Biclique Cryptanalysis of Lightweight Block Ciphers PRESENT, Piccolo and LED

Kitae Jeong[1], HyungChul Kang[1], Changhoon Lee[2], Jaechul Sung[3], and Seokhie Hong[1]

[1] Center for Information Security Technologies(CIST), Korea University, Seoul, Korea
kite.jeong@gmail.com, kanghc@korea.ac.kr, shhong@korea.ac.kr
[2] Department of Computer Science and Engineering, Seoul National University of Science and Technology, Seoul, Korea
chlee@seoultech.ac.kr
[3] Department of Mathematics, University of Seoul, Seoul, Korea
jcsung@uos.ac.kr

Abstract. In this paper, we evaluate the security of lightweight block ciphers PRESENT, Piccolo and LED against biclique cryptanalysis. To recover the secret key of PRESENT-80/128, our attacks require $2^{79.76}$ full PRESENT-80 encryptions and $2^{127.91}$ full PRESENT-128 encryptions, respectively. Our attacks on Piccolo-80/128 require computational complexities of $2^{79.13}$ and $2^{127.35}$, respectively. The attack on a 29-round reduced LED-64 needs $2^{63.58}$ 29-round reduced LED-64 encryptions. In the cases of LED-80/96/128, we propose the attacks on two versions. First, to recover the secret key of 45-round reduced LED-80/96/128, our attacks require computational complexities of $2^{79.45}, 2^{95.45}$ and $2^{127.45}$, respectively. To attack the full version, we require computational complexities of $2^{79.37}, 2^{95.37}$ and $2^{127.37}$, respectively. However, in these cases, we need the full codebook. These results are superior to known biclique cryptanalytic results on them.

Keywords: Block cipher, PRESENT, Piccolo, LED, Biclique, Cryptanalysis.

1 Introduction

Biclique cryptanalysis was first proposed at Asiacrypt 2011 [2]. To recover the secret keys of the full AES-128/192/256, the authors applied this technique to them in [2]. This is a kind of meet-in-the-middle attack such that bicliques improve an efficiency. After the proposal of biclique cryptanalysis, it brings new cryptanalytic techniques on block ciphers, which were known mainly in cryptanalysis of hash functions. The most attractive point is that this approach does not use related keys. Because of this property, many biclique cryptanalytic results on block ciphers were proposed [4, 5, 7, 8].

In this paper, we apply biclique cryptanalysis to the most popular lightweight block ciphers PRESENT [3], Piccolo [9] and LED [6]. In [2], two concepts of bicliques for AES were considered. One is the long-biclique and the other is the independent-biclique. We use the concept of independent-biclique. This is composed of constructing bicliques from independent related-key differentials and matching with precomputations. We find that a slow and limited diffusion of the keyschedule and encryption process in the target algorithm leads to relatively long bicliques with high dimension and an efficient matching check with precomputations. As the results, our attacks can recover the secret key of target algorithms with computational complexities smaller than an exhaustive search.

Our results are summarized in Table 1. In [1], biclique cryptanalysis of PRESENT and LED-64/128 were proposed. In detail, the attack on a 28-round reduced PRESENT-80 requires 2^{60} chosen plaintexts and a computational complexity of $2^{79.54}$, and the attack on the full PRESENT-128 requires 2^{56} chosen plaintexts and a computational complexity of $2^{127.42}$. Compared with this result, our attack on PRESENT-128 needs the smaller data complexity. In the case of LED, the attack on a 31.5-round reduced LED-64 requires 2^{56} chosen plaintexts and a computational complexity of $2^{63.4}$, and the attack on the full LED-128 requires 2^{64} chosen plaintexts and a computational complexity of $2^{127.25}$. With respect to the number of attack rounds, the attack result on LED-64 proposed in [1] is better than ours. However, the authors considered LED-64 without the postwhitening key.

On the other hand, biclique cryptanalysis of Piccolo-80/128 were proposed in [10]. The attack on a 25-round reduced Piccolo-80 requires 2^{48} chosen plaintexts and a computational complexity of $2^{78.95}$. Note that this is the attack result on Piccolo-80 without the postwhitening key. In the case of a 28-round reduced Piccolo-128, this attack requires 2^{24} chosen plaintexts and a computational complexity of $2^{126.79}$. Compared to these results, our attack results are superior to them.

This paper is organized as follows. In Section 2, we describe the structures of PRESENT, Piccolo and LED. In Section 3, we introduce briefly biclique cryptanalysis. Biclique cryptanalysis on PRESENT, Piccolo and LED are proposed in Section 4, 5 and 6, respectively. Finally, we give our conclusion in Section 7.

2 Description of PRESENT, Piccolo and LED

In this section, we present the structures of PRESENT, Piccolo and LED briefly.

2.1 PRESENT

PRESENT is a 64-bit block cipher with 80/128-bit secret keys and 31 iterative rounds. According to the length of secret keys, we call this algorithm PRESENT-80/128, respectively. PRESENT has the SPN structure and it is composed of the round function and the postwhitening. Both versions of PRESENT have the similar structure except the key schedule. In detail, PRESENT-80 takes a 64-bit plaintext $P = (P_{15}, P_{14}, \cdots, P_0)$ and the 80-bit secret key $K = (k_{79}, k_{78}, \cdots, k_0)$ as input values and generates a 64-bit ciphertext $C = (C_{15}, C_{14}, \cdots, C_0)$. Similarly, PRESENT-128 takes a 64-bit plaintext P and the 128-bit secret key $K = (k_{127}, k_{126}, \cdots, k_0)$ as input values and generates a 64-bit ciphertext C.

Table 1. Summary of biclique cryptanalytic results on PRESENT, Piccolo and LED.

Target algorithm	Rounds	Data complexity	Computational complexity	Reference
PRESENT-80	28	2^{60}	$2^{79.54}$	[1]
	Full(31)	2^{23}	$2^{79.76}$	This paper
PRESENT-128	Full(31)	2^{56}	$2^{127.42}$	[1]
	Full(31)	2^{19}	$2^{127.81}$	This paper
Piccolo-80	25*	2^{48}	$2^{78.95}$	[10]
	Full(25)	2^{48}	$2^{79.13}$	This paper
Piccolo-128	28	2^{24}	$2^{126.79}$	[10]
	Full(31)	2^{24}	$2^{127.35}$	This paper
LED-64	31.5**	2^{56}	$2^{63.40}$	[1]
	29	2^{40}	$2^{63.58}$	This paper
LED-80	45	2^{32}	$2^{79.45}$	This paper
	Full(48)	2^{64}	$2^{79.37}$	This paper
LED-96	45	2^{32}	$2^{95.45}$	This paper
	Full(48)	2^{64}	$2^{95.37}$	This paper
LED-128	Full(48)	2^{64}	$2^{127.25}$	[1]
	45	2^{32}	$2^{127.45}$	This paper
	Full(48)	2^{64}	$2^{127.37}$	This paper

$*$: Attack result on Piccolo-80 without the postwhitening key.

$**$: Attack result on LED-64 without the postwhitening key.

As depicted in Fig. 1, there are three subfunctions involved in the round function. The first subfunction is addRoundKey. At the beginning of round r, a 64-bit input value X^r is XORed with a round key $RK^r = (RK^r_{15}, \cdots, RK^r_0)$ ($r = 0, \cdots, 30$). The second subfunction is sBoxLayer. Sixteen identical 4×4 S-boxes are used in parallel as a nonlinear substitution layer. In the third subfunction, pLayer, a bit permutation is performed to provide diffusion. See [3] for the detailed description of the round function.

The key schedule of PRESENT takes the 80/128-bit secret key and generates thirty two 64-bit round keys RK^r and RK^{31} ($r = 0, \cdots, 30$). Note that RK^{31} is used for post-whitening. To generate 32 round keys, the key schedule of PRESENT-80 conducts the following procedure. First, the 80-bit secret key $K = (k_{79}, \cdots, k_0)$ is loaded to a 80-bit register $SK = (sk_{79}, \cdots, sk_0)$: $sk_i = k_i$ ($i = 0, \cdots, 79$). Then, SK is updated as follows.

1. $(sk_{79}, \cdots, sk_0) = (sk_{18}, \cdots, sk_0, sk_{79}, \cdots, sk_{19})$.
2. $(sk_{79}, sk_{78}, sk_{77}, sk_{76}) = Sbox[(sk_{79}, \cdots, sk_{76})]$.
3. $(sk_{19}, sk_{18}, sk_{17}, sk_{16}, sk_{15}) = (sk_{19}, \cdots, sk_{15}) \oplus (r + 1)$.

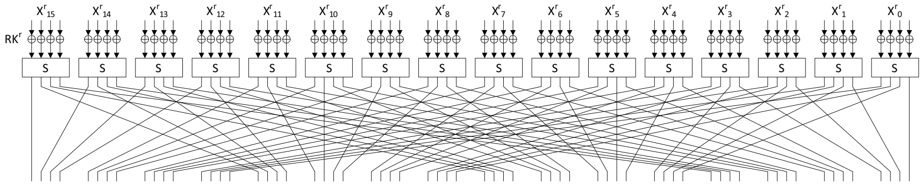

Fig. 1. Round function of PRESENT.

Applying the above procedure repeatedly, a 64-bit round key RK^r consists of the 64 leftmost bits of SK. That is, at round r, RK^r is computed as follows.

$$RK^r = (sk_{79}, sk_{78}, \cdots, sk_{16}).$$

The key schedule of PRESENT-128 is similar to that of PRESENT-80. After loading the 128-bit secret key $K = (k_{127}, \cdots, k_0)$, a 128-bit register $SK = (sk_{127}, sk_{126}, \cdots, sk_0)$ is updated as follows.

1. $(sk_{127}, \cdots, sk_0) = (sk_{66}, \cdots, sk_0, sk_{127}, \cdots, sk_{67})$.
2. $(sk_{127}, \cdots, sk_{124}) = Sbox[(sk_{127}, \cdots, sk_{124})]$.
3. $(sk_{123}, \cdots, sk_{120}) = Sbox[(sk_{123}, \cdots, sk_{120})]$.
4. $(sk_{66}, sk_{65}, sk_{64}, sk_{63}, sk_{62}) = (sk_{66}, \cdots, sk_{62}) \oplus (r + 1)$.

Applying the above procedure repeatedly, RK^r is computed as follows.

$$RK^r = (sk_{127}, sk_{126}, \cdots, sk_{64}).$$

2.2 Piccolo

Piccolo-80/128 is a 64-bit block cipher and supports 80/128-bit secret keys. As shown in Fig. 2, the structure of Piccolo-80/128 is a variant of generalized Feistel network. Here, the number of rounds r is 25 for Piccolo-80 and 31 for Piccolo-128. First, with a 64-bit plaintext $P = (P_0, P_1, P_2, P_3)$ and a prewhitening key (wk_0, wk_1), the input value $I_0 = (I_{0,0}, I_{0,1}, I_{0,2}, I_{0,3})$ of round 0 is computed as follows.

$$I_{0,0} = P_0 \oplus wk_0,\ I_{0,1} = P_1,\ I_{0,2} = P_2 \oplus wk_1,\ I_{0,3} = P_3.$$

To generate I_{i+1} from I_i $(i = 0, \cdots, r-2)$, each round is made up of a function F and a 64-bit round permutation RP. Fig. 3-(a) presents the structure of F function. Since our attack do not use the property of 4×4 S-box S and 4×4 matrix M, we omit the descriptions of them in this paper. See [9] for the detailed descriptions of them. As shown in Fig. 3-(b), a round permutation RP takes a 64-bit input value $X = (x_0, x_1, x_2, x_3)$ and generates a 64-bit output

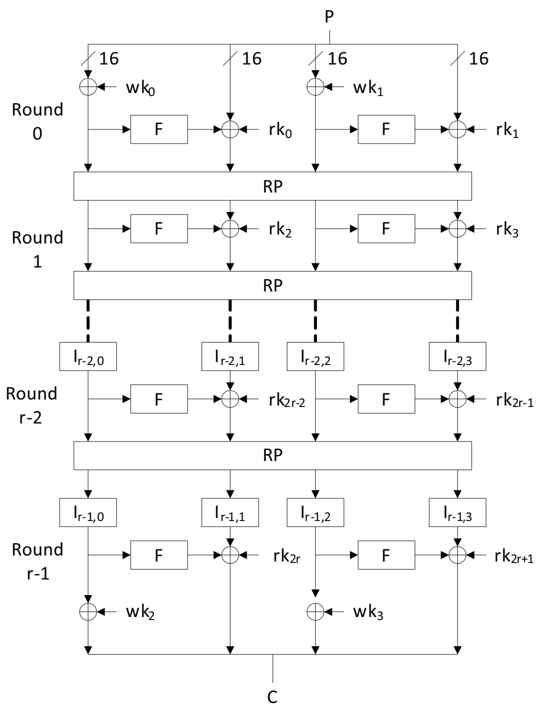

Fig. 2. The structure of Piccolo.

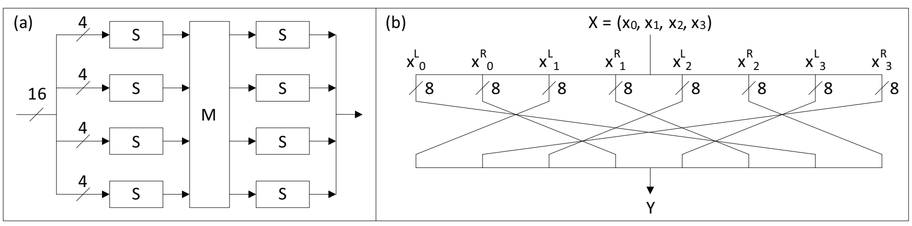

Fig. 3. (a) F function and (b) round permutation RP of Piccolo.

value $Y = (y_0, y_1, y_2, y_3)$. Here, a 16-bit x_i is divided into (x_i^L, x_i^R). A 64-bit ciphertext $C = (C_0, C_1, C_2, C_3)$ is generated as follows.

$$C_0 = I_{r-1,0} \oplus wk_2, \quad C_1 = F(I_{r-1,0}) \oplus I_{r-1,1} \oplus rk_{2r},$$
$$C_2 = I_{r-1,2} \oplus wk_3, \quad C_3 = F(I_{r-1,2}) \oplus I_{r-1,3} \oplus rk_{2r+1}.$$

The keyschedule of Piccolo-80 is simple. First, the 80-bit secret key K is computed as follows. Here, $k_j = (k_j^L, k_j^R)$ $(j = 0, 1, 2, 3, 4)$.

$$K = (k_0, k_1, k_2, k_3, k_4).$$

Four whitening keys (wk_0, wk_1, wk_2, wk_3) and 25 round keys (rk_{2r}, rk_{2r+1}) are generated as follows $(r = 0, 1, \cdots, 24)$. Here, $(con_{2r}^{80}, con_{2r+1}^{80})$ is a 16-bit round constant. See [9] for the detailed descriptions of them.

– Whitening key

$$wk_0 = k_0^L \| k_1^R, \quad wk_1 = k_1^L \| k_0^R,$$
$$wk_2 = k_4^L \| k_3^R, \quad wk_3 = k_3^L \| k_4^R.$$

– Round key

$$(rk_{2r}, rk_{2r+1}) = (con_{2r}^{80}, con_{2r+1}^{80}) \oplus \begin{cases} (k_2, k_3), (r \mod 5) \equiv 0 \text{ or } 2, \\ (k_0, k_1), (r \mod 5) \equiv 1 \text{ or } 4, \\ (k_4, k_4), (r \mod 5) \equiv 3. \end{cases}$$

The keyschedule of Piccolo-128 is similar to that of Piccolo-80. By using the 128-bit secret key $K = (k_0, k_1, \cdots, k_7)$, four whitening keys and 31 round keys are generated as follows.

– Whitening key

$$wk_0 = k_0^L \| k_1^R, \quad wk_1 = k_1^L \| k_0^R,$$
$$wk_2 = k_4^L \| k_7^R, \quad wk_3 = k_7^L \| k_4^R.$$

– Round key $(i = 0, 1, \cdots, 61)$
  - if $((i + 2) \mod 8 \equiv 0)$ then

$$(k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7) = (k_2, k_1, k_6, k_7, k_0, k_3, k_4, k_5).$$

  - $rk_i = k_{(i+2) \mod 8} \oplus con_i^{128}$.

Table 2 presents the partial secret keys used in each round key. For example, in Piccolo-80, the round key (rk_{48}, rk_{49}) of round 24 includes the partial secret key (k_0, k_1). See [9] for the detailed descriptions of them.

Table 2. The partial secret key used in each round key of Piccolo.

Piccolo-80		Piccolo-128	
Round i	Partial secret key	Round i	Partial secret key
Prewhitening	$(k_0^L \| k_1^R, k_1^L \| k_0^R)$	Prewhitening	$(k_0^L \| k_1^R, k_1^L \| k_0^R)$
0	(k_2, k_3)	0	(k_2, k_3)
1	(k_0, k_1)	1	(k_4, k_5)
$\vdots$	$\vdots$	$\vdots$	$\vdots$
22	(k_2, k_3)	28	(k_0, k_7)
23	(k_4, k_4)	29	(k_6, k_3)
24	(k_0, k_1)	30	(k_2, k_5)
Postwhitening	$(k_4^L \| k_3^R, k_3^L \| k_4^R)$	Postwhitening	$(k_4^L \| k_7^R, k_7^L \| k_4^R)$

2.3 LED

LED is a 64-bit block cipher which supports 64/80/96/128-bit secret keys. The number of rounds is 32 (LED-64) and 48 (LED-80/96/128). A 64-bit internal state is treated as the following nibble matrix of size 4×4 where each nibble represents an element from $GF(2^4)$ with the underlying polynomial for field multiplication given by $x^4 + x + 1$. Here, $I[i]$ is an i-th nibble value of I ($i = 0, \cdots, 15$).

$$I = \begin{pmatrix} I[0] & I[1] & I[2] & I[3] \\ I[4] & I[5] & I[6] & I[7] \\ I[8] & I[9] & I[10] & I[11] \\ I[12] & I[13] & I[14] & I[15] \end{pmatrix}.$$

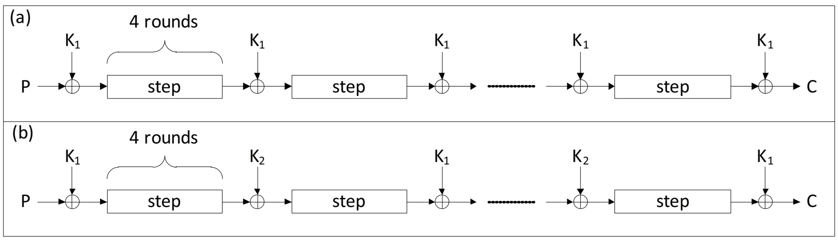

Fig. 4. The encryption process of (a) LED-64 and (b) LED-80/96/128.

Fig. 4 presents the encryption process of LED-64 and LED-80/96/128, respectively. The encryption process is described by using $addRoundKey(I, K)$ and $step(I)$. In $addRoundKey(I, K)$, nibbles of round key K are combined with an internal state I. Note that there is no keyschedule. According to the length of secret keys, they are arranged in one or two matrices of size 4×4 over $GF(2^4)$. In detail, the secret key K is repeatedly used without modification as follows.

– The 64-bit secret key $K = (K[0], K[1], \cdots, K[15])$:

$$K_1 = \begin{pmatrix} K[0] & K[1] & K[2] & K[3] \\ K[4] & K[5] & K[6] & K[7] \\ K[8] & K[9] & K[10] & K[11] \\ K[12] & K[13] & K[14] & K[15] \end{pmatrix}.$$

– The 80-bit secret key $K = (K[0], K[1], \cdots, K[19])$:

$$K_1 \| K_2 = \begin{pmatrix} K[0] & K[1] & K[2] & K[3] \\ K[4] & K[5] & K[6] & K[7] \\ K[8] & K[9] & K[10] & K[11] \\ K[12] & K[13] & K[14] & K[15] \end{pmatrix} \begin{pmatrix} K[16] & K[17] & K[18] & K[19] \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

– The 96-bit secret key $K = (K[0], K[1], \cdots, K[23])$:

$$K_1 \| K_2 = \begin{pmatrix} K[0] & K[1] & K[2] & K[3] \\ K[4] & K[5] & K[6] & K[7] \\ K[8] & K[9] & K[10] & K[11] \\ K[12] & K[13] & K[14] & K[15] \end{pmatrix} \begin{pmatrix} K[16] & K[17] & K[18] & K[19] \\ K[20] & K[21] & K[22] & K[23] \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}.$$

– The 128-bit secret key $K = (K[0], K[1], \cdots, K[31])$:

$$K_1 \| K_2 = \begin{pmatrix} K[0] & K[1] & K[2] & K[3] \\ K[4] & K[5] & K[6] & K[7] \\ K[8] & K[9] & K[10] & K[11] \\ K[12] & K[13] & K[14] & K[15] \end{pmatrix} \begin{pmatrix} K[16] & K[17] & K[18] & K[19] \\ K[20] & K[21] & K[22] & K[23] \\ K[24] & K[25] & K[26] & K[27] \\ K[28] & K[29] & K[30] & K[31] \end{pmatrix}.$$

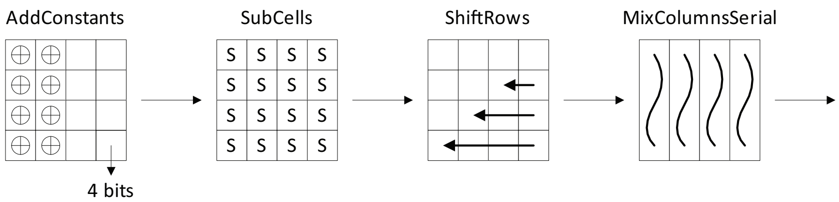

Fig. 5. Overview of a single round of LED.

The second operation $step(I)$ consists of four rounds of the encryption process. Each of these four rounds uses, in sequence, the operations $AddConstants$, $SubCells$, $ShiftRows$ and $MixColumnsSerial$ as illustrated in Fig. 5.

In $AddConstants(AC)$, a round constant is defined as follows. At each round, the six bits $(rc_5, rc_4, rc_3, rc_2, rc_1, rc_0)$ are shifted one position to the left with the new value to rc_0 being computed as $rc_5 \oplus rc_4 \oplus 1$. The six bits are initialized to zero, and updated "before" use in a given round. The constant, when used in a given round, is arranged into an array as follows:

$$\begin{pmatrix} 0 \ (rc_5\|rc_4\|rc_3) \ 0 \ 0 \\ 1 \ (rc_2\|rc_1\|rc_0) \ 0 \ 0 \\ 2 \ (rc_5\|rc_4\|rc_3) \ 0 \ 0 \\ 3 \ (rc_2\|rc_1\|rc_0) \ 0 \ 0 \end{pmatrix}.$$

The round constants are combined with an internal state, respecting array positioning, using bitwise exclusive-or.

In $SubCells(SC)$, each nibble in an internal state I is replaced by the nibble generated after using the following 4×4 S-box used in PRESENT [3].

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	C	5	6	B	9	0	A	D	3	E	F	8	4	1	7	2

In $ShiftRows(SR)$, row i of an internal state I is rotated i cell positions to the left, for $i = 0, 1, 2, 3$.

In $MixColumnsSerial(MC)$, each column of an internal state I is viewed as a column vector and replaced by the column vector that results after post-multiplying the vector by the following MDS matrix M.

$$M = \begin{pmatrix} 4 & 2 & 1 & 1 \\ 8 & 6 & 5 & 6 \\ B & E & A & 9 \\ 2 & 2 & F & B \end{pmatrix}.$$

3 Biclique cryptanalysis

Biclique cryptanalysis is a kind of meet-in-the-middle attack. The main idea of this attack is to construct bicliques on the target subcipher, and use this to enhance the efficiency of computations.

First, the concept of biclique is as follows. Let f be a subcipher that maps an internal state S to a ciphertext C: $f_K(S) = C$. We consider 2^d internal states $\{S_0, \cdots, S_{2^d-1}\}$, 2^d ciphertexts $\{C_0, \cdots, C_{2^d-1}\}$ and 2^{2d} keys $\{K_{<i,j>}\}$:

$$\{K_{<i,j>}\} = \begin{bmatrix} K_{<0,0>} & K_{<0,1>} & \cdots & K_{<0,2^d-1>} \\ \vdots & \vdots & \ddots & \vdots \\ K_{<2^d-1,0>} & K_{<2^d-1,1>} & \cdots & K_{<2^d-1,2^d-1>} \end{bmatrix}. \quad (1)$$

The 3-tuple $[\{C_i\}, \{S_j\}, \{K_{<i,j>}\}]$ is called a d-dimensional biclique, if

$$C_i = f_{K_{<i,j>}}(S_j) \text{ for all } i, j \in \{0, \cdots, 2^d - 1\}. \quad (2)$$

Biclique cryptanalysis consists of the following steps.

1. **[Preparation]** Partition the whole key space into 2^{k-2d} sets of 2^{2d} keys each, where k is the size of secret key. Each key in a set is indexed as an element of a $2^d \times 2^d$ matrix like Equation (1): $\{K_{<i,j>}\}$.
2. **[Constructing bicliques]** For each set of keys, build the structure $\{C_0, \cdots, C_{2^d-1}\}, \{S_0, \cdots, S_{2^d-1}\}, \{K_{<i,j>}\}$ satisfying Equation (2).
3. **[Collecting data]** Obtain plaintexts $\{P_i\}$ from ciphertexts $\{C_i\}$ through the decryption oracle.
4. **[Testing keys]** The right key K maps a plaintext P_i to an intermediate S_j. Thus, check
$$\exists\, i, j : P_i \xrightarrow[g]{K_{<i,j>}} S_j, \tag{3}$$
which proposes a key candidate. Here, g is a subcipher that maps a plaintext P to an internal state S: $g_K(P) = S$. If the right key is not found in the chosen key set, then choose another key set and repeat the above process.

3.1 Constructing bicliques from independent related-key differentials

In [2], two methods to construct bicliques were proposed. One is to use independent related-key differentials and the other is to interleave related-key differentials. In this paper, we focus on the former, called the independent-biclique.

We consider two sets of 2^d keys $A = \{K_{<i,0>} \mid 0 \le i \le 2^d - 1\}$ and $B = \{K_{<0,j>} \mid 0 \le j \le 2^d - 1\}$ such that $A \cap B = \{K_{<0,0>}\}$. We assume that S_0 is an intermediate string randomly chosen. Let $K_{<0,0>}$ maps S_0 to a ciphertext C_0 with f, that is $S_0 \xrightarrow[f]{K_{<0,0>}} C_0$. Then, $\{C_i\}$ and $\{S_j\}$ are obtained by using the following computations.

$$S_0 \xrightarrow[f]{K_{<i,0>}} C_i \text{ and } S_j \xleftarrow[f^{-1}]{K_{<0,j>}} C_0.$$

Let $\Delta_i = C_0 \oplus C_i$, $\Delta_i^K = K_{<0,0>} \oplus K_{<i,0>}$, $\nabla_j = S_0 \oplus S_j$ and $\nabla_j^K = K_{<0,0>} \oplus K_{<0,j>}$. Then, we can construct the Δ_i-differential $0 \xrightarrow{\Delta_i^K} \Delta_i$ where a related-key difference is Δ_i^K, and the ∇_j-differential $\nabla_j \xrightarrow{\nabla_j^K} 0$ where a related-key difference is ∇_j^K.

If these two related-key differentials do not share active nonlinear components for all i and j, the following relation is satisfied.

$$S_0 \oplus \nabla_j \xrightarrow[f]{K_{<0,0>} \oplus \Delta_i^K \oplus \nabla_j^K} C_0 \oplus \Delta_i \text{ for all } i, j \in \{0, \cdots, 2^d - 1\}.$$

3.2 Matching with precomputations

The matching with precomputations is an efficient method to check Equation (3) in the attack procedure. Let v be a part of an internal state between $\{P_i\}$

and $\{S_j\}$. v is called the matching variable. First, an attacker computes and stores the followings in memory.

$$\text{for all } i \in \{0, \cdots, 2^d - 1\},\ P_i \xrightarrow{K_{<i,0>}} \overrightarrow{v},$$
$$\text{for all } j \in \{0, \cdots, 2^d - 1\},\ \overleftarrow{v} \xleftarrow{K_{<0,j>}} S_j.$$

Then, for particular i and j, he checks the matching at v by recomputing only those parts of the cipher which differ from the stored ones. The cost of recomputation depends on the diffusion properties of both internal rounds and the keyschedule of the cipher.

4 Biclique cryptanalysis of PRESENT

In this section, we propose biclique cryptanalysis of PRESENT-80/128.

4.1 Biclique cryptanalysis of PRESENT-80

First, we explain how to construct a 4-dimensional biclique for round 28 $\sim$ 30 of PRESENT-80. The partial secret keys used in $(RK^{28}, RK^{29}, RK^{30}, RK^{31})$ are as follows.

- RK^{28}: $(k_{51}, k_{50}, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{68})$.
- RK^{29}: $(k_{70}, k_{69}, \cdots, k_7)$.
- RK^{30}: $(k_9, k_8, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{26})$.
- RK^{31}: $(k_{28}, k_{27}, \cdots, k_0, k_{79}, k_{78}, \cdots, k_{45})$.

From the above relation, we found that varying $(k_{58}, k_{57}, k_{56}, k_{55})$ and $(k_{37}, k_{36}, k_{35}, k_{34})$ gives bicliques for the attack on the full PRESENT-80. In detail, to construct the Δ_i-differential and the ∇_j-differential, we consider $(k_{58}, k_{57}, k_{56}, k_{55})$ and $(k_{37}, k_{36}, k_{35}, k_{34})$, respectively. Let f be a subcipher from round 28 to round 30 (see Fig. 6). An attacker fixes $C_0 = 0$ and derives $S_0 = f^{-1}_{K_{<0,0>}}(C_0)$. The Δ_i-differentials are based on the difference Δ_i^K where the difference of $(k_{58}, k_{57}, k_{56}, k_{55})$ is i and the other bits have zero differences. Similarly, the ∇_j-differentials are based on the difference ∇_j^K where the difference of $(k_{37}, k_{36}, k_{35}, k_{34})$ is j and the other bits have zero differences. Since the Δ_i-differential affects only 23 bits of the ciphertext from Fig. 6, all ciphertexts can be forced to share the same values in other bits. As a result, the data complexity does not exceed 2^{23}.

Now we are ready to describe our attack on the full PRESENT-80. We rewrite the full PRESENT-80 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 14$, round $15 \sim 27$ and round $28 \sim 30$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

We assume that the plaintext set $\{P_i\}$ corresponding to a 3-round biclique is obtained through the decryption oracle. Applying Equation (3) to $g_2 \circ g_1$, an

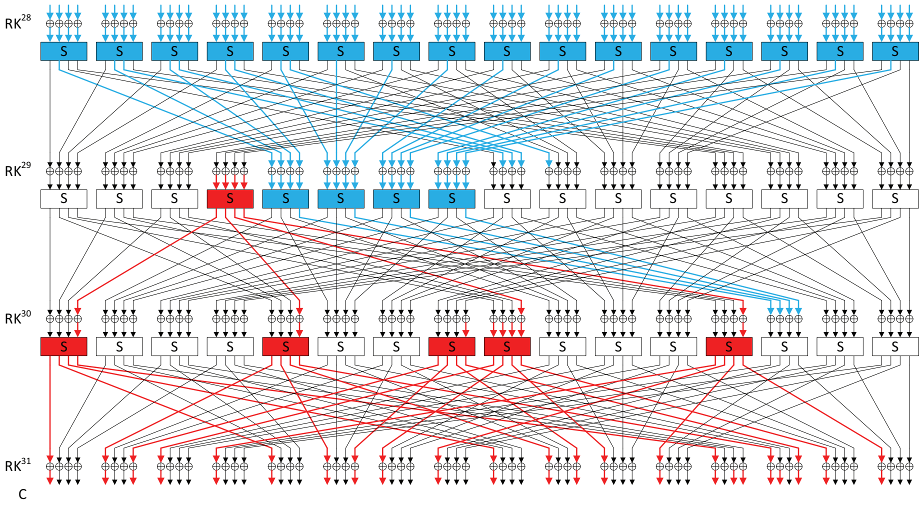

Fig. 6. 4-dimensional biclique for PRESENT-80.

attacker detects the right secret key by computing an intermediate variable v in both directions.

$$P_i \xrightarrow[g_1]{K_{<i,j>}} \overrightarrow{v} \stackrel{?}{=} \overleftarrow{v} \xleftarrow[g_2^{-1}]{K_{<i,j>}} S_j. \tag{4}$$

The attack procedure on the full PRESENT-80 is as follows.

1. **[Precomputation]** For all $i = 0, \cdots, 2^4 - 1 (d = 4)$, an attacker computes the most significant 4 bits of the output value of round 15 from P_i and $K_{<i,0>}$ in the forward direction, and store it as $\overrightarrow{v}_i$, together with intermediate states and round keys in memory (see Fig. 7). For all $j = 0, \cdots, 2^4-1$, an attacker computes the most significant 4 bits of the input value of round 16 from S_i and $K_{<0,j>}$ in the backward direction, and store it as $\overleftarrow{v}_j$, together with intermediate states and round keys in memory.
2. **[Computation in the backward direction]** In the backward direction, an attacker should compute $\overleftarrow{v}_j$ from S_j and $K_{<i,j>}$ for all i and j, and store them in memory. Recomputations are performed according to red/blue lines in Fig. 7, and the values on the other lines are reused from the precomputation table.
3. **[Computation in the forward direction]** In the forward direction, an attacker should compute $\overrightarrow{v}_i$ from P_i and $K_{<i,j>}$. Recomputations are performed according to red/blue lines in Fig. 7, and the values on the other lines are reused from the precomputation table.

For each computed $\overrightarrow{v}$, an attacker checks whether the corresponding key candidate $K_{<i,j>}$ satisfies Equation (4). If he finds such one, he should check the matching on the whole input value of round 16 for $K_{<i,j>}$, P_i and S_j. This matching step yields the right secret key K with a high probability. If a biclique does not give the right secret key, an attacker should choose another biclique and repeat the above procedure until the right secret key is found.

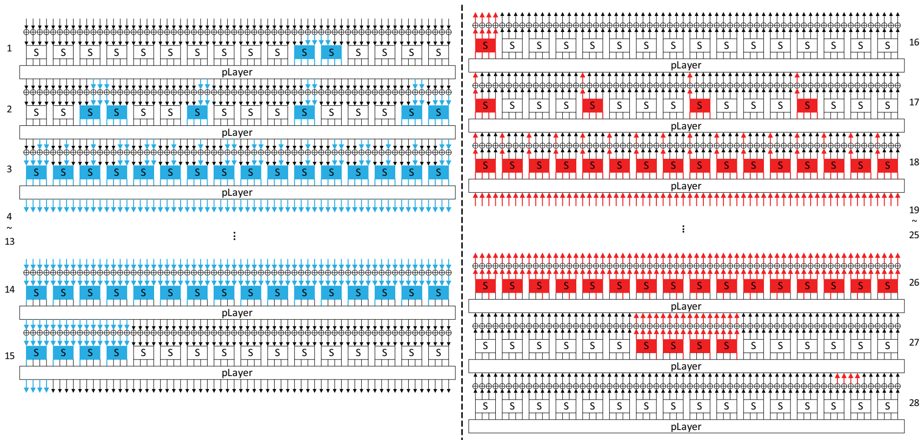

Fig. 7. Recomputations in forward and backward directions for PRESENT-80.

A total computational complexity of our attack on PRESENT-80 is computed as follows.

$$C_{total} = 2^{k-2d}\left(C_{biclique} + C_{precomp} + C_{recomp} + C_{falsepos}\right). \tag{5}$$

- $k = 80$ and $d = 4$.
- $C_{biclique}$ is a computational complexity of constructing a single biclique. In our attack, it is $2^{1.63}\left(\approx 2^{4+1}\cdot(3/31)\right)$ full PRESENT-80 encryptions.
- $C_{precomp}$ is a computational complexity of preparing the precomputation for the matching check in Equation (4). Applying it to our attack, it is $2^{3.85}\left(\approx 2^{4}\cdot(28/31)\right)$ full PRESENT-80 encryptions.
- C_{recomp} is a computational complexity of recomputing the internal variable v $2^{2d}\left(= 2^{8}\right)$ times. In our attack, it is $2^{7.53}\left(\approx 2^{2\cdot 4}\cdot(22.31/31)\right)$ full PRESENT-80 encryptions.
- $C_{falsepos}$ is a computational complexity caused by false positives, which have to be matched on other bit positions. Since the matching check is performed on four bits in our attack, $C_{falsepos}$ is $2^{4}\left(\approx 2^{2\cdot 4-4}\right)$ full PRESENT-80 encryptions.

Hence, a computational complexity of our attack on the full PRESENT-80 is computed as follows.

$$C_{total} = 2^{79.86}\left(\approx 2^{80-2\cdot 4}\left(2^{1.63} + 2^{3.85} + 2^{7.53} + 2^{4}\right)\right).$$

4.2 Biclique cryptanalysis of PRESENT-128

Since biclique cryptanalysis of the full PRESENT-128 is similar to that of the full PRESENT-80, we briefly introduce our attack on the full PRESENT-128.

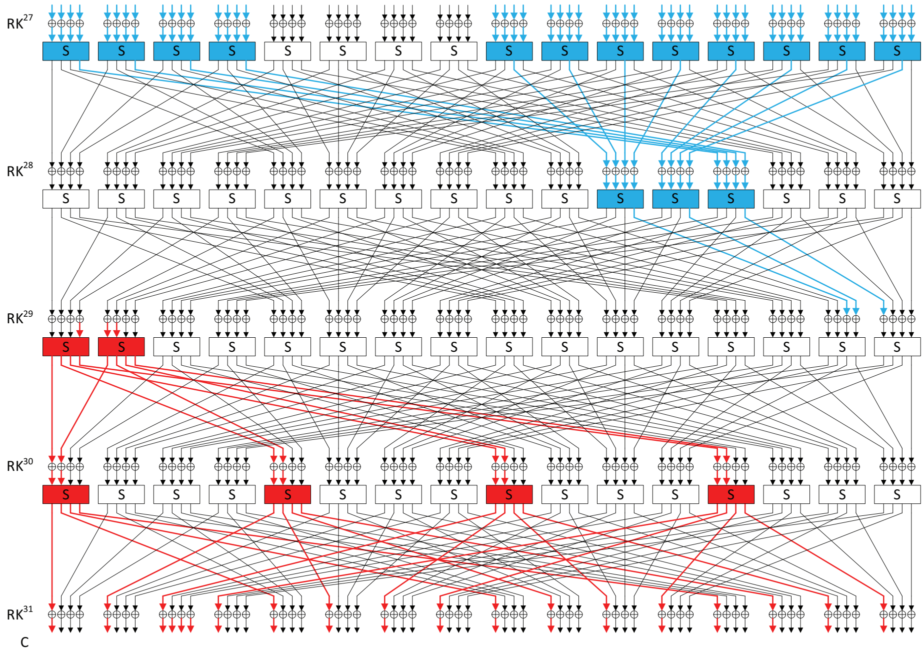

Fig. 8. 4-dimensional biclique for PRESENT-128.

To recover the 128-bit secret key, we construct a 3-dimensional biclique for round 27 $\sim$ 31 of PRESENT-128 as shown in Fig. 8. The partial secret keys used in $(RK^{27}, RK^{28}, RK^{29}, RK^{30}, RK^{31})$ are as follows.

- RK^{27}: $(k_{16}, k_{15}, \cdots, k_0, k_{127}, k_{126}, \cdots, k_{81})$.
- RK^{28}: $(k_{83}, k_{82}, \cdots, k_{20})$.
- RK^{29}: $(k_{22}, k_{21}, \cdots, k_0, k_{127}, k_{126}, \cdots, k_{87})$.
- RK^{30}: $(k_{89}, k_{88}, \cdots, k_{26})$.
- RK^{31}: $(k_{28}, k_{27}, \cdots, k_0, k_{127}, k_{126}, \cdots, k_{93})$.

From the above relation, to construct the Δ_i-differential and the ∇_j-differential, we consider (k_{19}, k_{18}, k_{17}) and (k_{80}, k_{79}, k_{78}), respectively. The Δ_i-differential affects only 19 bits of the ciphertext from Fig. 8. As a result, the data complexity does not exceed 2^{19}.

We rewrite the full PRESENT-128 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 13$, round $14 \sim 26$ and round $27 \sim 30$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

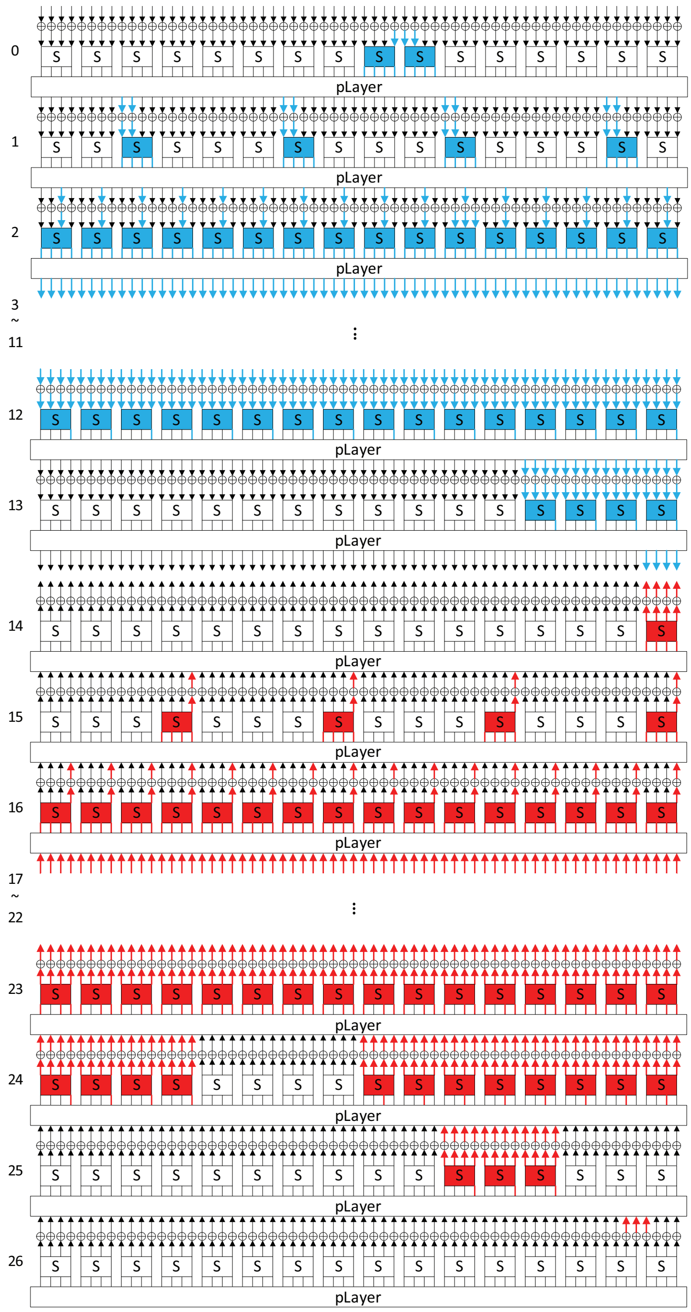

Fig. 9. Recomputations in forward and backward directions for PRESENT-128.

As depicted in Fig. 9, the matching variable v is the least significant 4 bits of the output value of round 13 (or the input value of round 14). Then, the complexities of our attack are computed as follows (see Equation (5)).

– Computational complexity: $2^{127.81}$ full PRESENT-128 encryptions.
  - $k = 128$ and $d = 3$.
  - $C_{biclique} = 2^{1.05} \left(\approx 2^{3+1} \cdot (4/31)\right)$ full PRESENT-128 encryptions.
  - $C_{precomp} = 2^{2.8} \left(2^{3} \cdot (27/31)\right)$ full PRESENT-128 encryptions.
  - $C_{recomp} = 2^{5.43} \left(2^{2 \cdot 3} \cdot (20.88/31)\right)$ full PRESENT-128 encryptions.
  - $C_{falsepos} = 2^{2} \left(2^{2 \cdot 3 - 4}\right)$ full PRESENT-128 encryptions.

5 Biclique cryptanalysis of Piccolo

In this section, we explain key recovery attacks on the full Piccolo-80/128 by constructing 8-dimensional bicliques.

5.1 Biclique cryptanalysis of Piccolo-80

As presented in Section 2.2, two 2-byte round keys are used at each round. Each 2-byte round keys are generated by a 2-byte secret key k_i (see Table 2). From Table 2, we found that k_4^L and k_2^L give the construction of a 8-dimensional biclique. Let f be a subcipher from round 19 to round 24. Then, as shown in Fig. 10, we can construct an 8-dimensional biclique. The Δ_i-differential affects only 6 bytes of the ciphertext from Fig. 10. As a result, the data complexity does not exceed 2^{48}.

We rewrite the full Piccolo-80 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 9$, round $10 \sim 18$ and round $19 \sim 24$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

The matching variable v is an 8-bit $I_{10,3}^L$ of round 10 as shown in Fig. 11. Then, the complexities of our attack are computed as follows (see Equation (5)).

– Computational complexity: $2^{79.13}$ full Piccolo-80 encryptions.
  - $k = 80$ and $d = 8$.
  - $C_{biclique} = 2^{6.94} \left(\approx 2^{8+1} \cdot (6/25)\right)$ full Piccolo-80 encryptions.
  - $C_{precomp} = 2^{7.6} \left(2^{8} \cdot (19/25)\right)$ full Piccolo-80 encryptions.
  - $C_{recomp} = 2^{15.11} \left(2^{2 \cdot 8} \cdot (13.5/25)\right)$ full Piccolo-80 encryptions.
  - $C_{falsepos} = 2^{8} \left(2^{2 \cdot 8 - 8}\right)$ full Piccolo-80 encryptions.

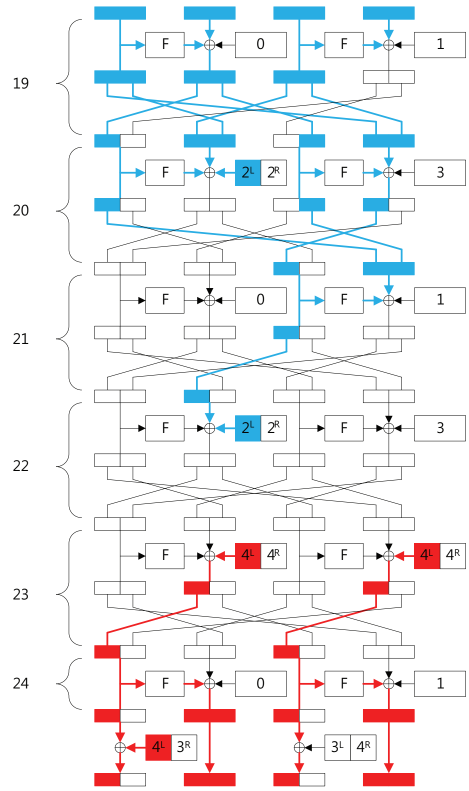

Fig. 10. 8-dimensional biclique for Piccolo-80.

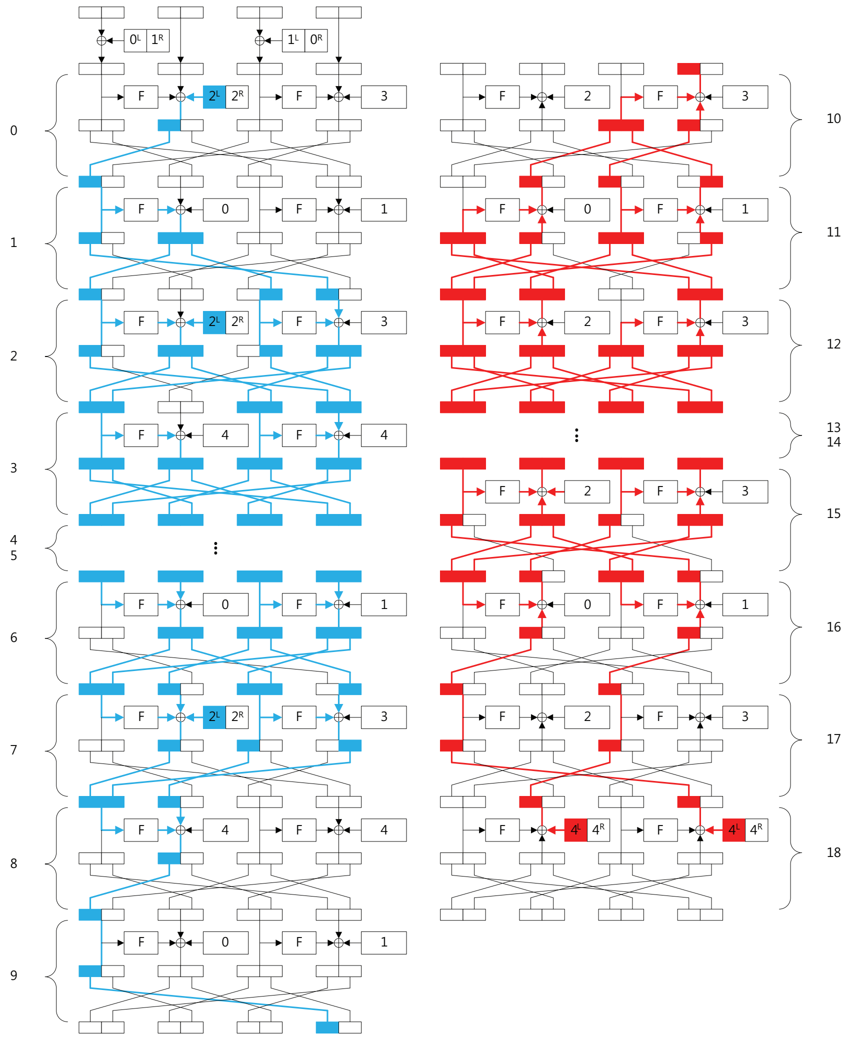

Fig. 11. Recomputations in forward and backward directions for Piccolo-80.

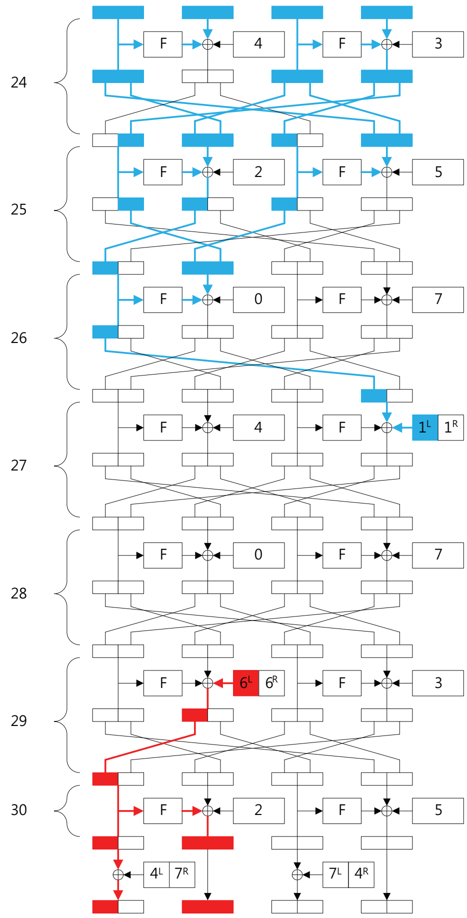

Fig. 12. 8-dimensional biclique for Piccolo-128.

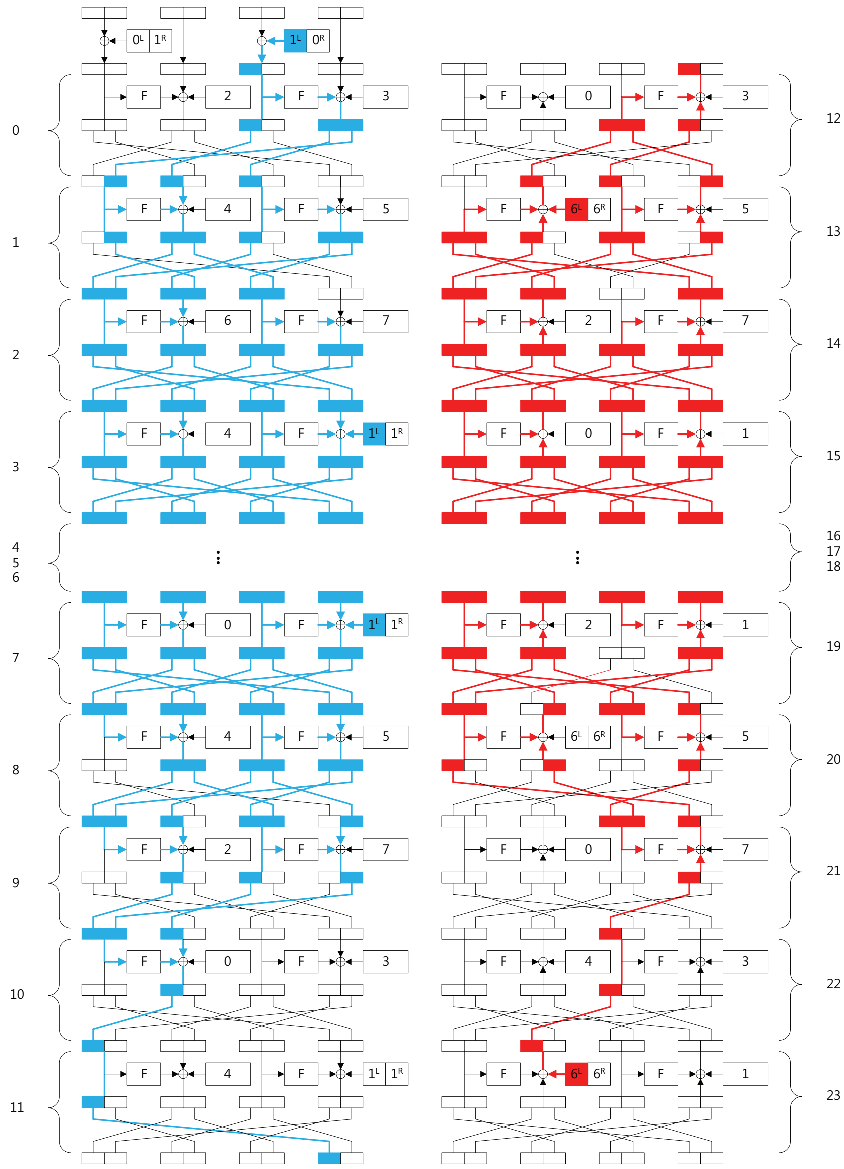

Fig. 13. Recomputations in forward and backward directions for Piccolo-128.

5.2 Biclique cryptanalysis of Piccolo-128

To recover the 128-bit secret key of Piccolo-128, we consider k_6^L and k_1^L from Table 2. Then, we can construct a 8-dimensional biclique for round $24 \sim 30$ as shown in Fig. 12. The Δ_i-differential affects only 3 bytes of the ciphertext from Fig. 12. As a result, the data complexity does not exceed 2^{24}.

We rewrite the full Piccolo-128 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 11$, round $12 \sim 23$ and round $24 \sim 30$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

As depicted in Fig. 13, the matching variable v is an 8-bit $I_{10,3}^L$ of round 12. Then, the complexities of our attack are computed as follows (see Equation (5)).

- Computational complexity: $2^{127.35}$ full Piccolo-128 encryptions.
  - $k = 128$ and $d = 8$.
  - $C_{biclique} = 2^{6.85} \left(\approx 2^{8+1} \cdot (7/31)\right)$ full Piccolo-128 encryptions.
  - $C_{precomp} = 2^{7.63} \left(2^8 \cdot (24/31)\right)$ full Piccolo-128 encryptions.
  - $C_{recomp} = 2^{15.33} \left(2^{2 \cdot 8} \cdot (19.5/31)\right)$ full Piccolo-128 encryptions.
  - $C_{falsepos} = 2^8 \left(2^{2 \cdot 8 - 8}\right)$ full Piccolo-128 encryptions.

6 Biclique cryptanalysis of LED

We propose two versions of our attacks on LED. First, we present the attacks on a 29-round reduced LED-64 and 45-round reduced LED-80/96/128. Then the attacks on the full LED-80/96/128 are introduced.

6.1 Biclique cryptanalysis of reduced versions of LED

The attack procedures on 45-round reduced LED-80/96/128 are similar to that on a 29-round reduced LED-64. Thus, we focus on the attack on a 29-round reduced LED-64 in this subsection.

For a 29-round reduced LED-64, we consider $(K[10], K[11])$ and $(K[2], K[3])$ (see Section 2.3). Then, we can construct a 8-dimensional biclique for round $24 \sim 28$ (see Fig. 14). The Δ_i-differential affects only 10 nibbles of the ciphertext from Fig. 14. As a result, the data complexity does not exceed 2^{40}. We rewrite a 29-round reduced LED-64 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 12$, round $13 \sim 23$ and round $24 \sim 28$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

The matching variable v is the 4-bit input value $I[0]$ of round 13 as shown in Fig. 15. Then, the complexities of our attack are computed as follows (see Equation (5)).

- Computational complexity: $2^{63.58}$ 29-round reduced LED-64 encryptions.

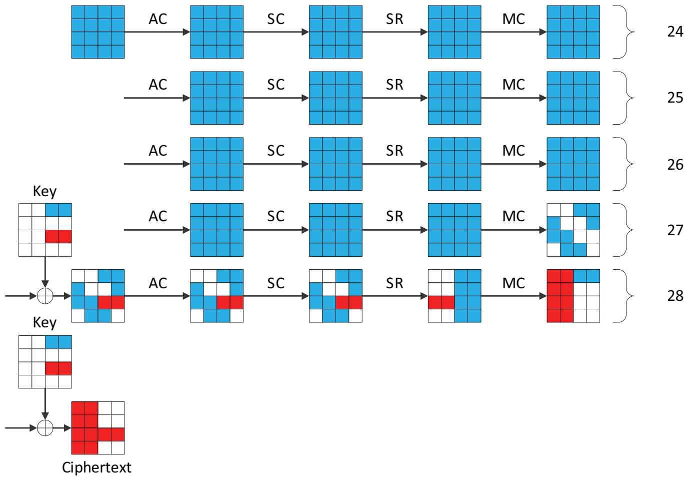

Fig. 14. 8-dimensional biclique for a 29-round reduced LED-64.

- $k = 64$ and $d = 8$.
- $C_{biclique} = 2^{6.46}$ $\left(\approx 2^{8+1} \cdot (5/29)\right)$ 29-round reduced LED-64 encryptions.
- $C_{precomp} = 2^{7.73}$ $\left(2^{8} \cdot (24/29)\right)$ 29-round reduced LED-64 encryptions.
- $C_{recomp} = 2^{15.44}$ $\left(2^{2 \cdot 8} \cdot (19.69/29)\right)$ 29-round reduced LED-64 encryptions.
- $C_{falsepos} = 2^{12}$ $\left(2^{2 \cdot 8 - 4}\right)$ 29-round reduced LED-64 encryptions.

The attacks on 45-round reduced LED-80/96/128 are explained in a similar fashion. In these cases, we consider $(K[16], K[17])$ and $(K[18], K[19])$ (see Section 2.3). Then, we can construct a 8-dimensional biclique for round $36 \sim 44$. We rewrite 45-round reduced LED-80/96/128 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 17$, round $18 \sim 35$ and round $36 \sim 44$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

The matching variable v is the 4-bit input value $I[0]$ of round 18. Then, the complexities of our attacks are computed as follows (see Equation (5)).

- Data complexity: 2^{32} chosen plaintexts.
- Computational complexity: $2^{79.45}/2^{95.45}/2^{127.45}$ 45-round reduced LED-80/96/128 encryptions.
  - $k = 80/96/128$ and $d = 8$.
  - $C_{biclique} = 2^{6.68}$ $\left(\approx 2^{8+1} \cdot (9/45)\right)$ 45-round reduced LED-80/96/128 encryptions.
  - $C_{precomp} = 2^{7.68}$ $\left(2^{8} \cdot (36/45)\right)$ 45-round reduced LED-80/96/128 encryptions.
  - $C_{recomp} = 2^{15.3}$ $\left(2^{2 \cdot 8} \cdot (27.69/45)\right)$ 45-round reduced LED-80/96/128 encryptions.
  - $C_{falsepos} = 2^{12}$ $\left(2^{2 \cdot 8 - 4}\right)$ 45-round reduced LED-80/96/128 encryptions.

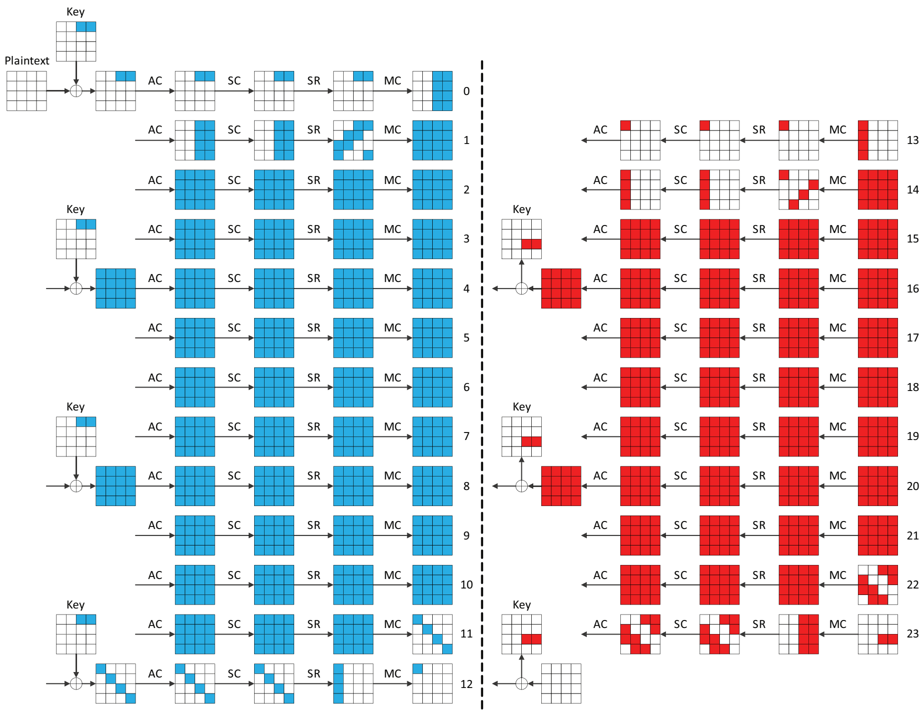

Fig. 15. Recomputations in forward and backward directions for a 29-round reduced LED-64.

6.2 Biclique cryptanalysis of the full LED

We found that it is possible to apply biclique cryptanalysis to the full LED-80/96/128. Similarly to the attacks on 45-round reduced LED-80/96/128, we consider $(K[16], K[17])$ and $(K[18], K[19])$ to construct a 8-dimensional biclique for round $36 \sim 47$ as shown in Fig. 16. The Δ_i-differential affects all nibbles of the ciphertext from Fig. 16. As a result, this attack require the full codebook (2^{64} chosen plaintexts). We rewrite the full LED-80/96/128 as follows. Here, g_1, g_2 and f are subciphers for round $0 \sim 17$, round $18 \sim 35$ and round $36 \sim 47$, respectively.

$$E : P \underset{g_1}{\longrightarrow} V \underset{g_2}{\longrightarrow} S \underset{f}{\rightarrow} C.$$

As depicted in Fig. 17, the matching variable v is the 4-bit input value $I[0]$ of round 18. Then, the complexities of our attacks are computed as follows (see Equation (5)).

- Data complexity: 2^{64} chosen plaintexts.
- Computational complexity: $2^{79.37}/2^{95.37}/2^{127.37}$ full LED-80/96/128 encryptions.
  - $k = 80/96/128$ and $d = 8$.
  - $C_{biclique} = 2^7 \left(\approx 2^{8+1} \cdot (12/48)\right)$ full LED-80/96/128 encryptions.
  - $C_{precomp} = 2^{7.58} \left(2^8 \cdot (36/48)\right)$ full LED-80/96/128 encryptions.
  - $C_{recomp} = 2^{15.21} \left(2^{2 \cdot 8} \cdot (27.69/48)\right)$ full LED-80/96/128 encryptions.
  - $C_{falsepos} = 2^{12} \left(2^{2 \cdot 8 - 4}\right)$ full LED-80/96/128 encryptions.

7 Conclusion

In this paper, we proposed biclique cryptanalysis of lightweight block ciphers PRESENT, Piccolo and LED. Our attack results are summarized in Table 1. From this table, our results are superior to known biclique cryptanalytic results on them.

References

1. F. Abed, C. Forler, E. List, S. Lucks and J. Wenzel, *Biclique Cryptanalysis of the PRESENT and LED Lightweight Ciphers*, Cryptology ePrint Archive, Report 2012/591, 2012.
2. A. Bogdanov, D. Khovratovich and C. Rechberger, *Biclique Cryptanalysis of the Full AES*, ASIACRYPT 2011, LNCS 7073, pp. 344–371, IACR, 2011.
3. A. Bogdanov, L. Knudsen, G. Leander, C. Paar, A. Poschmann, M. Robshaw, Y. Seurin and C. Vikkelsoe, *PRESENT: An Ultra-Lightweight Block Cipher*, CHES 2007, LNCS 4727, pp. 450–466, Springer-Verlag, 2007.
4. S. Chen, *Biclique Attack of the Full ARIA-256*, Cryptology ePrint Archive, Report 2012/011, 2012.
5. M. Çoban, F. Karakoç and Ö. Biztaş, *Biclique Cryptanalysis of TWINE*, Cryptology ePrint Archive, Report 2012/422, 2012.

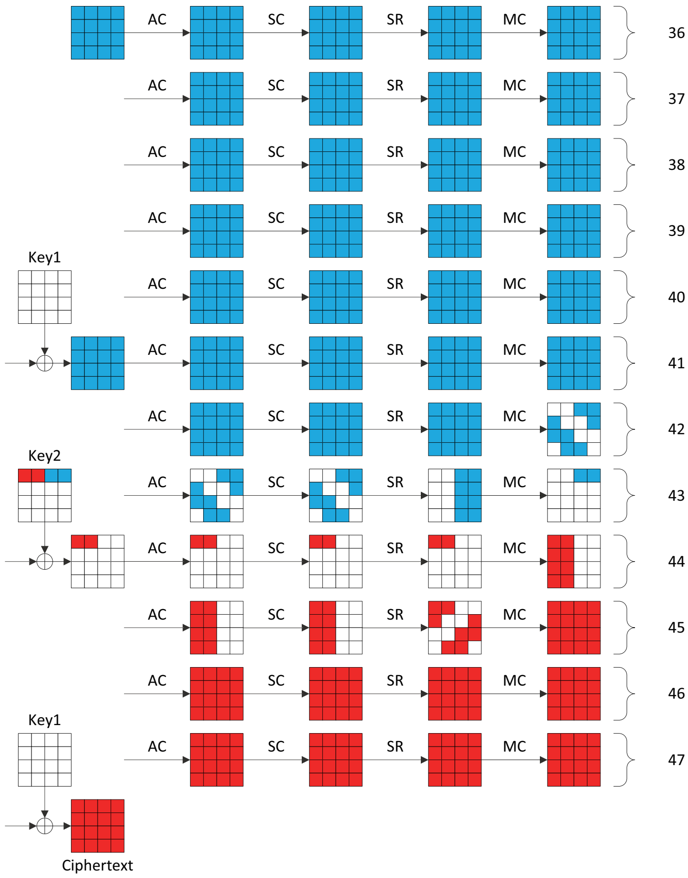

Fig. 16. 8-dimensional biclique for the full LED-80/96/128.

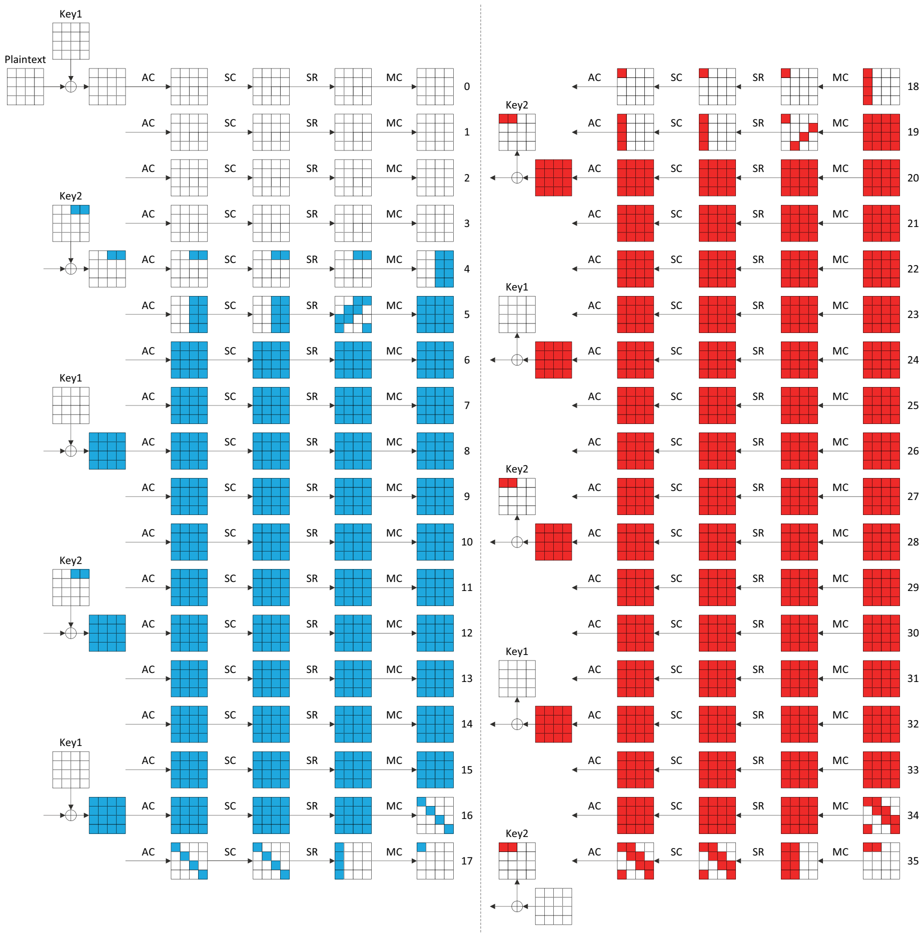

Fig. 17. Recomputations in forward and backward directions for the full LED-80/96/128.

6. J. Guo, T. Peyrin, A. Poschmann and M. Robshaw, *The LED Block Cipher*, CHES 2011, LNCS 6917, pp. 326–341, IACR, 2011.
7. D. Hong, B. Koo and D. Kwon, *Biclique Attack on the Full HIGHT*, ICISC 2011, LNCS 7259, pp. 365–374, Springer-Verlag, 2012.
8. D. Khovratovich, G. Leurent and C. Rechberger, *Narrow-Bicliques: Cryptanalysis of Full IDEA*, EUROCRYPT 2012, LNCS 7237, pp. 392–410, IACR, 2012.
9. K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita and T. Shirai, *Piccolo: An Ultra-Lightweight Blockcipher*, CHES 2011, LNCS 6917, pp. 342–357, IACR, 2011.
10. Y. Wang, W. Wu and X. Yu, *Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher*, ISPEC 2012, LNCS 7232, pp. 337–352, Springer-Verlag, 2012.