FUTURE AEAD

Gaurav Bansal[2], Manas Ghai[2], Kishan Chand Gupta[1], Rajat Khanna[2], Sumit Kumar Pandey[2] and Susanta Samanta[1]

[1] Indian Statistical Institute, Kolkata- 700108, INDIA.
[2] Indian Institute of Technology Jammu, Jagti, PO Nagrota, Jammu-181221, INDIA.
gauravb834@gmail.com, manasg46@gmail.com, kishan@isical.ac.in,
rajatkhanna1999@gmail.com, emailpandey@gmail.com,
susanta.math94@gmail.com

1 Introduction

Authenticated encryption (AE) is a symmetric-key cryptographic primitive for providing both confidentiality and authenticity. Due to the recent rise in communication networks operated on small devices, the era of the so-called Internet of Things, AE is expected to play a key role in securing these networks.

In this document, we propose a new nonce-based authenticated encryption with associated data (NAEAD) which instantiates the OFB block cipher based AEAD mode with our designed block cipher FUTURE.

2 Specification

In this chapter, we present the specification of the FUTURE AEAD along with its underlying block cipher FUTURE.

The AEAD receives an encryption key $K \in \{0,1\}^{128}$, a nonce $N \in \{0,1\}^{64}$, an associated data $A \in \{0,1\}^*$, and a message $M \in \{0,1\}^*$as inputs, and returns a ciphertext $C \in \{0,1\}^{|M|+128-(|M| \ mod \ 64)}$and a tag $t \in \{0,1\}^{64}$.

Key and Block cipher. The underlying cryptographic primitive is an 64-bit block cipher FUTURE, E, whose specification is discussed in Section 3. The key of the scheme is the key of the block cipher FUTURE.

Padding Function. For $M \in \{0,1\}^*$, we define padding function $Pad: \ \{0,1\}^{|M|} \rightarrow \{0,1\}^{|M|+128-(|M| \ mod \ 64)}$ as follows:

$$Pad(M) = \Big\{ M||0^{64-(|M| \ mod \ 64))}||0^{57}||z_{bin} \quad ,$$

where z_{bin} is the 7 bit binary number which represents the count of zeros appended.

Complete specification of the AEAD is presented below and the corresponding pictorial description can be found in Figure 1 and Figure 2.

1. Ciphertext and Tag Generation $(Nonce, AD, M = m_0||m_1||\cdots||m_r)$
   - $s_0 = E(K, Nonce)$.
   - $s_i = E(K \oplus_2 c_{i-1}, s_{i-1})$ for $1 \leq i \leq r$.
   - $c_i = m_i \oplus s_i$.
   - Let $Nonce||c_{r-1}||m_{r-1}||AD = x_0||x_1||\cdots||x_l$.
   - $t_0 = x_0 \oplus E(K, c_r)$.
   - $t_i = x_i \oplus E(K, t_{i-1})$ for $1 \leq i \leq l$.
   - $t = E(k, t_l)$.
   - Output: $(Nonce, AD, C = (c_0||c_1||\cdots||c_r), t)$.
2. Decryption and Tag Verification $(Nonce, AD, C = (c_0||c_1||\cdots||c_r), t))$
   - $s'_0 = E(K, Nonce)$.
   - $s'_i = E(K \oplus_2 c_{i-1}, s_{i-1})$ for $1 \leq i \leq r$.

- $m'_i = c_i \oplus s'_i$.
- Let $Nonce||c_{r-1}||m'_{r-1}||AD = x'_0||x'_1||\cdots||x'_l$. (See padding function).
- $t'_0 = x'_0 \oplus E(K, c_r)$.
- $t'_i = x'_i \oplus E(K, t'_{i-1})$ for $1 \leq i \leq l$.
- $t' = E(K, t'_l)$.
- If $t' \stackrel{?}{=} t$, output $m_0||m_1||\cdots||m_r$, else $\perp$.

Specifications:

- $|K| = 128$ bits.
- $|m_i| = |s_i| = |c_i| = |t_i| = 64$ bits.
- $K \oplus_2 c_i$ means xoring c_i at even bit positions of K, i.e. $k_{2i} \oplus c_i$ where $K = k_0||k_1||\cdots||k_{127}$ where $k_i \in \{0, 1\}$.

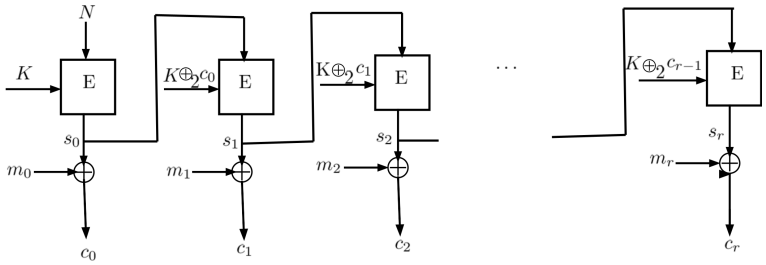

Fig. 1. Encryption of AEAD: $K \oplus_2 c_i$ means xoring c_i at even bit positions of K, i.e. $k_{2i} \oplus c_i$ where $K = k_0||k_1||\cdots||k_{127}$ where $k_i \in \{0, 1\}$.

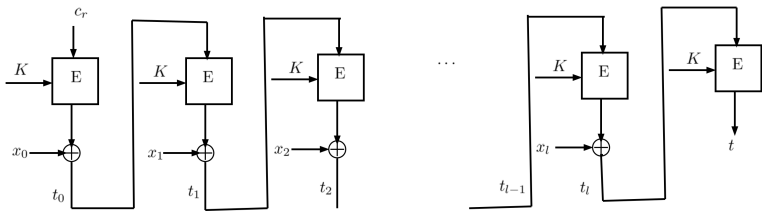

Fig. 2. Tag Generation

3 Specifications of FUTURE

FUTURE is a new SPN based block cipher and consists of 10 rounds in a fully unrolled fashion. It accept 128-bit keys and have a block size of 64-bit.

3.1 Round Function.

One encryption round of FUTURE is composed of four operations in the following order: SubCells, MixColumns, ShiftRows and AddRoundKey (see illustration in Figure 3). The cipher receives an 64-bit plaintext $P = b_0b_1b_2 \dots b_{62}b_{63}$ as the cipher state S, where b_0 being the most significant bit. The cipher state can also be expressed as 16 many 4-bit cells as follows:

$$S = \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{bmatrix},$$

i.e. $s_i \in \{0,1\}^4$. The i-th round output state is defined as S_i , namely $S_0 = P$.

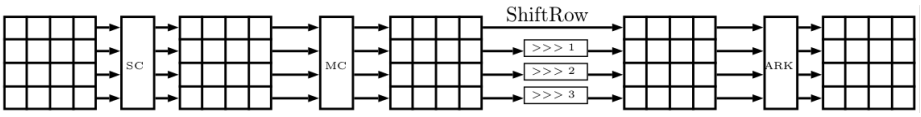

Fig. 3. The round function applies four different transformations: SubCells (SC), MixColumns (MC), ShiftRows (SR) and AddRoundKey (ARK).

SubCells. A 4-bit Sbox S is applied to every cell of the cipher internal state.

$$s_i \leftarrow S(s_i) \quad \text{for } i = 0, 1, \dots, 15.$$

The Sbox S is a composition of two low hardware cost Sboxes S_1 and S_2 i.e. $S(s_i) = S_1(S_2(s_i))$ for $i = 0, 1, \dots, 15$. The Sboxes in hexadecimal notation is given by the following table.

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S_1(x)$	1	3	0	2	7	5	4	6	9	a	8	b	f	e	c	d
$S_2(x)$	0	1	2	3	4	d	6	f	8	9	e	7	c	5	a	b
$S(x)$	1	3	0	2	7	e	4	d	9	a	c	6	f	5	8	b

MixColumns. FUTURE uses an MDS matrix M for the MixColumns operation. M is applied to every 16-bit column of the state S,

$$(s_i, s_{i+1}, s_{i+2}, s_{i+3}) \leftarrow M \cdot (s_i, s_{i+1}, s_{i+2}, s_{i+3})^t$$

for $i = 0, 4, 8, 12$.

The MDS matrix M is constructed by composition of 4 sparse matrices M_1, M_2, M_3 and M_4 of order 4 i.e., $M = M_1M_2M_3M_4$, where

$$M_1 = \begin{bmatrix} 0\,0\,1\,1 \\ 1\,0\,0\,0 \\ 1\,1\,0\,0 \\ 0\,0\,1\,0 \end{bmatrix}, M_2 = \begin{bmatrix} 0 & 0\,1\,\alpha \\ 1 & 0\,0\,0 \\ \alpha^3+1 & 1\,0\,0 \\ 0 & 0\,1\,0 \end{bmatrix}, M_3 = \begin{bmatrix} 0 & 0\,1\,1 \\ 1 & 0\,0\,0 \\ \alpha^3+1 & 1\,0\,0 \\ 0 & 0\,1\,0 \end{bmatrix} \text{ and } M_4 = M_1 \tag{1}$$

The multiplications between matrices and vectors are performed over $\mathbb{F}_{2^4}$ defined by the primitive polynomial $x^4 + x + 1$ and α is the primitive element of the field.

ShiftRows. Row i of the array state is rotated i cell positions to the right, for $i = 0, 1, 2, 3$, i.e.,

$$\begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{bmatrix} \leftarrow \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_{13} & s_1 & s_5 & s_9 \\ s_{10} & s_{14} & s_2 & s_6 \\ s_7 & s_{11} & s_{15} & s_3 \end{bmatrix}.$$

Note that in the ShiftRows operation of AES [4] and LED [5] the row i of the array state is rotated i cell positions to the left, for $i = 0, 1, 2, 3$.

AddRoundKey. Given round key RK_i for $1 \leq i \leq 10$, the i-th 64-bit round key RK_i is XORed to the state S.

3.2 Data Processing

The data processing part of FUTURE for encryption consisting of 10 rounds, F, takes a 64-bit data $X \in \{0,1\}^{64}$, whitening keys $WK \in \{0,1\}^{64}$ and 10 round keys $RK_i \in \{0,1\}^{64}$ $(1 \leq i \leq 10)$ as the inputs, and outputs a 64-bit data $Y \in \{0,1\}^{64}$. F is defined as follows:

$$F = \begin{cases} \{0,1\}^{64} \times \{0,1\}^{64} \times \left\{\{0,1\}^{64}\right\}^{10} \rightarrow \{0,1\}^{64} \\ (X, WK, RK_1, RK_2, \ldots, RK_{10}) \rightarrow Y \end{cases}$$

Input: X and $WK, RK_1, RK_2, \ldots, RK_{10}$
Initialization: $S \leftarrow$ **KeyAdd**(X, WK) ;
for $i \leftarrow 1$ **to** 9 **do**
  $S \leftarrow$ **SubCell**(S);
  $S \leftarrow$ **MixColumn**(S);
  $S \leftarrow$ **ShiftRows**(S);
  $S \leftarrow$ **AddRoundKey**(S, RK_i);
end
$S \leftarrow$ **SubCell**(S);
$S \leftarrow$ **ShiftRows**(S);
$S \leftarrow$ **AddRoundKey**$(S, RK10)$;
Output: Y

Algorithm 1: Encryption Function of FUTURE

3.3 Key schedule and round constants.

FUTURE uses a 128-bit secret key $K = k_0k_1 \dots k_{127}$. It splits K in two equal parts K_0 and K_1 for the round key and whitening key generation i.e. $K = K_0||K_1$, where $K_0 = k_0k_1 \dots k_{63}$ and $K_1 = k_{64}k_{65} \dots k_{127}$ are two 64-bit key. It uses K_0 as whitening key and the round key RK_i $(1 \leq i \leq 10)$ generation is as follows (see Figure 4):

$$RK_i = \begin{cases} K_0 \leftarrow K_0 <<< 5\,\lfloor i/2 \rfloor & \text{if } 2 \mid i \\ K_1 \leftarrow K_1 <<< 5\,\lfloor i/2 \rfloor & \text{if } 2 \nmid i \end{cases}$$

where $<<< j$ is an j bits left rotation within a 64-bit word.

For FUTURE a single bit "1" is XORed into the $4i+1$-th $(i = 0, 1, 2, \dots 15)$ bit position into the cipher state in the 2nd and 6th. Whereas "1" is XORed into the $(4i+2)$-th $(i = 0, 1, 2, \dots 15)$ bit position into the cipher state in the 3rd and 7th. In round 4-th and 8-th, "1" is XORed into the $(4i+3)$-th $(i = 0, 1, 2, \dots 15)$ bit position. I.e. we are adding a NOT gate in the respective position in each round except round 1st, 5th and 9th.

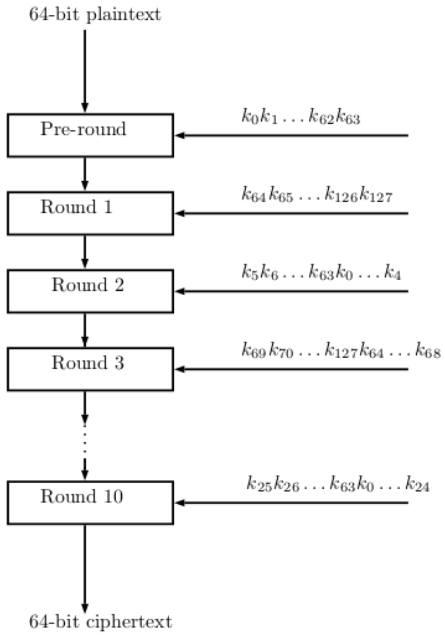

Fig. 4. Round Key Generation

In the following section we justify the decisions we took during the design of FUTURE.

4 Design Decision

SPN based block cipher is preferable over a Feistel-cipher, since a Feistel-cipher operates only on half the state resulting often in a higher number of rounds.

Also in an unrolled implementation offer the best performance due to the computation of single encryption within one clock cycle. The entire encryption or decryption function is implemented as a combinatorial circuit at the disadvantage of increasing the critical path. However, in this implementation, there is no need for registers to store the intermediate states resulting in a low implementation hardware cost with a very small time of delay for full encryption to the low round block ciphers.

4.1 SubCell

The cost of the Sbox, i.e., its area and critical path, is a major part of the overall cost. Thus, choosing an Sbox which minimizes those costs is crucial for obtaining standard S-box criteria. FUTURE Sbox satisfies the following condition:

1. Nonlinearity of the Sbox S is 4.
2. The maximal probability of a differential is 1/4.
3. There are exactly 24 differentials with probability 1/4.
4. The maximal absolute bias of a linear approximation is 1/4.
5. There are exactly 36 linear approximations with absolute bias 1/4.
6. There is no fixed point.

FUTURE Sbox S is a composition of two Sbox S_1 and S_2. During the search of an Sbox for FUTURE with this composition method, we only concentrate on the nonlinearity of the resulting Sbox. The nonlinearity of the Sboxes S_1 and S_2 are zero, whereas the resulting Sbox S has 4, which is the maximum value for a balanced 4-bit Sbox. The main concern for choosing such composition method was to reduce implementation cost for the Sbox S. The hardware cost for S_1 and S_2 are 6 GE, resulting a total 12 GE cost for the Sbox S.

4.2 MixColumn

Almost MDS matrix (or binary matrix with low branch number) has efficient implementation properties, whereas its diffusion speed is slower and the minimum number of active S-boxes in each round is smaller than those of ciphers uses MDS matrices in their MixColumn. The diffusion speed is measured by the number of rounds taken to achieve full diffusion i.e. all output cells are affected by all input cells.

Most of the lightweight block cipher in the literature uses almost MDS matrix or binary matrix with low branch number for hardware efficiency resulting more rounds for achieving the security against several attacks including impossible differential, saturation, differential and linear attacks. Whereas FUTURE needs only 10 rounds for this.

MDS matrices are not sparse. But they can be constructed from sparse matrices by recursive method i.e. using a sparse matrix several times (in general the order of the matrix) resulting a very low hardware cost.

Whereas, the MDS matrix in FUTURE is a composition of 4 different lightweight sparse matrices M_1, M_2, M_3 and M_4 (see 1). The idea of choosing MDS matrix in such a fashion was first introduced in [7]. But we have used a different form of sparse matrix structure to construct an MDS matrix in this composition methods. The implementation cost for the MDS matrix M is minimized due to the low implementation cost of M_1, M_2, M_3 and M_4. Note that to construct MDS matrix in this method, the implementation cost is calculated by the sum of the implementation cost of M_1, M_2, M_3 and M_4. Therfore FUTURE requires 35 XOR count for the MDS matrix.

4.3 Round Key

Our main goal for the key schedule is to minimize the hardware cost. Thus the key schedule function in FUTURE is a bit permutation of the master key, and so this module is constructed by a simple wire shuffle and takes no area at all.

5 Security Analysis of Block cipher FUTURE

In this section, we provide an analysis of the security of FUTURE against various cryptanalysis.

5.1 Differential and Linear Cryptanalysis

Analyzing the resistance of a cipher against differential and linear cryptanalysis of a block cipher is perhaps the most common and fundamental security analysis. In order to argue for the resistance against differential and linear attacks, we computed lower bounds on the minimum number of active Sboxes involved in a differential or linear characteristic. In this work, we use the Mixed Integer Linear Programming (MILP) to compute the lower bounds on the minimum number of active Sboxes in both Differential and linear cryptanalysis for various numbers of rounds, the results are summaries in Table 1. The MILP solution provides us the actual differential or linear characteristics, which allows us to compute the actual differential probability and correlation contribution from the DDT (Table 3) and LAT (Table 4) of FUTURE Sbox.

Rounds (N)	1	2	3	4	5
Differential (Linear) cryptanalysis	1	5	9	25	26

Table 1. The minimum number of active Sbox for N rounds of FUTURE

Differential cryptanalysis. Generally, for an adversary to mount differential cryptanalysis on an n-bit block cipher, there must be some differential

propagation with differential probability larger than 2^{1-n}. To have a better estimation about the probability of differentials, we found different single characteristics which follow the same Sbox activity pattern with the minimum number of active Sboxes. Then by summing all the probabilities of each single characteristics, we found a lower bound for the probability of corresponding differential.

For 4-round FUTURE, which has at least 25 active Sboxes, we have found an optimal differentials, having a probability of $2^{-62.405}$ and we expect that the differential probability will be lower than 2^{-63} when we have 5 round. Therefore, we believe that full round FUTURE is enough to resist against Differential cryptanalysis.

Linear cryptanalysis. Given a linear characteristic with a bias ϵ, the square of the correlation contribution (so-called correlation potential) is defined as $4\epsilon^2$. For an adversary to mount linear cryptanalysis on an n-bit block cipher, the correlation potential must be larger than 2^{-n}.

Similar to differential, we first find an optimal linear characteristic with the minimum number of active Sboxes, then find the linear characteristics which follow the same Sbox activity pattern and take the summation of the correlation potentials. For 4 round FUTURE, we have found an optimal linear hull having an average square correlation of $2^{-75.677}$. Therefore, we believe 10-round FUTURE is enough to resist against linear cryptanalysis.

5.2 Impossible Differential Attacks

A pair of differences $(\Delta x, \Delta y)$ is said to be an impossible differential over an encryption function F if, for all plaintexts x, $F(x) + F(x + \Delta x) \neq \Delta y$. Such a distinguisher over a reduced-round version of the cipher might be used for a key-recovery attack over a larger number of rounds by filtering all the key candidates which lead to the intermediate state values with differences Δx and Δy, i.e., the intermediate state values fulfilling the impossible differential. With the Mixed-Integer Linear Programming approach (see [3,8]), we searched for impossible differentials over reduced-round versions of FUTURE. Thereby, we exhaustively tested input and output differences satisfying the following conditions.

1. The input difference activates only one of the first four Sboxes.
2. The output difference activates only one Sbox.

For the first condition, there are $4 \times 15 = 60$ such input differences. For the second condition, there are $16 \times 15 = 240$ such output differences. Hence, we tested $60 \times 240 = 14,400$ pairs of input and output differences.

The search results show that for 4-round FUTURE, there are only 267 choices out of 14,400 choices are impossible. We then extend this search procedure to 5 rounds, and obtained that there does not exist any impossible differential from the $14,400$ pairs for 5 rounds. Thus we believe that the full round FUTURE are quite sufficient to resist the impossible differential attack.

5.3 Integral Attack

We first search for integral distinguishers by using the (bit-based) division property [9] using the Mixed-Integer Linear Programming approach described in [10]. We first evaluate the propagation of the division property for the Sbox. The algebraic normal form of FUTURE Sbox is given by

$$\begin{aligned} y_3 &= x_0x_1x_3 + x_0x_2 + x_3 \\ y_2 &= x_1x_3 + x_2 \\ y_1 &= x_0x_2x_3 + x_0x_2 + x_0 + x_1x_2 + x_2 \\ y_0 &= x_0x_1x_3 + x_0x_2 + x_0x_3 + x_1 + 1. \end{aligned}$$

and the propagation of the division property is summarized as Table 2.

	v															
u	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
1		×	×	×		×	×	×	×	×	×	×	×	×	×	×
2		×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
3		×		×		×	×	×	×	×	×	×	×	×	×	×
4		×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
5		×	×	×		×	×	×	×	×	×	×	×	×	×	×
6			×	×		×	×	×		×	×	×		×	×	×
7								×		×	×	×		×	×	×
8		×	×	×	×	×	×	×	×	×	×	×	×	×	×	×
9		×	×	×		×	×	×	×	×	×	×	×	×	×	×
a		×		×	×	×	×	×	×	×	×	×	×	×	×	×
b		×		×		×	×	×	×	×	×	×	×	×	×	×
c			×	×		×	×	×			×	×	×	×	×	×
d			×	×		×	×	×			×	×		×	×	×
e											×	×		×	×	×
f																×

Table 2. The possible propagation of the division property for FUTURE Sbox

Here, let u and v be the input and output division property, respectively. The propagation from u to v labeled $\times$ is possible. Otherwise, the propagation is impossible.

Taking into account the effect of MixColumn, we evaluated the propagation of the division property on reduced-round FUTURE. To search for the longest integral distinguisher, we choose only one bit in plaintext as constant, and the others are active. With that, we did not find distinguishers for more than 6 rounds. So we are expecting that full round FUTURE is secure against integral attack.

	ΔO															
ΔI	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	4	0	0	0	0	0	4	4	0	0	0	0	0
2	0	4	0	4	0	2	0	2	0	0	0	0	2	0	2	0
3	0	0	0	4	2	0	2	0	0	0	4	0	0	2	0	2
4	0	0	0	0	4	0	4	0	0	0	0	0	0	4	0	4
5	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
6	0	4	0	4	0	2	0	2	0	0	0	0	2	0	2	0
7	0	0	4	0	0	2	0	2	0	4	0	0	2	0	2	0
8	0	0	0	0	2	0	2	0	4	2	0	2	4	0	0	0
9	0	2	2	0	0	2	2	0	0	0	2	2	0	0	2	2
a	0	0	0	0	0	4	0	0	4	2	0	2	0	2	0	2
b	0	2	2	0	0	0	2	2	0	0	2	2	2	0	0	2
c	0	0	0	0	2	0	2	0	4	2	0	2	0	0	4	0
d	0	2	2	0	2	0	0	2	0	0	2	2	2	2	0	0
e	0	0	0	0	0	0	0	4	4	2	0	2	0	2	0	2
f	0	2	2	0	2	2	0	0	0	0	2	2	0	2	2	0

Table 3. Differential Distribution Table (DDT) of FUTURE Sbox

	β															
α	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	4	0	0	4	-4	0	0	0	0	0	0	0	0
2	0	-4	-2	-2	0	0	2	-2	0	-4	2	2	0	0	-2	2
3	0	-4	-2	-2	0	0	2	-2	0	4	-2	-2	0	0	2	-2
4	0	2	0	-2	4	-2	0	-2	2	0	-2	0	2	0	2	4
5	0	-2	4	-2	0	-2	0	2	-2	0	2	0	2	4	2	0
6	0	-2	2	0	4	-2	-2	0	-2	0	0	2	-2	-4	0	-2
7	0	2	2	-4	0	-2	2	0	2	0	0	-2	-2	0	-4	-2
8	0	0	0	0	0	0	0	0	4	0	0	4	4	0	0	-4
9	0	0	0	0	0	0	0	0	4	0	4	0	-4	0	4	0
a	0	0	-2	2	0	-4	2	2	0	-4	-2	-2	0	0	2	-2
b	0	0	2	-2	0	4	-2	-2	0	-4	-2	-2	0	0	2	-2
c	0	-2	0	2	4	2	0	2	2	0	2	-4	2	0	-2	0
d	0	2	0	-2	0	2	4	2	-2	0	2	0	2	-4	2	0
e	0	-2	2	0	-4	-2	-2	0	2	0	0	-2	2	-4	0	2
f	0	2	-2	0	0	-2	-2	-4	-2	0	4	-2	2	0	0	-2

Table 4. Linear approximation table (LAT) of FUTURE Sbox. Each entry represents $\#\{x \in \mathbb{F}_{2^4} : x \cdot \alpha \oplus S(x) \cdot \beta = 0\} - 8$.Linear approximation table (LAT) of FUTURE Sbox.

References

1. E. Biham, A. Shamir Differential cryptanalysis of DES-like cryptosystems. In A.J. Menezes, S.A. Vanstone, eds., CRYPTO'90. Volume 537 of LNCS., Springer, Heidelberg (August 1991) 2–21
2. J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knežević, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C Rechberger, P Rombouts, S. S. Thomsen, and T. Yalçın, PRINCE- A low-latency block cipher for pervasive computing applications - extended abstract. In Wang, X., Sako, K., eds.: ASIACRYPT 2012. Volume 7658 of LNCS., Springer, Heidelberg (December 2012) 208–225
3. T. Cui, K. Jia, K. Fu, S. Chen, and M. Wang. New Automatic Search Tool for Impossible Differentials and Zero-Correlation Linear Approximations.. Cryptology ePrint Archive, Report 2016/689 (2016) .
4. J. Daemen and V. Rijmen, The Design of Rijndael: AES - The Advanced Encryption Standard, Springer-Verlag, 2002.
5. J. Guo, T. Peyrin, A. Poshmann and M. J. B. Robshaw, The LED block cipher, In *CHES 2011*, LNCS, vol. 6917, p. 326–341, Springer, 2011.
6. M. Matsui, Linear cryptoanalysis method for DES cipher. In T. Helleseth, ed., EUROCRYPT'93. Volume 765 of LNCS., Springer, Heidelberg (May 1994) 386–397
7. M. Sajadieh and M. Mousavi. Construction of MDS Matrices from Generalized Feistel Structures, Available at `https://eprint.iacr.org/2018/1072.pdf`.
8. Y. Sasaki, Y. Todo, New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In J. Coron, J.B. Nielsen, eds., EUROCRYPT 2017, Part III. Volume 10212 of LNCS. (2017) 185–215
9. Y. Todo, M. Morii, Bit-based division property and application to Simon family. In T. Peyrin ed., FSE 2016. Volume 9783 of LNCS., Springer (2016) 357–377.
10. Z. Xiang, W. Zhang, Z. Bao, and D. Lin, Applying MILP Method to Searching Integral Distinguishers Based on Division Property for 6 Lightweight Block Ciphers, In J.H. Cheon, T. Takagi, eds., ASIACRYPT 2016 Part I. Volume 10031 of LNCS(2016) 648–678.