INTERNATIONAL STANDARD

ISO 20038

First edition
2017-11

Banking and related financial services — Key wrap using AES

Banque et autres services financiers — Enveloppe de clé utilisant AES

Reference number
ISO 20038:2017(E)

ISO copyright office
Ch. de Blandonnet 8 • CP 401
CH-1214 Vernier, Geneva, Switzerland
Tel. +41 22 749 01 11
Fax +41 22 749 09 47
copyright@iso.org
www.iso.org

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

The procedures used to develop this document and those intended for its further maintenance are described in the ISO/IEC Directives, Part 1. In particular the different approval criteria needed for the different types of ISO documents should be noted. This document was drafted in accordance with the editorial rules of the ISO/IEC Directives, Part 2 (see www.iso.org/directives).

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights. Details of any patent rights identified during the development of the document will be in the Introduction and/or on the ISO list of patent declarations received (see www.iso.org/patents).

Any trade name used in this document is information given for the convenience of users and does not constitute an endorsement.

For an explanation on the voluntary nature of standards, the meaning of ISO specific terms and expressions related to conformity assessment, as well as information about ISO's adherence to the World Trade Organization (WTO) principles in the Technical Barriers to Trade (TBT) see the following URL: www.iso.org/iso/foreword.html.

This document was prepared by Technical Committee ISO/TC 68, *Financial services*, Subcommittee SC 2, *Financial Services, security*.

Introduction

The secure management of cryptographic keys requires that their values and usage constraints be protected for both confidentiality and integrity. This is especially true for keys used with the 64-bit block cipher triple data encryption algorithm (TDEA) and the 128-bit block cipher advanced encryption standard (AES) because these block ciphers allow the use of key sizes that are larger than the block size.

This document provides a method of wrapping cryptographic keys in order to provide confidentiality and integrity protection for the keys when being transmitted or stored. The mechanism is designed to use AES as the wrapping cipher.

Banking and related financial services — Key wrap using AES

1 Scope

This document defines a method for packaging cryptographic keys for transport. This method can also be used for the storage of keys under an AES key. The method uses the block cipher AES as the wrapping cipher algorithm.

Other methods for wrapping keys are outside the scope of this document but can use the authenticated encryption algorithms specified in ISO/IEC 19772.

2 Normative references

The following documents are referred to in the text in such a way that some or all of their content constitutes requirements of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO 11568-2, *Financial services — Key management (retail) — Part 2: Symmetric ciphers, their key management and life cycle*

ISO/IEC 9797-1, *Information technology — Security techniques — Message Authentication Codes (MACs) — Part 1: Mechanisms using a block cipher*

ISO/IEC 10116, *Information technology — Security techniques — Modes of operation for an n-bit block cipher*

ANS X9 TR-31, *Interoperable Secure Key Exchange Key Block Specification for Symmetric Algorithms*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

ISO and IEC maintain terminological databases for use in standardization at the following addresses:

— ISO Online browsing platform: available at http://www.iso.org/obp

— IEC Electropedia: available at http://www.electropedia.org/

3.1
advanced encryption standard
AES
algorithm specified in ISO/IEC 18033-3

3.2
bit
binary digit

3.3
byte
sequence of 8 *bits* (3.2)

3.4
ciphertext
encrypted (enciphered) data

3.5
cryptographic key
key
sequence of symbols that controls the operation of a cryptographic transformation (e.g. *encryption* (3.7), *decryption* (3.6), cryptographic check function computation, signature generation, or signature verification)

3.6
decryption
process of transforming *ciphertext* (3.4) into *plaintext* (3.13)

3.7
encryption
process of transforming *plaintext* (3.13) into *ciphertext* (3.4)

3.8
exclusive-OR
bit-by-bit modulo-2 addition of binary vectors of equal length

3.9
initialization vector
binary vector used as the input to initialize the algorithm for the *encryption* (3.7) of a plaintext block sequence to increase security by introducing additional cryptographic variance and to synchronize cryptographic equipment

Note 1 to entry: See ISO/IEC 10116.

3.10
key block
block containing a protected key, its usage constrains and other data, that is wrapped (encrypted) using a key wrapping mechanism

3.11
key wrap
symmetric *encryption* (3.7) algorithm designed to encapsulate (encrypt) cryptographic key material

3.12
nibble
half a *byte* (3.3), which can be represented by a single hexadecimal digit

3.13
plaintext
intelligible data that has meaning and can be read or acted upon without the application of *decryption* (3.6)

Note 1 to entry: Also known as cleartext. In the context of this document, the plaintext is the key being wrapped.

3.14
secure cryptographic device
SCD
device that provides secure storage for secret information, such as keys, and provides security services based on this secret information

3.15
triple data encryption algorithm
TDEA
algorithm specified in ISO/IEC 18033-3

4 Symbols and abbreviated terms

AES	advanced encryption standard
CBC	cipher block chaining (mode of encryption)
CMAC	cipher-based MAC
CTR	counter (mode of encryption)
IV	initialization vector for CBC mode or starting value for CTR mode
K	cryptographic key
MAC	message authentication code
TDEA	triple data encryption algorithm
SCD	secure cryptographic device
$\oplus$	exclusive-OR

5 Key wrap method characteristics

Key management according to ISO 11568-2 requires that symmetric keys be protected by physical protection, by splitting the key into components, or by cryptographic protection. Cryptographic protection can be achieved using an authenticated encryption algorithm such as one standardized in ISO/IEC 19772. However, most of the authenticated encryption algorithms in ISO/IEC 19772 are designed for protecting generic payloads such as long messages or large databases rather than symmetric keys that are short and have high entropy. A clear exception to this is mechanism 2 of ISO/IEC 19772:2009 which is called Key Wrap. As stated in ISO/IEC 19772, "This scheme was originally designed for authenticated-encryption of keys and associated information. This mode is known as AES Key Wrap when the AES block cipher is used". It is also noted in ISO/IEC 19772 that AES Key Wrap is also specified in NIST, *AES Key Wrap Specification* and Reference [5].

The method defined in this document uses the MAC as IV (compared with Algorithm 5 in ISO/IEC 19772 which is an encrypt-then-MAC authenticated encryption algorithm) and as such it could theoretically support any symmetric encryption algorithm mode (e.g. taken from ISO/IEC 10116) or MAC algorithm (e.g. taken from ISO/IEC 9797-1). However, for the purposes of this document, the key wrap method supports only CBC or CTR mode encryption (as defined in ISO/IEC 10116) and CMAC (Method 5 in ISO/IEC 9797-1 and NIST/SP 800-38B) for MAC generation.

The key usage attributes from ANS/TR 31 shall be included in the wrapping process as defined in Annex A. Other methods include but are not limited to authenticated encryption algorithms in ISO/IEC 19772, RFC 3394[5], ANSI CBC MAC[4] and TDEA Key Wrap[4].

6 Key Block Binding key wrap method

6.1 General

When a key is encrypted with a block cipher that has a block size less than the size of the key, this forces the key to be represented by several blocks resulting in a danger of substitution or misuse of a fragment of the overall key cryptogram. Binding the blocks of the encrypted key may be achieved through various methods.

The Key Block Binding method protects the secrecy of the key blocks and protects the integrity of the association between the key blocks and the key block header (see Annex A for a definition of a key block header). The method uses an AES Key Block Protection Key that was previously exchanged (using

secure, possibly manual, methods as described in ISO 11568-2) between the two communicating parties and used for deriving keys used for MACing and encrypting the key blocks. The method can be used for wrapping any cryptographic key (see Table A.4).

The processing components of the Key Block Binding key wrap method are as follows.

— Key derivation as described in 6.3:

  — derivation of the MAC and encryption keys from the protection key.

— Binding and encryption as described in 6.2:

  — binding of the key to be wrapped and its header using the derived MAC key;

  — encryption of the key to be wrapped and its length using the derived encryption key.

— Decryption and validation as described in 6.4:

  — decryption of the wrapped key and its length using the derived encryption key;

  — validation of the associated header data using the derived MAC key.

6.2 Key block binding and encryption

The key block binding and encryption proceeds as follows.

— The confidential portion is constructed using one of the following methods.

  — For CBC mode encryption, the confidential portion (key length and key) is padded on the right with random pad bytes until the resulting string is a multiple of 16 bytes. Additional padding may be used to mask the true length of the key data as long as the resulting length is a multiple of 16 bytes.

  — For CTR mode encryption, there is no padding. Note that although CTR does not require padding, the confidential portion may be padded in the same way as CBC mode in order to disguise the key length.

— CMAC is applied to the entire payload, that is, the header concatenated with the confidential part, including padding if present, using the derived MAC key (see 6.3) to yield a MAC, m. The MAC is not truncated and is 16 bytes.

— The confidential part (key length, key and random padding if present) is encrypted in either CBC or CTR mode (depending on which mode is chosen) with no additional padding applied and using the MAC m as IV and the derived encryption key (see 6.3) in accordance with ISO/IEC 10116. This yields a ciphertext, c.

— The ciphertext c is transmitted along with the MAC m and the unencrypted portion (the header).

Figure 1 illustrates the Key Block Binding and encryption described above.

Details of the key block header and key length encoding can be found in Annex A.

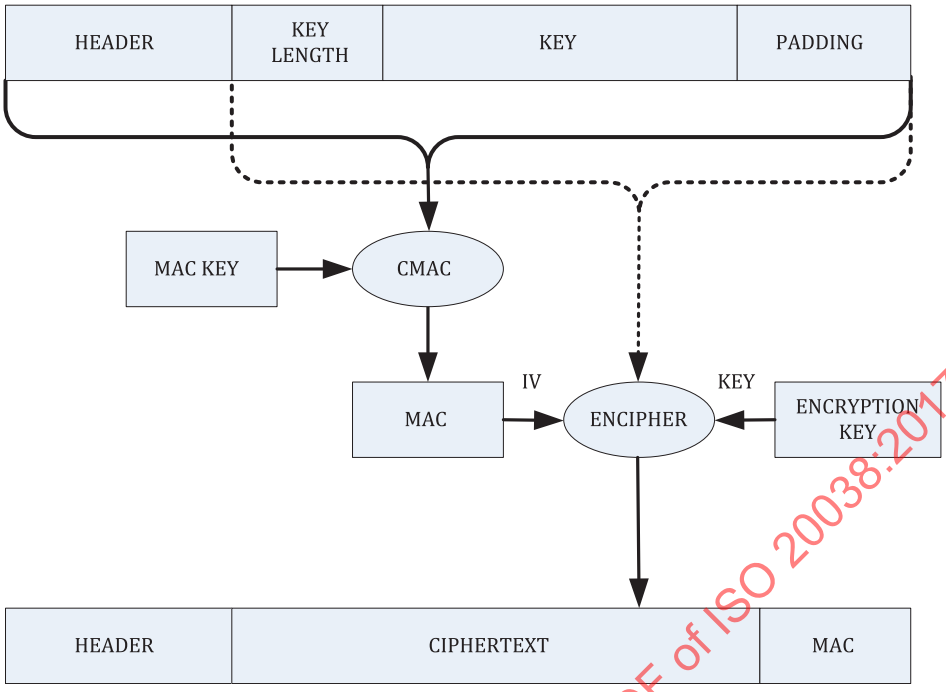


Figure 1 — Key Block Binding

The MAC key and the encryption key are derived keys as described in the next section.

6.3 Key derivation

The encryption key and MAC key are derived from the Key Block Protection Key using CMAC (algorithm 5 in ISO/IEC 9797-1) as detailed in the remainder of this subclause. Table 1 shows the input data to the CMAC function.

Table 1 — Key Derivation Input Data

Nibble #	Field name	Description	Encoding	Range of values
0–1	Counter	A counter that is incremented for each CMAC operation	2H	0x01–0x02
2–5	Key Usage Indicator	Indicates whether the key to be derived is to be used for encryption/decryption or MAC generation/verification	4H	0x0000 = encryption CBC mode 0x0001 = MAC 0x0002 = encryption CTR mode
6–7	Separator	A 1-byte separator, shall be zero	2H	0x00
8–11	Algorithm Indicator	Indicates the encryption and MAC block cipher algorithm that is going to use the two derived keys (and is used to derive those keys)	4H	0x0002 = AES 128 bit 0x0003 = AES 192 bit 0x0004 = AES 256 bit
12–15	Length	Length, in bits, of the keying material being generated for the pair of encryption and MAC keys	4H	0x0080 if AES-128 keys are being generated 0x00C0 if AES-192 keys are being generated 0x0100 if AES-256 keys are being generated
NOTE The counter value in nibbles 0–1 is set to 1 when deriving the first bytes of the encryption key, then is reset to 1 again when deriving the first bytes of the MAC key.				

The Counter is incremented for each call to CMAC as part of deriving an encryption or a MAC key from a Key Block Protection Key. The Counter starts at 0x01. The Key Usage Indicator tells if the key generated is a MAC key or an encryption key for CBC mode or CTR mode encryption. The Algorithm Indicator tells which algorithm is used.

Key Block Protection Keys derive keys of the same length. That is, a 128-bit AES key can only be used to derive 128-bit encryption and MAC keys, a 192-bit AES key can only be used to derive 192-bit encryption and MAC keys and a 256-bit AES key can only be used to derive 256-bit encryption and MAC keys.

"Length" indicates how many bits of keying material is to be derived for the encryption and MAC keys. If the derived key is a 128-bit key, then a total of 128 bits (0x0080) are to be derived, if it is a 192-bit key, then a total of 192 bits (0x00C0) are to be derived and if it is a 256-bit key, then a total of 256 bits (0x0100) are to be derived. Note that because wrapping is only allowed using AES, the length information can be derived from the Algorithm Indicator.

In any sequence of calls to CMAC where the counter is incremented, the Key Usage Indicator and the Algorithm Indicator should remain unchanged. Hence, when deriving an encryption key and a MAC key, that should be done in two distinct sequences of calls to CMAC, each starting with the Counter as 0x01.

Figure 2 illustrates how to derive a 128-bit AES CBC encryption key and MAC key from a 128-bit AES Key Block Protection Key, K.

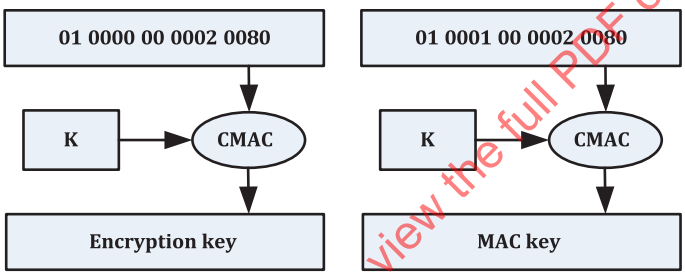
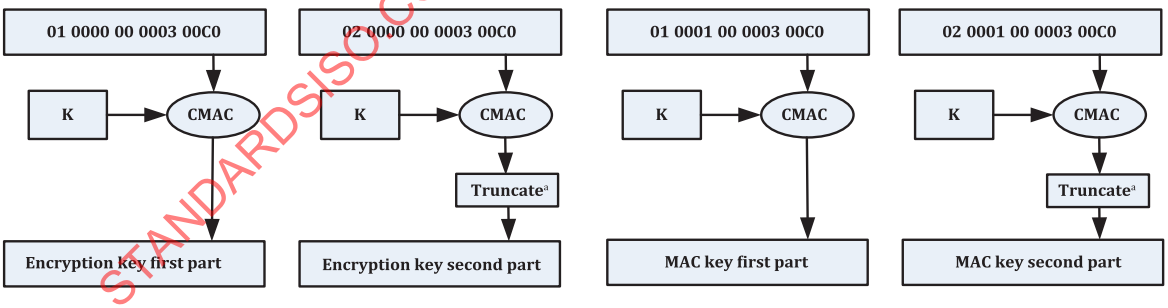


Figure 2 — Deriving AES 128 MAC and CBC encryption keys

Figure 3 illustrates how to derive a 192-bit AES CBC encryption key and MAC key from a 192-bit AES Key Block Protection Key, K.



a Select leftmost 64 bits.

Figure 3 — Deriving 192-bit AES MAC and CBC encryption keys

Figure 4 illustrates how to derive a 256-bit AES CBC encryption key and MAC key from a 256-bit AES Key Block Protection Key, K.

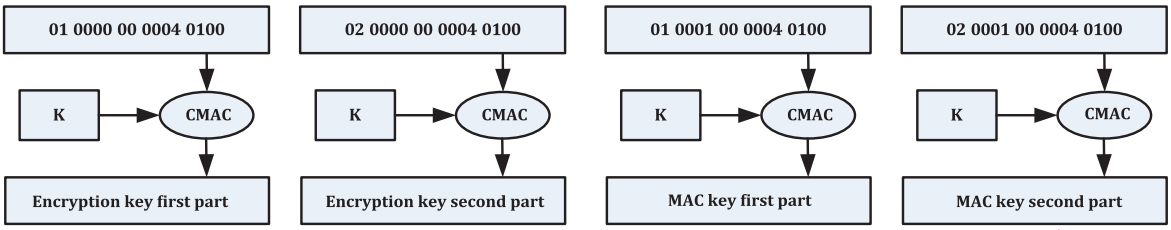


Figure 4 — Deriving 256-bit AES MAC and CBC encryption keys

6.4 Key Block Decryption and MAC Validation

Upon receiving the key block, the SCD decrypts and authenticates the key block and verifies that the contents of the header and the structure of the header are valid. If any verification fails, the key block is rejected. The steps of this process are described as follows.

a) The received key block header is checked for formatting errors (e.g. invalid data in any fields) and for CBC mode only, the total length of the received key block ciphertext is validated to be a multiple of 16; if these checks fail, the received key block is rejected.

b) If the above checks pass, the MAC and encryption keys are derived from the Key Block Protection Key as per 6.3.

c) The received MAC is stored into a temporary buffer (RCVD_MAC).

d) The cleartext key block header is stored into a temporary buffer (BLK_BUF).

e) The encrypted portion of the received key block is decrypted using the algorithm indicated with the encryption key and the contents of RCVD_MAC as the IV and the decrypted data are stored in BLK_BUF appended to the Key Block Header and using the derived encryption key.

f) A MAC is calculated across the entire contents of the temporary buffer BLK_BUF using CMAC with the derived MAC key.

g) The calculated MAC is compared with the contents of RCVD_MAC. If the contents are identical, the received key block is considered to be valid and the contents of BLK_BUF may be used; if the calculated MAC and the contents of RCVD_MAC are not identical, the key block is rejected.

Annex A
(normative)

Key Block with Optional Block

A.1 General

This document requires that the entities using the key wrap method have a pre-agreed Key Block Header definition. Such a definition shall support the parsing, interpretation and processing of the wrapped Key Block contents (header, ciphertext and MAC) and shall at a minimum identify the encryption mode (CBC or CTR) and the algorithm (see Table A.4) to be used with the protected key. It should also identify the key usage of the protected key (e.g. encrypting a PIN, calculating a MAC, see Table A.3) and its mode of use (e.g. restricted to verifying a MAC, see Table A.5).

The remainder of this annex defines the Key Block Header to be used with the Key Block Binding methods defined in Clause 5. This header is based on ANS/TR 31.

Figure A.1 outlines the overall format of the Key Block.

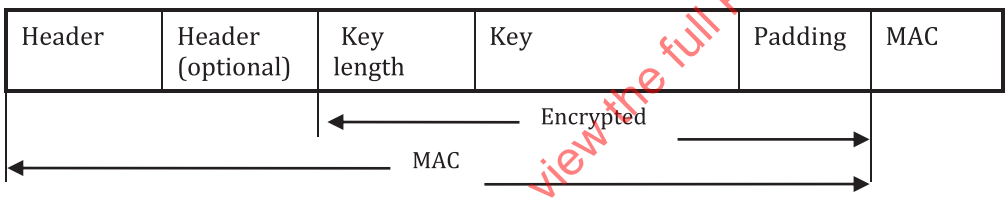


Figure A.1 — CBC MAC Key Block

A.2 Key Block Header

A.2.1 General

The header contains attribute information about the key. For better supportability (i.e. human readability), the header bytes use uppercase ASCII printable characters, although in some cases other characters may be necessary. The specific encoding and acceptable characters are defined individually for each field, for example in the "Encoding" column of Table A.1. When the term "numeric" is used to describe an ASCII header byte, it means an ASCII character in the range "0" to "9" (0x31 to 0x39). A multi-byte header field is considered numeric only when all bytes of the field contain numeric values.

Table A.1 shows the format of the Key Block Header.

Table A.1 — Key Block Header for Key Block Binding method

Byte #	Field name	Description	Encoding
0	Key Block Version ID	Identifies the version of the key block, which defines the method by which it is cryptographically protected and the content and layout of the block: — "D" (0x44) — Key block protected using the AES Key Derivation Binding Method for CBC mode; — "E" (0x45) — Key block protected using the AES Key Derivation Binding Method for CTR mode. Note that numeric key block Version IDs are reserved for proprietary key block definitions. Also note that multiple key block versions may be in use at any time. Future versions of this document may define other values. The values "A", "B" and "C" are not used by this document.	1AN
1–4	Key Block Length	ASCII numeric digits providing key block length (number of characters) after encoding. Length includes the entire block (Header + encrypted confidential data + MAC) in decimal; e.g. a 112-byte key block would contain "0" in byte #1, "1" in byte #2, "1" in byte #3, and "2" in byte #4. See A.2.9.	4N
5–6	Key Usage	Provides information about the intended function of the protected key/sensitive data. Common functions include encrypting data, encrypting PINs, and calculating a MAC. See Table A.3.	2AN
7	Algorithm	The approved algorithm for which the protected key may be used. See Table A.4.	1AN
8	Mode of Use	Defines the operation the protected key can perform. For example, a MAC key may be limited to verify-only. See Table A.5.	1AN
9–10	Key Version Number	Two-digit ASCII character version number, optionally used to indicate that contents of the key block is a component or to prevent re-injection of old keys. See Table A.6.	2AN
11	Exportability	Defines whether the protected key may be transferred outside the cryptographic domain in which the key is found. See Table A.7.	1AN
12–13	Number of Optional Blocks	Defines the number of Optional Blocks included in the key block. The minimum value is zero and the maximum is 99, but it is also limited by the maximum number of bytes in the entire key block. For example, five Optional Blocks would be represented by "0" (0x30) in byte 12 and "5" (0x35) in byte 13.	2N
14–15	Reserved for future use	This field is reserved for future use and is filled with ASCII zero (0x30) characters.	2N
16–17	First Optional Block ID	ID field for the First Optional Block, if one is present as indicated by a value other than "00" in bytes 12 and 13. See Table A.8.	2AN

Table A.1 *(continued)*

Byte #	Field name	Description	Encoding
18–19	Optional Block 1 Length	If the first Optional Block is present, indicated by a value other than "00" in bytes 12 and 13, then this field contains the length of that Optional Block in bytes, including the field's ID, length, and data. Byte 18 contains the high-order byte of the length and byte 19 contains the low-order byte, both in hex-ASCII form. For example, for a 24-byte Optional Block (18 in hexadecimal notation), byte 18 would contain a "1" (0x31) and byte 19 would contain an "8" (0x38). NOTE Optional Block Data may be omitted, in which case the Optional Block Length will be "04".	2H
20-n	Optional Block 1 Data	If the first Optional Block is present, indicated by a value other than "00" in bytes 12–13, then this field contains the Data for that Optional Block. It will contain only printable ASCII characters.	PA
(n+1)-m	Additional Optional Blocks, if present	A variable number of Optional Blocks can follow the first Optional Block, as indicated by the count in bytes 12–13. Each Optional Block contains a 2-byte ID, 2-byte length, and variable-length data field following the same format described above for Optional Block 1.	

NOTE 1 Before a key in the key block format is used in a secure cryptographic device (SCD), the contents of the header block are validated to ensure the correct usage is enforced. First, the SCD makes sure that the length of the key block matches the contents of bytes 1 to 4. If the lengths do not match, the SCD returns an error and stops processing the block. The "Key Usage" byte is typically checked next followed by the "Algorithm" byte. The processing of the other header bytes depends on the key usage and the algorithm used.

From 0 to 99, optional blocks can be included in the key block, with the limitation that the total length of the key block cannot exceed the maximum length that can be represented by the Key Block Length field in bytes 1 to 4. The total number of optional blocks is indicated by a count in bytes 12 to 13. Each optional block consists of three fields: a 2-byte identifier that indicates the content of the optional block, a 2-byte length that indicates the total number of bytes in all of the optional block's fields, and the variable-length data carried in the optional block.

As an example, consider an optional block that contains a Key Set Identifier. A sample block could contain the following data.

Identifier: "KS"

Length: "18"

Data: "00604B120F9292800000"

NOTE 2 The length value equals 2 + 2 + 20 (24). The length value is encoded in Hex-ASCII; 18 hex equals 24 decimal.

If more than one optional block is present, each one immediately follows the previous one in the key block structure.

The total length of all optional blocks in the key block will be a multiple of the encryption block size. This may require padding, and if padding is needed, it is included in a special final optional block that is filled with an appropriate number of padding characters. For example, if the total length of all optional blocks before padding is 90, six additional bytes are needed in order to bring the total to the next multiple of the encryption block size, 96. A padding block would be added as the last optional block, containing the 2-byte identifier "PB", a 2-byte length field "06", and two characters of padding.

A.2.2 Encryption

The data to be encrypted is the key length, the key, and any padding of the key field. The Key Length is binary encoded on two bytes in the same way as the length field for the derivation data in Table A.1. Table A.2 shows an example of formatting for a 16-byte TDEA key where encryption uses CBC mode and thus involves padding. Other amounts of random padding can also be used, as long as the total confidential data field is a multiple of 16 bytes in length. Refer to Annex B for a full worked example.

Table A.2 — Example of confidential data for a 16 byte TDEA key using CBC

Length before encryption	Field name	Description	Encoding	Encrypted
2 bytes	Key Length	Key length in bits represented in binary with the most significant byte transmitted in the leftmost byte (byte 16 if no optional data are present). In this example, a 128-bit key is represented as 0x00 in byte 16 and 0x80 in byte 17.	2B	Yes
16 bytes	Key	Key/sensitive data to be protected. In this example, the TDEA key (including parity bits).	16B	Yes
14 bytes	Pad	Random data	14B	Yes

A.2.3 Key usage

Key usage defines the type of the key, whether it is used to encrypt data, calculate a MAC, etc. The key usage is identified by bytes 5 and 6. Table A.3 shows the currently defined key usage values.

Table A.3 — Defined key usage values

Value	Hex	Definition	Mode(s) of use
"B0"	0x42, 0x30	BDK Base Derivation Key	"X"
"B1"	0x42, 0x31	DUKPT Initial Key (also known as IPEK)	"X"
"C0"	0x43, 0x30	CVK Card Verification Key	"C", "G", "V"
"D0"	0x44, 0x30	Data Encryption using ECB, CBC, CFB, OFB, CCM, or CTR	"B", "D", "E"
"D1"	0x44, 0x31	Asymmetric key Data Encryption	"B", "D", "E"
"E0"	0x45, 0x30	EMV/chip Issuer Master Key: Application cryptograms	"X"
"E1"	0x45, 0x31	EMV/chip Issuer Master Key: Secure Messaging for Confidentiality	"X"
"E2"	0x45, 0x32	EMV/chip Issuer Master Key: Secure Messaging for Integrity	"X"
"E3"	0x45, 0x33	EMV/chip Issuer Master Key: Data Authentication Code	"X"
"E4"	0x45, 0x34	EMV/chip Issuer Master Key: Dynamic Numbers	"X"
"E5"	0x45, 0x35	EMV/chip Issuer Master Key: Card Personalization	"X"
"E6"	0x45, 0x36	EMV/chip Issuer Master Key: Other	"X"
"I0"	0x49, 0x30	Initialization Vector (IV)	"N"
"K0"	0x4B, 0x30	Key Encryption or wrapping	"B", "D", "E"
"K1"	0x4B, 0x31	TR-31 Key Block Protection Key	"B", "D", "E"
"K2"	0x4B, 0x32	TR-34 Asymmetric Key	"B", "D", "E", "S", "V"
"K3"	0x4B, 0x33	Asymmetric Key for key agreement/key wrapping	"B", "D", "E", "X"
"K4"	0x4B, 0x34	ISO 20038 Key Block Protection Key	"B", "D", "E"
"M0"	0x4D, 0x30	ISO 16609 MAC algorithm 1 (using TDEA)	"C", "G", "V"
"M1"	0x4D,0x31	ISO 9797-1 MAC Algorithm 1	"C", "G", "V"

Table A.3 *(continued)*

Value	Hex	Definition	Mode(s) of use
"M2"	0x4D, 0x32	ISO 9797-1 MAC Algorithm 2	"C", "G", "V"
"M3"	0x4D, 0x33	ISO 9797-1 MAC Algorithm 3	"C", "G", "V"
"M4"	0x4D, 0x34	ISO 9797-1 MAC Algorithm 4	"C", "G", "V"
"M5"	0x4D, 0x35	ISO 9797-1 MAC Algorithm 5	"C", "G", "V"
"M6"	0x4D, 0x36	CMAC	"C", "G", "V"
"M7"	0x4D, 0x36	HMAC	"C", "G", "V"
"P0"	0x50, 0x30	PIN encryption	"B", "D", "E"
"S0"	0x53, 0x30	Asymmetric key for digital signatures	"S", "V"
"S1"	0x53, 0x31	Asymmetric key pair, CA key	"S", "V"
"S2"	0x53, 0x32	Asymmetric key pair, non-X9.24 key	"S", "V", "T", "B", "D", "E"
"V0"	0x56, 0x30	PIN verification, KPV, other algorithm	"C", "G", "V"
"V1"	0x56, 0x31	PIN verification, IBM 3624	"C", "G", "V"
"V2"	0x56, 0x32	PIN Verification, VISA PVV	"C", "G", "V"
All numeric values		Reserved for proprietary use (Acceptable modes of use vary depending on rules of use for each such key)	

Note that some of these usages may work for both symmetric and asymmetric keys. Usage "K0" is appropriate for a TDEA KEK and an RSA key exchange key. This list of usages is not intended to be exhaustive. Future versions may include other key usage types.

Precise usage restrictions for each of these usage values are described below.

Asymmetric Key Data Encryption — The public and private key pair used to directly encipher or decipher data (e.g. encrypting a PAN using RSA or Elliptic Curve ECIES). If ECIES is used, the derived encryption and MAC keys are ephemeral and never disclosed outside the SCD. Note that the D1 key usage value should never be used when wrapping a symmetric key in a digital envelope, even if the symmetric key is not stored or returned.

Asymmetric key pair — Asymmetric public/private keys used for computing and verifying digital signatures, key exchange or both.

Asymmetric key pair, key exchange — Asymmetric public/private keys used to encrypt or decrypt other keys for transport. Mode of use "T" implies that the key can also be used to authenticate key exchanges (e.g. computing the TR-34 Freshness Challenge), but never for signing only.

BDK Base Derivation Key — This key is used to derive the Initial Key (IPEK) in the DUKPT process defined in X9.24-1. It cannot be used in any other key derivation process. The Initial Key is subsequently used by the PED to derive any of the three DUKPT key types: PIN keys, MAC keys, or data encipherment keys. This key type is always used in conjunction with the Derivation Key Usage, "X".

CVK Card Verification Key — This key is used to compute a card verification code such as those using the CVV, CVC, CSC, CVC2 and CVV2 algorithms. It can also be used to verify those codes, which consists of computing the code and comparing the result to a supplied value. Note that key usages appropriate for the CVK are the same as those appropriate for MAC Keys.

Data Encryption — This key can be used to encipher or decipher data, but it cannot be used in any more specific cryptographic operations such as PIN block encryption or key encryption.

DUKPT Initial Key (also known as IPEK) — This key is sent to a PIN Entry Device as the initial key in a DUKPT key management scheme. This key type is always used in conjunction with the Derivation Key Usage, "X".

EMV/chip Issuer Master Key: Application cryptograms — Keys with this type are used for deriving the EMV chip card key used to produce Application Cryptograms (TC, ARQC or AAC).

EMV/chip Issuer Master Key: Card Personalization — Keys with this type are used for deriving the EMV chip card Master Key used to protect card data during personalization.

EMV/chip Issuer Master Key: Data Authentication Code — Keys with this type are used for deriving the EMV chip card Master Key used to produce the DAC.

EMV/chip Issuer Master Key: Dynamic Numbers — Keys with this type are used for deriving the EMV chip card Master Key used for producing Dynamic Numbers.

EMV/chip Issuer Master Key: Secure Messaging for Confidentiality — Keys with this type are used for deriving the EMV chip card Master Key used to ensure messaging confidentiality.

EMV/chip Issuer Master Key: Secure Messaging for Integrity — Keys with this type are used for deriving the EMV chip card Master Key used to ensure messaging integrity.

EMV/chip Issuer Master Key: Other — Keys with this type are used for deriving the EMV chip card Master Key used for any other purposes.

NOTE Requirements for the use, generation, and management of EMV keys are defined by the standards written by EMV and by the card associations that make use of EMV cards. These requirements are not defined in this document.

Initialization Vector (IV) — This is the IV used as input to any cipher mode of operation requiring an IV (e.g. MAC, Data encryption/Decryption, etc.). It is not a key and cannot be used for any purpose other than as an IV. Note that Key Usage appropriate for the IV is "N" since the IV is not a key. The algorithm for which the IV can be used is defined by the algorithm identifier in the header.

ISO 16609 MAC algorithm 1 (using TDEA) — This key can be used to either compute (generate) or verify a MAC using the ISO 16609 algorithm 1. It cannot be used with any other MAC algorithms.

ISO 9797-1 MAC Algorithm 1 — This key can be used to either compute (generate) or verify a MAC using the ISO 9797-1 algorithm 1. It cannot be used with any other MAC algorithms.

ISO 9797-1 MAC Algorithm 2 — This key can be used to either compute (generate) or verify a MAC using the ISO 9797-1 algorithm 2. It cannot be used with any other MAC algorithms.

ISO 9797-1 MAC Algorithm 3 — This key can be used to either compute (generate) or verify a MAC using the ISO 9797-1 algorithm 3. It cannot be used with any other MAC algorithms.

ISO 9797-1 MAC Algorithm 4 — This key can be used to either compute (generate) or verify a MAC using the ISO 9797-1 algorithm 4. It cannot be used with any other MAC algorithms.

ISO 9797-1 MAC Algorithm 5 — This key can be used to either compute (generate) or verify a MAC using the ISO 9797-1 algorithm 5. It cannot be used with any other MAC algorithms.

Key Encryption or wrapping — This is a Key Encryption Key (KEK) which is used only to encrypt or decrypt other keys, or as a key used to derive keys that are used for that purpose. Note that this can be a key used for an authenticated encryption operation in which the key itself or keys derived from it are used for both encryption and integrity computations. Note that this key usage is not used for TR-31 Key Block Protection Keys.

PIN Encryption — This key is used to protect PIN blocks. It can be used with any PIN block format and with any PIN block encryption method. It cannot be used with any type of data other than a PIN or PIN block.

TR-31 Key Block Protection Key — The derivation key from which the Key Block Encryption Key and the Key Block MAC Key are derived; this key is used for no other purpose.

PIN verification, KPV, other algorithm — A key of this type is used to verify a PIN using some algorithm other than IBM 3624 or Visa PVV, each of which have their own specific key types. These keys cannot be used for the IBM 3264 or Visa PVV verification algorithms, or with any function other than PIN verification.

PIN verification, IBM 3624 — A key of this type is used in the IBM 3624 PIN offset generation process or to verify a PIN using the IBM 3624 algorithm. It cannot be used to verify a PIN with any other algorithm, or with any function other than PIN verification.

PIN Verification, VISA PVV — A key of this type is used to generate the PIN Verification Value (PVV) or to verify a PIN using the Visa PVV algorithm. It cannot be used to verify a PIN with any other algorithm, or with any function other than PIN verification.

A.2.4 Algorithm

The algorithm byte defines what algorithms can be used with this key. The algorithm is described by byte 7 of the header. Table A.4 shows the currently defined algorithm values.

Table A.4 — Defined algorithm values

Value	Hex	Definition
"A"	0x41	AES
"D"	0x44	DEA
"E"	0x45	Elliptic Curve
"H"	0x48	HMAC-SHA-1
"I"	0x49	HMAC-SHA-2
"J"	0x4A	HMAC-SHA-3
"R"	0x52	RSA
"S"	0x53	DSA
"T"	0x54	Triple DEA (TDEA)
Numeric values		Reserved for proprietary use

A.2.5 Mode of use

The Mode of Use byte defines the operation the key can perform. The mode is identified in byte 8. Table A.5 shows the currently defined mode of use values.

Table A.5 — Defined mode of use values

Value	Hex	Definition
"B"	0x42	Both encrypt and decrypt/wrap and unwrap
"C"	0x43	Both generate and verify
"D"	0x44	Decrypt/unwrap only
"E"	0x45	Encrypt/wrap only
"G"	0x47	Generate only
"N"	0x4E	No special restrictions (other than restrictions implied by the Key Usage)
"S"	0x53	Signature only
"V"	0x56	Verify only
"X"	0x58	Key used to derive other key(s)
Numeric values		Reserved for proprietary use

Precise usage restrictions for each of these mode values are described below.

Both Encrypt and Decrypt — The key can be used in encryption or decryption. This includes a variety of key usages and not only data encryption and decryption. For example, if the key has usage "Key Encryption or wrapping", this mode indicates that the key can be used for both wrapping and unwrapping.

MAC Calculate (Generate or Verify) — The key can be used to generate or verify a MAC. This mode is compatible with keys having MAC key usage attributes, as well as other attributes in which the key is used to compute or verify a MAC

Decrypt Only — The key can be used in a decryption process. This includes a variety of key usages and not only data decryption. For example, if the key has usage "Key Encryption or wrapping", this mode indicates that the key can be used for unwrapping but cannot be used for wrapping. If the Key Type is "P0", PIN Encryption, a Key Usage value of Decrypt Only means that this key is only used to decrypt PINs as the first half of a translate or in preparation for a verify operation.

Encrypt Only — The key can be used in an encryption process. This includes a variety of key usages and not only data encryption. For example, if the key has usage "Key Encryption or wrapping", this mode indicates that the key can be used for wrapping but cannot be used for unwrapping. If the Key Type is "P0", PIN Encryption, a Key Usage value of Encrypt Only may be used to encrypt a PIN in a PED or as the second half of a PIN translate operation.

MAC Generate Only — The key can be used to generate a MAC, but it cannot be used to verify a MAC. This mode value is only compatible with keys that have one of the MAC key usage attributes.

No special restrictions or not applicable — If the key has this mode, it can be used in any way that is compatible with the key usage attribute associated with that key. This value is typically used (but not limited to use) with IV's and for proprietary key usages.

Signature Only — The key can only be used in digital signature operations. This means that the key shall be a public key (signature verify) or a private key (signature generate) from a public-key algorithm such as RSA or Elliptic Curve.

MAC Verify Only — The key can be used to verify a MAC, but it cannot be used to generate a MAC. This mode value is only compatible with keys that have one of the MAC key usage attributes.

Key used to Derive other key(s) — The key is used only in a key derivation process that produces one or more derived keys.

A.2.6 Key Version Number

The key version number is a 2-byte field that can be used for one of two purposes, depending on the type of sensitive data being protected in the key block. The key version number can have the values shown in Table A.6.

Table A.6 — Key Version Number definition

First character	Second character	Key value field meaning
"0" (0x30)	"0" (0x30)	Key versioning is not used for this key.
"c" (0x63)	Any character	The value carried in this key block is a component of a key. Local rules will dictate the proper use of a component. Typically, a key with a specific header will be derived from two or more components with the same header (with the obvious exception of the key version bytes).
Any other combination of characters		The key version field indicates the version of the key carried in the key block. The value may optionally be used to prevent re-injection of old keys.

This field should not be confused with byte 0, the Key Block Version ID. Byte 0 indicates which version of the key block description is being used. It differentiates vendor proprietary key blocks from key blocks that comply with this interoperable definition. It also provides future flexibility, since a different version number could be used for an interoperable key block with different fields or different key block protection mechanisms. In contrast, the key version number is a tool for enforcement of local key change rules.

A.2.7 Exportability

"Exportability" refers to transfer outside the cryptographic domain in which the key is found. Secure backup of a key may be allowed, depending on local rules. Flags in this field indicate special types of keys that require unusual handling. Any key that does not follow normal security assumptions should have a notation in this field. This may mean that the key should be less trusted, like a CAPI key, or that the key assumes a high degree of trust, such as a CA key. In general, a letter other than "E" in this column means that future developers should check the definition of this type of key carefully.

The exportability byte is byte 11 of the header. Table A.7 shows the currently defined values for the exportability byte.

Table A.7 — Defined values for exportability byte

Value	Hex	Definition
"E"	0x45	Exportable under a KEK in a form meeting the requirements of ISO 11568-2
"N"	0x4E	Non-exportable
"S"	0x53	Sensitive, exportable under a KEK in a form not necessarily meeting the requirements of ISO 11568-2, e.g. backwards compatibility during key migration
Numeric values		Reserved for proprietary use

A.2.8 Optional Block ID

Each Optional Block begins with a 2-byte identifier that indicates the type of optional data. Table A.8 shows the currently defined values for the Optional Block ID. If a device encounters an Optional block ID value that it does not understand, it will reject the key block containing that Optional Block.

An optional block ID value cannot be repeated within a TR-31 key block. No two optional blocks in a key block can have the same block ID.

Figure A.2 shows simplified examples of the three possible configurations of the key block header in terms of Optional Blocks.

— The left part of the diagram shows the case when there are no Optional Blocks. In this case, the "Number of Optional Blocks" in the header is set to zero by making its value ASCII "00" (0x3030). No Optional Block information at all is appended to the base Key Block Header structure.

— The centre of the diagram shows the case where Optional Blocks are present and the total length of all Optional Blocks combined happens to be a multiple of 8, the encryption block size. In this specific example, two Optional Blocks are shown. In this case, the presence of two Optional Blocks is shown by the value 2 (ASCII "02" 0x3032) in the "Number of Optional Blocks", and the information for the two Optional Blocks immediately follows the base Key Block Header structure. Note that the order of the Optional Blocks is unconstrained in this case; they can appear in any order.

— The right part of the diagram shows the case where the creator of the key block needs to include Optional Blocks, but the total length of those Optional Blocks combined is not a multiple of 8, the encryption block length. In the example, the creator needs to include two Optional Blocks, shown as "First optional block" and "2nd optional block". Since the total length of the entire header, including Optional Blocks, will be a multiple of 8 bytes, padding is required. The padding is included by appending a third Optional Block called the Padding Block to the two Optional Blocks that were required for functional purposes. The padding block is formed so that its total length, when added to the combined total lengths of all other Optional Blocks in the header, is a multiple of 8. If used, the padding block will always be the last Optional Block in the header. There is no constraint on the order of the Optional Blocks that may precede it in the header; they can be in any order.

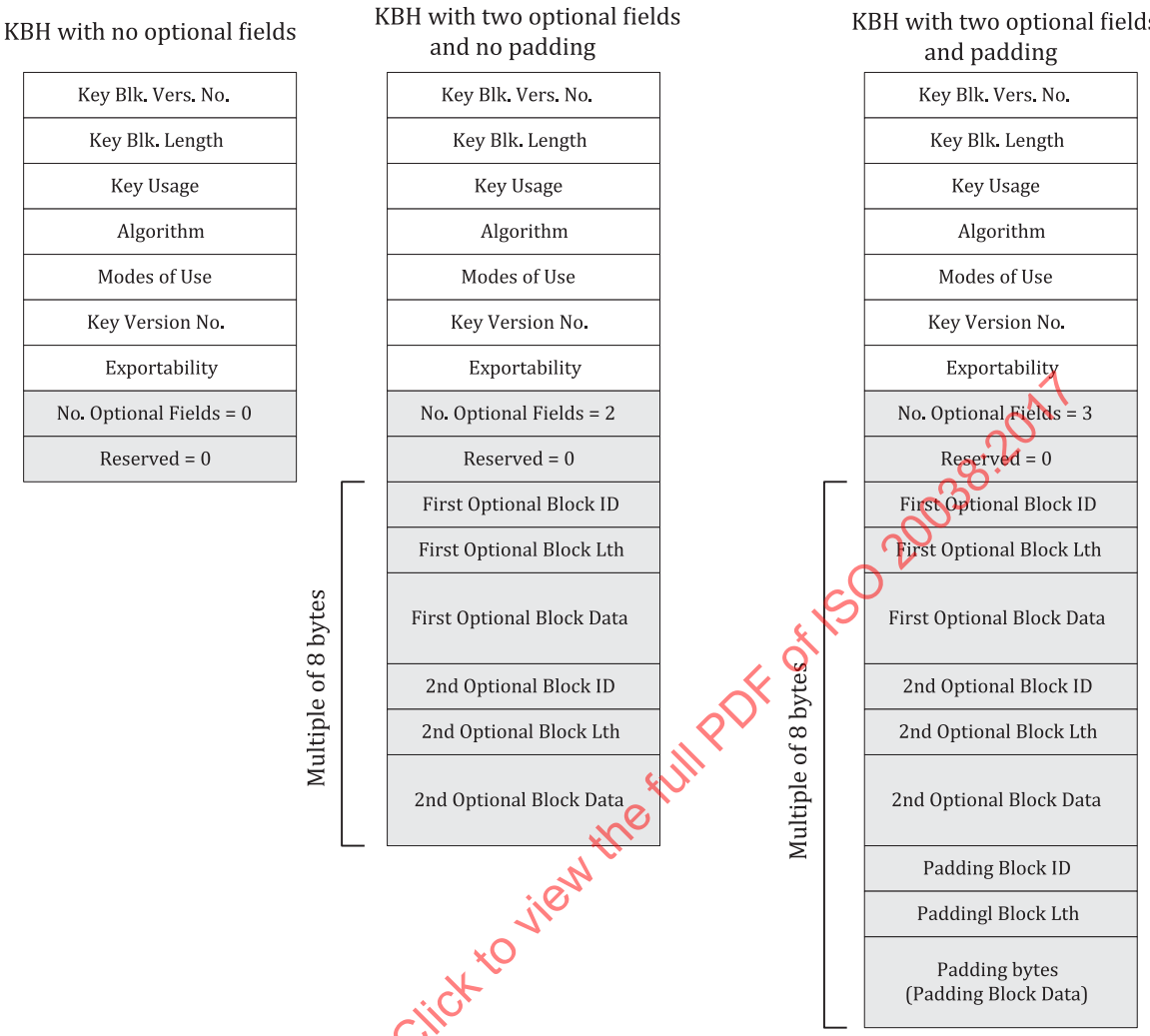


Figure A.2 — Examples of KBH and Optional Blocks

Table A.8 — Defined values for Optional Block ID

Value	Hex	Definition
"KC"	0x4B43	Key Check Value of wrapped key: computed according to X9.24-1. Not used as an integrity mechanism.
"KP"	0x4B50	Key Check Value of KBPK; computed according to X9.24-1. Not used as an integrity mechanism.
"KS"	0x4B53	Key Set Identifier, encoded in hex-ASCII; optionally used to identify the key within a system.
"KV"	0x4B56	Key Block Values: Informational field indicating the version of the key block field values. Sub-fields are defined in Table A.9.

Table A.8 *(continued)*

Value	Hex	Definition
"PB"	0x5042	A variable-length padding field used as the last Optional Block. The padding block is used to bring the total length of all Optional Blocks in the key block to a multiple of the encryption block length. The data bytes in this block are filled with readable ASCII characters.
"TS"	0x5453	Time stamp: the time and date (in UTC time format) that indicates when the key block was formed.
Numeric values		These identifiers are reserved for proprietary use and will not be defined by this report. If proprietary identifiers are used, it is the responsibility of the application owner to ensure that all necessary devices support the proprietary Optional Block identifiers that they will use.

Table A.9 — Key Block Values Version IDs Optional Block

Bytes	Definition
0–1	Key Block Field Values Version ID This informational field identifies the version of the set of approved key block field values. Each byte contains a printable ASCII character. A value of "00" indicates that this field is unused.
2–3	Provisional Values Version ID This informational field identifies that the key block contains provisional values that are not yet approved by ISO. Each byte contains a printable ASCII character. A value of "00" indicates that this field is unused.

A.2.9 Encoding

The results of the encryption and the MAC are raw binary data. For transport, that binary data will be encoded as hex-ASCII characters. For example, the 128-bit MAC result 0x12345678123456781234567812345678 (16 binary bytes) will be encoded as 32 hex-ASCII bytes hex 31323334353637383132333435363738313233343536373831323334353637383132333435363738.

The header is printable information and is not encoded.

The minimum key block length using CBC encryption for a 16-byte TDEA key can be calculated as follows:

— The KBH is 16 bytes.

— The confidential data are 32 bytes:

  — 2 bytes key length,

  — 16 bytes key,

  — 14 bytes pad.

— CMAC 16 bytes.

After encoding, the key block length is 16 + 64 +32 = 112 bytes. In this case, bytes 1 to 4 of the KBH will be "0112" (0x30313132).