (Hash)
Store Hash value
Send signature for verification
PSO : Verify Digital Signature (Signature)
OK / KO
Compute M' = EQT.PK[Signature]
Verify M' has the form 00ll01llPSll00llDER(H')
Verify Hash=H'

]]

Changes to legislation:
There are currently no known outstanding effects for the Council Regulation (EEC) No 3821/85, Appendix 11.