INTERNATIONAL STANDARD

ISO/IEC 18033-3

Second edition
2010-12-15

AMENDMENT 1
2021-06

Information technology — Security techniques — Encryption algorithms —

Part 3:
Block ciphers

AMENDMENT 1: SM4

Technologies de l'information — Techniques de sécurité — Algorithmes de chiffrement —

Partie 3: Chiffrement par blocs

AMENDEMENT 1: SM4

Reference number
ISO/IEC 18033-3:2010/Amd.1:2021(E)

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work.

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

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

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

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

This document was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *Information security, cybersecurity and privacy protection*.

Any feedback or questions on this document should be directed to the user's national standards body. A complete listing of these bodies can be found at www.iso.org/members.html.

Information technology — Security techniques — Encryption algorithms —

Part 3: Block ciphers

AMENDMENT 1: SM4

Clause 1

In the first paragraph, replace "seven different block ciphers" with "eight different block ciphers".

Replace Table 1 with the following:

Block length	Algorithm name (see #)	Key length
64 bits	TDEA (4.2)	128 or 192 bits
	MISTY (4.3)	128 bits
	CAST-128 (4.4)	
	HIGHT (4.5)	
128 bits	AES (5.2)	128, 192 or 256 bits
	Camellia (5.3)	
	SEED (5.4)	128 bits
	SM4 (5.5)	

5.1

Replace the sentence with the following:

> In this clause, four 128-bit block ciphers are specified: AES in 5.2, Camellia in 5.3, SEED in 5.4, and SM4 in 5.5.

5.5

Add new subclause 5.5 as follows:

5.5 SM4

5.5.1 The SM4 algorithm

The SM4 algorithm is a symmetric block cipher that can process data blocks of 128 bits, using a cipher key with length of 128 bits under 32 rounds.

5.5.2 SM4 encryption

A 128-bit block P is transformed into a 128-bit block C using the following procedure, where for i = 0, 1, 2, 3 the X_i are 32-bit variables, and for i = 0, 1, ..., 31 the rk_i are 32-bit subkeys:

(1) $P = X_0 \,||\, X_1 \,||\, X_2 \,||\, X_3$

(2) for i = 0 to 31:

$$X_{i+4} = \mathrm{F}(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i)$$

(3) $C = X_{35} \,||\, X_{34} \,||\, X_{33} \,||\, X_{32}$

5.5.3 SM4 decryption

The decryption operation is identical to the encryption operation, except that the rounds (and therefore the subkeys) are used in reverse order:

(1) $C = X_{35} \,||\, X_{34} \,||\, X_{33} \,||\, X_{32}$

(2) for i = 31 to 0:

$$X_i = \mathrm{F}(X_{i+4}, X_{i+1}, X_{i+2}, X_{i+3}, rk_i)$$

(3) $P = X_0 \,||\, X_1 \,||\, X_2 \,||\, X_3$

5.5.4 SM4 functions

5.5.4.1 Function F

The function F is used for both encryption and decryption. The function F is defined as follows:

$$\mathrm{F}(X_0, X_1, X_2, X_3, rk) = X_0 \oplus \mathrm{T}(X_1 \oplus X_2 \oplus X_3 \oplus rk)$$

where X_i (i = 0, 1, 2, 3) and rk are bit strings of length 32, T is a permutation defined in 5.5.4.2.

5.5.4.2 Permutation T and T'

5.5.4.2.1 General

The permutation T is used both for encryption and decryption. T is a composition of a nonlinear transformation τ and a linear transformation L, that is T(·) = L(τ(·)). The permutation T' is used for the key schedule. T' is a composition of the nonlinear transformation τ and a linear transformation L', that is T'(·) = L'(τ(·)). T, T', L, L' and τ are all transformations on 32-bit strings.

5.5.4.2.2 Nonlinear transformation τ

The nonlinear transformation τ is defined as follows, where for i = 0, 1, 2, 3 the a_i are bytes and S is an S-box defined in 5.5.4.2.4:

$$\tau(a_0 \,||\, a_1 \,||\, a_2 \,||\, a_3) = \mathrm{S}(a_0) \,||\, \mathrm{S}(a_1) \,||\, \mathrm{S}(a_2) \,||\, \mathrm{S}(a_3).$$

5.5.4.2.3 Linear transformation L and L'

The linear transformation L is defined as follows (B is a 32-bit variable):

$$\mathrm{L}(B) = B \oplus (B{<}{<}{<}_2) \oplus (B{<}{<}{<}_{10}) \oplus (B{<}{<}{<}_{18}) \oplus (B{<}{<}{<}_{24}).$$

The linear transformation L' is defined as follows (B is a 32-bit variable):

$$\mathrm{L}'(B) = B \oplus (B{<}{<}{<}_{13}) \oplus (B{<}{<}{<}_{23}).$$

5.5.4.2.4 S-box S

The S-box S used in the transformation τ is presented in hexadecimal form in Table 17.

Table 17 — SM4 S-box

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	d6	90	e9	fe	cc	e1	3d	b7	16	b6	14	c2	28	fb	2c	05
1	2b	67	9a	76	2a	be	04	c3	aa	44	13	26	49	86	06	99
2	9c	42	50	f4	91	ef	98	7a	33	54	0b	43	ed	cf	ac	62
3	e4	b3	1c	a9	c9	08	e8	95	80	df	94	fa	75	8f	3f	a6
4	47	07	a7	fc	f3	73	17	ba	83	59	3c	19	e6	85	4f	a8
5	68	6b	81	b2	71	64	da	8b	f8	eb	0f	4b	70	56	9d	35
6	1e	24	0e	5e	63	58	d1	a2	25	22	7c	3b	01	21	78	87
7	d4	00	46	57	9f	d3	27	52	4c	36	02	e7	a0	c4	c8	9e
8	ea	bf	8a	d2	40	c7	38	b5	a3	f7	f2	ce	f9	61	15	a1
9	e0	ae	5d	a4	9b	34	1a	55	ad	93	32	30	f5	8c	b1	e3
a	1d	f6	e2	2e	82	66	ca	60	c0	29	23	ab	0d	53	4e	6f
b	d5	db	37	45	de	fd	8e	2f	03	ff	6a	72	6d	6c	5b	51
c	8d	1b	af	92	bb	dd	bc	7f	11	d9	5c	41	1f	10	5a	d8
d	0a	c1	31	88	a5	cd	7b	bd	2d	74	d0	12	b8	e5	b4	b0
e	89	69	97	4a	0c	96	77	7e	65	b9	f1	09	c5	6e	c6	84
f	18	f0	7d	ec	3a	dc	4d	20	79	ee	5f	3e	d7	cb	39	48

5.5.5 SM4 key schedule

The key scheduling part accepts a 128-bit master key $MK = MK_0 \| MK_1 \| MK_2 \| MK_3$, and yields 32 subkeys, as shown below.

(1) $K_0 \| K_1 \| K_2 \| K_3 = (MK_0 \oplus FK_0) \| (MK_1 \oplus FK_1) \| (MK_2 \oplus FK_2) \| (MK_3 \oplus FK_3)$

(2) for $i = 0$ to 31:

$$rk_i = K_{i+4} = K_i \oplus \mathrm{T}'(K_{i+1} \oplus K_{i+2} \oplus K_{i+3} \oplus CK_i)$$

The constants FK_i ($i = 0, 1, 2, 3$) are as follows (in hexadecimal form):

FK_0 = `a3b1bac6`, FK_1 = `56aa3350`, FK_2 = `677d9197`, FK_3 = `b27022dc`.

The constants CK_i ($i = 0, 1, ..., 31$) are defined as follows. Suppose $CK_i = ck_{i,0} \| ck_{i,1} \| ck_{i,2} \| ck_{i,3}$, where $ck_{i,j}$ are bytes, and $ck_{i,j} = (4i+j)\times 7 \pmod{256}$ ($i = 0, 1, ..., 31, j = 0, 1, 2, 3$).

Thus, the values of CK_i ($i = 0, 1, ..., 31$) are (in hexadecimal form):

```
00070e15,   1c232a31,   383f464d,   545b6269,
70777e85,   8c939aa1,   a8afb6bd,   c4cbd2d9,
e0e7eef5,   fc030a11,   181f262d,   343b4249,
50575e65,   6c737a81,   888f969d,   a4abb2b9,
c0c7ced5,   dce3eaf1,   f8ff060d,   141b2229,
30373e45,   4c535a61,   686f767d,   848b9299,
a0a7aeb5,   bcc3cad1,   d8dfe6ed,   f4fb0209,
10171e25,   2c333a41,   484f565d,   646b7279.
```

Annex B

Insert the following line after `id-bc128-seed`:

```
id-bc128-sm4 OID ::= {id-bc128 sm4(4)}
```

Replace `{ OID id-bc128-seed PARMS KeyLength }` , with the following:

```
{ OID id-bc128-seed PARMS KeyLength } |
{ OID id-bc128-sm4  PARMS KeyLength },
```

Annex D

Change the title to "Numerical examples".

Replace "test vectors" with "numerical examples".

D.1

Change the heading to "General".

Replace the text with the following *:*

> This annex provides numerical examples for TDEA, MISTY1, CAST-128, HIGHT, AES, Camellia, SEED, and SM4 ciphers. In these examples, all data are expressed in hexadecimal.

D.9

Add new clause D.9 as follows:

D.9 SM4 numerical examples

D.9.1 SM4 encryption

Given inputs (plaintext and key), output (ciphertext and subkeys) and intermediate values are described.

Input plaintext: `01 23 45 67 89 ab cd ef fe dc ba 98 76 54 32 10`.

Input key: `01 23 45 67 89 ab cd ef fe dc ba 98 76 54 32 10`.

The subkeys and the values of the output of each round:

rk_0 = f12186f9	X_4 = 27fad345
rk_1 = 41662b61	X_5 = a18b4cb2
rk_2 = 5a6ab19a	X_6 = 11c1e22a
rk_3 = 7ba92077	X_7 = cc13e2ee
rk_4 = 367360f4	X_8 = f87c5bd5
rk_5 = 776a0c61	X_9 = 33220757

rk_6 = b6bb89b3	X_{10} = 77f4c297
rk_7 = 24763151	X_{11} = 7a96f2eb
rk_8 = a520307c	X_{12} = 27dac07f
rk_9 = b7584dbd	X_{13} = 42dd0f19
rk_{10} = c30753ed	X_{14} = b8a5da02
rk_{11} = 7ee55b57	X_{15} = 907127fa
rk_{12} = 6988608c	X_{16} = 8b952b83
rk_{13} = 30d895b7	X_{17} = d42b7c59
rk_{14} = 44ba14af	X_{18} = 2ffc5831
rk_{15} = 104495a1	X_{19} = f69e6888
rk_{16} = d120b428	X_{20} = af2432c4
rk_{17} = 73b55fa3	X_{21} = ed1ec85e
rk_{18} = cc874966	X_{22} = 55a3ba22
rk_{19} = 92244439	X_{23} = 124b18aa
rk_{20} = e89e641f	X_{24} = 6ae7725f
rk_{21} = 98ca015a	X_{25} = f4cba1f9
rk_{22} = c7159060	X_{26} = 1dcdfa10
rk_{23} = 99e1fd2e	X_{27} = 2ff60603
rk_{24} = b79bd80c	X_{28} = eff24fdc
rk_{25} = 1d2115b0	X_{29} = 6fe46b75
rk_{26} = 0e228aeb	X_{30} = 893450ad
rk_{27} = f1780c81	X_{31} = 7b938f4c
rk_{28} = 428d3654	X_{32} = 536e4246
rk_{29} = 62293496	X_{33} = 86b3e94f
rk_{30} = 01cf72e5	X_{34} = d206965e
rk_{31} = 9124a012	X_{35} = 681edf34

The output ciphertext: `68 1e df 34 d2 06 96 5e 86 b3 e9 4f 53 6e 42 46`.

D.9.2 SM4 encryption 1 000 000 times

Given inputs (plaintext and key), output (ciphertext) after encryption iteratively 1 000 000 times is described.

Input plaintext: `01 23 45 67 89 ab cd ef fe dc ba 98 76 54 32 10`.