TECHNICAL REPORT

ISO/TR 19038

First edition
2005-06-15

Banking and related financial services — Triple DEA — Modes of operation — Implementation guidelines

Banque et autres services financiers — Triple DEA — Modes d'opération — Lignes directrices pour la mise en œuvre

Reference number
ISO/TR 19038:2005(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

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.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In exceptional circumstances, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example), it may decide by a simple majority vote of its participating members to publish a Technical Report. A Technical Report is entirely informative in nature and does not have to be reviewed until the data it provides are considered to be no longer valid or useful.

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.

ISO/TR 19038 was prepared by Technical Committee ISO/TC 68, *Financial services*, Subcommittee SC 2, *Security management and general banking operations*.

Introduction

In order to significantly strengthen DEA (Data Encryption Algorithm) and extend its useful lifetime, the use of Triple Data Encryption Algorithm (TDEA) modes of operation has been recommended. These TDEA modes of operation not only provide greatly increased cryptographic protection, but because they are based on DEA, the TDEA learning curve for users and vendors is reduced. Since certain TDEA modes of operation can be made backward compatible with existing DEA modes of operation, the financial community may leverage its investment in standard DEA technology by using TDEA to extend its secure lifetime.

Each mode of operation provides different benefits and has different characteristics. The selection, implementation and use of a particular mode of operation is dependent upon the security requirements, risk acceptance posture, and operational needs of the financial institution and are beyond the scope of this Technical Report. This Technical Report is necessary to provide the basis for interoperability between different parties using any of the TDEA modes specified herein, provided that they use the same mode of operation and share the same secret cryptographic key(s).

This Technical Report does not replace the Data Encryption Algorithm Standard nor the Triple Data Encryption Algorithm specified in ISO/IEC 18033. DEA is the basis for the TDEA modes of operation. TDEA provides increased security in keeping with advances in computing technology and cryptanalytic techniques. TDEA may be implemented in hardware, software or a combination of hardware and software.

This Technical Report provides implementation guidelines for the modes of operation specified in ISO/IEC 10116.

It is the responsibility of the financial institution to put overall security procedures in place with the necessary controls to ensure that the process is implemented in a secure manner. Furthermore, the process should be audited to ensure compliance with the procedures.

Banking and related financial services — Triple DEA — Modes of operation — Implementation guidelines

1 Scope

This Technical Report provides the user with technical support and details for the safe and efficient implementation of the Triple Data Encryption Algorithm (TDEA) modes of operation for the enhanced cryptographic protection of digital data. The modes of operation described herein are specified for both enciphering and deciphering operations. The modes described in this Technical Report are implementations of the block cipher modes of operation specified in ISO/IEC 10116 using the Triple DEA algorithm (TDEA) specified in ISO/IEC 18033-3.

The TDEA modes of operation may be used in both wholesale and retail financial applications. The use of this Technical Report provides the basis for the interoperability of products and facilitates the development of application standards that use the TDEA modes of operation. This Technical Report is intended for use with other ISO standards using DEA.

2 Normative references

The following referenced documents are indispensable for the application 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/IEC 10116, *Information technology — Security techniques — Modes of operation for an n-bit block cipher*

ISO/IEC 18033-3, *Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers*

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

3 Terms and definitions

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

3.1
birthday phenomenon
phenomenon whereby at least two people out of a relatively small group of n people will likely share the same birthday

EXAMPLE: when $n = 23$, the probability is over ½. Generally, if one randomly picks up a number from m possible numbers with replacement, the probability to get at least one coincidence in n experiments ($n < m$) is approximated by:

$$p = 1 - e^{-n^2/2m}$$

In the above experiment, the expected number of trials before a coincidence is found is approximately $(\pi m/2)^{1/2}$. It implies that for a 64-bit block encryption operation with a fixed key, if one has a text dictionary of 2^{32} plaintext/ciphertext pairs and

2^{32} blocks of ciphertext produced from random input, then it should be expected that one block of unknown ciphertext will be found in the dictionary (see [11]).

3.2
block
binary string

EXAMPLE: a plaintext or a ciphertext, is segmented with a given length. Each segment is called a block. A plaintext (ciphertext) is encrypted (decrypted) block by block from left to right. In this Technical Report, for TCBC, TCBC-I, TOFB, TOFB-I modes, the plaintext and ciphertext are segmented into 64-bit blocks, while for TCFB and TCFB-P modes, the encryption and decryption support 1-bit, 8-bit and 64-bit plaintext and ciphertext block sizes.

3.3
bundle
collection of elements comprising a TDEA (K) key

NOTE A bundle may consist of two elements (k_1,k_2) or three elements (k_1,k_2,k_3).

3.4
ciphertext
encrypted (enciphered) data

3.5
clock cycle
time unit used in this Technical Report to define the time period for executing DEA operation once by one DEA functional block

3.6
cryptographic initialization
process of entering the initialization vector(s) into the TDEA to initialize the algorithm prior to the commencement of encryption or decryption

3.7
cryptographic key
key
parameter that determines the transformation from plaintext to ciphertext and vice versa

NOTE A DEA key is a 64-bit parameter consisting of 56 independent bits and 8 parity bits.

3.8
cryptoperiod
time span during which a specific (bundle of) key(s) is authorized for use

3.9
data encryption algorithm
DEA
algorithm specified in ISO/IEC 18033-3

NOTE The term “single DEA” implies DEA, whereas TDEA implies triple DEA as defined in this Technical Report.

3.10
DEA encryption operation
enciphering of 64-bit blocks by DEA with a key K

3.11
DEA decryption operation
deciphering of 64-bit blocks by DEA with a key K

3.12
DEA functional block
that which performs either a DEA encryption operation or a DEA decryption operation with a specified key

NOTE In this Technical Report, each DEA functional block is represented by DEA_j.

3.13
decryption
process of transforming ciphertext into plaintext

3.14
encryption
process of transforming plaintext into ciphertext

3.15
exclusive-OR
bit-by-bit modulo 2 addition of binary vectors of equal length

3.16
initialization vector
binary vector used as the input to initialize the algorithm for the encryption of a plaintext block sequence to increase security by introducing additional cryptographic variance and to synchronize cryptographic equipment

NOTE The initialization vector need not be secret.

3.17
key
see **3.7** cryptographic key

3.18
plaintext
intelligible data that has meaning and can be read or acted upon without the application of decryption

NOTE Also known as cleartext.

3.19
propagation delay
delay between the presentation of a plaintext block to a TDEA mode and the availability of the resulting ciphertext block

3.20
re-synchronization
synchronization, after being lost because of the addition or deletion of bits in one or more ciphertext blocks

EXAMPLE: if the additions or deletions can be detected, and if the appropriate number of bits can be deleted or added to the ciphertext so that the block boundaries are re-established correctly starting at block C_i such that the succeeding decrypted plaintext is correct from block P_{i+r} for some r, then we say that it is re-synchronized at C_{i+r}.

3.21
self-synchronization
automatic re-synchronization

EXAMPLE: the TCBC mode exhibits self-synchronization in the sense that if an error (including the loss of one or more entire blocks) occurs in ciphertext block C_i but no further error occurs, then C_{i+2} and succeeding ciphertext blocks are correctly decrypted to P_{i+2} and succeeding plaintext blocks (see [11] and [12]).

3.22
synchronization
where, for a plaintext with blocks P_1, P_2, … P_n if it is encrypted as a ciphertext with blocks C_1, C_2, … C_n, then for any i, $1 \leqslant i \leqslant n$, P_1, P_2, … P_i can be correctly decrypted from C_1, C_2, … C_i.

NOTE If some error occurs in the transmission of the ciphertext or if some bits are added or lost from the ciphertext, then synchronization is lost.

4 Symbols and abbreviations

C_i	i-th ciphertext block consisting of k bits, where k = 1, 8, 64.
$C^{(j)}$	j-th ciphertext substream in TCBC-I mode.
$C_{j,i}$	i-th block in j-th ciphertext substream.
CBC	Cipher block chaining.
CFB	Cipher feedback.
D_{K_j}	A DEA decryption operation with key "K_j".
DEA	The data encryption algorithm specified in ISO/IEC 18033-3.
DEA_j	j-th DEA functional block.
E_{K_j}	A DEA encryption operation with key "K_j".
ECB	Electronic codebook.
I_i	i-th input block of encryption operation consisting of 64 bits in TCFB, TCFB-P, TOFB, and TOFB-I modes of operation.
i	Index of blocks.
IV	Initialization vector.
j	Index of functional blocks, index of keys, and index of plaintext substreams (ciphertext substreams) in TCBC-I.
h	A given counter value of a clock cycle. It is for describing the actions of each DEA functional block at $t = h - 1$, $t = h$, and $t = h + 1$. In the interleaved or pipelined mode, h is used to describe at clock cycle $t = 3(h - 1) + j$, j = 1, 2, 3, the simultaneous actions of three functional blocks. In the interleaved mode, h is used as an index of blocks for tripartition of a plaintext.
k	Size of blocks, a parameter for shifting functions S_k, k = 1, 8, 64.
K	Cryptographic key.
n	Number of blocks in a plaintext.
O_i	i-th output block of encryption operation consisting of 64 bits in TCFB, TCFB-P, TOFB, and TOFB-I modes of operation.
$\{O_i\}_k$	Leftmost k bits of O_i, k =1, 8, 64. When k = 64, $\{O_i\}_k = O_i$.
OFB	Output feedback.

P_i — i-th plaintext block consisting of k bits, where k = 1, 8, 64.

$P^{(j)}$ — j-th plaintext substream in TCBC-I mode.

$P_{j,i}$ — i-th plaintext block in j-th plaintext substream.

S_k — "k-Shifting" function, defined as follows:

Given a 64-bit block I = (i_1, i_2, …, i_{64}) and a k-bit block C = (c_1, c_2, … c_k) where k = 1, 8, 64, the shifting function S_k(I | C) produces a 64-bit block:

$S_k(I \mid C) = \{i_{k+1}, i_{k+2}, ..., i_{64}, c_1, c_2, ... c_k\}$

where the bits of I have been shifted left by k places, discarding i_1, i_2, ... i_k and placing the k bits of C in the rightmost k places of I. When k = 64, S_k(I | C) = C.

t — Counter of clock cycle starting from 1.

TCBC — TDEA cipher block chaining.

TCBC-I — TDEA cipher block chaining-interleaved.

TCFB — TDEA cipher feedback.

TCFB-P — TDEA cipher feedback-pipelined.

TDEA — Triple data encryption algorithm.

TECB — TDEA electronic codebook.

TOFB — TDEA output feedback.

TOFB-I — TDEA output feedback-interleaved.

$X \oplus Y$ — "Exclusive-or" operation of X and Y.

X || Y — Concatenation of X and Y.

|X| — Length of binary string X.

5 Specifications

5.1 TDEA encryption/decryption operation

In this Technical Report, each TDEA encryption/decryption operation is a compound operation of DEA encryption and decryption operations as specified in ISO/IEC 18033-3. The following operations are to be used in this Technical Report.

a) TDEA encryption operation: the transformation of a 64-bit block I into a 64-bit block O that is defined as follows:

$O = E_{K3}(D_{K2}(E_{K1}(I)))$.

b) TDEA decryption operation: the transformation of a 64-bit block I into a 64-bit block O that is defined as follows:

$O = D_{K1}(E_{K2}(D_{K3}(I)))$.

5.2 Keying options

This Technical Report uses the following keying options for the TDEA key.

a) Keying Option 1: K1, K2 and K3 are independent keys;

b) Keying Option 2: K1 and K2 are independent keys and K3 = K1;

c) Keying Option 3: K1 = K2 = K3.

NOTE Keying option 3 is not recommended as its use reduces the strength of the TDEA operation to that of DEA.

5.3 TDEA modes of operation

This Technical Report discusses:

a) TDEA Electronic Codebook Mode (TECB);

b) TDEA Cipher Block Chaining Mode (TCBC);

c) TDEA Cipher Block Chaining Mode — Interleaved (TCBC-I);

d) TDEA Cipher Feedback Mode (TCFB);

e) TDEA Cipher Feedback Mode — Pipelined (TCFB-P);

f) TDEA Output Feedback Mode (TOFB);

g) TDEA Output Feedback Mode — Interleaved (TOFB-I).

These are triple DEA implementations of the ECB, CBC, CFB, and OFB modes of operation specified in ISO/IEC 10116. For applications in which high TDEA encryption/decryption throughput is important or in which propagation delay must be minimized, the new interleaved (for TCBC and TOFB) and pipelined (for TCFB) modes are provided.

5.4 Backward compatibility

In this Technical Report, a TDEA mode of operation is backward compatible with its single DEA counterpart if, with a proper keying option for TDEA operation,

a) an encrypted plaintext computed using a single DEA mode of operation can be decrypted correctly by a corresponding TDEA mode of operation;

b) an encrypted plaintext computed using a TDEA mode of operation can be decrypted correctly by a corresponding single DEA mode of operation.

When using Keying Option 3, TECB, TCBC, TCFB and TOFB modes are backward compatible with single DEA modes of operation ECB, CBC, CFB, OFB respectively. It should be noted that backward compatibility with single DEA reduces the security of the TDEA mode to that of the single DEA mode.

5.5 Schedule of DEA functional blocks

In this Technical Report, one clock cycle is defined as the time period for a DEA functional block to perform $E_K(I)$ or $D_K(I)$. In a schedule of DEA functional blocks, $O = E_{K3}(D_{K2}(E_{K1}(I)))$ is broken down into three actions. Each action is finished in one clock cycle by a functional block. The following table shows the schedule for three DEA functional blocks in performing $E_{K3}(D_{K2}(E_{K1}(I)))$.

	Input	DEA_1	DEA_2	DEA_3	Output
t = 1	I	$E_{K1}(I)$			
t = 2			$D_{K2}(E_{K1}(I))$		
t = 3				$E_{K3}(D_{K2}(E_{K1}(I)))$	O

5.6 Improving throughput and minimizing propagation

As is shown in 5.5, a valid TDEA output block, O, is produced only after the input block, I, has propagated through the three individual DEA functional blocks. That is, it takes three clock cycles to get the output. Within each clock cycle, only one DEA functional block is actively encrypting/decrypting data. This configuration provides the slowest throughput speed and greatest propagation delay.

In order to improve the throughput and minimize the propagation, interleaved and pipelined modes of operation are provided. They are TCBC-I, TCFB-P, and TOFB-I modes. In an interleaved mode, the plaintext sequence is split into three subsequences of plaintext. The encryption can be done simultaneously. In a pipelined mode, the encryption is initiated with three IVs at three clock cycles so that after initialization, the three DEA functional blocks can process the data simultaneously. The interleaved and pipelined configurations are intended for systems equipped with multiple DEA processors.

In a mode of operation, which is interleaved or pipelined, a schedule defines simultaneous actions of multiple DEA functional blocks within each clock cycle.

5.7 Keys and initialization vectors

The following specifications for keys and initialization vectors shall be met in implementing the TDEA modes of operation.

a) For all TDEA modes of operation, the three cryptographic keys (K_1, K_2, K_3) define a TDEA key bundle. The bundle and the individual keys shall:

   1) be secret;

   2) be generated randomly;

   3) have integrity whereby each key in the bundle has not been altered in an unauthorized manner since the time it was generated, transmitted, or stored by an authorized source;

   4) be used in the appropriate order as specified by the particular mode;

   5) be considered a fixed quantity in which an individual key cannot be manipulated while leaving the other two keys unchanged;

   6) cannot be unbundled for any purpose.

b) IVs shall meet the following attributes:

   1) for TECB, no IV is used;

   2) for all modes using IV(s), the IV(s) may be public information;

   3) in the cryptoperiod of a given bundle of keys, a new IV or three new IVs shall be generated whenever the encryption process is reinitialized.

c) IVs shall be generated by one of the following methods, which are given in order of preference:

   1) generate randomly or

   2) use values of a monotonically increasing counter such that the values will not be repeated during the cryptoperiod of the keys.

d) When three IVs are required, then generate IV by method 1) or method 2) in item c) such that

   1) $IV_1 = IV$;

   2) $IV_2 = IV_1 + R_1 \bmod 2^{64}$, where R_1 = (5555555555555555);

   3) $IV_3 = IV_1 + R_2 \bmod 2^{64}$, where R_2 = (AAAAAAAAAAAAAAAA).

— In the above equations for IV_2 and IV_3, the binary strings or hexadecimal strings are converted to integers. The operation is integer addition modulo 2^{64}. The operation results shall be converted back to binary strings or hexadecimal strings.

— When the IV is generated by method 2), i.e. values of a monotonically increasing counter are used, the IV value, once converted to an integer, shall be smaller than R_1. R_1 is considered as the integer converted from (5555555555555555).

5.8 Input and output

For the input and output of the TDEA modes of operation, the following specification applies.

a) The input and output of a TDEA operation are 64-bit blocks. For TCFB and TCFB-P modes, the plaintext/ciphertext block size may be 1 bit, 8 bits, or 64 bits. For TECB, TCBC, TCBC-I, TOFB, TOFB-I modes, the plaintext/ciphertext requires complete data blocks of 64 bits for its operation. Blocks of less than 64 bits require special handling, which is not addressed in this Technical Report.

b) As knowledge of intermediate results reduces the strength of the TDEA to that of DEA, implementations of any TDEA mode of operation should ensure that the intermediate results between the different DEA functional blocks are not revealed. Thus to protect against attacks on the device implementing TDEA the device itself must be a physically secure device and must not reveal intermediate results.

c) The initial output data shall be suppressed because it is invalid and may create a security risk if revealed. Each mode of operation shall specify how many bits of output should be suppressed.

6 TDEA modes of operation

6.1 TDEA electronic codebook mode of operation

6.1.1 TECB definition

6.1.1.1 General

Three keying options are defined for TECB mode as described in Section 6.2.

6.1.1.2 TECB encryption

— **Input**: $P_1, P_2, \ldots P_n$; $|P_i| = 64$.

— **Output**: $C_1, C_2, \ldots C_n$; $|C_i| = 64$.

For $i = 1, 2, \ldots n$, do

1) $C_i = E_{K3}(D_{K2}(E_{K1}(P_i)))$;

2) Output C_i.

The TECB encryption is shown in Figure 1.

Suppose that three DEA functional blocks, DEA_1, DEA_2, and DEA_3, are simultaneously clocked. Let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation and DEA_3 perform the E_{K3} operation. At each clock cycle, each DEA_j performs the specified operation with the input from DEA_{j-1} (or input buffer) and passes the result to DEA_{j+1} (or output buffer). Table 1 shows how three DEA functional blocks are scheduled. At the first two clock cycles, the 128-bit output of the TDEA should be suppressed since valid output is not produced.

Table 1 — Schedule of TECB encryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
$t = 1$	P_1	$E_{K1}(P_1)$	idle	idle	N/A
$t = 2$	P_2	$E_{K1}(P_2)$	$D_{K2}(E_{K1}(P_1))$	idle	N/A
$t = 3$	P_3	$E_{K1}(P_3)$	$D_{K2}(E_{K1}(P_2))$	$E_{K3}(D_{K2}(E_{K1}(P_1)))$	C_1
$t = 4$	P_4	$E_{K1}(P_4)$	$D_{K2}(E_{K1}(P_3))$	$E_{K3}(D_{K2}(E_{K1}(P_2)))$	C_2
		…	…	…	
$t = h$	P_h	$E_{K1}(P_h)$	$D_{K2}(E_{K1}(P_{h-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{h-2})))$	C_{h-2}
		…	…	…	
$t = n$	P_n	$E_{K1}(P_n)$	$D_{K2}(E_{K1}(P_{n-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{n-2})))$	C_{n-2}
$t = n + 1$	N/A	idle	$D_{K2}(E_{K1}(P_n))$	$E_{K3}(D_{K2}(E_{K1}(P_{n-1})))$	C_{n-1}
$t = n + 2$	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_n)))$	C_n

For example:

If the plaintext to be enciphered is “Now is the time for all good men” which when encoded in ASCII is represented in hexadecimal as:

`X'4E6F772069732074 68652074696D6520 666F7220616C6C20 676F6F64206D656E'`

is enciphered using TECB mode with Key `X'0123456789ABCDEFFEDCBA9876543210'` the following results.

Table 2 — Example of TECB encryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
t = 1	P_1 4E6F772069732074	$E_{K1}(P_1)$ 3FA40E8A984D4815	idle	idle	N/A
t = 2	P_2 68652074696D6520	$E_{K1}(P_2)$ 6A271787AB8883F9	$D_{K2}(E_{K1}(P_1))$ 0EF220F064194595	idle	N/A
t = 3	P_3 666F7220616C6C20	$E_{K1}(P_3)$ 893D51EC4B563B53	$D_{K2}(E_{K1}(P_2))$ 174B332E073DE8AF	$E_{K3}(D_{K2}(E_{K1}(P_1)))$ D80A0D8B2BAE5E4E	C_1 D80A0D8B2BAE5E4E
t = 4	P_4 676F6F64206D656E	$E_{K1}(P_4)$ 73C1ADB2171F7894	$D_{K2}(E_{K1}(P_3))$ 47B3F7F0E82E1F35	$E_{K3}(D_{K2}(E_{K1}(P_2))$ 6A0094171ABCFC27	C_2 6A0094171ABCFC27
t = 5	N/A	idle	$D_{K2}(E_{K1}(P_4))$ 7A1E4ABD1DA455C6	$E_{K3}(D_{K2}(E_{K1}(P_3)))$ 75D2235A706E232C	C_3 75D2235A706E232C
t = 6	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_4)))$ 41B637F9AB83FFD4	C_4 41B637F9AB83FFD4

6.1.1.3 TECB decryption

— **Input**: $C_1, C_2, \ldots C_n$; $|C_i| = 64$.

— **Output**: $P_1, P_2, \ldots P_n$; $|P_i| = 64$.

For $i = 1, 2, \ldots, n$, do

1) $P_i = D_{K1}(E_{K2}(D_{K3}(C_i)))$;

2) Output P_i.

The TECB decryption is shown in Figure 1.

Suppose that three DEA functional blocks, DEA_1, DEA_2, and DEA_3, are simultaneously clocked. Let DEA_1 perform the D_{K3} operation, DEA_2 perform the E_{K2} operation, and DEA_3 perform the D_{K1} operation. At each clock cycle, each DEA_j performs the specified operation with the input from DEA_{j-1} (or input buffer) and passes the result to DEA_{j+1} (or output buffer). Table 2 shows how three DEA functional blocks are scheduled. At the first two clock cycles, the 128-bit output of the TDEA should be suppressed since valid output is not produced.

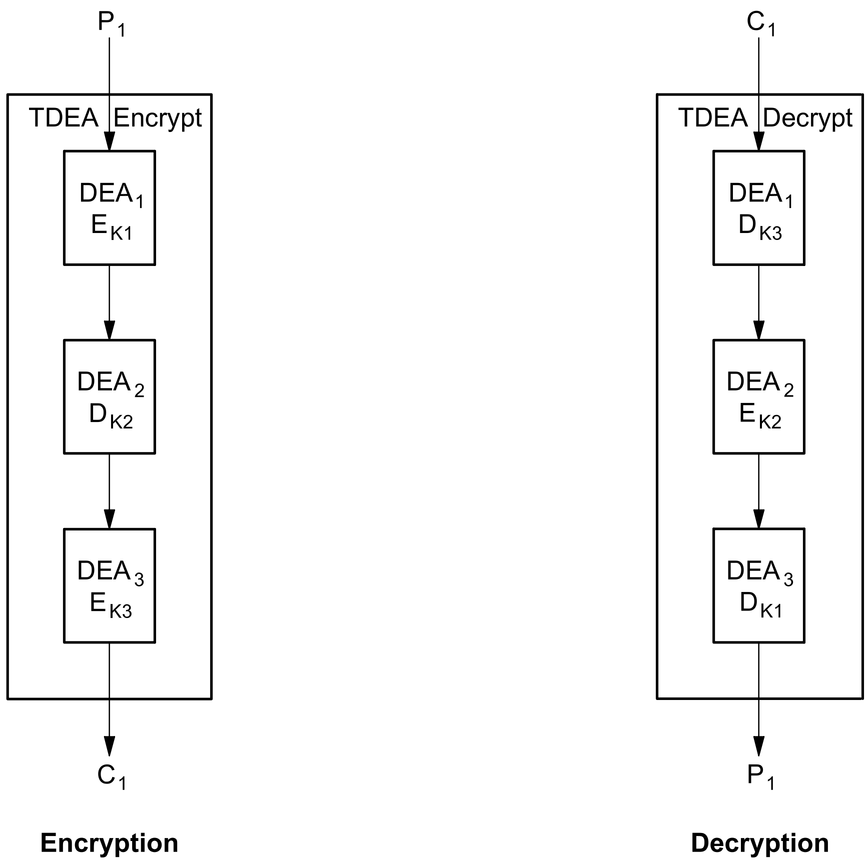

Figure 1 — TDEA electronic codebook

Table 3 — Schedule of TECB decryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
$t = 1$	C_1	$D_{K3}(C_1)$	idle	idle	N/A
$t = 2$	C_2	$D_{K3}(C_2)$	$E_{K2}(D_{K3}(C_1))$	idle	N/A
$t = 3$	C_3	$D_{K3}(C_3)$	$E_{K2}(D_{K3}(C_2))$	$D_{K1}(E_{K2}(D_{K3}(C_1)))$	P_1
$t = 4$	C_4	$D_{K3}(C_4)$	$E_{K2}(D_{K3}(C_3))$	$D_{K1}(E_{K2}(D_{K3}(C_2)))$	P_2
		…	…	…	
$t = h$	C_h	$D_{K3}(C_h)$	$E_{K2}(D_{K3}(C_{h-1}))$	$D_{K1}(E_{K2}(D_{K3}(C_{h-2})))$	P_{h-2}
		…	…	…	
$t = n$	C_n	$D_{K3}(C_n)$	$E_{K2}(D_{K3}(C_{n-1}))$	$D_{K1}(E_{K2}(D_{K3}(C_{n-2})))$	P_{n-2}
$t = n + 1$	N/A	idle	$E_{K2}(D_{K3}(C_n))$	$D_{K1}(E_{K2}(D_{K3}(C_{n-1})))$	P_{n-1}
$t = n + 2$	N/A	idle	idle	$D_{K1}(E_{K2}(D_{K3}(C_n)))$	P_n

6.1.2 TECB properties

a) When the three keys are set to be the same (see Keying Option 3), the TECB mode of operation is backward compatible with the single DEA ECB mode using the same key.

b) In TECB decryption, a single bit error in a ciphertext input block C_i will result, upon decryption, in a maximum of 64 bits of error in plaintext block P_i. The average error rate for such a plaintext block P_i will be 50 %. However, there is no error propagation to other blocks, i.e. the plaintext error brought about by C_i only occurs in P_i.

c) Synchronization is required for the TECB mode.

If less than 64 bits are added or deleted in a ciphertext block C_i, then synchronization will be lost. If the bit additions or deletions are detected and if the proper number of bits are removed from or added to C_i, then the decryption may be re-synchronized such that, except for P_i, the succeeding decrypted blocks are correct. Otherwise, the decryption of C_i and succeeding decrypted blocks are all in error.

If one or several entire blocks are lost or added, then the same number of blocks is lost or added in the decrypted plaintext. However, the succeeding decrypted blocks after the additions or deletions are correct if no further error occurs.

d) As for the single DEA ECB mode, the TECB mode will produce identical ciphertext blocks for identical plaintext blocks under the action of the same key. This characteristic makes TECB unsuitable for general data encryption where the pattern of plaintext block repetitions will reveal significant information about the plaintext (e.g. digitized pictures). It is suitable for those applications where the input data has high variability or the data consists of a single block.

e) TECB is a block method of encryption, and therefore requires complete data blocks of 64 bits for its operation. Blocks of less than 64 bits require special handling, which is not addressed in this Technical Report.

6.2 TDEA cipher block chaining mode of operation

6.2.1 TCBC definition

6.2.1.1 General

This mode of operation is the CBC mode (with parameter m equal to 1) defined by ISO 10116 using TDEA as the n-bit block cipher. See Figures 2 and 3.

Three keying options are defined for the TCBC mode as described in 5.2.

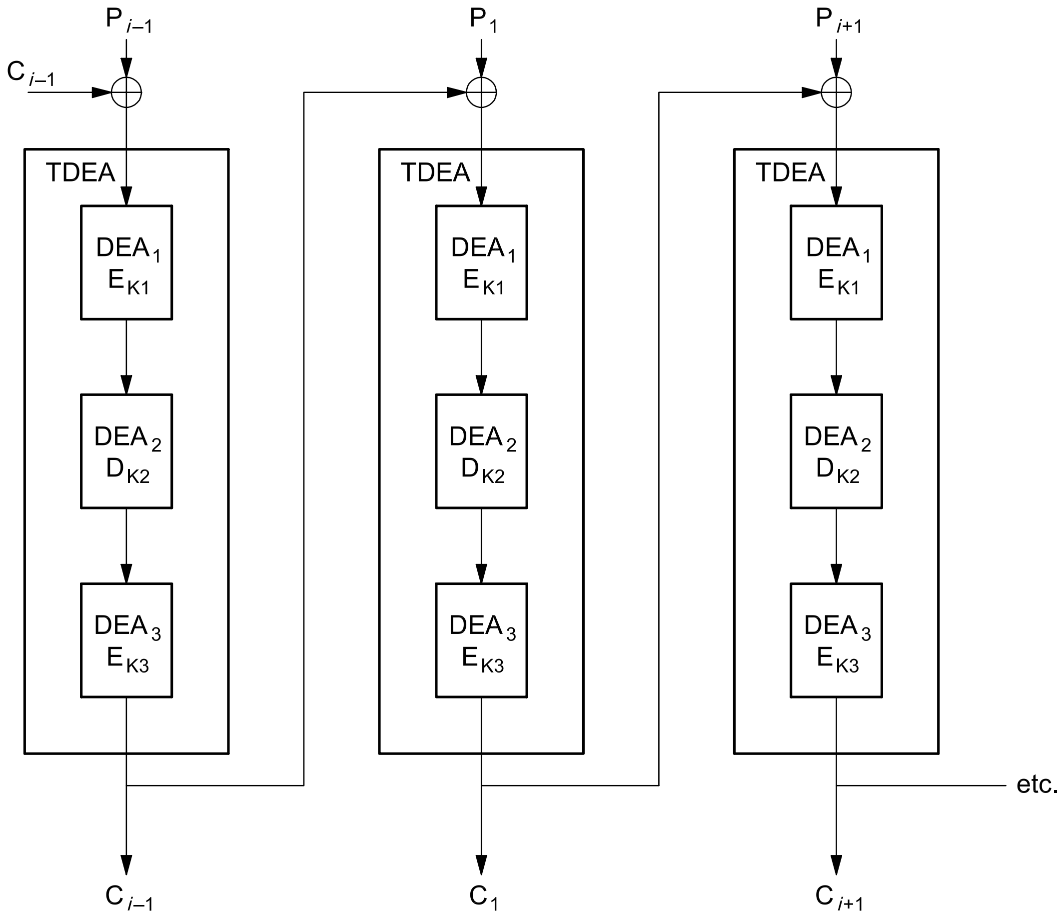

Figure 2 — TDEA cipher block chaining — Encryption

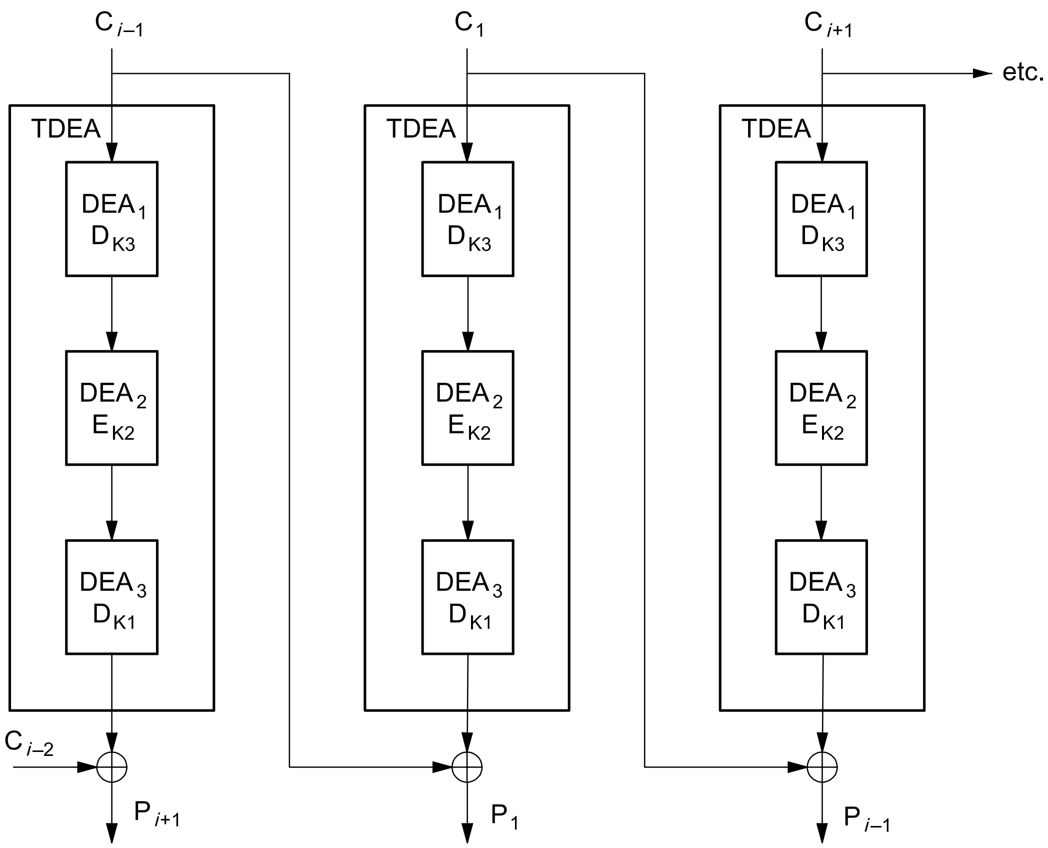

Figure 3 — TDEA cipher block chaining — Decryption

6.2.1.2 TCBC encryption

— **Input**: P_1, P_2, …, P_n; IV; $|P_i| = 64$, $|IV| = 64$.

— **Output**: C_1, C_2, …, C_n; $|C_i| = 64$.

a) $C_0 = IV$.

b) For $i = 1, 2, \ldots n$, do

1) $C_i = E_{K3}(D_{K2}(E_{K1}(P_i \oplus C_{i-1})))$;

2) Output C_i.

In TCBC encryption, let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. If at clock cycle t =1, DEA_1 performs $E_{K1}(P_1)$, then at $t = 2$ and $t = 3$, DEA_1 must be idle, since the next input for DEA_1 is $P_2 \oplus C_1$ where C_1 is the output at $t = 3$. So it is impossible for DEA_1, DEA_2, DEA_3 to perform DEA operations simultaneously for TCBC encryption. The schedule for DEA_1, DEA_2, and DEA_3 is such that at time $t = 3(h - 1) + j$, where $j = 1, 2, 3$, and $h = 1, 2, \ldots n$, only DEA_j is activated and the other two have to be idle.

For example:

If the plaintext to be enciphered is "Now is the time for all good men" which when encoded in ASCII is represented in hexadecimal as:

`X'4E6F772069732074 68652074696D6520 666F7220616C6C20 676F6F64206D656E'`

is enciphered using TCBC mode with Key `X'0123456789ABCDEFFEDCBA9876543210'` and an IV of `X'0000000000000000'` the following results.

Table 4 — Example of TCBC encryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
t = 1	P_1 4E6F772069732074 ⊕ 0000000000000000	$E_{K1}(P_1)$ 3FA40E8A984D4815	idle	idle	N/A
t = 2	N/A	idle	$D_{K2}(E_{K1}(P_1))$ 0EF220F064194595	idle	N/A
t = 3	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_1)))$ D80A0D8B2BAE5E4E	C_1 D80A0D8B2BAE5E4E
t = 4	P_2 68652074696D6520 ⊕ D80A0D8B2BAE5E4E	$E_{K1}(P_2)$ 38D4E8D33C87C3F3	idle	idle	N/A
t = 5	N/A	idle	$D_{K2}(E_{K1}(P_2))$ 2259D3308D542A00	idle	N/A
t = 6	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_2)))$ 319E5E68C3E8891B	C_2 319E5E68C3E8891B
t = 7	P_3 666F7220616C6C20 ⊕ 319E5E68C3E8891B	$E_{K1}(P_3)$ 9A08D22BD5F1B809	idle	idle	N/A
t = 8	N/A	idle	$D_{K2}(E_{K1}(P_3))$ 810A66AC66A0A869	idle	N/A
t = 9	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_3)))$ 93462A6DB9B4A4D1	C_3 93462A6DB9B4A4D1
t = 10	P_4 676F6F64206D656E ⊕ 93462A6DB9B4A4D1	$E_{K1}(P_4)$ 2A389A64B537E2D5	idle	idle	N/A
t = 11	N/A	idle	$D_{K2}(E_{K1}(P_4))$ C8247F029F02DFBD	idle	N/A
t = 12	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_4)))$ 976E095D6DA30EE9	C_4 976E095D6DA30EE9

6.2.1.3 TCBC decryption

— **Input**: C_1, C_2, … C_n; IV; $|C_i| = 64$, $|IV| = 64$.

— **Output**: P_1, P_2, … P_n; $|P_i| = 64$.

a) $C_0 = IV$.

b) For $i = 1, 2, ..., n$, do

1) $P_i = D_{K1}(E_{K2}(D_{K3}(C_i))) \oplus C_{i-1}$;

2) Output P_i.

TCBC decryption differs from TCBC encryption, in that, if DEA_1 performs the D_{K3} operation, DEA_2 performs the E_{K2} operation, and DEA_3 performs the D_{K1} operation, DEA_1, DEA_2, DEA_3 can perform DEA operations simultaneously. Refer to Table 2 in 6.1.1.2 to get the schedule of DEA functional blocks. Notice that if TCBC decryption is implemented with multiple DEA processors according to Table 2, the output of the DEA_3 needs to be XORed with C_{i-1} in order to get plaintext block P_i.

6.2.2 TCBC properties

a) When the three keys are set to be the same (see keying option 3), the TCBC mode of operation is backward compatible with the single DEA CBC mode using the same key.

b) For this mode, one or more bit errors within a single ciphertext block will affect the decryption of two blocks: the block in which the error occurs and the succeeding block. If the error(s) occur in ciphertext block C_{i-1}, then each bit of plaintext block P_{i-1} will have an average error rate of 0,5. The plaintext block P_i will have only those bits in error which correspond directly to the ciphertext bits in error. If no error occurs in C_i, then P_{i+1} will be decrypted correctly, i.e. limited error propagation.

c) Synchronization is required for the TCBC mode of operation. If less than 64 bits are added or are lost in ciphertext block C_{i-1}, then synchronization is lost. If the bit additions or deletions are detected and if the proper number of bits is removed from or added to C_{i-1}, then the decryption may be resynchronized such that, except for P_{i-1} and P_i, the succeeding decrypted blocks are correct. Otherwise, P_{i-1} and the succeeding decrypted blocks are all in error.

   If r entire blocks are added or lost right after C_{i-1} (i.e., blocks C_i to C_{i+r-1} are added or lost), then P_i is an error block, and r blocks are added or lost after P_i (i.e., P_{i+1} to P_{i+r} are added or lost). However, the blocks after the added or lost r blocks can be correctly decrypted if no further error occurs.

d) If the same IV is used with each new plaintext, then TCBC will produce identical ciphertext for identical plaintext using exactly the same key bundle. A new IV may be used with each new plaintext under the action of the same key.

e) Since TCBC is a block method of encryption, it needs to operate on complete blocks of 64 bits. Blocks of less than 64 bits require special handling, which is not addressed in this Technical Report.

6.3 TDEA cipher block chaining mode of operation — Interleaved

6.3.1 TCBC-I definition

6.3.1.1 General

To increase the performance of TCBC, the mode can be modified by dividing the plaintext into three plaintext substreams. Three keying options are defined for TCBC-I mode as in 5.2.

This mode of operation is the CBC mode (with parameter m equal to 3) defined by ISO/IEC 10116 using TDEA as the n-bit block cipher.

6.3.1.2 Plaintext division

Let P = (P_1, P_2, … P_n) be a plaintext with n blocks. P's blocks are re-indexed in the following way.

For each P_i, $1 \leqslant i \leqslant n$, first find a pair of integers (j, h), j = 1, 2, or 3 and $h > 0$ such that $i = 3(h - 1) + j$. Then re-index P_i as $P_{j,h}$.

For example, for P_8, since $8 = 3(3 - 1) + 2$, $j = 2$ and $h = 3$. P_8 is re-indexed as $P_{2,3}$.

The plaintext P = (P_1, P_2, … P_n) is re-indexed as

$$P = (P_{1,1}, P_{2,1}, P_{3,1}; P_{1,2}, P_{2,2}, P_{3,2}; P_{1,3}, P_{2,3}, P_{3,3}; ...; P_{1,h}, P_{2,h}, P_{3,h}; \ldots; \ldots P_{j', nj'}),$$

where the last block $P_{j',nj'} = P_n$ and $n = 3(n_{j'}\text{-}1) + j'$, j' = 1, 2, or 3.

Then divide P to three plaintext sub-streams

$$P^{(1)} = P_{1,1}; P_{1,2}; ...; P_{1,n1};$$

$$P^{(2)} = P_{2,1}; P_{2,2}; ...; P_{2,n2};$$

$$P^{(3)} = P_{3,1}; P_{3,2}; ...; P_{3,n3},$$

where the three plaintext substreams may not have the same length and depend on the number n in the following ways:

if $n = 0 \bmod 3$, then $n_1 = n_2 = n_3 = n/3$;

if $n = 1 \bmod 3$, then $n_1 = (n + 2)/3$, $n_2 = n_3 = (n - 1)/3$;

if $n = 2 \bmod 3$, then $n_1 = n_2 = (n + 1)/3$, and $n_3 = (n - 2)/3$.

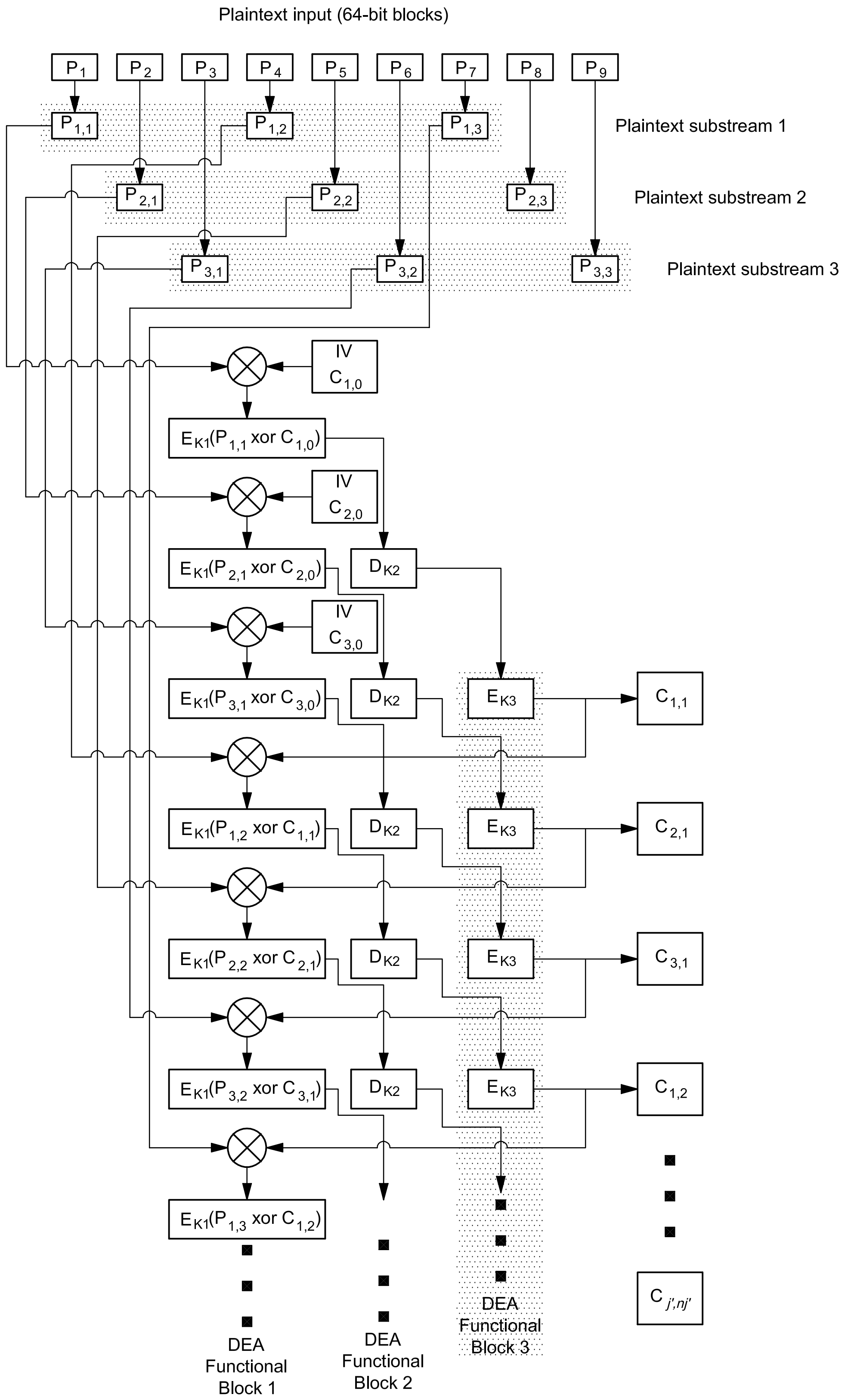

Figure 4 — TCBC-I encryption

6.3.1.3 TCBC-I encryption

In TCBC-I encryption, each plaintext substream $P^{(j)}$ is encrypted by the algorithm in 7.2.1.1 with initialization vector IV_j.

— **Input**: $P^{(1)} = P_{1,1}; P_{1,2}; ...; P_{1,n_1}$;

$P^{(2)} = P_{2,1}; P_{2,2}; ...; P_{2,n_2}$;

$P^{(3)} = P_{3,1}; P_{3,2}; ...; P_{3,n_3}$;

IV_1, IV_2, IV_3, where $|P_{j,i}| = 64$ and $|IV_j| = 64$.

— **Output**: $C^{(1)} = C_{1,1}; C_{1,2}; ...; C_{1,n_1}$;

$C^{(2)} = C_{2,1}; C_{2,2}; ...; C_{2,n_2}$;

$C^{(3)} = C_{3,1}; C_{3,2}; ...; C_{3,n_3}$, $|C_{i,j}| = 64$.

For $j = 1, 2, 3$, do

$C_{j,0} = IV_j$.

For $h = 1, 2, \ldots n_j$, do

$C_{j,h} = E_{K3}(D_{K2}(E_{K1}(P_{j,h} \oplus C_{j,h-1})))$;

Output $C_{j,h}$.

The above algorithm gives the relationship of plaintext blocks and ciphertext blocks in terms of three plaintext substreams and three ciphertext substreams.

This results in the following ciphertext stream

$C = (C_{1,1}, C_{2,1}, C_{3,1}; C_{1,2}, C_{2,2}, C_{3,2}; C_{1,3}, C_{2,3}, C_{3,3}; ...; C_{1,h}, C_{2,h}, C_{3,h}; \ldots; \ldots C_{j', nj'})$.

With three DEA functional blocks, DEA_1, DEA_2, and DEA_3, which are simultaneously clocked, the encryption of three plaintext substreams $P^{(1)}$, $P^{(2)}$, $P^{(3)}$ can be interleaved. Let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. Table 5 shows how three DEA functional blocks are scheduled (as an example, suppose that n mod 3 = 0. In this case, $n_1 = n_2 = n_3 = n/3$). At the first two clock cycles, the 128-bit output of the TDEA should be suppressed since valid output is not produced.

Table 5 — Schedule of TCBC-I encryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
$t = 1$	$P_{1,1} \oplus C_{1,0}$	$E_{K1}(P_{1,1} \oplus C_{1,0})$	idle	idle	N/A
$t = 2$	$P_{2,1} \oplus C_{2,0}$	$E_{K1}(P_{2,1} \oplus C_{2,0})$	$D_{K2}(E_{K1}(P_{1,1} \oplus C_{1,0}))$	idle	N/A
$t = 3$	$P_{3,1} \oplus C_{3,0}$	$E_{K1}(P_{3,1} \oplus C_{3,0})$	$D_{K2}(E_{K1}(P_{2,1} \oplus C_{2,0}))$	$E_{K3}(D_{K2}(E_{K1}(P_{1,1} \oplus C_{1,0})))$	$C_{1,1}$
$t = 4$	$P_{1,2} \oplus C_{1,1}$	$E_{K1}(P_{1,2} \oplus C_{1,1})$	$D_{K2}(E_{K1}(P_{3,1} \oplus C_{3,0}))$	$E_{K3}(D_{K2}(E_{K1}(P_{2,1} \oplus C_{2,0})))$	$C_{2,1}$
$t = 5$	$P_{2,2} \oplus C_{2,1}$	$E_{K1}(P_{2,2} \oplus C_{2,1})$	$D_{K2}(E_{K1}(P_{1,2} \oplus C_{1,1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{3,1} \oplus C_{3,0})))$	$C_{3,1}$
$t = 6$	$P_{3,2} \oplus C_{3,1}$	$E_{K1}(P_{3,2} \oplus C_{3,1})$	$D_{K2}(E_{K1}(P_{2,2} \oplus C_{2,1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{1,2} \oplus C_{1,1})))$	$C_{1,2}$
		…	…	…	
$t = 3(h-1) + 1$	$P_{1,h} \oplus C_{1,h-1}$	$E_{K1}(P_{1,h} \oplus C_{1,h-1})$	$D_{K2}(E_{K1}(P_{3,h-1} \oplus C_{3,h-2}))$	$E_{K3}(D_{K2}(E_{K1}(P_{2,h-1} \oplus C_{2,h-2})))$	$C_{2,h-1}$
$t = 3(h-1) + 2$	$P_{2,h} \oplus C_{2,h-1}$	$E_{K1}(P_{2,h} \oplus C_{2,h-1})$	$D_{K2}(E_{K1}(P_{1,h} \oplus C_{1,h-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{3,h-1} \oplus C_{3,h-2})))$	$C_{3,h-1}$
$t = 3(h-1) + 3$	$P_{3,h} \oplus C_{3,h-1}$	$E_{K1}(P_{3,h} \oplus C_{3,h-1})$	$D_{K2}(E_{K1}(P_{2,h} \oplus C_{2,h-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{1,h} \oplus C_{1,h-1})))$	$C_{1,h}$
		…	…	…	
$t = n = 3n_3$	$P_{3,n_3} \oplus C_{3,n3-1}$	$E_{K1}(P_{3,n_3} \oplus C_{3,n3-1})$	$D_{K2}(E_{K1}(P_{2,n_3} \oplus C_{2,n3-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{1,n_3} \oplus C_{1,n3-1})))$	$C_{1,n3}$
$t = n + 1$	N/A	idle	$D_{K2}(E_{K1}(P_{3,n_3} \oplus C_{3,n3-1}))$	$E_{K3}(D_{K2}(E_{K1}(P_{2,n_3} \oplus C_{2,n3-1})))$	$C_{2,n3}$
$t = n + 2$	N/A	idle	idle	$E_{K3}(D_{K2}(E_{K1}(P_{3,n_3} \oplus C_{3,n3-1})))$	$C_{3,n3}$

Note that even though the plaintext is divided into three plaintext substreams, in TCBC-I mode, the order of input blocks is the same as that of the original plaintext $P_1, P_2, \dots P_n$. If the output blocks are indexed according to the order of output as $C_1, C_2, \dots C_n$, then three corresponding ciphertext substreams $C^{(1)}, C^{(2)}, C^{(3)}$ are derived from ciphertext $C_1, C_2, \dots C_n$ by division as described in 6.3.1.1.

6.3.1.4 TCBC-I decryption

In TCBC-I decryption, the ciphertext $C_1, C_2, \dots, C_n$ are divided to three ciphertext substreams $C^{(1)}, C^{(2)}, C^{(3)}$. The method of ciphertext division is the same as the method of the plaintext division as described in 6.3.1.1. Each ciphertext substream $C^{(j)}$ is decrypted by the algorithm as described in Section 6.2.1.2 with initialization vector IV_j.

— **Input**: $C^{(1)} = C_{1,1}; C_{1,2}; ...; C_{1,n1}$;

$C^{(2)} = C_{2,1}; C_{2,2}; ...; C_{2,n2}$;

$C^{(3)} = C_{3,1}; C_{3,2}; ...; C_{3,n3}$;

IV_1, IV_2, IV_3; where $|C_{j,i}| = 64$ and $|IV_j| = 64$.

— **Output**: $P^{(1)} = P_{1,1}; P_{1,2}; ...; P_{1,n1}$;

$P^{(2)} = P_{2,1}; P_{2,2}; ...; P_{2,n2}$;

$P^{(3)} = P_{3,1}; P_{3,2}; ...; P_{3,n3}$; $|P_{i,j}| = 64$.

For $j = 1, 2, 3$, do

$C_{j,0} = IV_j$.

For $h = 1, 2, \ldots, n_j$, do

$P_{j,h} = D_{K1}(E_{K2}(D_{K3}(C_{j,h}))) \oplus C_{j,h-1}$;

Output $P_{j,h}$.

This results in the following plaintext stream

$P = (P_{1,1}, P_{2,1}, P_{3,1}; P_{1,2}, P_{2,2}, P_{3,2}; P_{1,3}, P_{2,3}, P_{3,3}; \ldots; P_{1,h}, P_{2,h}, P_{3,h}; \ldots; \ldots P_{j',nj'})$.

With three DEA functional blocks, DEA_1, DEA_2, DEA_3, which are simultaneously clocked, the decryption of three plaintext substreams $C^{(1)}$, $C^{(2)}$, $C^{(3)}$ can be interleaved. Let DEA_1 perform the D_{K3} operation, DEA_2 perform the E_{K2} operation, and DEA_3 perform the D_{K1} operation. Table 6 shows how three DEA functional blocks are scheduled (as an example, suppose that n mod 3 = 0. In this case, $n_1 = n_2 = n_3 = n/3$). At the first two clock cycles, the 128-bit output of the TDEA should be suppressed since valid output is not produced.

Table 6 — Schedule of TCBC-I decryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output
$t = 1$	$C_{1,1}$	$D_{K3}(C_{1,1})$	idle	idle	N/A
$t = 2$	$C_{2,1}$	$D_{K3}(C_{2,1})$	$E_{K2}(D_{K3}(C_{1,1}))$	idle	N/A
$t = 3$	$C_{3,1}$	$D_{K3}(C_{3,1})$	$E_{K2}(D_{K3}(C_{2,1}))$	$D_{K1}(E_{K2}(D_{K3}(C_{1,1})))$	$P_{1,1}$
$t = 4$	$C_{1,2}$	$D_{K3}(C_{1,2})$	$E_{K2}(D_{K3}(C_{3,1}))$	$D_{K1}(E_{K2}(D_{K3}(C_{2,1})))$	$P_{2,1}$
$t = 5$	$C_{2,2}$	$D_{K3}(C_{2,2})$	$E_{K2}(D_{K3}(C_{1,2}))$	$D_{K1}(E_{K2}(D_{K3}(C_{3,1})))$	$P_{3,1}$
$t = 6$	$C_{3,2}$	$D_{K3}(C_{3,2})$	$E_{K2}(D_{K3}(C_{2,2}))$	$D_{K1}(E_{K2}(D_{K3}(C_{1,2})))$	$P_{1,2}$
		…	…	…	
$t = 3(h-1) + 1$	$C_{1,h}$	$D_{K3}(C_{1,h})$	$E_{K2}(D_{K3}(C_{3,h-1}))$	$D_{K1}(E_{K2}(D_{K3}(C_{2,h-1})))$	$P_{2,h-1}$
$t = 3(h-1) + 2$	$C_{2,h}$	$D_{K3}(C_{2,h})$	$E_{K2}(D_{K3}(C_{1,h}))$	$D_{K1}(E_{K2}(D_{K3}(C_{3,h-1})))$	$P_{3,h-1}$
$t = 3(h-1) + 3$	$C_{3,h}$	$D_{K3}(C_{3,h})$	$E_{K2}(D_{K3}(C_{2,h}))$	$D_{K1}(E_{K2}(D_{K3}(C_{1,h})))$	$P_{1,h}$
		…	…	…	
$t = n = 3n_3$	C_{3,n_3}	$D_{K3}(C_{3,n_3})$	$E_{K2}(D_{K3}(C_{2,n_3}))$	$D_{K1}(E_{K2}(D_{K3}(C_{1,n_3})))$	P_{1,n_3}
$t = n + 1$	N/A	idle	$E_{K2}(D_{K3}(C_{3,n_3}))$	$D_{K1}(E_{K2}(D_{K3}(C_{2,n_3})))$	P_{2,n_3}
$t = n + 2$	N/A	idle	idle	$D_{K1}(E_{K2}(D_{K3}(C_{3,n_3})))$	P_{3,n_3}

6.3.2 TCBC-I properties

a) TCBC-I mode is not backward compatible with the single DEA CBC mode.

b) For the TCBC-I mode, one or more bit errors within a single ciphertext block $C_{j,h-1}$ will affect the decryption of two blocks: the block $P_{j,h-1}$ and the succeeding block $P_{j,h}$ in the same plaintext substream. Each bit of the plaintext block $P_{j,h-1}$ will have an average error rate of 50 %. The plaintext block $P_{j,h}$ will have only those bits in error which correspond directly to the ciphertext bits in error. However, if no error occurs other than the error in $C_{j,h-1}$, then the blocks, except for $P_{j,h-1}$ and $P_{j,h}$, will be correctly decrypted, i.e. limited error propagation.

Note that $P_{j,h-1}$ and $P_{j,h}$ are two successive blocks in jth plaintext substream $P^{(j)}$. But they are not two successive blocks in the plaintext P_1, P_2, … P_n. There are two blocks between $P_{j,h-1}$ and $P_{j,h}$; e.g. if $j = 2$, then the blocks between $P_{2,h-1}$ and $P_{2,h}$ are $P_{3,h-1}$ and $P_{1,h}$. In this case, except for $P_{2,h-1}$ and $P_{2,h}$, the other blocks will be decrypted correctly.

c) Synchronization is required for the TCBC-I mode of operation.

If block boundaries are lost between encipherment and decipherment (e.g. due to loss or insertion of a ciphertext bit), synchronization between the encipherment and decipherment operations will be lost until the correct bit boundaries are re-established. The result of all decipherment operations will be incorrect while the block boundaries are lost.

d) If the same IVs are always used then TCBC-I will always produce the same ciphertext for a given plaintext and key. Therefore (to avoid this) new IVs should be used with each new plaintext.

e) Since TCBC-I is a block method of encryption, it needs to operate on complete data blocks of multiples of 64 bits. Blocks of less than 64 bits require special handling, which is not addressed in this Technical Report.

6.4 TDEA cipher feedback mode of operation

6.4.1 TCFB definition

6.4.1.1 General

The TDEA cipher feedback (TCFB) mode of operation shown in Figures 5 and 6 is based upon the CFB mode of ISO 10116.

TCFB is different from the other modes in this Technical Report because the plaintext/ciphertext block length can be smaller than 64 bits. Note that the input/output data block to TDEA encryption/decryption operation is still 64 bits. The IV consists of 64 bits. Three keying options are defined for TCFB mode as described in 5.2.

For this Technical Report, the following implementations of TCFB are defined:

a) TCFB1, the1-bit plaintext/ciphertext block implementation;

b) TCFB8, the 8-bit plaintext/ciphertext block implementation;

c) TCFB64, the 64-bit plaintext/ciphertext block implementation.

With the above k-bit TCFB implementations, the plaintext data is divided into a sequence of n plaintext blocks P_1, P_2, ... P_n, each of k bits, where k = 1, 8 or 64.

6.4.1.2 TCFB encryption

— **Input**: P_1, P_2, … P_n; IV; $|P_i| = k$, $|IV| = 64$.

— **Output**: C_1, C_2, … C_n; $|C_i| = k$.

$I_0 = IV$;

$O_1 = E_{K3}(D_{K2}(E_{K1}(I_0)))$;

$C_1 = P_1 \oplus \{O_1\}_k$;

Output and feedback C_1.

For i = 2, ... n, do

$I_{i-1} = S_k(I_{i-2} \mid C_{i-1})$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$C_i = P_i \oplus \{O_i\}_k$;

Output and feedback C_i.

The input to the TDEA encryption operation is a 64-bit block I_{i-1}. The output is a 64-bit block O_i. The leftmost k bits of O_i, denoted as $\{O_i\}_k$, are exclusive–ored with the plaintext block P_i to get ciphertext C_i, which is also used as an input to the shifting function $S_k(I_{i-1} \mid C_i)$.

In TCFB encryption, let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. If at clock cycle $t = 1$, DEA_1 performs $E_{K1}(I_0)$, then at $t = 2$ and $t = 3$, DEA_1 must be idle, since the next input for DEA_1 should be $S_k(I_0 \mid C_1)$. But C_1 is the output occurring at $t = 3$. Therefore it is impossible for DEA_1, DEA_2, DEA_3 to perform DEA operations simultaneously for TCFB encryption.

6.4.1.3 TCFB decryption

— **Input**: $C_1, C_2, \ldots C_n$; IV; $|C_i| = k$, $|IV| = 64$.

— **Output**: $P_1, P_2, \ldots P_n$; $|P_i| = k$.

$I_0 = IV$;

$O_1 = E_{K3}(D_{K2}(E_{K1}(I_0)))$;

$P_1 = C_1 \oplus \{O_1\}_k$;

Output P_1 and feedback C_1.

For $i = 2, \ldots n$, do

$I_{i-1} = S_k(I_{i-2} \mid C_{i-1})$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus \{O_i\}_k$;

Output P_i and feedback C_i.

NOTE In TCFB mode, the TDEA encryption operation is used for both encryption and decryption to produce $O_1, O_2, \ldots O_n$.

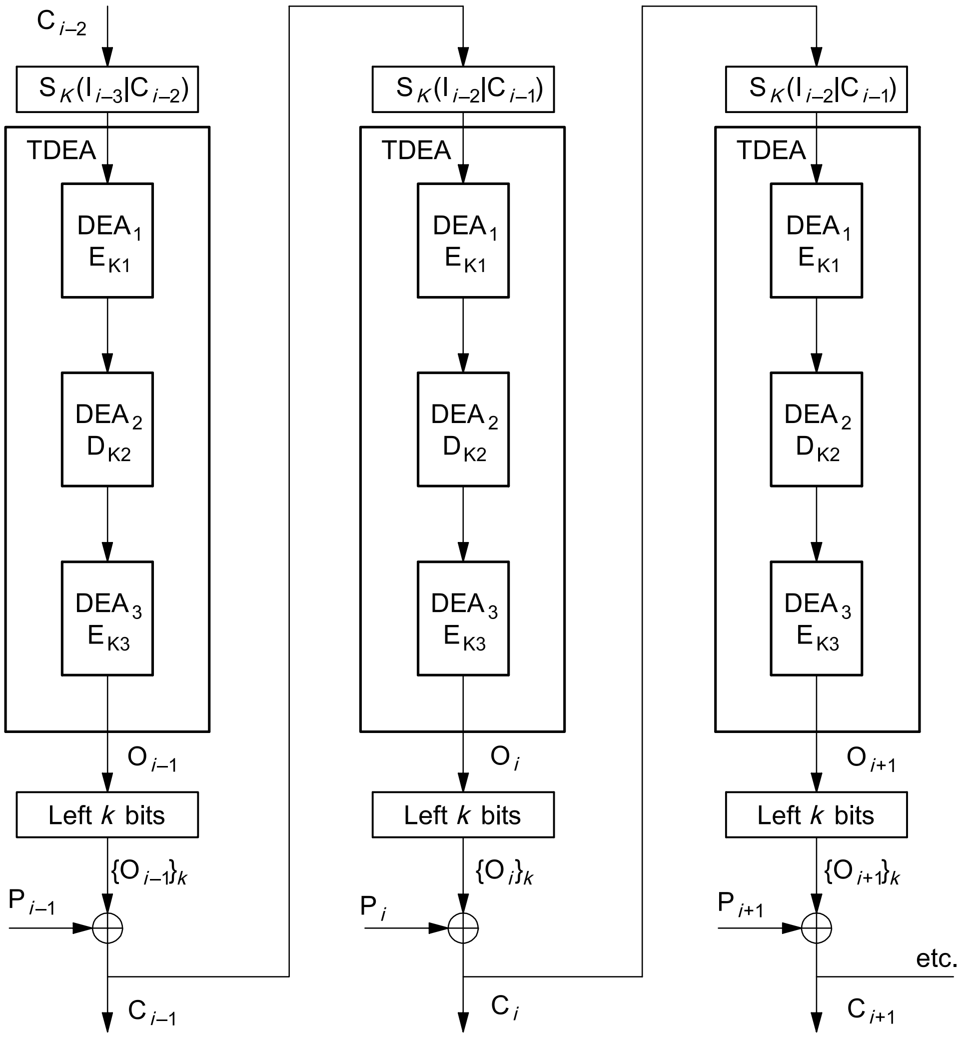

Figure 5 — TDEA cipher feedback-encryption

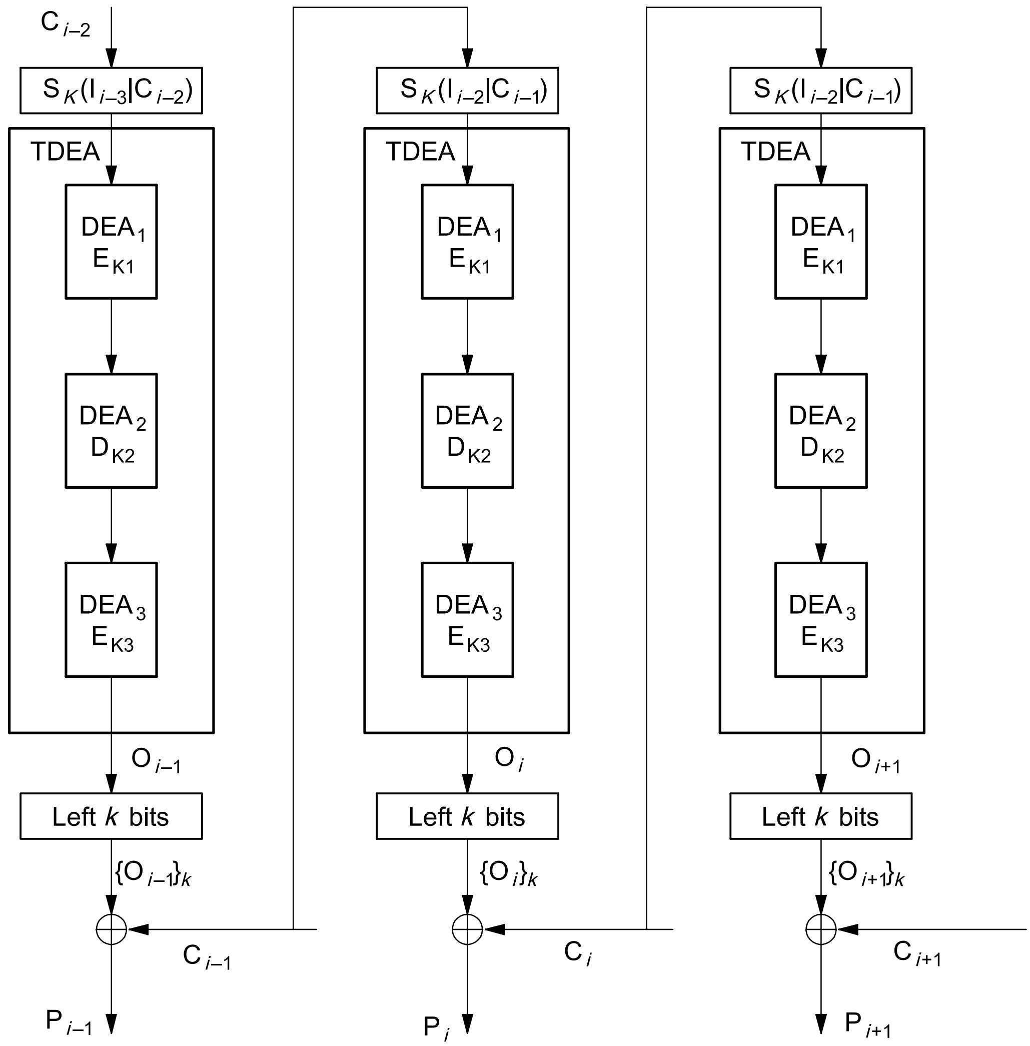

Figure 6 — TDEA cipher feedback-decryption

6.4.2 TCFB properties

a) When the three keys are set to be the same (see Keying Option 3), the TCFB mode of operation is backward compatible with the single DEA CFB mode using the same key.

b) In this mode, bit errors in any k-bit ciphertext block C_i will affect the decryption of $(64/k) + 1$ blocks. The first affected k-bit block P_i of plaintext will have errors in exactly those places where the ciphertext is in error. Succeeding decrypted plaintext blocks will have an average error rate of 0,5 until the bits in error are no longer used (i.e. they have been removed by the action of the shifting function $S_k(I_{i-2} \| C_{i-1})$). Assuming that no additional errors are encountered during this time, the correct plaintext blocks will then be obtained; e.g. with $k = 8$, if C_i is the ciphertext block with errors, then P_i will have errors at the same places as C_i. Each of P_{i+1}, P_{i+2}, … P_{i+8} will have an average error rate of 50 %. If no additional errors are encountered after C_i, then P_{i+9} and the succeeding blocks will be correct.

c) For the TCFB mode, synchronization is required.

If less than k bits are added or are lost in a ciphertext block C_i, then synchronization is lost. If the bit additions or deletions are detected and if the proper number of bits are removed from or added to C_i, then decryption may be re-synchronized such that, except for P_i, P_{i+1}, P_{i+2}, … $P_{i+(64/k)}$, the succeeding decrypted blocks are correct. Otherwise, the decryption of C_i and succeeding blocks are all in error.

If r entire blocks are added or lost right after C_i, then r blocks are added or lost after P_i. After the added or lost r blocks, $64/k$ error blocks follow. However, the succeeding blocks after $64/k$ error blocks can be decrypted correctly if no further error occurs. For example, if $k = 8$ and if C_{i+1} is lost, then block P_{i+1} will be lost, and P_{i+2}, P_{i+3}, … P_{i+9} are in error. If no further error occurs, then P_{i+10} and succeeding blocks are correctly decrypted.

d) If the same IV is used with each new plaintext, then TCFB will produce identical ciphertext for identical plaintext. Therefore a new IV shall be used with each new plaintext under the action of the same key.

6.5 TDEA cipher feedback mode of operation — pipelined

6.5.1 TCFB-P definition

6.5.1.1 General

In the pipelined configuration of TCFB, three IVs, generated as described in 6.7, shall be used. Three keying options are defined for TCFB-P mode as described in 5.2.

Prior to commencing TCFB-P encryption or decryption, the mode shall be initialized as described below. With the feedback path disconnected, IV_1 is clocked as input I_0. Then IV_2 is clocked as I_1. Finally, IV_3 is clocked as I_2. The feedback path is now connected and encryption/decryption can commence.

6.5.1.2 TCFB-P encryption

— **Input**: $P_1, P_2, \ldots P_n; IV_1, IV_2, IV_3. |P_i| = k, |IV_j| = 64$.

— **Output**: $C_1, C_2, \ldots C_n; |C_i| = k$.

For $i = 1, 2, 3$, do

$I_{i-1} = IV_i;$

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})));$

$C_i = P_i \oplus \{O_i\}_k;$

Output and feedback C_i.

For $i = 4, 5, \ldots n$, do

$I_{i-1} = S_k(I_{i-2} \mid C_{i-3});$

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})));$

$C_i = P_i \oplus \{O_i\}_k;$

Output and feedback C_i.

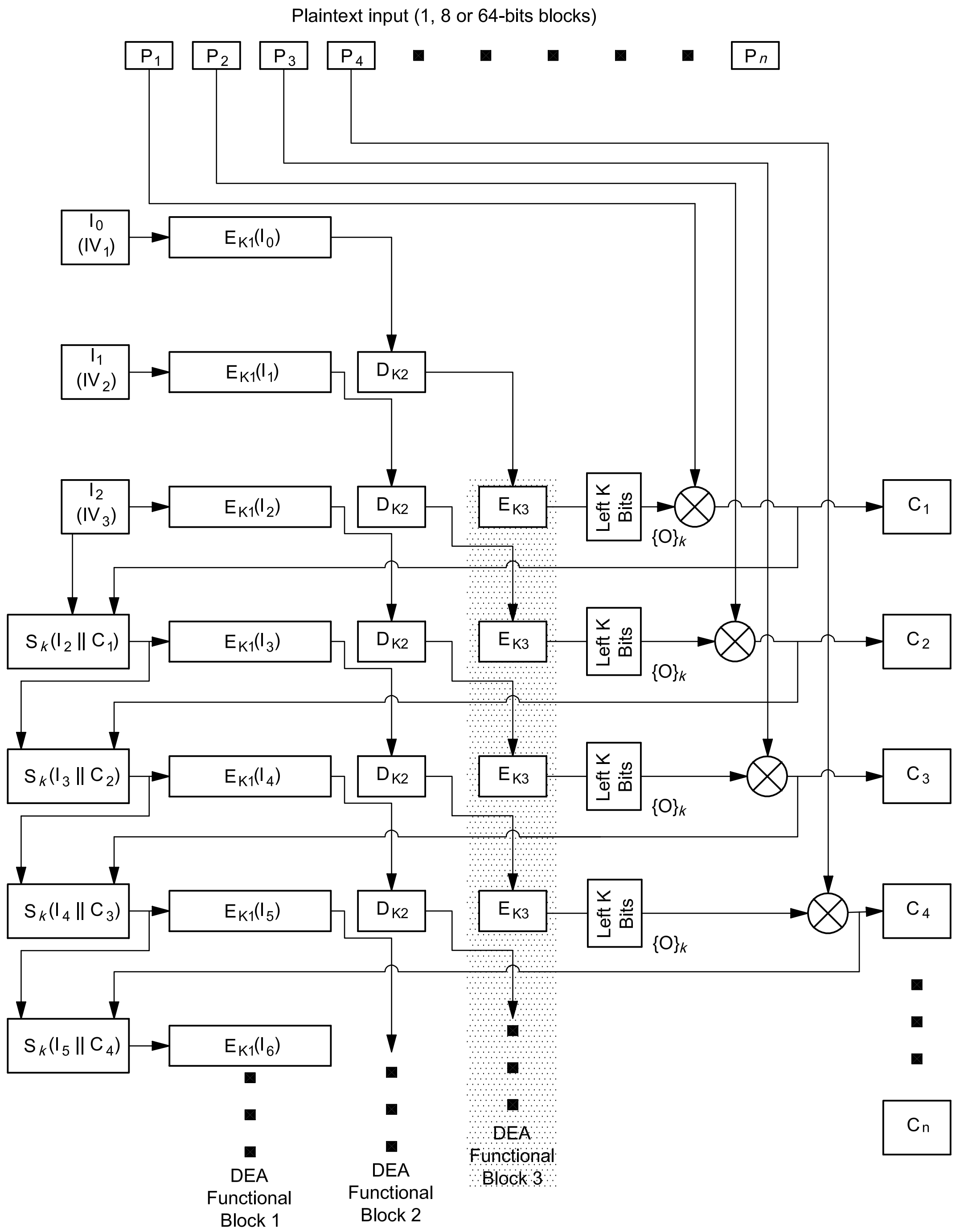

Figure 7 — TCFB-P encryption

With three DEA functional blocks, DEA_1, DEA_2, DEA_3, which are simultaneously clocked, and with three initialization vectors IV_1, IV_2, IV_3, the TCFB encryption can be pipelined. Let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. Table 7 shows how three DEA functional blocks are scheduled. Table 7 includes the feedback path connection or disconnection information. $F = 0$ is used for feedback path disconnection, and $F = 1$ is used for feedback path connection. At the first two clock cycles, the 128-bit output of the TDEA should be suppressed, since valid output is not produced.

Table 7 — Schedule of TCFB-P encryption

Clock	Input	DEA$_1$	DEA$_2$	DEA$_3$	Out-put	F
t =1	$I_0 = IV_1$	$E_{K1}(I_0)$	idle	idle	N/A	0
t = 2	$I_1 = IV_2$	$E_{K1}(I_1)$	$D_{K2}(E_{K1}(I_0))$	idle	N/A	0
t = 3	$I_2 = IV_3$	$E_{K1}(I_2)$	$D_{K2}(E_{K1}(I_1))$	$E_{K3}(D_{K2}(E_{K1}(I_0)))$	O_1	1
t = 4	$I_3 = S_k(I_2 \parallel C_1)$	$E_{K1}(I_3)$	$D_{K2}(E_{K1}(I_2))$	$E_{K3}(D_{K2}(E_{K1}(I_1)))$	O_2	1
t = 5	$I_4 = S_k(I_3 \parallel C_2)$	$E_{K1}(I_4)$	$D_{K2}(E_{K1}(I_3))$	$E_{K3}(D_{K2}(E_{K1}(I_2)))$	O_3	1
t = 6	$I_5 = S_k(I_4 \parallel C_3)$	$E_{K1}(I_5)$	$D_{K2}(E_{K1}(I_4))$	$E_{K3}(D_{K2}(E_{K1}(I_3)))$	O_4	1
	…	…	…	…		
$t = h$	$I_{h-1} = S_k(I_{h-2} \parallel C_{h-3})$	$E_{K1}(I_{h-1})$	$D_{K2}(E_{K1}(I_{h-2}))$	$E_{K3}(D_{K2}(E_{K1}(I_{h-3})))$	O_{h-2}	1
	…	…	…	…		
$t = n - 2$	$I_{n-3} = S_k(I_{n-4} \parallel C_{n-5})$	$E_{K1}(I_{n-3})$	$D_{K2}(E_{K1}(I_{n-4}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-5})))$	O_{n-4}	1
$t = n - 1$	$I_{n-2} = S_k(I_{n-3} \parallel C_{n-4})$	$E_{K1}(I_{n-2})$	$D_{K2}(E_{K1}(I_{n-3}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-4})))$	O_{n-3}	1
$t = n$	$I_{n-1} = S_k(I_{n-2} \parallel C_{n-3})$	$E_{K1}(I_{n-1})$	$D_{K2}(E_{K1}(I_{n-2}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-3})))$	O_{n-2}	1
$t = n + 1$	$I_n = S_k(I_{n-1} \parallel C_{n-2})$	$E_{K1}(I_n)$	$D_{K2}(E_{K1}(I_{n-1}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-2})))$	O_{n-1}	1
$t = n + 2$	$I_{n+1} = S_k(I_n \parallel C_{n-1})$	$E_{K1}(I_{n+1})$	$D_{K2}(E_{K1}(I_n))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-1})))$	O_n	1

6.5.1.3 TCFB-P decryption

— **Input**: $C_1, C_2, \ldots C_n$; IV_1, IV_2, IV_3. $|C_i| = k$, $|IV_j| = 64$.

— **Output**: $P_1, P_2, \ldots P_n$; $|P_i| = k$.

a) For i = 1, 2, 3, do

$I_{i-1} = IV_i$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus \{O_i\}_k$;

Output P_i and feedback C_i.

b) For i = 4, 5, … n, do

$I_{i-1} = S_k(I_{i-2} \parallel C_{i-3})$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus \{O_i\}_k$;

Output P_i and feedback C_i.

NOTE In the TCFB-P mode, the TDEA encryption operation is used for both encryption and decryption to produce the same $O_1, O_2, \ldots O_n$. Therefore, Table 7 can be used to schedule the work of DEA$_1$, DEA$_2$ and DEA$_3$ within each clock cycle.

6.5.2 TCFB-P properties

a) TCFB-P is not compatible with the single DEA CFB mode.

b) In this mode, bit errors in ciphertext block C_i will affect the decryption of C_i and of the 64/k succeeding blocks after C_{i+2} until the bits in error are no longer used (i.e, the bits have been removed by the action of the shifting function $S_k(I_{i-2} \| C_{i-3})$). The error bits of P_i are at the same positions as the error bits in C_i. The 64/k succeeding affected blocks after P_{i+2} will have an average error rate of 50 %, e.g. in TCFB1-P mode, bit errors in C_i will produce corresponding errors in P_i, followed by two properly decrypted plaintext bits, P_{i+1}, P_{i+2}, followed by 64 error bits, P_{i+3}, P_{i+4}, … P_{i+66} with an average error rate of 50 %. If no further error occurs, P_{i+67} and the succeeding blocks are correctly decrypted blocks.

c) For the TCFB-P mode, synchronization is required.

If less than k bits are added or are lost in a ciphertext block C_i, then synchronization is lost. If the bit additions or deletions are detected and if the proper number of bits are removed from or added to C_i, then decryption may be re-synchronized such that, except for P_i, P_{i+3}, P_{i+4}, … $P_{i+2+(64/k)}$, the succeeding decrypted blocks are correct. Otherwise, the decryption of C_i and succeeding blocks are all in error.

If r entire blocks are added or lost immediately after C_i, then r blocks are added or lost after P_i. After the added or lost r blocks, (64/k) + 2 error blocks are decrypted. However, the succeeding blocks, after the (64/k) + 2 error blocks, can be decrypted correctly if no further error occurs; e.g. if k = 8 and C_{i+1} is lost, then block P_{i+1} will be lost and P_{i+2}, P_{i+3}, … P_{i+10}, P_{i+11} are error blocks. If no further error occurs, the P_{i+11} and succeeding blocks will be correctly decrypted.

d) If the same IVs are used with each new plaintext, then TCFB-P will produce identical ciphertext for identical plaintext. Therefore new IVs shall be used with each new plaintext under the action of the same key.

6.6 TDEA output feedback mode of operation

6.6.1 TOFB definition

6.6.1.1 General

The TOFB mode of operation is based upon the OFB mode defined by ANSI X3.106 and ISO 8372, and is created by substituting the TDEA encryption operation (see 6.1) for the DEA encryption operation in that definition of the mode. See Figures 8 and 9.

The IV shall consist of 64 bits. Three keying options are defined for TOFB mode as described in 6.2. As with the TCFB mode, encryption and decryption use the same TDEA encryption operation. The only difference from TCFB in 7.4 is that the feedback to the shifting function is O_i instead of C_i, and k = 64; in this case, $I_i = S_k(I_{i-1} \| O_i) = O_i$.

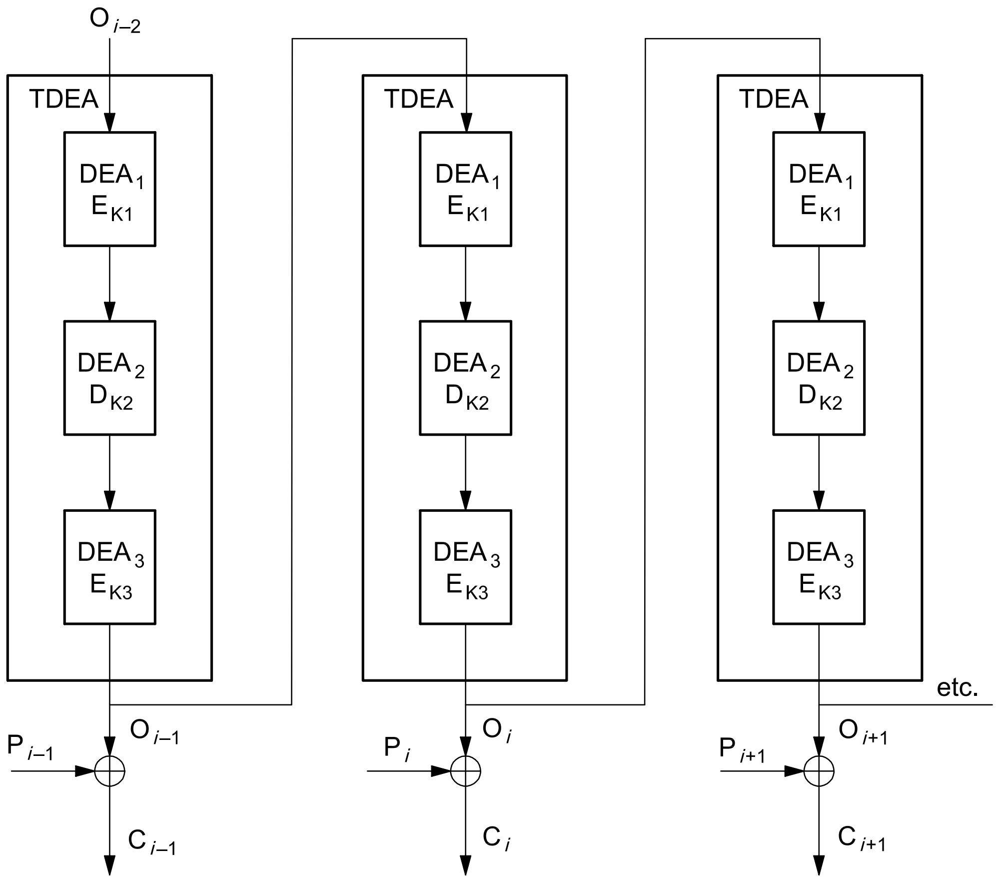

Figure 8 — TDEA output feedback-encryption

6.6.1.2 TOFB encryption

— **Input**: P_1, P_2, … P_n; IV; $|P_i| = 64$; $|IV| = 64$.

— **Output**: C_1, C_2, … C_n; $|C_i| = 64$.

$I_0 = IV$;

$O_1 = E_{K3}(D_{K2}(E_{K1}(I_0)))$;

$C_1 = P_1 \oplus O_1$;

Output C_1 and feedback O_1.

For $i = 2, ..., n$, do

$I_{i-1} = O_{i-1}$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$C_i = P_i \oplus O_i$;

Output C_i and feedback O_i.

In TOFB encryption, let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. If at clock cycle $t = 1$, DEA_1 performs $E_{K1}(I_0)$, then at $t = 2$ and $t = 3$, DEA_1 must be idle, since the next input for DEA_1 needs to be O_1. But the output O_1 does not occur until $t = 3$. Therefore it is impossible for DEA_1, DEA_2, DEA_3 to perform DEA operations simultaneously in TOFB encryption.

6.6.1.3 TOFB decryption

— **Input**: $C_1, C_2, \dots C_n$; IV; $|C_i| = 64$, $|IV| = 64$.

— **Output**: $P_1, P_2, \dots P_n$.

$I_0 = IV$;

$O_1 = E_{K3}(D_{K2}(E_{K1}(I_0)))$;

$P_1 = C_1 \oplus O_1$;

Output P_1 and feedback O_1.

For $i = 2, \dots n$, do

$I_{i-1} = O_{i-1}$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus O_i$;

Output P_i and feedback O_i.

NOTE In TOBF decryption, the TDEA encryption operation is used with K_1, K_2, K_3, and IV to produce the same key stream $O_1, O_2, \dots O_n$ that was produced during encryption: DEA_1, DEA_2 and DEA_3 each perform the same operation as is done for encryption. For the same reason as cited in 6.4.1.1, DEA_1, DEA_2 and DEA_3 cannot perform the operations simultaneously for TOFB decryption.

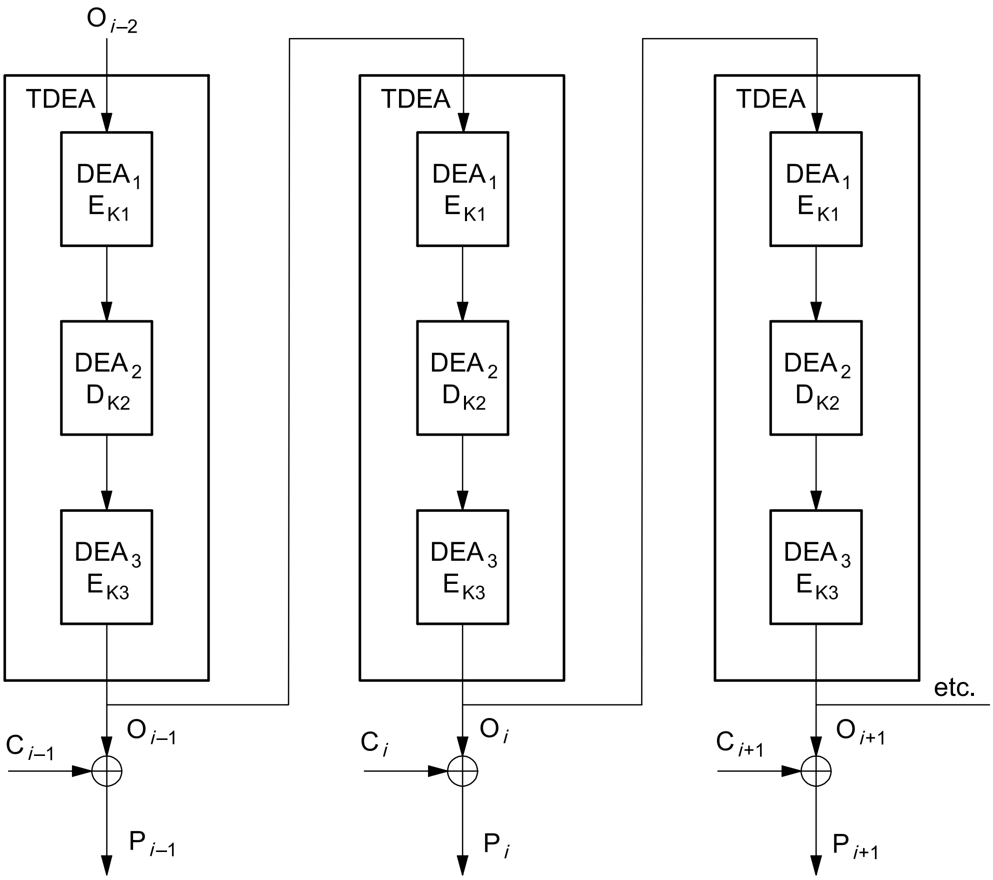

Figure 9 — TDEA output feedback — Decryption

6.6.2 TOFB properties

a) When the three keys are set to be the same (see Keying Option 3), the TOFB mode of operation is backward compatible with the single DEA OFB mode using the same key.

b) There is no error propagation when using TOFB, but an error in the ciphertext causes a corresponding error in the plaintext; i.e., if ciphertext block C_i has some error bits, then it only affects the same bits of plaintext block P_i.

c) For the TOFB mode, synchronization is required.

If bits are added or lost in ciphertext, then synchronization will be lost. If the bit additions or deletions are detected and if the proper number of bits is removed from or added to a suitable position in the ciphertext, then the decryption may be re-synchronized. Otherwise, the decryption of the succeeding blocks is in error.

d) For the TOFB mode, if the same IV is ever re-used, then the same stream cipher key will be produced for different plaintexts P and P′. As a result, $C \oplus C' = P \oplus P'$. Assuming the plaintext is structured, this event will leak information. Therefore, a newly selected IV shall be used for encryption of each new plaintext under the action of the same key.

6.7 TDEA output feedback mode of operation — Interleaved

6.7.1 TOFB-I definition

6.7.1.1 General

In order to interleave the TOFB mode of operation, three 64-bit IVs are required. These shall be generated as described in 5.7. In Figures 8 and 9, the TDEA is initialized by disconnecting the feedback path and sequentially loading each IV into the input block of the TDEA encryption operation. Once the IVs have been loaded, the feedback path should be connected and the mode started.

The plaintext is not explicitly divided into three plaintext substreams. The reason for this is that, for the encryption and decryption algorithm, the process can be considered as either that three plaintext substreams are encrypted/decrypted separately with the three given IVs, or that the encryption and decryption are pipelined as they are in TCFB-P mode. In other words, for TOFB mode, interleaved mode and pipelined mode are the same.

6.7.1.2 TOFB-I encryption

— **Input**: P_1, P_2, … P_n; IV_1, IV_2, IV_3. $|P_i| = 64$, $|IV_j| = 64$.

— **Output**: C_1, C_2, … C_n; $|C_i| = 64$.

For i = 1, 2, 3, do

$I_{i-1} = IV_i$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$C_i = P_i \oplus O_i$;

Output C_i and feedback O_i.

For i = 4, 5, … n, do

$I_{i-1} = O_{i-3}$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$C_i = P_i \oplus O_i$;

Output C_i and feedback O_i.

With three DEA functional blocks, DEA_1, DEA_2, DEA_3, which are simultaneously clocked, and with three initialization vectors IV_1, IV_2, IV_3, the TOFB encryption can be interleaved. Let DEA_1 perform the E_{K1} operation, DEA_2 perform the D_{K2} operation, and DEA_3 perform the E_{K3} operation. Table 8 shows how the three DEA functional blocks are scheduled. Table 6 includes the feedback path connection or disconnection information: $F = 0$ is used to imply disconnection, and $F = 1$ is used to imply connection. At the first two clock cycles, the 128-bit output of the TDEA should be suppressed since valid output is not produced.

Table 8 — Schedule of TOFB-I encryption

Clock	Input	DEA_1	DEA_2	DEA_3	Output	*F*
$t = 1$	$I_0 = IV_1$	$E_{K1}(I_0)$	idle	idle	N/A	0
$t = 2$	$I_1 = IV_2$	$E_{K1}(I_1)$	$D_{K2}(E_{K1}(I_0))$	idle	N/A	0
$t = 3$	$I_2 = IV_3$	$E_{K1}(I_2)$	$D_{K2}(E_{K1}(I_1))$	$E_{K3}(D_{K2}(E_{K1}(I_0)))$	O_1	1
$t = 4$	$I_3 = O_1$	$E_{K1}(I_3)$	$D_{K2}(E_{K1}(I_2))$	$E_{K3}(D_{K2}(E_{K1}(I_1)))$	O_2	1
$t = 5$	$I_4 = O_2$	$E_{K1}(I_4)$	$D_{K2}(E_{K1}(I_3))$	$E_{K3}(D_{K2}(E_{K1}(I_2)))$	O_3	1
$t = 6$	$I_5 = O_3$	$E_{K1}(I_5)$	$D_{K2}(E_{K1}(I_4))$	$E_{K3}(D_{K2}(E_{K1}(I_3)))$	O_4	1
	…	…	…	…		
$t = h$	$I_{h-1} = O_{h-3}$	$E_{K1}(I_{h-1})$	$D_{K2}(E_{K1}(I_{h-2}))$	$E_{K3}(D_{K2}(E_{K1}(I_{h-3})))$	O_{h-2}	1
	…	…	…	…		
$t = n - 2$	$I_{n-3} = O_{n-5}$	$E_{K1}(I_{n-3})$	$D_{K2}(E_{K1}(I_{n-4}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-5})))$	O_{n-4}	1
$t = n - 1$	$I_{n-2} = O_{n-4}$	$E_{K1}(I_{n-2})$	$D_{K2}(E_{K1}(I_{n-3}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-4})))$	O_{n-3}	1
$t = n$	$I_{n-1} = O_{n-3}$	$E_{K1}(I_{n-1})$	$D_{K2}(E_{K1}(I_{n-2}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-3})))$	O_{n-2}	1
$t = n + 1$	$I_n = O_{n-2}$	$E_{K1}(I_n)$	$D_{K2}(E_{K1}(I_{n-1}))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-2})))$	O_{n-1}	1
$t = n + 2$	$I_{n+1} = O_{n-1}$	$E_{K1}(I_{n+1})$	$D_{K2}(E_{K1}(I_n))$	$E_{K3}(D_{K2}(E_{K1}(I_{n-1})))$	O_n	1

6.7.1.3 TOFB-I decryption

— **Input**: $C_1, C_2, \ldots C_n$; IV_1, IV_2, IV_3. $|C_i| = 64$, $|IV_j| = 64$.

— **Output**: $P_1, P_2, \ldots P_n$; $|P_i| = 64$.

For $i = 1, 2, 3$, do

$I_{i-1} = IV_i$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus O_i$;

Output P_i and feedback O_i.

For $i = 4, 5, \ldots n$, do

$I_{i-1} = O_{i-3}$;

$O_i = E_{K3}(D_{K2}(E_{K1}(I_{i-1})))$;

$P_i = C_i \oplus O_i$;

Output P_i and feedback O_i.

For TOFB-I decryption, the TDEA encryption operation is used to produce the same O_1, O_2, … O_n as is produced for encryption. Table 6 as provided in 6.3.1.4 can be used to schedule the work of DEA_1, DEA_2 and DEA_3 within each clock cycle.

6.7.2 TOFB-I properties

a) TOFB-I is not backward compatible with the single DEA OFB mode.

b) There is no error propagation when using TOFB-I, but an error causes an error. A bit flip in ciphertext block C_i will result in an error for that particular bit in plaintext block P_i.

c) For the TOFB-I mode, synchronization is required.

If bits are added or lost in the ciphertext, then synchronization will be lost. If the bit additions or deletions are detected, and if the proper number of bits is removed from or added to a suitable position in the ciphertext, then decryption may be re-synchronized. Otherwise, the decryption of the block where additions or deletions start and the succeeding blocks is in error.

d) For the TOFB-I mode, if the same IVs are ever re-used, then the same stream cipher key will be produced for different plaintexts P and P′. As a result, $C \oplus C' = P \oplus P'$. Assuming the plaintext is structured, this event will leak information. Therefore, newly selected IVs shall be used for each re-initialized encryption under the action of the same key.

Annex A
(informative)

ASN.1 syntax for TDEA modes of operation

A.1 Overview

This annex provides ASN.1 syntax (see [4 - 7]) for the Triple DEA modes of operation defined in this Technical Report.

In cases where interoperability is a requirement, implementations shall use the following ASN.1 encoding (see [8] and [9]). Use of ASN.1 encoding is not mandated in situations where interoperability is not required or where equally robust rules for syntax and encoding are defined.

An algorithm identifier is defined by an object identifier and parameters. The parameters define keying options, IV generation preference and feedback parameters. Together these algorithm and parameter pairs define TDEA modes of operation.

A.2 Syntax for TDEA modes of operation

In this clause, the general syntax definitions for modes of operation are provided. A Triple DEA mode of operation is defined by ASN.1 type `TDEAIdentifier`:

```
TDEAIdentifier ::= AlgorithmIdentifier {{ TDEAModes }}
```

The Triple DEA modes of operation defined in this Technical Report are specified by objects of class `ALGORITHM-ID`. The information object set `TDEAModes` is used as the single parameter in a reference to type `AlgorithmIdentifier` and contains seven objects followed by the extension marker ("…"). Each object that represents a Triple DEA operation mode contains a unique object identifier and its associated type. The values of these objects define all of the valid values that may appear in `TDEAIdentifier`. The extension marker allows backward compatibility with future versions of this Technical Report which may define objects to represent additional operation modes. The set of Triple DEA modes defined in this Technical Report are:

```
TDEAModes ALGORITHM-ID ::= {

   { OID  tECB     PARMS  ECBParms  } |    -- mode 1 --

   { OID  tCBC     PARMS  TDEAParms } |    -- mode 2 --

   { OID  tCBC-I   PARMS  TDEAParms } |    -- mode 3 --

   { OID  tCFB     PARMS  CFBParms  } |    -- mode 4 --

   { OID  tCFB-P   PARMS  CFBParms  } |    -- mode 5 --

   { OID  tOFB     PARMS  TDEAParms } |    -- mode 6 --

   { OID  tOFB-I   PARMS  TDEAParms },     -- mode 7 --

...

}
```

Values of type `TDEAIdentifier` are constrained to the object identifier and parameter pairs defined by the objects listed in the information object set `TDEAModes`. For some instances of those operation modes which do not require `CFBParms`, the `parameters` component of `TDEAIdentifier` need not be present in an encoding of a value of that type. This may occur when none of the optional components of types `ECBParms` or `TDEAParms` are present. For operation modes associated with type `CFBParms`, the width of the feedback path must always be provided.

```
ECBParms ::= TDEAParms (WITH COMPONENTS {                              ...,
ivGeneration ABSENT })

TDEAParms ::= SEQUENCE {

   keyingOptions  KeyingOptions  OPTIONAL,

   ivGeneration   [0] IVGeneration  OPTIONAL

}

CFBParms ::= SEQUENCE {

   keyingOptions  KeyingOptions  OPTIONAL,

   feedbackSize   FeedbackSize

}
```

A user can further restrict the keying options allowed for a given X9.52 mode. Keying Option 3 can be used on some modes for which backward compatiblity is needed. Keying Option 2 uses two independent keys. Keying Option 1 requires that three independent keys be used.

```
KeyingOptions ::= BIT STRING {

   option-1 (0), -- (3-key) K1, K2 and K3 are independent keys

   option-2 (1), -- (2-key) K1 and K2 are independent and K3 = K1

   option-3 (2)  -- (1-key) K1 = K2 = K3

}
```

When the optional Keying Options component is not specified, it is assumed that the number of independent keys is decided elsewhere, perhaps by negotiation, prior arrangement or because in some context the number is obvious. Example values of type `KeyingOptions` are:

```
three-Key  KeyingOptions ::= { option-1 }

three-or-two-Key  KeyingOptions ::= { option-1, option-2 }

dea-compatible  KeyingOptions ::= { option-3 }
```

In X9.52, an IV or three IVs can be generated by a random number generator or by a monotonically increasing counter. The former is preferred to the latter.

```
IVGeneration ::= BIT STRING {

   random   (0),

   counter  (1)

 }
```

When the optional IV generation component is not specified, it is assumed that the method in which IVs are generated is decided elsewhere, or that the order of preference specified in this Technical Report and the capability of the user will decide. Example values of type `IVGeneration` are:

```
randomOnly  IVGeneration ::= { random }

eitherMethod  IVGeneration ::= { random, counter }
```

TCFB mode and TCFB-P mode require 1-bit, 8-bit, and 64-bit feedback path widths, which are specified in the `parameters` component of type `TDEAIdentifier` for these modes as values of type `FeedbackSize`:

```
FeedbackSize  ::=  INTEGER {   -- Feedback path widths in bits

   one       (1),

   eight     (8),

   sixtyfour(64)

} ( one | eight | sixtyfour )
```

A.3 Object identifiers

The object identifier `id-ansi-x952` represents the tree containing all object identifiers defined in this Technical Report, and has the following value:

```
id-ansi-x952 OBJECT IDENTIFIER ::= {

iso(1) member-body(2) us(840) ansi-x952(10047) }
```

The object identifier `mode` represents the tree containing object identifiers representing all of the Triple DEA modes of operation defined in this Technical Report. It has the following value:

```
mode  OBJECT IDENTIFIER ::= { id-ansi-x952 1 }
```

The following object identifiers represent each of the seven Triple DEA modes of operation defined in this Technical Report. These modes have the following values:

```
tECB     OBJECT IDENTIFIER ::= { mode 1 }

tCBC     OBJECT IDENTIFIER ::= { mode 2 }

tCBC-I   OBJECT IDENTIFIER ::= { mode 3 }

tCFB     OBJECT IDENTIFIER ::= { mode 4 }

tCFB-P   OBJECT IDENTIFIER ::= { mode 5 }

tOFB     OBJECT IDENTIFIER ::= { mode 6 }

tOFB-I   OBJECT IDENTIFIER ::= { mode 7 }
```

A.4 Supporting definitions

A parameterized version of X.509 (see [10]) type `AlgorithmIdentifier` is defined in this Technical Report. A single parameter is required in a reference to this type, an information object set of class `ALGORITHM-ID`. The reference which defines type `TDEAIdentifier` above specifies the parameter `TDEAModes`.

```
AlgorithmIdentifier { ALGORITHM-ID:IOSet } ::= SEQUENCE {

   algorithm   ALGORITHM-ID.&id({IOSet}),

   parameters  ALGORITHM-ID.&Type({IOSet}{@algorithm}) OPTIONAL

}
```

Type `AlgorithmIdentifier` is composed of two components, `algorithm` and `parameters`, which are defined in terms of the information object class `ALGORITHM-ID`, and are specified by the fields of that class, `&id` and `&Type`.

```
ALGORITHM-ID ::= CLASS {

   &id    OBJECT IDENTIFIER UNIQUE,

   &Type  OPTIONAL

}

  WITH SYNTAX { OID &id [PARMS &Type] }
```

These fields form a template for defining sets of information objects, instances of the class `ALGORITHM-ID`. This class is similar to the useful information object `class TYPE-IDENTIFIER` used to define class `ALGORITHM` in X.509 (see [10]), but differs in allowing the `parameters` component of `AlgorithmIdentifier` to be absent in an encoding of a value of that type. In an instance of `ALGORITHM-ID`, “`algorithm`” will contain an object identifier value that uniquely identifies the type contained in “`parameters`”. The effect of referencing “`algorithm`” in both components of the `AlgorithmIdentifier` sequence is to tightly bind the object identifier and its type.

A.5 ASN.1 module

A complete ASN.1 module is provided below, which contains all of the notation defined in this Technical Report.

```
ANSI-X9-52 {

   iso(1) member-body(2) us(840) ansi-x952(10047) module(4) 1 }

      DEFINITIONS EXPLICIT TAGS ::= BEGIN

-- X9.52 TDEA Modes of Operation

-- EXPORTS All;

-- IMPORTS None;

TDEAIdentifier ::= AlgorithmIdentifier {{ TDEAModes }}

TDEAModes ALGORITHM-ID ::= {
```

```
   { OID  tECB     PARMS  ECBParms  } |    -- mode 1 --
   { OID  tCBC     PARMS  TDEAParms } |    -- mode 2 --
   { OID  tCBC-I   PARMS  TDEAParms } |    -- mode 3 --
   { OID  tCFB     PARMS  CFBParms  } |    -- mode 4 --
   { OID  tCFB-P   PARMS  CFBParms  } |    -- mode 5 --
   { OID  tOFB     PARMS  TDEAParms } |    -- mode 6 --
   { OID  tOFB-I   PARMS  TDEAParms },     -- mode 7 --
   ...
}
ECBParms ::= TDEAParms (WITH COMPONENTS {
                           ..., ivGeneration ABSENT })
TDEAParms ::= SEQUENCE {
   keyingOptions  KeyingOptions  OPTIONAL,
   ivGeneration   [0] IVGeneration  OPTIONAL
}
CFBParms ::= SEQUENCE {
   keyingOptions  KeyingOptions  OPTIONAL,
   feedbackSize   FeedbackSize
}
KeyingOptions ::= BIT STRING {
   option-1 (0), -- (3-key) K1, K2 and K3 are independent keys
   option-2 (1), -- (2-key) K1 and K2 are independent and K3 = K1
   option-3 (2)  -- (1-key) K1 = K2 = K3
}
 IVGeneration ::= BIT STRING {
   random   (0),
   counter  (1)
 }
FeedbackSize  ::=  INTEGER {   -- Feedback path widths in bits
   one      (1),
```

```
   eight     (8),
   sixtyfour(64)
} ( one | eight | sixtyfour )
-- Object identifiers
id-ansi-x952 OBJECT IDENTIFIER ::= {
   iso(1) member-body(2) us(840) ansi-x952(10047) }
mode  OBJECT IDENTIFIER ::= { id-ansi-x952 1 }
tECB     OBJECT IDENTIFIER ::= { mode 1 }
tCBC     OBJECT IDENTIFIER ::= { mode 2 }
tCBC-I   OBJECT IDENTIFIER ::= { mode 3 }
tCFB     OBJECT IDENTIFIER ::= { mode 4 }
tCFB-P   OBJECT IDENTIFIER ::= { mode 5 }
tOFB     OBJECT IDENTIFIER ::= { mode 6 }
tOFB-I   OBJECT IDENTIFIER ::= { mode 7 }
-- Supporting definitions
AlgorithmIdentifier { ALGORITHM-ID:IOSet } ::= SEQUENCE {
   algorithm   ALGORITHM-ID.&id({IOSet}),
   parameters  ALGORITHM-ID.&Type({IOSet}{@algorithm}) OPTIONAL
}
ALGORITHM-ID ::= CLASS {
   &id    OBJECT IDENTIFIER UNIQUE,
   &Type  OPTIONAL
}
  WITH SYNTAX { OID &id [PARMS &Type] }
END
```

Annex B
(informative)

TDEA modes of operation cryptographic attributes

B.1 Modes of operation

This annex describes, in a general nature, the major cryptographic attributes of the TDEA modes of operation. Unless marked with (*), the identified attributes are also applicable to the interleaved or pipelined modes.

Table B.1 — Modes of operation

Attribute	TECB	TCBC	TCFB1	TCFB8	TCFB64	TOFB
Error propagation	1 Block	2 Blocks	65 Blocks	9 Blocks	2 Blocks	None
Masks block pattern in plaintext	No	Yes	Yes	Yes	Yes	Yes
Backward compatible (*)	Yes	Yes	Yes	Yes	Yes	Yes
Number of DEA per block	3	3	3	3	3	3

B.2 Key attacks

A key attack attempts to recover the value of the key and thereby enable the recovery of all data encrypted using that key.

With Keying Options 1 and 2, if there are a few known plaintext/ciphertext block pairs then the best known attacks for TECB, TCBC, TOFB and TCFB64 take 2^{112} single DEA encryptions.

If there are many known plaintext/ciphertext block pairs, then with Keying Option 2 (see 5.2), the best attack takes $(2^{120})/r$ single DEA encryptions, where r is the number of known plaintext/ciphertext block pairs. But with Keying Option 1 (see 5.2), the attacks are not known to be easier by knowing many plaintext/ciphertext pairs.

Currently, there are no known feasible key attacks on any of these modes, when using Keying Options 1 or 2.

B.3 Text attacks

B.3.1 General

A text attack attempts to recover some plaintext or information about some plaintext from the ciphertext; the key is not recovered.

B.3.2 Stream cipher cycle length

B.3.2.1 General

For TOFB, a concern is the length of the key stream before it repeats. TOFB has an average cycle length of 2^{63} blocks. Once TOFB repeats, the conservative assumption is that all encrypted data using a repeated key stream can be recovered.

After the generation of approximately 2^{32} IVs for the same set of keys, the expectation is that the IV will repeat, thus causing the same key stream to be produced. In this event, the conservative assumption is that all plaintext can be recovered.

It is strongly recommended that the set of TDEA keys be changed well before either of these events occurs.

B.3.2.2 Text dictionary

An attacker may build a dictionary of known plaintext/ciphertext pairs and seek to find at least one entry corresponding to encrypted text where the plaintext is (supposed to be) secret.

Let m be the number of different 64-bit plaintext blocks to be encrypted in the TECB mode; m is at most 2^{64}. Let the crossover point be the number of blocks at which there is an expectation that one encrypted block of secret plaintext is revealed by being in a dictionary due to the birthday phenomenon. The crossover points for the TDEA modes of operation are given in Table B.2.

Table B.2 — Crossover points

Mode of Operation	Crossover Point
TECB	$2(m^{1/2})$ encrypted blocks
TCBC	2^{33} encrypted blocks
TCFB64	2^{33} encrypted blocks

The prudent implementer should consider changing the bundle of TDEA keys well before reaching the crossover points.

B.3.2.3 Matching ciphertext

After about 2^{32} blocks have been encrypted, the birthday phenomenon predicts that one block of ciphertext will match another block.

For TECB, matching ciphertext blocks indicate that the same plaintext blocks occur in differing locations; this may result in an information leak. As TECB does not randomize, or "pre-whiten", the block (as does TCBC) by using an IV, the chance of matching ciphertext is dependent only on the number of different plaintext blocks being encrypted.

For TCBC, matching ciphertext blocks $C_i = C_{i'}$ implies that $P_i \oplus P_{i'} = C_{i-1} \oplus C_{i'-1}$; assuming that the plaintext blocks are structured, this event will leak information.

For TCFB64, matching ciphertext blocks $C_i = C_{i'}$ implies $P_{i+1} \oplus P_{i'+1} = C_{i+1} \oplus C_{i'+1}$; assuming that the plaintext blocks are structured, this event will leak information.

For TOFB, one block of matching ciphertext should not be significant, as the key stream is independent of the previous ciphertext.

The prudent implementer should consider changing the set of TDEA keys well before this event becomes likely.

B.4 Guidance on the authentication of data

The TDEA modes described in this Technical Report are designed to provide data confidentiality between two parties sharing a cryptographic key. These modes by themselves do not provide for the authentication of the underlying integrity of the data; e.g. an untrusted third party may intentionally garble ciphertext in order to cause a garble in the plaintext after decryption. In some cases, an untrusted third party who knows a plaintext message may be able to modify the cipher or the IV so that another incorrect message will result upon decryption.

Techniques are available to authenticate the integrity of decrypted messages. Guidance on the use of these techniques is beyond the scope of this Technical Report.

Annex C
(informative)

Key bundle encryption precautions

C.1 Characteristics

Where a key is to be encrypted with a block cipher that has a block size less than the size of the key, precautions need to be taken to prevent the substitution or use of a fragment of the overall key cryptogram. Binding between the blocks of the enciphered key bundle may be achieved through the use of message digests or through the use of specific modes of operation. This annex presents three alternative methods, RFC 3217, Authenticated Key Block and Three Pass Outer CBC encipherment.

Table C.1 — Characteristics

	Encrypted Key size	Cipher/hash operations	Comment
RFC3217	Fixed 40-octet output	27	Requires SHA-1
AKB	Fixed 80-octet output	42	Provides key tags
3CPO	Same as input size	18	

C.2 RFC 3217

C.2.1 General

This method, based on RFC3217, expands all TDEA keys to a fixed length, provides a strong checksum of the key and includes two passes of CBC encipherment to provide a fixed length, 40-octet key cryptogram. It differs from the method specified in RFC3217 in that it permits the encryption of a 192-bit TDEA key with a 128-bit TDEA key.

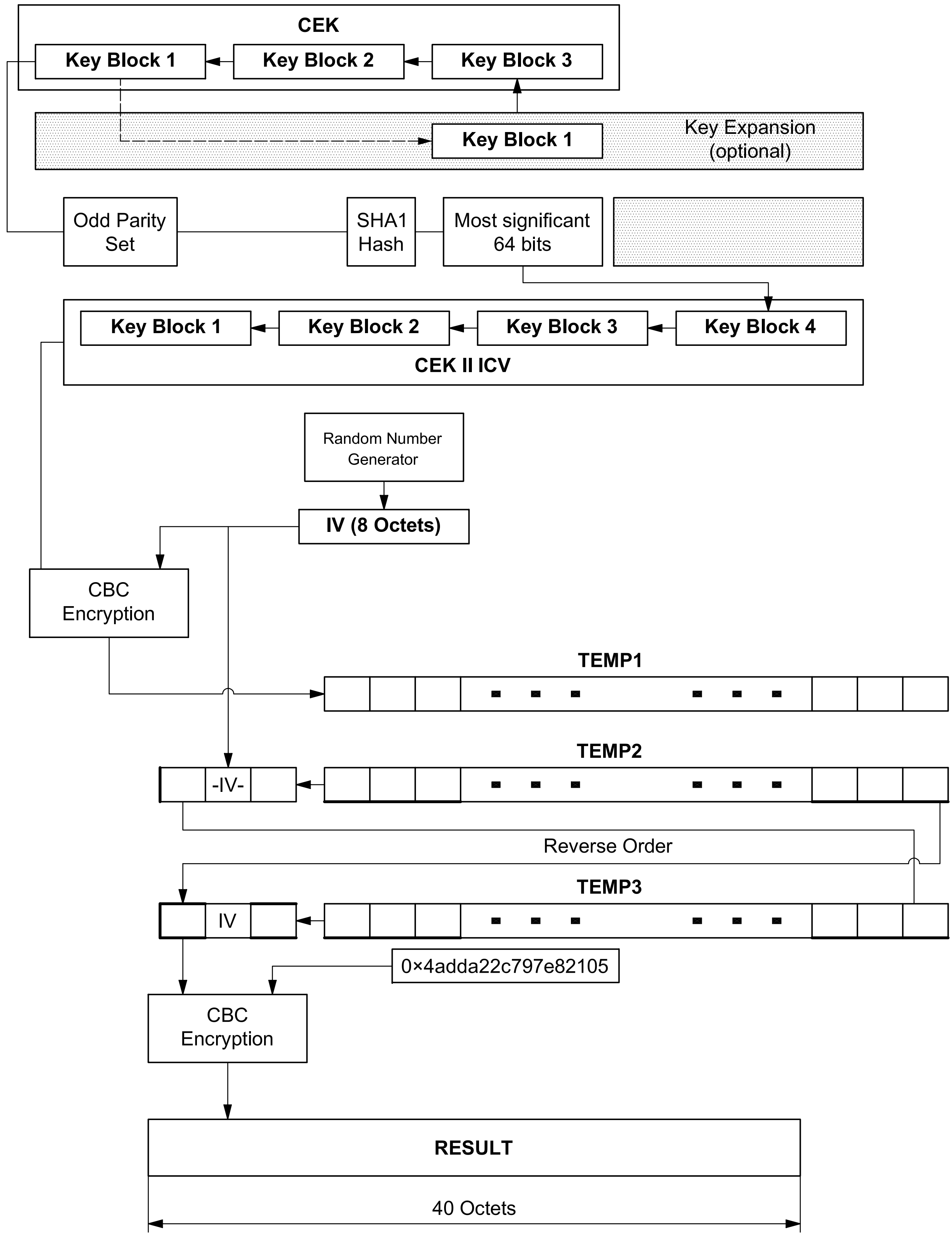

Figure C.1 — RFC3217 Key binding

C.2.2 Functional elements — Key checksum

C.2.2.1 General

The key checksum algorithm is used to provide a key integrity check value.

The algorithm is:

— Compute a 20-octet SHA-1 [SHA1] message digest on the key that is to be wrapped.

— Use the most significant (first) eight bytes of the message digest value as the checksum value.

C.2.2.2 Key expansion

The same key wrap algorithm is used for both two-key TDEA (128-bit) and three-key TDEA (192-bit) keys. When a two-key TDEA key is to be wrapped, a third DEA key with the same value as the first DEA key is created. Thus, all wrapped TDEA keys are 192 bits in length.

It is permissible to encrypt a 192-bit TDEA key with a 128-bit TDEA key as a 128-bit DEA key provides near equivalent protection.

C.2.2.3 TDEA key wrap

The TDEA key wrap algorithm encrypts a TDEA key with a TDEA key-encryption key. The TDEA key wrap algorithm is:

a) expand any 128-bit TDEA keys to 192 bits by appending the leftmost 64 bits of the TDEA key to itself;

b) set odd parity for each of the DEA key octets comprising the TDEA key that is to be wrapped; call the result CEK;

c) compute an 8-octet key checksum value on CEK as described in C.2.1, call the result ICV;

d) let CEKICV = CEK || ICV;

e) generate 8 bytes at random, call the result IV;

f) encrypt CEKICV in CBC mode using the key-encryption key; use the random value generated in the previous step as the initialization vector (IV); call the ciphertext TEMP1;

g) let TEMP2 = IV || TEMP1;

h) reverse the order of the octets in TEMP2, i.e. the most significant (first) octet is wapped with the least significant (last) octet, and so on; call the result TEMP3;

i) encrypt TEMP3 in CBC mode using the key-encryption key; use an initialization vector (IV) of 0x4ADDA22C79E82105, the ciphertext is 40 bytes long.

NOTE When the same 192-bit TDEA key is wrapped in different key-encryption keys, a fresh initialization vector (IV) must be generated for each invocation of the key wrap algorithm.

C.2.2.4 TDEA key unwrap

The TDEA key unwrap algorithm decrypts a TDEA key using a TDEA key-encryption key. The TDEA key unwrap algorithm is:

a) if the wrapped key is not 40 bytes, then error;

b) decrypt the wrapped key in CBC mode using the key-encryption key; use an initialization vector (IV) of 0x4ADDA22C79E82105; call the output TEMP3;

c) reverse the order of the bytes in TEMP3, i.e. the most significant (first) octet is swapped with the least significant (last) octet and so on; call the result TEMP2;

d) decompose TEMP2 into IV and TEMP1; IV is the most significant (first) 8 bytes and TEMP1 is the least significant (last) 32 bytes;

e) decrypt TEMP1 in CBC mode using the key-encryption key; use the IV value from the previous step as the initialization vector; call the ciphertext CEKICV;

f) decompose CEKICV into CEK and ICV; CEK is the most significant (first) 24 bytes and ICV is the least significant (last) 8 bytes;

g) compute an 8-octet key checksum value on CEK as described in C.2.1; if the computed key checksum value does not match the decrypted key checksum value, ICV, then error;

h) check for odd parity each of the DES key bytes comprising CEK. If parity is incorrect, then error;

i) use CEK as a TDEA key.

C.3 Authenticated key block method (AKB)

C.3.1 General

The Authenticated Key Block has a fixed format. It contains a header of length 16 bytes, an encrypted key field (in hex-ASCII format) padded to the maximum length of a TDEA key in order to hide the true length of short keys) followed by a MAC field of 16 bytes, resulting in an 80-byte key block.

Table C.2 — AKB key binding

Header	Encrypted Key	MAC

C.3.2 Key block header (KBH)

C.3.2.1 General

The header is a fixed length of 16 bytes and contains attribute information about the key. For better supportability (i.e. human readability), the 16 bytes of the header shall only contain uppercase ASCII printable characters. Tables are provided that list specific headers for defined key types.

C.3.2.2 Key block header definition

Table C.3 — Key block header definition

Byte #	Definition	Contents
0	Version number	“2” (Current version)
1-4	Key block length	ASCII number digits providing key block length; e.g., a 72-byte key block would contain “0” in Byte #1, “0” in Byte #2, “7” in Byte #3, and “2” in Byte #4
5	Key usage	“K” for key encryption, “D” for data encryption, etc.
6	Other information	Other information about the key
7	Algorithm	“D” for DES, “R” for RSA, “A” for AES
8	Mode of use	“E” for encrypt only, “D” for decrypt only, etc.
9	Exportability	“E” for exportable under trusted key, “N” not exportable, etc.
10-11	Reserved/random value length	For key blocks bound with the CBC MAC binding method, this field is reserved and is always filled with “R”
12-15	Reserved	“0”

NOTE Before a key in the Key Block format is used in a Tamper Resistant Security Module (TRSM), the content of the header block must be validated to ensure the correct usage is enforced. The “Key Usage” byte is typically checked first followed by the “Algorithm” byte. The other header bytes may or may not be checked depending on the key usage and the algorithm used.

C.3.2.3 Byte 5, key usage

Table C.4 — Byte 5 — Key usage

Value	Hex	Definition
“D”	0×44	Data encryption
“I”	0×49	IV or control vector Byte 6 = “0” for IV
“K”	0×4B	Key encryption or wrapping
“M”	0×4D	MAC
“P”	0×50	Pin encryption
“V”	0×56	PIN verification, KPV
“C”	0×43	CVK card verification key
“B”	0×42	BDK base derivation key

NOTE These usages work for both symmetric and asymmetric keys. Usage “K” is appropriate for a DES KEK and an RSA Key exchange key.

C.3.2.4 Byte 6, other information

The value in this byte is used to provide additional information of the key. C.2.2.3 has more details about the possible values of this byte.

C.3.2.5 Byte 7, algorithm

Table C.5 — Byte 7 algorithm

Value	Hex	Definition
“D”	0×44	DES
“R”	0×52	RSA
“A”	0×41	AES
“S”	0×53	DSA
“U”	0×55	Unknown or unspecified
“E”	0×45	Elliptic curve

C.3.2.6 Byte 8, mode of use

Table C.6 — Byte 8 — Usage mode

Value	Hex	Definition
“N”	0×4E	No special restrictions
“E”	0×45	Encrypt only
“D”	0×44	Decrypt only
“0”	0×30	IV

C.3.2.7 Byte 9, exportability

Table C.7 — Byte 9 — Exportability

Value	Hex	Definition
“S”	0x53	Sensitive
“E”	0x45	Exportable
“N”	0x4E	Non-exportable

Flags in this field indicate special types of key that require unusual handling. Any key that does not follow normal security assumptions should have a notation in this field. In general, a letter in the “Value” column means that future developers should check the definition of this type of key carefully.

C.3.2.8 Bytes 12-15, reserved

Table C.8 — Reserved bytes

Value	Hex	Definition
“0”	0×30	Reserved

C.3.2.9 Key to be exchanged/stored

The key to be exchanged and/or stored is represented in the key block in hex-ASCII format. Single DES keys and double length TDEA keys are padded to a full 48-byte length in order to mask the true length of the key.

Padding, if used, is specific to DES and triple-DES implementations. It is not used with any other key types. All pad characters are random data with their parity bits forced to even parity to identify that they are padding bytes.

C.3.2.10 Key separation

Key separation is maintained by deriving the encryption and MAC keys from the base Key Encrypting Key using predefined variants.

C.3.2.11 Key block encryption

The key block encryption method uses TDEA CBC encryption for the purpose of maintaining the secrecy of the key being exchanged and/or stored. The key and any random and/or pad characters are TDEA CBC encrypted, with bytes 5-12 of the header used as the IV for the CBC encryption.

The encrypting key is the result of an exclusive OR operation between the Key Encrypting Key and a constant of X‘4545454545454545’ (8 bytes of ASCII “E”) expanded, by repetition, to equal the length of the Key Encrypting Key.

C.3.2.12 CBC MAC binding method

The CBC MAC binding method consists of calculating a TDEA CBC MAC across the entire key block using bytes 5-12 of the KBH as the IV. The CBC MAC is computed according to ISO/IEC 9797-1 MAC algorithm number 1 and padding method 1 using the TDEA block cipher specified in ISO/IEC 18033.

The MAC Key is the result of an exclusive OR operation between the Key Encrypting Key and a constant of X‘4D4D4D4D4D4D4D4D’ (8 bytes of ASCII “M”) expanded, by repetition, to equal the length of the Key Encrypting Key.

This results in a MAC key distinctly different from the encryption key. The MAC, calculated over the clear header and the encrypted key block, binds those two parts together and prevents any alteration among them.

The size of MAC is 8 bytes long (16 hex-ASCII characters).

C.3.2.13 Key validation

Upon receiving the authenticated key block, the key block must be validated by ensuring the validity of the MAC and the contents of the header.

C.4 3CPO — Three, CBC pass outer encryption

C.4.1 Introduction

This method, illustrated in Figure C.2 with a typical two-key key bundle, achieves key binding between the elements of the key bundle through the use of CBC encryption and with the initialization vectors influenced by the other key bundle component.

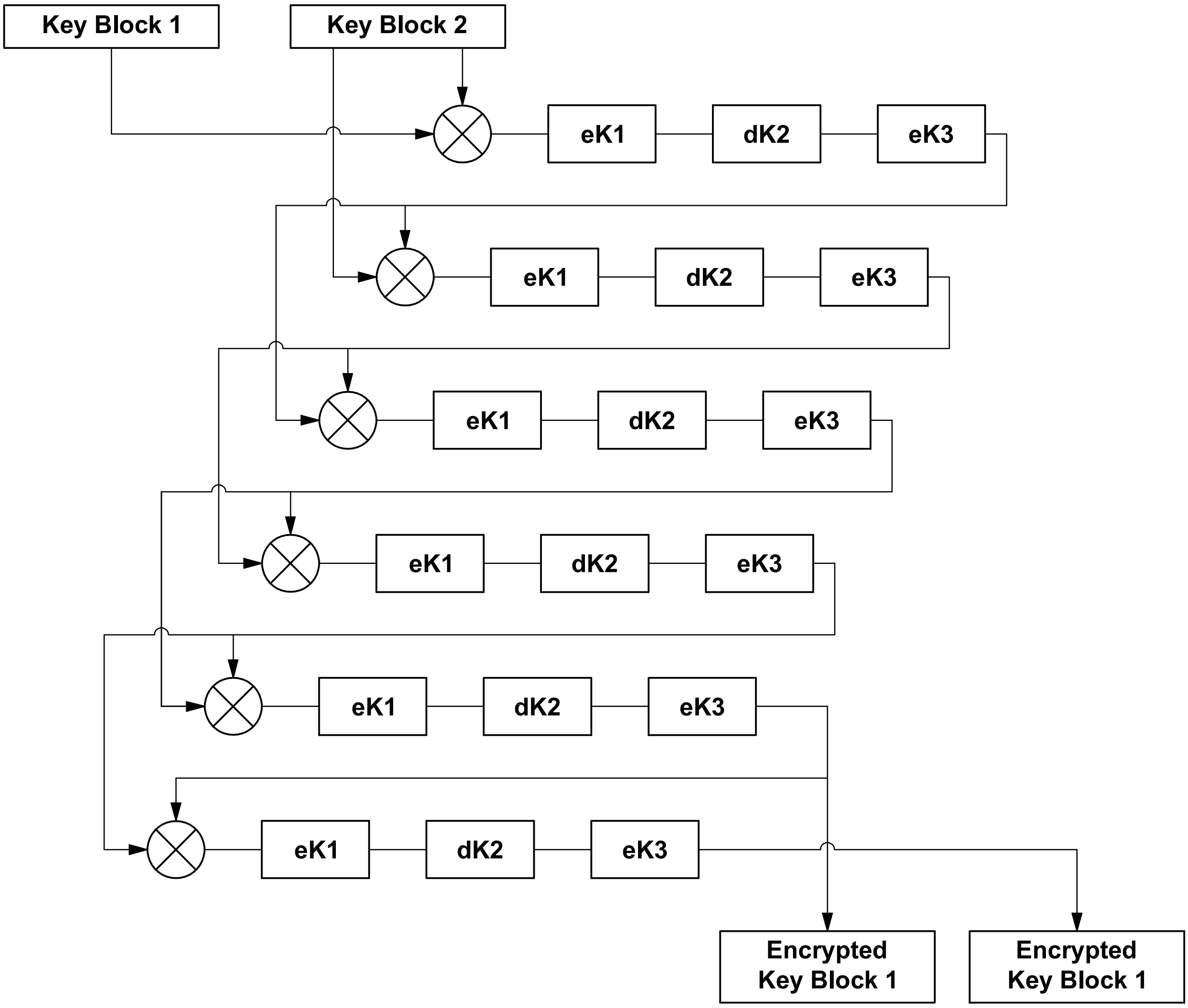

Figure C.2 — 3CPO Key block binding

C.4.2 Method

In this method, three passes of CBC encryption are performed with the first pass chaining into the IV of the second and the second into the third. The method is suitable for *n*-block encipherment and extension to additional passes of encipherment.

C.4.3 Encipherment formulas

C.4.3.1 2-block encipherment

p = number of passes of encipherment (three recommended)

$T_{-1} = K1$

$T_0 = K2$

$T_i = e(K1(dK2(eK3(T_{i-2} \oplus T_{i-1}))), I = 1, 2, \ldots 2p$

Encrypted Key Block 1 = T_{2p-1}

Encrypted Key Block 2 = T_{2p}

C.4.3.2 *n*-block encipherment

$T_{i-n} = K_i$, $i = 1, 2, \ldots n$

$T_i = eK1(dK2(eK3(T_{i-n \oplus T1-I}))$, $i = 1, 2, \ldots np$

$EKB_i = T_{i+n(p-1)}$, $i = 1, 2, \ldots n$

C.4.3.3 Decipherment Formulae

C.4.3.3.1 2-block decipherment

T_{2p} = Encrypted Key Block 1

T_{2p-1} = Encrypted Key Block 2

$T_{i-2} = dK1(eK2(dK3(T_i))) \oplus T_{i-1}$, $i = 2p, 2p-1, \ldots 1$

$KB1 = T_0$

$KB2 = T_{-1}$

C.4.3.3.2 *n*-block decipherment

$T_{i+n(p-1)} = EKB_i$, $i = 1, 2, \ldots n$

$T_{i-n} = dK1(eK2(dK3(T_i))) \oplus T_{i-1}$, $i = np, np-1, \ldots 1$

$KB_i = T_{i-n}$, I = 1, 2, … n

Bibliography

[1] ANSI X3.92-1981, *Data Encryption Algorithm*

[2] ANSI X9.52-1998, *Triple Data Encryption Algorithm — Modes of Operation*

[3] ISO/IEC 8372:1987, *Information processing — Modes of operation for 64-bit block cipher algorithm*

[4] X.680, ITU-T Recommendation X.680 (1997) | ISO/IEC 8824-1:1998, *Information Technology — Abstract Syntax Notation One (ASN.1): Specification of basic notation — Part 1*

[5] X.681, ITU-T Recommendation X.681 (1997) | ISO/IEC 8824-2:1998, *Information Technology — Abstract Syntax Notation One (ASN.1): Information object specification — Part 2*

[6] X.682, ITU-T Recommendation X.682 (1997) | ISO/IEC 8824-3:1998, *Information Technology — Abstract Syntax Notation One (ASN.1): Constraint specification — Part 3*

[7] X.683, ITU-T Recommendation X.683 (1997) | ISO/IEC 8824-4:1998, *Information Technology — Abstract Syntax Notation One (ASN.1): Parameterization of ASN.1 specifications — Part 4*

[8] X.690, ITU-T Recommendation X.690 (1997) | ISO/IEC 8825-1:1998, *Information Technology — ASN.1 Encoding rules: Specification of Basic Encoding Rules (BER), Canonical Encoding Rules (CER) and Distinguished Encoding Rules (DER) — Part 1*

[9] X.691, ITU-T Recommendation X.691 (1997) | ISO/IEC 8825-2:1998, *Information Technology — ASN.1 Encoding rules: Specification of Packed Encoding Rules (PER) — Part 2*

[10] ITU-T REC. X.509, *Information technology — Open Systems Interconnection — The Directory — Authentication framework*, International Communication Union, Geneva, Switzerland, 1997

[11] MENEZES, Alfred J., VAN OORSCHOT, Paul C. and VANSTONE, Scott A., *Handbook of Applied Cryptography*, CRC Press, 1997

[12] MEYER, Carl H. and MATYAS, Stephen M., *Cryptography: A New Dimension in Computer Data Security*, John Wiley & Sons, New York, 1982

ICS 35.240.40

Price based on 54 pages