SMS4 Encryption Algorithm for Wireless Networks

Translated and typeset by

Whitfield Diffie of Sun Microsystems

and

George Ledin of Sonoma State University

15 May 2008

Version 1.03

SMS4 is a Chinese block cipher standard, mandated for use in protecting wireless networks, and issued in January 2006. The input, output, and key of SMS4 are each 128 bits. The algorithm has 32 rounds, each of which modifies one of the four 32-bit words that make up the block by xoring it with a keyed function of the other three words. Encryption and decryption have the same structure except that the round key schedule for decryption is the reverse of the round key schedule for encryption.

SMS4 Encryption Algorithm for Wireless Networks

The SMS4 algorithm is a block cipher with 128-bit key and 128-bit input block. Encryption and decryption take 32 rounds of nonlinear substitutions. Encryption and decryption have the same structure, but the round key schedule for decryption is the reverse (goes in the opposite order) of the round key schedule for encryption.

1. Terminology (Definitions)

1.1 Zi and ZiJie (Word and Byte)

Z_2^e is the set of e-bit vectors. Specifically, the elements of Z_2^{32} are called Zi (32-bit words), and the elements of Z_2^8 are called ZiJie (8-bit characters, or bytes).

1.2 S box

The S (substitution) box takes in 8 bits and outputs 8 bits. It is written Sbox(.).

1.3 Fundamental Operations

The two fundamental operations used by this algorithm are:

$\oplus$ the bitwise XOR of two 32-bit vectors,

$<<< i$ the circular shift of a 32-bit word, with i bits shifted left.

1.4 Input and output blocks, and key

The 128-bit input block consists of four 32-bit words $MK = (MK_1, MK_2, MK_3, MK_4)$ or $MK_i (i = 0, 1, 2, 3)$.

The round key schedule, derived from the encryption key, is represented by $(rk_0, rk_1, \ldots, rk_{31})$, where each $rk_i (i = 0, \ldots, 31)$ is 32 bits long.

The 128-bit output block consists of four 32-bit words $FK = (FK_0, FK_1, FK_2, FK_3)$. For decryption, the round key schedule is represented by $CK = (CK_0, CK_1, \ldots, CK_{31})$ or $FK_i (i = 0, \ldots, 3)$, $CK_i (i = 0, \ldots, 31)$.

2. The round function F

This algorithm uses a nonlinear substitution structure, encrypting 32 bits at a time. This is called a *one-round exchange.* To illustrate, consider a one-round-substitution:

Let the 128-bit input block be the four 32-bit elements $(X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$, with $rk \in Z_2^{32}$, then F is given by

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

2.1 Mixer-substitution T

T is a substitution that generates 32 bits from 32 bits $T : Z_2^{32} \mapsto Z_2^{32}$. This substitution is a reversible process. It consists of a non-linear substitution, τ, and a linear substitution L, i.e., $T(.) = L(\tau(.))$.

2.1.1 Non-linear substitution τ

τ applies 4 S-boxes in parallel.

Let a 32-bit input word be $A = (a_0, a_1, a_2, a_3) \in (Z_2^8)^4$, where each a_i is an 8-bit character. Let the 32-bit output word be $B = (b_0, b_1, b_2, b_3) \in (Z_2^8)^4$, given by

$$(b_0, b_1, b_2, b_3) = \tau(A) = (\text{Sbox}(a_0), \text{Sbox}(a_1), \text{Sbox}(a_2), \text{Sbox}(a_3))$$

2.1.2 Linear substitution L

$B \in Z_2^{32}$, the 32-bit output word of the non-linear substitution τ will be the input word of the linear substitution L. Let $C \in Z_2^{32}$ be the 32-bit output word generated by L. Then

$$C = L(B) = B \oplus (B <\!\!<\!\!< 2) \oplus (B <\!\!<\!\!< 10) \oplus (B <\!\!<\!\!< 18) \oplus (B <\!\!<\!\!< 24)$$

2.2 S box

All Sbox numbers are in hexadecimal notation.

	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

For example, if the input to the Sbox is 'ef', then go to e-th row and f-th column, to find Sbox('ef')='84'.

3. Encryption and decryption

Let the reverse substitution R be:

$$R(A_0, A_1, A_2, A_3) = (A_3, A_2, A_1, A_0), A_i \in Z_2^{32}, i = 0, 1, 2, 3.$$

Let the plaintext input be $(X_0, X_1, X_2, X_3) \in (Z_2^{32})^4$, the ciphertext output be $(Y_0, Y_1, Y_2, Y_3) \in (Z_2^{32})^4$, and the encrypting key be $rk_i \in Z_2^{32}, i = 0, 1, 2, \ldots, 31$. Then encryption proceeds as follows:

$$X_{i+4} = F(X_i, X_{i+1}, X_{i+2}, X_{i+3}, rk_i) = X_i \oplus T(X_{i+1} \oplus X_{i+2} \oplus X_{i+3} \oplus rk_i), i = 0, 1, \ldots, 31$$

$$(Y_0, Y_1, Y_2, Y_3) = R(X_{32}, X_{33}, X_{34}, X_{35}) = (X_{35}, X_{34}, X_{33}, X_{32}).$$

This algorithm's encryption and decryption methods have the same structure, except the order in which the round keys are used is reversed.

The key order for encryption is: $(rk_0, rk_1, \ldots rk_{31})$. The key order for decryption is: $(rk_{31}, rk_{30}, \ldots rk_0)$.

4. Key expansion when encrypting

The rk_i round key used for encrypting in this algorithm is derived from the encryption key MK.

Let $MK = (MK_0, MK_1, MK_2, MK_3)$, $MK_i \in Z_2^{32}, i = 0, 1, 2, 3$; $K_i \in Z_2^{32}, i = 0, 1, \ldots, 31$; $rk_i \in Z_2^{32}$, $i = 0, 1, \ldots, 31$; the derivation proceeds as follows:

First,

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

Then for $i = 0, 1, 2, \ldots, 31$:

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

Notes:

(1) T' substitution uses the same T as in encryption, except the linear substitution L is changed to L':

$$L'(B) = B \oplus (B \lll 13) \oplus (B \lll 23);$$

(2) The system parameter FK, given in hexadecimal notation is

$$FK_0 = (a3b1bac6), FK_1 = (56aa3350), FK_2 = (677d9197), FK_3 = (b27022dc).$$

(3) The constant parameter CK is calculated as follows:

Let $ck_{i,j}$ be the j-th byte of $CK_{i,j} (i = 0, 1, \ldots, 31; j = 0, 1, 2, 3)$, i.e., $CK_i = (ck_{i,0}, ck_{i,1}, ck_{i,2}, ck_{i,3}) \in (Z_2^8)^4$, then $ck_{i,j} = (4i + j) \times 7 \pmod{256}$. The 32 constants CK_i are represented in hexadecimal as tabulated below.

```
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
```

5. Encryption examples

Below are encryption examples of this algorithm's ECB (electronic code book mode) calculation method. We use this to verify the correctness of this algorithm's encryption. The numbers are represented in hexadecimal notation.

Example 1: Encrypt plaintext with key once

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

rk and the output in each round:

```
rk[ 0]  =  f12186f9  X[ 0]  =  27fad345
rk[ 1]  =  41662b61  X[ 1]  =  a18b4cb2
rk[ 2]  =  5a6ab19a  X[ 2]  =  11c1e22a
rk[ 3]  =  7ba92077  X[ 3]  =  cc13e2ee
rk[ 4]  =  367360f4  X[ 4]  =  f87c5bd5
rk[ 5]  =  776a0c61  X[ 5]  =  33220757
rk[ 6]  =  b6bb89b3  X[ 6]  =  77f4c297
rk[ 7]  =  24763151  X[ 7]  =  7a96f2eb
rk[ 8]  =  a520307c  X[ 8]  =  27dac07f
rk[ 9]  =  b7584dbd  X[ 9]  =  42dd0f19
rk[10]  =  c30753ed  X[10]  =  b8a5da02
rk[11]  =  7ee55b57  X[11]  =  907127fa
rk[12]  =  6988608c  X[12]  =  8b952b83
rk[13]  =  30d895b7  X[13]  =  d42b7c59
rk[14]  =  44ba14af  X[14]  =  2ffc5831
rk[15]  =  104495a1  X[15]  =  f69e6888
rk[16]  =  d120b428  X[16]  =  af2432c4
rk[17]  =  73b55fa3  X[17]  =  ed1ec85e
rk[18]  =  cc874966  X[18]  =  55a3ba22
rk[19]  =  92244439  X[19]  =  124b18aa
rk[20]  =  e89e641f  X[20]  =  6ae7725f
rk[21]  =  98ca015a  X[21]  =  f4cba1f9
rk[22]  =  c7159060  X[22]  =  1dcdfa10
rk[23]  =  99e1fd2e  X[23]  =  2ff60603
rk[24]  =  b79bd80c  X[24]  =  eff24fdc
rk[25]  =  1d2115b0  X[25]  =  6fe46b75
rk[26]  =  0e228aeb  X[26]  =  893450ad
rk[27]  =  f1780c81  X[27]  =  7b938f4c
rk[28]  =  428d3654  X[28]  =  536e4246
rk[29]  =  62293496  X[29]  =  86b3e94f
rk[30]  =  01cf72e5  X[30]  =  d206965e
rk[31]  =  9124a012  X[31]  =  681edf34
```

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

Example 2: Use the same encryption key and encrypt the plaintext again and again 1,000,000 times.

```
plaintext:        01 23 45 67 89 ab cd ef fe dc ba 98 76 54 32 10
encrypting key:   01 23 45 67 89 ab cd ef fe dc ba 98 76 54 32 10
ciphertext:       59 52 98 c7 c6 fd 27 1f 04 02 f8 04 c3 3d 3f 66
```