Specification: Hierocrypt-3

Toshiba Corporation

August 31, 2000

Contents

1 Algorithm of Hierocrypt-3

1.1 Notations

An n-bit value is basically expressed with the subscript (n). For example, the value $X_{(n)}$ is an element of $\mathrm{GF}(2)^n$. A value expressed by a capital(s) describes an element of no less than 16 bits A value expressed by a small letter(s) describes an element of less than 16 bits.

We adopt the bigendian convention. When a value $X_{(mn)}$ is expressed as a concatenation of m pieces of n-bit length, each piece is expressed with the subscript $i(n)$ $(i = 1, 2, \ldots, m)$, that is $X_{(mn)} = X_{1(n)} \| X_{2(n)} \| \cdots \| X_{m(n)}$. Furthermore, $X_{(mn)}$ and $X_{i(n)}$ are expressed as the following concatenations: $X_{(mn)} = x_{1(1)} \| x_{2(1)} \| \cdots \| x_{mn(1)}$, $X_{i(n)} = x_{ni-n+1(1)} \| x_{ni-n+2(1)} \| \cdots \| x_{ni-n+n(1)}$. The following shows an example of concatenation expression of a 128-bit value $X_{(128)}$.

$$X_{(128)} = X_{1(32)} \| X_{2(32)} \| X_{3(32)} \| X_{4(32)} \ ,$$

$$X_{i(32)} = x_{4i-4+1(8)} \| x_{4i-4+2(8)} \| \cdots \| x_{4i-4+4(8)} \ , i = 1, 2, \ldots, 4,$$

$$X_{j(8)} = x_{8j-8+1(1)} \| x_{8j-8+2(1)} \| \cdots \| x_{8j-8+8(1)} \ , j = 1, 2, \ldots, 16.$$

Note that the LSB of the value $X_{i(n)}$ (i=1,2,...,m) is $x_{in(1)}$, which is the in-th MSB of $X_{(mn)}$.

2 Structure of the algorithm

The structures of data randomization part and the key scheduling part are described in this section. Fundamental operations used there are described in the next section.

2.1 Encryption

The T-round encryption of Hierocrypt-3 consists of $(T-1)$ operations of round function ρ, an operation of XS-function, and the final key addition (AK) as shown below.

$$P_{(128)} \equiv X^{(0)}_{(128)} \stackrel{\rho}{\longmapsto} X^{(1)}_{(128)} \stackrel{\rho}{\longmapsto} \cdots \stackrel{\rho}{\longmapsto} X^{(T-1)}_{(128)} \stackrel{XS}{\longmapsto} X^{(T)}_{(128)} \stackrel{AK}{\longmapsto} C_{(128)}$$

T is 6,7, or 8 for 128-, 192-, or 256-bit, respectively.

The 128-bit value $X^{(i)}_{(128)}$ is the output of the i-th operation of round function ρ $(i = 1, 2, \ldots, T-1)$. The plaintext $P_{(128)}$ is assigned to the 0-th value $X^{(0)}_{(128)}$.

The value $X^{(t)}_{(128)}$ is the output of the t-th operation of ρ-function for the input $X^{(t-1)}_{(128)}$ and the round key $K^{(t)}_{(256)}$.

$$X^{(t)}_{(128)} = \rho(X^{(t-1)}_{(128)}, K^{(t)}_{(256)}), \quad t = 1, 2, \cdots, T-1.$$

Similarly, $X^{(T)}_{(128)}$ is the output of XS-function for the input $X^{(T-1)}_{(128)}$ and the final key $K^{(T)}_{(256)}$.

$$X^{(T)}_{(128)} = XS(X^{(T-1)}_{(128)}, K^{(T)}_{(256)}) \ .$$

The ciphertext $C_{(128)}$ is given as the addition (XOR, exclusive or) between the T-th round output $X^{(T)}_{(128)}$ and the first half of the final key $K^{(T+1)}_{1(128)}$

$$C_{(128)} = X^{(T)}_{(128)} \oplus (K^{(T+1)}_{1(64)} \| K^{(T+1)}_{2(64)}) \ .$$

2.2 decryption

The decryption of Hierocrypt-3 is the inverse of encryption, and consists of the final key addition, the inverse of XS-function (XS^{-1}), and $(T-1)$ inverse operations of round function (ρ^{-1}).

$$\begin{aligned}
&X^{(T)}_{(128)} = C_{(128)} \oplus (K^{(T+1)}_{1(64)} \| K^{(T+1)}_{2(64)}) \ , \\
&X^{(T-1)}_{(128)} = XS^{-1}(X^{(T)}_{(128)}, K^{(T)}_{(256)}) \ , \\
&X^{(t-1)}_{(128)} = \rho^{-1}(X^{(t)}_{(128)}, K^{(t)}_{(256)}) \ , \ t = T-1, \ldots, 2, 1.
\end{aligned}$$

The plaintext $P_{(128)}$ is given as the final output $X^{(0)}_{(128)}$.

$$P_{(128)} = X^{(0)}_{(128)} \ .$$

2.3 Key scheduling

The main part of key scheduling consists of the intermediate key generation part and the round key generation part, preceded by the intermediate key initialization. The intermediate key part recursively generates intermediate key outputs $Z^{(t)}_{(256)}$ $(t = 1, 2, \ldots, T+1)$, and the round key generation part gerates round keys $K^{(t)}_{(256)}$ $(t = 1, 2, \ldots, T+1)$ from the corresponding intermediate keys, as follows.

$$\begin{aligned} &K_{(\mathrm{length})} \overset{\mathrm{padding}}{\longmapsto} Z^{(-1)}_{(256)} \overset{\sigma_0}{\longmapsto} Z^{(0)}_{(256)} \overset{\sigma}{\longmapsto} Z^{(1)}_{(256)} \\ &\overset{\sigma}{\longmapsto} Z^{(2)}_{(256)} \overset{\sigma}{\longmapsto} \cdots \overset{\sigma}{\longmapsto} Z^{(t_{\mathrm{turn}})}_{(256)} \\ &\overset{\sigma^{-1}}{\longmapsto} Z^{(t_{\mathrm{turn}}+1)}_{(256)} \overset{\sigma^{-1}}{\longmapsto} \cdots \overset{\sigma^{-1}}{\longmapsto} Z^{(T+1)}_{(256)} \end{aligned}$$

The intermediate key $Z^{(t)}_{(128)}$ and the round key $K^{(t)}_{(128)}$ are divided into 4 pieces. $Z^{(t)}_{(128)}$ and $K^{(t)}_{(128)}$ are divided into 4 pieces.

$$\begin{aligned} Z^{(t)}_{(256)} &= Z^{(t)}_{1(64)} \| Z^{(t)}_{2(64)} \| Z^{(t)}_{3(64)} \| Z^{(t)}_{4(64)} \ , \\ K^{(t)}_{(256)} &= K^{(t)}_{1(64)} \| K^{(t)}_{2(64)} \| K^{(t)}_{3(64)} \| K^{(t)}_{4(64)} \ . \end{aligned}$$

To generate the intermediate keys, the σ-function is used for $1 \le t \le t_{\mathrm{turn}}$, and the σ^{-1}-function is used for $t_{\mathrm{turn}} + 1 \le t \le T + 1$, where $t_{\mathrm{turn}} = 4$ for the 128/192-bit key and where $t_{\mathrm{turn}} = 5$ for the 256-bit key. Under the recursion rule, the intermediate key values are symmetric with regard to the point $t = t_{\mathrm{turn}}$.

$$Z^{(t)}_{(256)} = Z^{(2t_{\mathrm{turn}}-t)}_{(256)} \ , \quad t_{\mathrm{turn}} + 1 \le t \le T + 1 \ .$$

2.3.1 round-dependent constants

To prevent periodic patterns from appearing in the intermediate key generation, and to improve resistance against the related key attack, we introduce round-dependent key additions to the intermediate key generation part. The round-dependent keys have been made by combining two from the four 32-bit values which are given as binary expansions of irrational numbers.

$$\begin{aligned} H_0 &= \ \mathrm{0x5A827999} = \ trunc(\sqrt{2}/4), \\ H_1 &= \ \mathrm{0x6ED9EBA1} = \ trunc(\sqrt{3}/4), \\ H_2 &= \ \mathrm{0x8F1BBCDC} = \ trunc(\sqrt{5}/4), \\ H_3 &= \ \mathrm{0xCA62C1D6} = \ trunc(\sqrt{10}/4), \\ trunc(x) &= \lfloor 2^{32} x \rfloor \ , \\ G_0(0) &= H_3 \| H_0 \ , \quad G_0(1) = H_2 \| H_1 \ , \\ G_0(2) &= H_1 \| H_3 \ , \quad G_0(3) = H_0 \| H_2 \ , \\ G_0(4) &= H_2 \| H_3 \ , \quad G_0(5) = H_1 \| H_0 \ . \end{aligned}$$

2.3.2 Preprocessing

The intermediate key $Z^{(-1)}_{(256)}$ is made of the encryption key $K_{(\mathrm{length})}$ (length = 128, 192, 256) with an initial operation. The intermediate key $Z^{(0)}_{(256)}$ is deriven from $Z^{(-1)}_{(256)}$ through the pre-whitening operation σ_0. The padding operation is done when length = 192, 256 where padded values are concatenations of the above mentioned 32-bit constants H_i. The padding operations are described as follows.

[**128-bit key**]

$$K_{1(64)} \| K_{2(64)} = K_{(128)} \ ,$$

$$Z^{(-1)}_{1(64)} = K_{1(64)} \ , \quad Z^{(-1)}_{2(64)} = K_{2(64)} \ ,$$

$$Z^{(-1)}_{3(64)} = K_{1(64)} \ , \quad Z^{(-1)}_{4(64)} = H_3 \| H_2 \ .$$

[**192-bit key**]

$$K_{1(64)}\|K_{2(64)}\|K_{3(64)} = K_{(192)} \ ,$$
$$Z^{(-1)}_{1(64)} = K_{1(64)} \ , \quad Z^{(-1)}_{2(64)} = K_{2(64)} \ ,$$
$$Z^{(-1)}_{3(64)} = K_{3(64)} \ , \quad Z^{(-1)}_{4(64)} = H_2\|H_3 \ .$$

[**256-bit key**]

$$K_{1(64)}\|K_{2(64)}\|K_{3(64)}\|K_{4(64)} = K_{(256)} \ ,$$
$$Z^{(-1)}_{1(64)} = K_{1(64)} \ , \quad Z^{(-1)}_{2(64)} = K_{2(64)} \ ,$$
$$Z^{(-1)}_{3(64)} = K_{3(64)} \ , \quad Z^{(-1)}_{4(64)} = K_{4(64)} \ .$$

[**pre-whitening**](ρ_0)

The pre-whitening is done, before iterative operation by the σ-function. The pre-whitening operation ρ_0 is made from ρ by removing $P^{(32)}$.

$$Z^{(0)}_{(256)} = \sigma_0(Z^{(-1)}_{(256)}, G^{(0)}_{(64)}),$$
$$\begin{aligned}
Z^{(0)}_{3(64)} &= M_{5E}(Z^{(-1)}_{3(64)}) \oplus G^{(0)}_{(64)} \ , \\
Z^{(0)}_{4(64)} &= M_{5E}(Z^{(-1)}_{4(64)}) \ , \\
Z^{(0)}_{1(64)} &= Z^{(-1)}_{2(64)} \ , \\
Z^{(0)}_{2(64)} &= Z^{(-1)}_{1(64)} \oplus F_\sigma(Z^{(-1)}_{2(64)} \oplus Z^{(0)}_{3(64)}) \ .
\end{aligned}$$

As the round-dependent constant $G^{(0)}_{(64)}$, the following 64-bit concatenated value is used.

$$G^{(0)}_{(64)} = G_0(5) = H_1\|H_0 \ .$$

2.3.3 Intermediate key update (σ-function)

The intermediate key $Z^{(t)}_{(256)}$ is generated by the operation σ up to $t = t_{\rm turn}$, and afterwards by the inverse operation σ^{-1}. The sequence of intermediate keys is symmetric with respect to the point $t = t_{\rm turn}$ for this round-trip-type scheduling.

$$Z^{(t)}_{(256)} = Z^{(2t_{\rm turn}-t)}_{(256)} \ , \quad t_{\rm turn} \le t \le T+1 \ .$$

We call the region: $(1 \le t \le t_{\rm turn})$ as the plaintext side, and the other region: $(t_{\rm turn}+1 \le t \le T+1)$ as the ciphertext side, corresponding to the position in the data randomizing part.

[**Iteration of intermediate key(plaintext side)**] $(1 \le t \le t_{\rm turn})$

$$Z^{(t)}_{(256)} = \sigma(Z^{(t-1)}_{(256)} \ , \quad G^{(t)}_{(64)}) \ ,$$
$$\begin{aligned}
W^{(t-1)}_{1(64)}\|W^{(t-1)}_{2(64)} &= P^{(32)}(Z^{(t-1)}_{3(64)}\|Z^{(t-1)}_{4(64)}) \ , \\
Z^{(t)}_{3(64)} &= M_{5E}(W^{(t-1)}_{1(64)}) \oplus G^{(t)}_{(64)} \ , \\
Z^{(t)}_{4(64)} &= M_{5E}(W^{(t-1)}_{2(64)}) \ , \\
Z^{(t)}_{1(64)} &= Z^{(t-1)}_{2(64)} \ , \\
Z^{(t)}_{2(64)} &= Z^{(t-1)}_{1(64)} \oplus F_\sigma(Z^{(t-1)}_{2(64)} \oplus Z^{(t)}_{3(64)}) \ .
\end{aligned}$$

[**Iteration of intermediate key(ciphertext side)**] $(t_{\rm turn}+1 \le t \le T+1)$

$$Z^{(t)}_{(256)} = \sigma^{-1}(Z^{(t-1)}_{(256)}, G^{(t-1)}_{(64)}) \ ,$$
$$\begin{aligned}
Z^{(t)}_{1(64)} &= Z^{(t-1)}_{2(64)} \oplus F_\sigma(Z^{(t-1)}_{1(64)} \oplus Z^{(t-1)}_{3(64)}) \ , \\
Z^{(t)}_{2(64)} &= Z^{(t-1)}_{1(64)} \ , \\
W^{(t)}_{1(64)} &= M_{B3}(Z^{(t-1)}_{3(64)} \oplus G^{(t-1)}_{(64)}) \ , \\
W^{(t)}_{2(64)} &= M_{B3}(Z^{(t-1)}_{4(64)}) \ , \\
Z^{(t)}_{3(64)}\|Z^{(t)}_{4(64)} &= {P^{(32)}}^{-1}(W^{(t)}_{1(64)}\|W^{(t)}_{2(64)}) \ .
\end{aligned}$$

2.3.4 Round key generation

The different rules are applied to generate a round key from the corresponding intermediate key for the plaintext side and the ciphertext side.

[**Round key generation(plaintext side)**] $(1 \le t \le t_{\mathrm{turn}})$

$$V_{(64)}^{(t)} = F_\sigma(Z_{2(64)}^{(t-1)} \oplus Z_{3(64)}^{(t)}) \ ,$$
$$K_{1(64)}^{(t)} = Z_{1(64)}^{(t-1)} \oplus V_{(64)}^{(t)} \ , \quad K_{2(64)}^{(t)} = Z_{3(64)}^{(t)} \oplus V_{(64)}^{(t)} \ ,$$
$$K_{3(64)}^{(t)} = Z_{4(64)}^{(t)} \oplus V_{(64)}^{(t)} \ , \quad K_{4(64)}^{(t)} = Z_{2(64)}^{(t-1)} \oplus Z_{4(64)}^{(t)} \ .$$

[**Round key generation(ciphertext side)**] $(t_{\mathrm{turn}} + 1 \le t \le T + 1)$

$$V_{(64)}^{(t)} = F_\sigma(Z_{1(64)}^{(t-1)} \oplus Z_{3(64)}^{(t-1)}) \ ,$$
$$K_{1(64)}^{(t)} = Z_{1(64)}^{(t)} \oplus Z_{3(64)}^{(t-1)} \ , \quad K_{2(64)}^{(t)} = W_{1(64)}^{(t)} \oplus V_{(64)}^{(t)} \ ,$$
$$K_{3(64)}^{(t)} = W_{2(64)}^{(t)} \oplus V_{(64)}^{(t)} \ , \quad K_{4(64)}^{(t)} = Z_{1(64)}^{(t-1)} \oplus W_{2(64)}^{(t)} \ .$$

Table 1: Key schedule for 128-bit key (6 rounds)

round key	t	operation	$G_{(64)}^{(t)}$
–	-1 (PAD)	–	$H_3 \| H_2$
–	0 (PW)	σ_0	$G_0(5)$
$K_{(256)}^{(1)}$	1	σ	$G_0(0)$
$K_{(256)}^{(2)}$	2	σ	$G_0(1)$
$K_{(256)}^{(3)}$	3	σ	$G_0(2)$
$K_{(256)}^{(4)}$	4	σ	$G_0(3)$
$K_{(256)}^{(5)}$	5	σ^{-1}	$G_0(3)$
$K_{(256)}^{(6)}$	6	σ^{-1}	$G_0(2)$
$K_{(256)}^{(7)}$	7	σ^{-1}	$G_0(1)$

Table 2: Key schedule for 192-bit key (7 rounds)

round key	t	operation	$G_{(64)}^{(t)}$
–	-1 (PAD)	–	$H_2 \| H_3$
–	0 (PW)	σ_0	$G_0(5)$
$K_{(256)}^{(1)}$	1	σ	$G_0(1)$
$K_{(256)}^{(2)}$	2	σ	$G_0(0)$
$K_{(256)}^{(3)}$	3	σ	$G_0(3)$
$K_{(256)}^{(4)}$	4	σ	$G_0(2)$
$K_{(256)}^{(5)}$	5	σ^{-1}	$G_0(2)$
$K_{(256)}^{(6)}$	6	σ^{-1}	$G_0(3)$
$K_{(256)}^{(7)}$	7	σ^{-1}	$G_0(0)$
$K_{(256)}^{(8)}$	8	σ^{-1}	$G_0(1)$

3 Fundamental Operations

3.1 Round function ρ

The ρ-function, which is the round function of the data randomization part, is a composite function of the XS-function and the MDS_{L}-function, where the input data are the 128-bit value $X_{(128)}$ and the 256-bit value

Table 3: Key schedule for 256-bit key (8 rounds)

round key	t	operation	$G^{(t)}_{(64)}$
$-$	-1 (PAD)	$-$	$-$
$-$	0 (PW)	σ_0	$G_0(5)$
$K^{(1)}_{(256)}$	1	σ	$G_0(4)$
$K^{(2)}_{(256)}$	2	σ	$G_0(0)$
$K^{(3)}_{(256)}$	3	σ	$G_0(2)$
$K^{(4)}_{(256)}$	4	σ	$G_0(1)$
$K^{(5)}_{(256)}$	5	σ	$G_0(3)$
$K^{(6)}_{(256)}$	6	σ^{-1}	$G_0(3)$
$K^{(7)}_{(256)}$	7	σ^{-1}	$G_0(1)$
$K^{(8)}_{(256)}$	8	σ^{-1}	$G_0(2)$
$K^{(9)}_{(256)}$	9	σ^{-1}	$G_0(0)$

$K_{(256)}$.

$$\rho(X_{(128)}, K_{(256)}) = MDS_{\mathrm{H}}(XS(X_{(128)}, K_{(256)})) \ .$$

3.2 MDS_{H}-function

The MDS_{H}-function is a linear transformation consisting of exclusive or's between 8-bit subdata $x_{i(8)}$ ($\in$ GF(2)^8; $i = 1, 2, \ldots, 16$), which is represented by the following matrix form.

$$Y_{(128)} = MDS_{\mathrm{H}}(X_{(128)}) \ ,$$

$$\begin{pmatrix} y_{1(8)} \\ y_{2(8)} \\ y_{3(8)} \\ y_{4(8)} \\ y_{5(8)} \\ y_{6(8)} \\ y_{7(8)} \\ y_{8(8)} \\ y_{9(8)} \\ y_{10(8)} \\ y_{11(8)} \\ y_{12(8)} \\ y_{13(8)} \\ y_{14(8)} \\ y_{15(8)} \\ y_{16(8)} \end{pmatrix} = \begin{pmatrix} 1\,0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1\,1\,1\,1\,1 \\ 1\,1\,0\,1\,1\,1\,0\,1\,1\,1\,1\,0\,0\,1\,1\,1 \\ 1\,1\,1\,0\,1\,1\,1\,0\,1\,1\,1\,1\,0\,0\,1\,1 \\ 0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1\,0\,1\,1\,1\,0 \\ 1\,1\,1\,1\,1\,1\,0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1 \\ 0\,1\,1\,1\,1\,1\,1\,0\,1\,1\,1\,0\,1\,1\,1\,1\,0 \\ 0\,0\,1\,1\,1\,1\,1\,1\,0\,1\,1\,1\,0\,1\,1\,1\,1 \\ 1\,1\,1\,0\,0\,1\,0\,1\,0\,1\,0\,1\,1\,0\,1\,0 \\ 1\,1\,0\,1\,1\,1\,1\,1\,1\,0\,1\,0\,1\,0\,1\,0 \\ 1\,1\,1\,0\,0\,1\,1\,1\,1\,1\,0\,1\,1\,1\,0\,1 \\ 1\,1\,1\,1\,0\,0\,1\,1\,1\,1\,1\,0\,1\,1\,1\,0 \\ 1\,0\,1\,0\,1\,1\,1\,0\,0\,1\,0\,1\,0\,1\,0\,1 \\ 1\,0\,1\,0\,1\,1\,0\,1\,1\,1\,1\,1\,1\,0\,1\,0 \\ 1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,1\,1\,1\,1\,1\,0\,1 \\ 1\,1\,1\,0\,1\,1\,1\,1\,1\,0\,0\,1\,1\,1\,1\,1\,0 \\ 0\,1\,0\,1\,1\,0\,1\,0\,1\,1\,1\,0\,0\,1\,0\,1 \end{pmatrix} \begin{pmatrix} x_{1(8)} \\ x_{2(8)} \\ x_{3(8)} \\ x_{4(8)} \\ x_{5(8)} \\ x_{6(8)} \\ x_{7(8)} \\ x_{8(8)} \\ x_{9(8)} \\ x_{10(8)} \\ x_{11(8)} \\ x_{12(8)} \\ x_{13(8)} \\ x_{14(8)} \\ x_{15(8)} \\ x_{16(8)} \end{pmatrix} \ .$$

3.3 XS-function

XS-function is a composite function of the S-function, 128-bit key addition, and MDS_{L}-function.

$$XS(X_{(128)}, K_{(256)}) = S(MDS_{\mathrm{L}}(S(X_{(128)} \oplus K_{1(128)})) \oplus K_{2(128)}) \ .$$

3.4 S-function

The S-functions consists of sixteen operations of s-function for 8-bit subdata of the 128-bit input data.

$$Y_{(128)} = S(X_{(128)}) \ ,$$

$$y_{i(8)} = s(x_{i(8)}) \ , \quad i = 1, 2, \ldots, 16 \ .$$

3.5 MDS_{L}-function

The MDS_{L}-function consists of four operations of mds_{L}-function for 32-bit subdata of the 128-bit input data.

$$Y_{(128)} = MDS_{\mathrm{L}}(X_{(128)}) \ ,$$

$$y_{i(32)} = mds_{\mathrm{L}}(x_{i(32)}) \ , \quad i = 1, 2, 3, 4 \ .$$

3.6 s-function

The s-function is a nonlinear transformation for 8-bit input/output value, which is given as the following table where all numbers are represented in hexadecimal.

$s(256) = ($

$(s(0)\ s(1)\ s(2)\ \ldots\ s(\mathrm{F})\ s(10)\ s(11)\ \ldots\ s(\mathrm{FF})) =$

(	07	FC	55	70	98	8E	84	4E	BC	75	CE	18	02	E9	5D	80
	1C	60	78	42	9D	2E	F5	E8	C6	7A	2F	A4	B2	5F	19	87
	0B	9B	9C	D3	C3	77	3D	6F	B9	2D	4D	F7	8C	A7	AC	17
	3C	5A	41	C9	29	ED	DE	27	69	30	72	A8	95	3E	F9	D8
	21	8B	44	D7	11	0D	48	FD	6A	01	57	E5	BD	85	EC	1E
	37	9F	B5	9A	7C	09	F1	B1	94	81	82	08	FB	C0	51	0F
	61	7F	1A	56	96	13	C1	67	99	03	5E	B6	CA	FA	9E	DF
	D6	83	CC	A2	12	23	B7	65	D0	39	7D	3B	D5	B0	AF	1F
	06	C8	34	C5	1B	79	4B	66	BF	88	4A	C4	EF	58	3F	0A
	2C	73	D1	F8	6B	E6	20	B8	22	43	B3	33	E7	F0	71	7E
	52	89	47	63	0E	6D	E3	BE	59	64	EE	F6	38	5C	F4	5B
	49	D4	E0	F3	BB	54	26	2B	00	86	90	FF	FE	A6	7B	05
	AD	68	A1	10	EB	C7	E2	F2	46	8A	6C	14	6E	CF	35	45
	50	D2	92	74	93	E1	DA	AE	A9	53	E4	40	CD	BA	97	A3
	91	31	25	76	36	32	28	3A	24	4C	DB	D9	8D	DC	62	2A
	EA	15	DD	C2	A5	0C	04	1D	8F	CB	B4	4F	16	AB	AA	A0) .

3.7 mds_{L}-function

The mds_{L}-function is a linear transformation which is represented by 4×4 matrix multiplication where all matrix and vector elements are regarded as elements of $\mathrm{GF}(2^8)$.

$$Y_{i(32)} = mds_{\mathrm{L}}(X_{i(32)}) \ ,$$

$$\begin{pmatrix} y_{4i-4+1(8)} \\ y_{4i-4+2(8)} \\ y_{4i-4+3(8)} \\ y_{4i-4+4(8)} \end{pmatrix} = \begin{pmatrix} \mathrm{C4} & 65 & \mathrm{C8} & \mathrm{8B} \\ \mathrm{8B} & \mathrm{C4} & 65 & \mathrm{C8} \\ \mathrm{C8} & \mathrm{8B} & \mathrm{C4} & 65 \\ 65 & \mathrm{C8} & \mathrm{8B} & \mathrm{C4} \end{pmatrix} \begin{pmatrix} x_{4i-4+1(8)} \\ x_{4i-4+2(8)} \\ x_{4i-4+3(8)} \\ x_{4i-4+4(8)} \end{pmatrix} \ .$$

Here, 8-bit data $x_{(8)}$ and the matrix element a (in hexadecimal) are regarded as elements of $\mathrm{GF}(2^8)$ related as follows.

$$x_{(8)} \Leftrightarrow \sum_{i=1}^{8} x_{i(1)} z^{8-i} \ ,$$

$$a = \sum_{i=0}^{7} a_i 2^i \Leftrightarrow \sum_{i=0}^{7} a_i z^i \ .$$

The following polynomial $p(x)$ is used as the primitive polynomial for the Galois field $\mathrm{GF}(2^8)$.

$$p(z) = z^8 + z^6 + z^5 + z + 1 \ .$$

3.8 $P^{(n)}$-function

The $P^{(n)}$ function consists of the a linear transformation for the input $X_{(4n)}$ which is a concatenation of four n-bit values $x_{i(n)}$ $(i = 1, 2, 3, 4)$ where each element is regarded as an element of $\mathrm{GF}(2)^n$.

$$Y_{(4n)} = P^{(n)}(X_{(4n)}) \ ,$$

$$X_{(4n)} = x_{1(n)} \| x_{2(n)} \| x_{3(n)} \| x_{4(n)} \ ,$$

$$Y_{(4n)} = y_{1(n)} \| y_{2(n)} \| y_{3(n)} \| y_{4(n)} \ ,$$

$$\begin{pmatrix} y_{1(n)} \\ y_{2(n)} \\ y_{3(n)} \\ y_{4(n)} \end{pmatrix} = \begin{pmatrix} 1\ 0\ 1\ 0 \\ 0\ 1\ 0\ 1 \\ 0\ 1\ 1\ 1 \\ 1\ 0\ 1\ 1 \end{pmatrix} \begin{pmatrix} x_{1(n)} \\ x_{2(n)} \\ x_{3(n)} \\ x_{4(n)} \end{pmatrix} .$$

The inverse function of $P^{(n)}$, ${P^{(n)}}^{-1}$, is given by the following equation.

$$X_{(4n)} = {P^{(n)}}^{-1}(Y_{(4n)}) ,$$

$$\begin{pmatrix} x_{1(n)} \\ x_{2(n)} \\ x_{3(n)} \\ x_{4(n)} \end{pmatrix} = \begin{pmatrix} 1\ 1\ 1\ 0 \\ 1\ 1\ 0\ 1 \\ 0\ 1\ 1\ 0 \\ 1\ 0\ 0\ 1 \end{pmatrix} \begin{pmatrix} y_{1(n)} \\ y_{2(n)} \\ y_{3(n)} \\ y_{4(n)} \end{pmatrix} .$$

3.9 M_{5E}-function

The M_{5E}-function consists of a concatenation of two 32-bit linear transformations, where each 8-bit subdata is regarded as an element of $\mathrm{GF}(2)^8$.

$$Y_{(64)} = M_{5E}(X_{(64)}) ,$$

$$X_{(64)} = x_{1(8)} \| x_{2(8)} \| \cdots \| x_{8(8)} ,$$

$$Y_{(64)} = y_{1(8)} \| y_{2(8)} \| \cdots \| y_{8(8)} ,$$

$$\begin{pmatrix} y_{1(8)} \\ y_{2(8)} \\ y_{3(8)} \\ y_{4(8)} \end{pmatrix} = \begin{pmatrix} 1\ 0\ 1\ 0 \\ 1\ 1\ 0\ 1 \\ 1\ 1\ 1\ 0 \\ 0\ 1\ 0\ 1 \end{pmatrix} \begin{pmatrix} x_{1(8)} \\ x_{2(8)} \\ x_{3(8)} \\ x_{4(8)} \end{pmatrix} ,$$

$$\begin{pmatrix} y_{5(8)} \\ y_{6(8)} \\ y_{7(8)} \\ y_{8(8)} \end{pmatrix} = \begin{pmatrix} 1\ 1\ 1\ 1 \\ 0\ 1\ 1\ 1 \\ 0\ 0\ 1\ 1 \\ 1\ 1\ 1\ 0 \end{pmatrix} \begin{pmatrix} x_{5(8)} \\ x_{6(8)} \\ x_{7(8)} \\ x_{8(8)} \end{pmatrix} .$$

3.10 M_{B3}-function

The M_{BE}-function consists of a concatenation of two 32-bit linear transformations, where each 8-bit subdata is regarded as an element of $\mathrm{GF}(2)^8$.

$$Y_{(64)} = M_{B3}(X_{(64)}) ,$$

$$X_{(64)} = x_{1(8)} \| x_{2(8)} \| \cdots \| x_{8(8)} ,$$

$$Y_{(64)} = y_{1(8)} \| y_{2(8)} \| \cdots \| y_{8(8)} ,$$

$$\begin{pmatrix} y_{1(8)} \\ y_{2(8)} \\ y_{3(8)} \\ y_{4(8)} \end{pmatrix} = \begin{pmatrix} 0\ 1\ 0\ 1 \\ 1\ 0\ 1\ 0 \\ 1\ 1\ 0\ 1 \\ 1\ 0\ 1\ 1 \end{pmatrix} \begin{pmatrix} x_{1(8)} \\ x_{2(8)} \\ x_{3(8)} \\ x_{4(8)} \end{pmatrix} ,$$

$$\begin{pmatrix} y_{5(8)} \\ y_{6(8)} \\ y_{7(8)} \\ y_{8(8)} \end{pmatrix} = \begin{pmatrix} 1\ 1\ 0\ 0 \\ 0\ 1\ 1\ 0 \\ 1\ 0\ 1\ 1 \\ 1\ 0\ 0\ 1 \end{pmatrix} \begin{pmatrix} x_{5(8)} \\ x_{6(8)} \\ x_{7(8)} \\ x_{8(8)} \end{pmatrix} .$$

3.11 F_σ-function

The F_σ-function is a nonlinear function for 64-bit input/output value, which consists of the s-functions and the $P^{(16)}$-functions.

$$\begin{aligned} Y_{(64)} \quad & = F_\sigma(X_{(64)}) , \\ & u_{i(8)} = s(x_{i(8)}) , \quad 1 \le i \le 8 , \\ & Y_{(64)} = P^{(16)}(U_{(64)}) . \end{aligned}$$

4 Test data

• The number of rounds = 6 , key length = 128 bits

Key = (0x4703c87e 0x817842c4 0xce6b167d 0x43701b76)
Plaintext = (0x85693846 0xdb4c1b34 0x87272e55 0x5761c7f5)
Ciphertext = (0x5c5f4b00 0xaec36d89 0x3cf1041e 0x7fa8bae8)

• The number of rounds = 7 , key length = 192 bits

Key = (0x7742a038 0x89b58601 0xf74d5513
0x88872377 0x324fbc1d 0x30c54fc6)
Plaintext = (0x54406620 0x9d931b33 0x0c9089fd 0xb4cb8259)
Ciphertext = (0x1d310598 0x8dbbd50c 0xb0a17193 0xea5ba244)

• The number of rounds = 8 , key length = 256 bits

Key = (0x11a18026 0x9a78dda4 0x99474621 0x3b5a6dd6
0xe34ffe0c 0xc465d583 0xaff66e13 0x29419c94)
Plaintext = (0xc16d7efc 0xa1cbafc7 0x625cbe9c 0x2593de2d)
Ciphertext = (0xc86cd3b4 0xa3185232 0xe3457d63 0x8c6515c9)