Biclique Cryptanalysis of the PRESENT and LED Lightweight Ciphers

Farzaneh Abed, Christian Forler, Eik List, Stefan Lucks, Jakob Wenzel

Bauhaus-University Weimar, Germany
{firstname.lastname}@uni-weimar.de

Abstract. In this paper, we propose the first full-round attacks on the PRESENT and LED lightweight ciphers. In our attacks, we use the independent-biclique approach which has been developed recently. The proposed attacks on PRESENT-80 and PRESENT-128 require 2^{60} and 2^{56} chosen plaintexts, and have time complexities of $2^{79.54}$ and $2^{127.42}$, respectively. Our attacks on LED-64 and LED-128 need 2^{56} and 2^{64} chosen plaintexts and the time complexities are equivalent to $2^{63.40}$ and $2^{127.25}$ encryptions.

Keywords: PRESENT, LED, lightweight block cipher, independent biclique, matching with precomputations

1 Introduction

Recently, the need for security constructions for limited devices like RFID tags, influenced the development of lightweight ciphers [3,22,4,9,4,12,10], which are efficient in hardware and require only small memory resources. At the time, 64 bits of key material are considered adequate to ensure sufficient security for limited devices or "very short-term protection against small organizations" [23]. Therefore, usual lightweight ciphers support key lengths of at least 64 bits.

Biclique cryptanalysis was originally introduced by Khovratovich et al. for preimage attacks on hash functions [16]. In [2], Bogdanov et al. demonstrated that biclique attacks can help to significantly reduce the effort for key-recovery attacks on block ciphers. In their work, the authors could construct the first attacks on full versions of the AES in the single-key model. Since then, biclique attacks have been applied on several further ciphers, including SQUARE [18], ARIA-256 [6], Piccolo [25], TWINE [5], and HIGHT [11]. In this paper, we describe our cryptanalytic results of biclique attacks on the PRESENT and LED lightweight ciphers

PRESENT was proposed by Bogdanov and Knudsen in 2007 [3]. The cipher supports two versions of 80 and 128 bits for the key, which we denote by PRESENT-80 and PRESENT-128. Because of its simple structure and the small S-box, the cipher has become a reference design in the field of lightweight ciphers. In 2008, Wang proposed a first differential attack on a reduced version with 16 out of 31 rounds with a time complexity of 2^{65} [24]. To the best of our knowledge, the most powerful attack on PRESENT is the work of Cho [7], which allows to distinguish 25 out of 28 rounds in PRESENT-80 from a random permutation. In the same work, Cho proposed an attack on 26 rounds which requires the entire codebook.

LED was designed by Jian et al. [10] in 2011 and supports key lengths of 64 and 128 bits. We denote the versions by LED-64 and LED-128, corresponding to the key lengths. In their security analysis, Jian et al. proposed two chosen-related-key attacks and they claimed, that the best probabilistic differential attacks on LED could only cover 15 out of 32 rounds of the 64-bit, and 27 out of 48 rounds of the 128-bit version. In [13], Isobe and Shibutani proposed a splice-and-cut

attack on eight rounds of LED-64 which requires 2^{56} encryptions using eight chosen plaintexts. They proposed another attack on 16 rounds of LED-128 which requires time equivalent to 2^{112} encryptions and 2^{16} chosen plaintexts. Recently, Mendel et al. published the best related-key attacks on up to 16 rounds for LED-64 and 24 rounds for LED-128 [19].

Our attacks cover full-round versions of PRESENT-80, PRESENT-128 and LED-128, and almost the full LED-64, except for the final key addition. Table 1 summarizes the previous attacks on PRESENT and LED and compares their complexities with our proposed attacks in this work.

We use an automation method to search for bicliques and an optimal matching. So, we could reduce the computational costs and achieve the best attack on PRESENT and LED ciphers at the time of writing this paper.

Cipher	Model	Rounds	Attack Type	Time	Data	Reference
PRESENT-80	SK	16/28	Differential	2^{65}	2^{64} CP	[24]
	SK	16/28	Differential + Algebraic	2^{85}	n.s.	[1]
	SK	24/28	Saturation	2^{57}	2^{57} CP	[8]
	SK	24/28	Linear	n. s.	$2^{63.5}$ KP	[20]
	SK	25/28	Linear	2^{65}	$2^{62.4}$ KP	[7]
	SK	26/28	Linear	2^{72}	2^{64} KP	[7]
	SK	**28 (full)**	**Biclique**	$\mathbf{2^{79.54}}$	$\mathbf{2^{60}}$ CP	**Section 4**
PRESENT-128	SK	19/28	Differential + Algebraic	2^{113}	n.s.	[1]
	SK	**31 (full)**	**Biclique**	$\mathbf{2^{127.42}}$	$\mathbf{2^{56}}$ CP	**Section 5**
LED-64	CRK	15/32	Rebound	2^{16}	2^{118} CP	[10]
	SK	8/32	Splice-and-cut	2^{56}	2^{8} CP	[13]
	RK	16/32	Differential	$2^{(n+\frac{1}{p})/2}$ (*)	2^{64} CP	[19]
	SK	**31.5/32**	**Biclique**	$\mathbf{2^{63.40}}$	$\mathbf{2^{56}}$ CP	**Section 6**
LED-128	CRK	27/48	Rebound	2^{16}	2^{118} CP	[10]
	SK	16/48	Splice-and-cut	2^{112}	2^{16} CP	[13]
	RK	24/48	Differential	$2^{3n/2}$ (*)	2^{64} CP	[19]
	SK	**48 (full)**	**Biclique**	$\mathbf{2^{127.25}}$	$\mathbf{2^{64}}$ CP	**Section 7**

Table 1: Attacks on PRESENT and LED (n.s.: not specified, CP: Chosen Plaintexts, KP: Known Plaintexts, RK: Related Keys, CRK: Chosen Related Keys, SK: Single Key, (*): n denotes a chosen number of key bits to recover in an attack).

1.1 Notation

- A: A bit string
- $A\|B$: A concatenation of A and B
- v_i: Internal states, indexed by i
- S_j: Internal states, indexed by j
- P_i: Plaintexts, indexed by i
- C_i: Ciphertexts, indexed by i. A ciphertext C_i corresponds to a plaintext P_i, i.e., $C_i = E_K(P_i)$
- E: Encryption process

– $K[i,j]$: Keys used in a biclique, indexed by i and j
– $\mathcal{B}$: A subcipher over which we construct a biclique; usually a few rounds
– Δ_i: Differences in the ending states of a biclique, indexed by i
– ∇_j: Differences in the starting states of a biclique, indexed by j
– Δ_i^K: Differences in the keys for forward computation of a biclique, indexed by i
– ∇_j^K: Differences in the keys for backward computation of a biclique, indexed by j

The organization of this paper is as follows. First, in Section 2, we give a brief summary of both lightweight ciphers to show how they work. In Section 3, we overview biclique cryptanalysis. Sections 4 to 7 describe our attacks including the biclique construction, the matching part and the resulting complexities in detail. Finally, we conclude our paper in Section 8.

2 Preliminaries

2.1 PRESENT

PRESENT has a state size of 64 bits and transforms the state in 31 rounds. After the final round, the state is XORed with an additional round key to generate the ciphertext. Each round has three operations: an XOR with the round key, a non-linear substitution layer and a permutation layer, as shown in Figure 1 (cf. [3]).

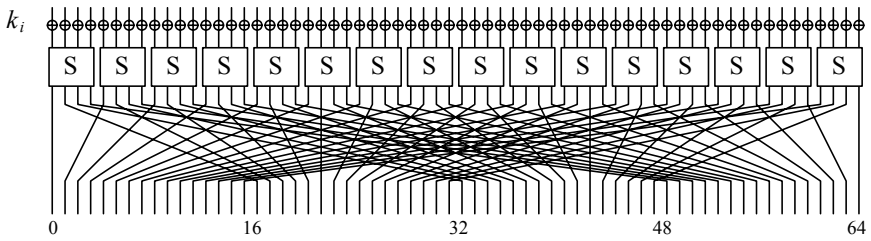

Fig. 1: Round structure of PRESENT.

The key schedule expands a secret key of 80 or 128 bits to 32 round keys with 64 bits each. The secret key of the 80-bit version is stored in a key register and represented by $k_{79}k_{78}...k_1k_0$, where k_i denotes the i-th bit in the register, and k_{79} is the leftmost bit. At round r, the key schedule uses the leftmost 64 bits of the key register as the round key. After extraction of the key, the register is transformed in three operations:

– $[k_{79}k_{78}\dots k_1k_0] = [k_{18}k_{17}\dots k_1k_0k_{79}k_{78}\dots k_{20}k_{19}]$
– $[k_{79}k_{78}k_{77}k_{76} = Sbox[k_{79}k_{78}k_{77}k_{76}]$
– $[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus r$

The key is rotated by 61 positions to the left, before the four leftmost bits are processed in the S-box, and a round counter r is XORed to five bits of the register. The key schedule for the 128-bit version works similar, but the S-box is used twice for the eight leftmost bits of the register. A more formal representation of the 128-bit transformation is given by:

– $[k_{127}k_{126}\ldots k_1k_0] = [k_{66}k_{65}\ldots k_1k_0k_{127}k_{126}\ldots k_{68}k_{67}]$
– $[k_{127}k_{126}k_{125}k_{124}] = Sbox[k_{127}k_{126}k_{125}k_{124}]$
– $[k_{123}k_{122}k_{121}k_{120}] = Sbox[k_{123}k_{122}k_{121}k_{120}]$
– $[k_{66}k_{65}k_{64}k_{63}k_{62}] = [k_{66}k_{65}k_{64}k_{63}k_{62}] \oplus r$

2.2 LED

LED has a state length of 64 bits and uses key lengths from 64 to 128 bits. The internal state is arranged in a 4×4-matrix, where each matrix cell represents a nibble. At the beginning, the state is initialized row-wise with the plaintext. The key K is also viewed nibble-wise and is loaded into one array K_1 or two arrays K_1 and K_2, depending on the key length. For the key lengths from 65 to 128 bits, the first 64 bits of the given key are used for K_1 and the remaining key is padded with zeroes to fill up K_2.

The encryption process of LED consists of two operations, `AddRoundKey` and `step`, as shown in Figure 2. The `step` operation has four rounds and each round includes `AddConstants`, `SubCells`, `ShiftRows`, and `MixColumnsSerial`:

$$\begin{aligned}\texttt{round} &= (\texttt{MixColumnsSerial} \circ \texttt{ShiftRows} \circ \texttt{SubCells} \circ \texttt{AddConstants}).\\ \texttt{step} &= (\texttt{round} \circ \texttt{round} \circ \texttt{round} \circ \texttt{round})\\ \texttt{LED} &= (\texttt{AddRoundKey} \circ \texttt{step} \circ \texttt{AddRoundKey} \circ \ldots \circ \texttt{AddRoundKey} \circ \texttt{step} \circ \texttt{AddRoundKey}).\end{aligned}$$

In the 64-bit version, only K_1 is used in each call of the `AddRoundKey` operation but in the 128-bit version, K_1 and K_2 are used alternately. For details on the individual operations, we refer to the original proposal [10].

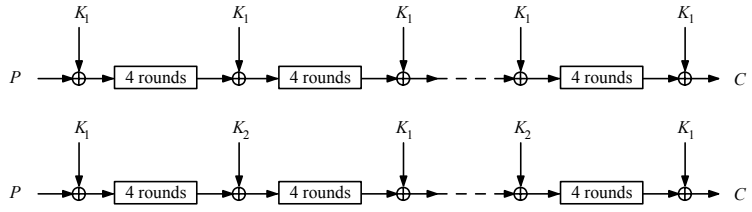

Fig. 2: Round structure of LED [10] with 64-bit key(top) or 128-bit key (bottom).

3 Biclique Cryptanalysis

In this section, we give an overview of biclique cryptanalysis proposed by Bogdanov et al. [2].

3.1 Definition

A biclique is a complete bipartite graph which covers some steps of a cipher, connecting every text in a set of starting states $\mathcal{S}$ with every text in a set of ending states $\mathcal{C}$. We represent the texts in $\mathcal{C}$ by C_i, and the texts in $\mathcal{S}$ by S_j ,where a path from S_j to C_i represents the encryption

under a key $K[i,j]$. The 3-tuple of sets $[\{S_j\},\{C_i\},\{K[i,j]\}]$ is called a *d-dimensional biclique*, if

$$\forall i,j \in \{0,\ldots,2^d-1\}: \quad S_j \xrightarrow[\mathcal{B}]{K[i,j]} C_i.$$

Figure 3 shows the schematic view of bicliques.

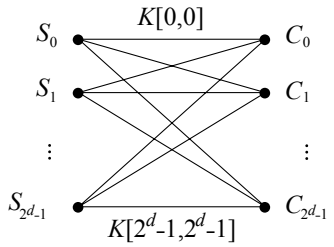

Fig. 3: Schematic view on bicliques.

A biclique defines a group of keys $K[i,j]$, in which every key can be represented relative to the base key of the group $K[0,0]$ and two differences Δ_i and ∇_j:

$$K[i,j] = K[0,0] \oplus \Delta_i \oplus \nabla_j.$$

In the beginning of an attack, the adversary divides the key space into 2^{k-2d} subspaces of 2^{2d} keys each, where k is the key size and d is the dimension of biclique. The cipher E is considered as a composition of three parts, $E = \mathcal{B} \circ E_2 \circ E_1$, where E_1 is the subcipher that maps a plaintext P to an internal state V, E_2 is the subcipher, that maps V to another internal state S, and $\mathcal{B}$ is the subcipher which maps the state S to the ciphertext C:

$$P \xrightarrow{E_1} V \xrightarrow{E_2} S \xrightarrow{\mathcal{B}} C.$$

The adversary can construct a biclique over the first or the last part of the cipher and use a meet-in-the-middle or similar attack for the remaining parts.

[16,2], and later [14] and [15] introduced different type of biclique attacks. This paper focuses on independent-biclique attacks which usually have a high dimension and a small number of rounds.

3.2 Independent Bicliques

Independent bicliques allow the construction of bicliques over some subcipher $\mathcal{B}$ from two differentials. First, we choose a base computation, that is a 3-tupel $\{S_0, C_0, K[0,0]\}$, where the key $K[0,0]$ maps the internal state S_0 to the ciphertext C_0 over $\mathcal{B}$:

$$S_0 \xrightarrow[\mathcal{B}]{K[0,0]} C_0.$$

Then, we choose 2^d Δ_i-differentials in the forward direction, which connect the state S_0 with ciphertexts C_i as follows:

$$S_0 \xrightarrow[\mathcal{B}]{K[0,0]\oplus\Delta_i^K} C_0 \oplus \Delta_i = C_i.$$

then, we choose 2^d ∇_j -differentials in backward direction, which connect the ciphertext C_0 with the states S_j as follows:

$$S_j = S_0 \oplus \nabla_j \xleftarrow[\mathcal{B}^{-1}]{K[0,0]\oplus\nabla_j^K} C_0.$$

The trails of the base computation conform to both differentials. If the trails of the Δ_i-differentials *do not share any active non-linear operations* with the ∇_j-differentials, then each of the 2^d input differences ∇_j can connect with each of the 2^d output differences Δ_i, by applying both differences to the key: $K[0,0] \oplus \Delta_i^k \oplus \nabla_j^k$. Therefore, we obtain a set of 2^{2d} *independent* (Δ_i,∇_j)-*differential trails*:

$$S_0 \oplus \nabla_j \xrightarrow[\mathcal{B}]{K[0,0]\oplus\Delta_i^K\oplus\nabla_j^K} C_0 \oplus \Delta_i \qquad \forall i,j \in \{0,\ldots,2^d-1\}.$$

A single biclique of dimension d allows to test 2^{2d} keys with 2^d computations in each direction. If the full candidate space can be partitioned into groups of 2^{2d} keys, then the complexity of an attack can be reduced to about 2^{n-2d} computations of E in addition to the effort to construct the biclique and the effort for the matching. According to [2], for the independent-biclique approach, the length of differentials is limited by the number of steps for one full diffusion of the cipher. Thus, when the biclique is quite short and too many rounds need to be covered, matching with precomputations method can be used instead of a meet-in-the middle attack.

3.3 Matching with Precomputations

In [16], Khovratovich et al. introduced matching-with-precomputations as an efficient approach to perform a matching on those parts which are not covered by the biclique.

For this approach, an adversary first chooses an internal state V between E_1 and E_2. Secondly, it computes and stores 2^d values $\overrightarrow{v_{i,0}}$ in forward direction from the plaintexts to V

$$P_i \xrightarrow[E_1]{K[i,0]} \overrightarrow{v_{i,0}},$$

and 2^d values $\overleftarrow{v_{0,j}}$ in backward direction from each of the starting states S_j:

$$\overleftarrow{v_{0,j}} \xleftarrow[E_2^{-1}]{K[0,j]} S_j.$$

In the end, the adversary re-uses the stored values for the remaining $2^{2d} - 2^d$ computations

$$P_i \xrightarrow[E_1]{K[i,j]} \overrightarrow{v_{i,j}}, \quad \text{and} \quad \overleftarrow{v_{i,j}} \xleftarrow[E_2^{-1}]{K[i,j]} S_j,$$

where it needs to recompute only those parts of the key schedule and the round transformation that differ from the stored values. By using this method, we could reduce the computational effort significantly. Further reduction is possible by using partial matching.

3.4 Complexity Calculations

The independent-biclique approach does not try to create the longest possible biclique. If the biclique is short and many rounds need to be covered, the adversary applies a matching based on precomputations. The adversary precomputes and stores 2^d values of $\overrightarrow{v_{i,0}}$ for $K[i,0]$ and $\overleftarrow{v_{0,j}}$ for $K[0,j]$, respectively. For all values of $K[i,j]$, only those parts of the key schedule and the cipher are recomputed which differ from the stored values. More positively, the adversary needs to have only access to either an encryption or a decryption oracle.

For every biclique, the adversary tests 2^{2d} keys with $2 \cdot 2^d$ computations. Hence, the effort for one such set of keys is upper bounded by 2^d computations of E. Therefore, the adversary needs to construct 2^{n-2d} bicliques to cover the full key space. Then, the full complexity is given as follows:

$$C_{full} = 2^{n-2d} \left(C_{biclique} + C_{decrypt} + C_{precompute} + C_{recompute} + C_{falsepos} \right),$$

where

- $C_{biclique}$ denotes the costs for constructing a biclique,
- $C_{decrypt}$ is the complexity of the oracle to decrypt 2^d ciphertexts,
- $C_{precompute}$ denotes the costs for the computation of v for 2^d computations of $E_2 \circ E_1$,
- $C_{recompute}$ is the complexity of recomputing 2^{2d} values $v_{i,j}$ in both directions,
- $C_{falsepos}$ is the complexity to eliminate false positives.

The complexity is dominated by the recomputations cost, $C_{recompute}$. The memory requirements are upper bounded by storing 2^d values of the intermediate states $v_{i,j}$.

4 New Independent-Biclique Attack on PRESENT-80

In this section, we describe the independent-biclique attack on the full PRESENT-80. The attack consists of three steps: partitioning the key space, constructing a biclique, and matching over the remaining rounds. As the final step, we show the complexity of the attack.

4.1 Key Space Partitioning

The 80-bit key space is divided into groups of 2^{16} keys each with respect to the secret key. The base keys $K[0,0]$ are all 80-bit secret keys with 16 bits fixed to 0, where the remaining 64 bits can take on all other possible values.

The 2^{16} keys in a group of $\{K[i,j]\}$ are defined relative to the base key $K[0,0]$ and two differences Δ_i^K and ∇_j^K, where $i,j \in \{0,\ldots,255\}$. Δ_i^K and ∇_j^K are used to manipulate eight bits ($k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0$) in forward direction, and other eight bits (k_{64},k_{63},k_{62},k_{61},k_{60},k_{59}, k_{58}, k_{57}) in backward direction to construct the bicliques. Therefore, in total, we have 2^{64} key groups.

$$\begin{aligned} K[0,0] &= (k_{79}, k_{78}, \ldots, k_{66}, k_{65}, 0, \ldots, 0, k_{56}, k_{55}, \ldots, k_9, k_8, 0, \ldots, 0), \\ \Delta_i^K &= (0, \ldots, 0, k_7, k_6, k_5, k_4, k_3, k_2, k_1, k_0), \\ \nabla_j^K &= (0, \ldots, 0, k_{64}, k_{63}, k_{62}, k_{61}, k_{60}, k_{59}, k_{58}, k_{57}, 0, \ldots, 0). \end{aligned}$$

4.2 4-Round Biclique of Dimension 8

We can construct a biclique over four rounds in the beginning of the cipher rather than three rounds at the end of the cipher. Otherwise, the key addition after the last round would cost a round. The resulting biclique transforms the plaintexts P_j to states S_i:

$$P_j \xrightarrow[\mathcal{B}]{K[i,j]} S_i.$$

The biclique construction is illustrated in Figure 4.

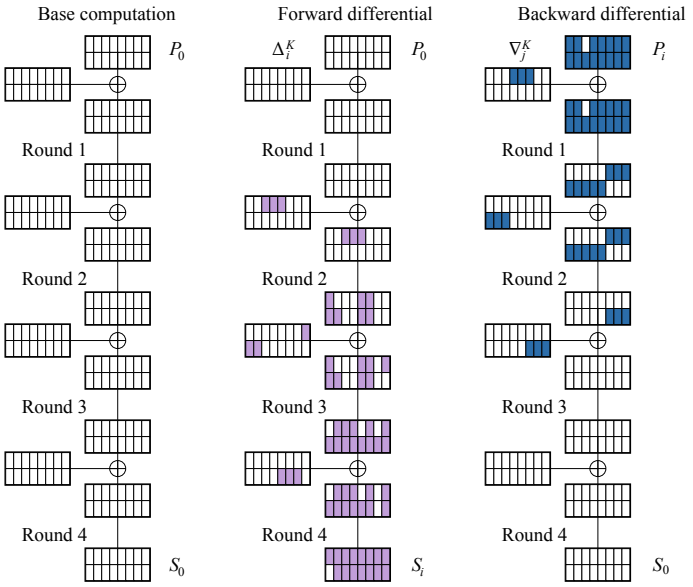

Fig. 4: Biclique for PRESENT-80 in rounds 1 - 4 with Δ_i- and ∇_j-differentials.

4.3 Matching over 27 Rounds

Figure 5 shows the matching part from rounds 5-7 in forward direction and 7-31 in backward direction. We match in two nibbles which consists of the eight bits $v_{23}v_{22}\dots v_{17}v_{16}$ of the state v after Round 7, where $v = v_{63}v_{62}\dots v_1v_0$. Figure 5 shows $3+8+8 = 19$ active nibbles in forward direction and $2+4+21\cdot 16+15 = 357$ active nibbles in backward direction which need to be recomputed.

For the key schedule, we concentrate on the S-box calls which need to be recomputed. PRESENT-80 uses the S-box for the leftmost four bits in each round key. We employ the S-box in the rounds 10, 14 and 31. Therefore, $19 + 357 + 3 = 379$ nibbles need to be considered in all steps of the matching part.

4.4 The Complexity of the Attack

PRESENT can be seen either as a bit-wise-operating cipher due to its diffusion layer, or as a nibble-wise-operating cipher due to its S-box. To calculate the complexity of our attacks on PRESENT, we consider it as a nibble-wise operating cipher and approximate the effort by counting the amount of nibbles which need to be recomputed in the matching part. In addition,

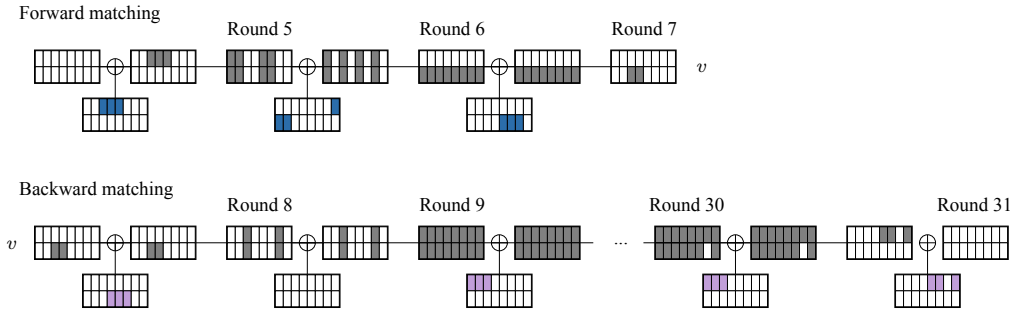

Fig. 5: Recomputations for PRESENT-80 in forward and backward direction.

we need to include the costs for computing the changes in the round keys $K[i,j]$ in the key schedule of PRESENT. There, we consider the S-box calls which need to be recomputed.

PRESENT-80 transforms the 16 nibbles of the state in each of its 31 rounds and updates 62 nibbles using the S-box in the key schedule. So, the number of nibble-wise operations sums up to $31 \cdot 16 + 31 = 527$. Since we test 2^{16} keys in a group, the recomputation complexity for one such group is $C_{recomp} = 2^{16} \cdot \frac{379}{527} = 2^{15.51}$ full encryptions. Since we match in eight bits in v, we can approximate the complexity to test false positives $C_{falsepos}$ by 2^8. Therefore, the full computational complexity is

$$2^{64} \cdot (2^{6.05} + 2^{7.8} + 2^{15.51} + 2^8 + 2^8) = 2^{79.54}.$$

The data complexity is 2^{60}, and we need to store 2^8 texts.

We can have an advantage of one power of two compared to exhaustive search by mounting a similar attack on PRESENT-80 with only the first 19 out of 28 rounds. Then, by using the same biclique structure, the time complexity for this attack on round-reduced PRESENT-80 is

$$2^{64} \cdot (2^{6.05} + 2^{7.8} + 2^{15} + 2^8 + 2^8) = 2^{79.04}.$$

5 New Independent-Biclique Attack on PRESENT-128

This section describes the attack on the full PRESENT-128.

5.1 Key Space Partitioning

The 128-bit key space is divided into groups of 2^{16} keys each with respect to the secret key. The base keys $K[0,0]$ are all 128-bit secret keys with 16 bits fixed to 0, where the remaining 112 bits can take on all other possible values.

The 2^{16} keys in a group of $\{K[i,j]\}$ are defined based on the base key $K[0,0]$ and two differences Δ_i^K and ∇_j^K, where $i,j \in \{0,\ldots,255\}$. Δ_i^K and ∇_j^K are used to manipulate eight bits ($k_{19}, k_{18}, k_{17}, k_{16}, k_{11}$, k_{10}, k_9, k_8) in forward direction, and other eight bits (k_{23},k_{22},k_{21},k_{20},k_{15}, k_{14}, k_{13}, k_{12}) in backward direction to construct the bicliques. So, we have 2^{112} key groups in total.

$$\begin{aligned} K[0,0] &= (k_{79}, k_{78}, \ldots, k_{25}, k_{24}, 0, \ldots, 0, k_7, k_6, \ldots, k_0), \\ \Delta_i^K &= (0, \ldots, 0, k_{19}, k_{18}, k_{17}, k_{16}, 0, \ldots, 0, k_{11}, k_{10}, k_9, k_8, 0, \ldots, 0), \\ \nabla_j^K &= (0, \ldots, 0, k_{23}, k_{22}, k_{21}, k_{20}, 0, \ldots, 0, k_{15}, k_{14}, k_{13}, k_{12}, 0, \ldots, 0). \end{aligned}$$

5.2 5-Round Biclique of Dimension 8

Similar to the attack on the 80-bit version, we construct a biclique at the beginning of the cipher. Here, we can construct a biclique over the first five rounds, which is one round more than in the attack on PRESENT-80. This aspect can be attributed to the longer (128-bit) key register and the rotation by 61 bits in each round in the key schedule. As a result, the round keys for Round 3 do not inject differences in neither the Δ_i- nor the ∇_j-differentials. Figure 6 shows the biclique construction in rounds 1-5. As we can see, the plaintexts P_j are active in 14 out of 16 nibbles. Thus, the data complexity over all key groups does not exceed 2^{56} chosen plaintexts.

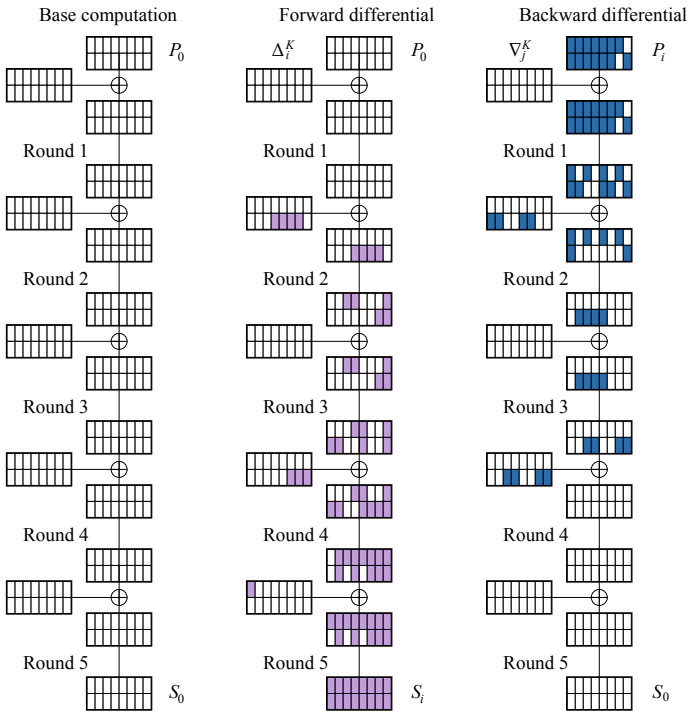

Fig. 6: Biclique for PRESENT-128 in rounds 1 - 5 with Δ_i- and ∇_j-differentials.

5.3 Matching over 26 Rounds

We match in two nibbles in the state v after Round 14. As we can see from Figure 7, we need to recompute $3+10+5\cdot 16+12+8 = 113$ nibbles from the plaintexts P_j to the matching states $v_{i,j}$ in forward direction, and $2+4+14\cdot 16+14 = 244$ nibbles in backward direction.

PRESENT-128 uses the S-box for the leftmost eight bits in each round key. Thus, we need to update one nibble in each of the rounds 9, 11, 12, 26, and 30, and two nibbles in the rounds 7 and 28. In total, we recompute $113 + 244 + 9 = 366$ nibbles in the attack.

5.4 The Complexity of the Attack

PRESENT-128 transforms the 16 nibbles of the state in each of its 31 rounds and updates 62 nibbles using the S-box in the key schedule. This sums up to $31\cdot 16+62 = 558$ nibbles in total. Thus, the recomputation complexity is equal to $C_{recomp} = 2^{16}\cdot\frac{366}{558} = 2^{15.39}$. Since we match in

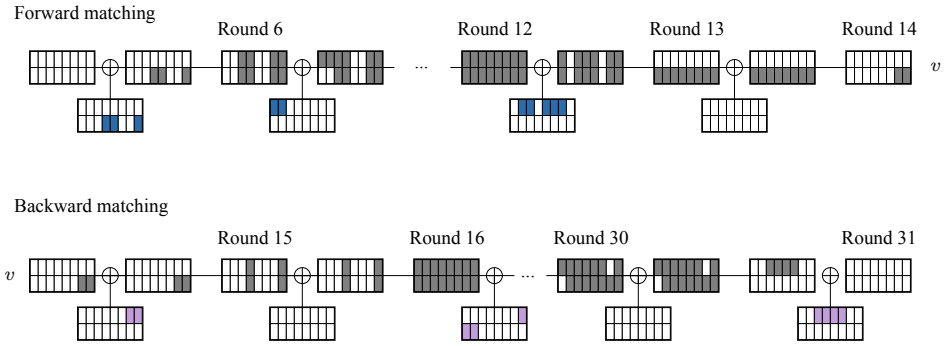

Fig. 7: Recomputations for PRESENT-128 in forward and backward direction.

eight bits of v, the complexity to test false positives $C_{falsepos}$ is approximated by 2^8. The full computational complexity is equivalent to

$$2^{112} \cdot (2^{6.37} + 2^{7.75} + 2^{15.39} + 2^8 + 2^8) = 2^{127.42}.$$

The data complexity is 2^{56}, and we need to store 2^8 texts per group.

To obtain a better advantage, we can mount an attack on a reduced version of the first 20 out of 31 rounds. Using the same biclique structure, the time complexity is then given by

$$2^{112} \cdot (2^{6.37} + 2^{7.75} + 2^{14.86} + 2^8 + 2^8) = 2^{126.9}.$$

6 New Independent-Biclique Attack on 31.5 Rounds of LED-64

This part includes an explanation of our attack on 31.5 out of 32 rounds of LED-64. The attack has three steps: key partitioning, biclique construction, and a matching part. The complexity of the attack comes at the end.

6.1 Key Space Partitioning

In this part, we choose the dimension of 8 for our biclique attack. The key space is partitioned into 2^{48} groups of 2^{16} keys each. The base keys $K[0,0]$ are all 16-nibble values K_1 with four nibbles fixed to 0 and all other nibbles running over all possible values. The keys in a group $K[i,j]$ are enumerated by all possible differences $i = (i_1 \| i_2)$ and $j = (j_1 \| j_2)$ with respect to $K[0,0]$, as shown in Figure 8.

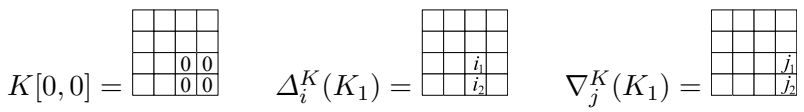

Fig. 8: Key differences used in the biclique for LED-64.

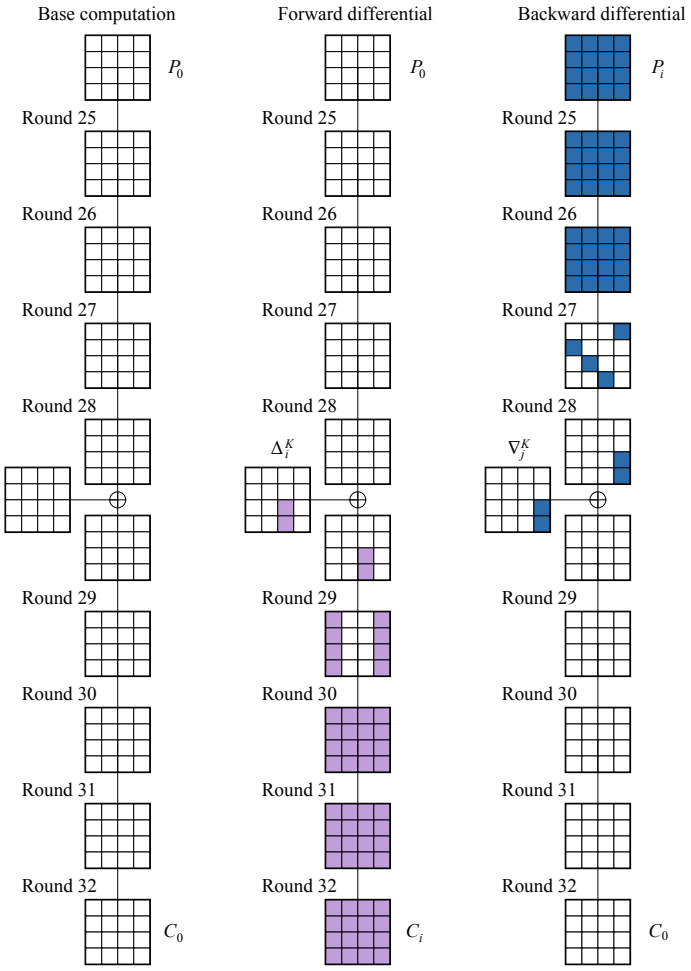

Fig. 9: Biclique for LED-64 in rounds 25 - 32 with Δ_i- and ∇_j-differentials.

6.2 7.5-Round Biclique of Dimension 8

Since LED omits any key schedule, every key injection will affect the state. Moreover, LED possesses a very fast diffusion which means a difference with only a single active nibble propagates to a full active state after only two rounds. As a consequence, the Δ_i- and ∇_j-differentials can not share a sequence of four consecutive rounds where they have both active differences. Thus, the length of bicliques in LED-64 is very limited. Furthermore, the key injection after the last round leads to the active ∇_j-trail after two rounds. Thus, a biclique which includes the final key addition can maximally cover four rounds. The same situation applies for a biclique at the beginning of the cipher. Figure 9 visualizes the biclique over almost eight rounds, from 25 to 32, without the final key addition.

6.3 Matching over 24 Rounds

In the matching phase, we check if the secret key K_{secret} belongs to the group defined by $K[i, j]$. We locate the matching state v at Round 3 and concentrate on two nibbles, as shown in Figure 10. There are $2 + 8 + 4 = 14$ S-boxes in forward direction and $2 + 10 + 18 \cdot 16 + 4 = 304$ S-boxes

in backward direction which need to be recomputed, which sum up to a total of 318 S-boxes in the matching phase.

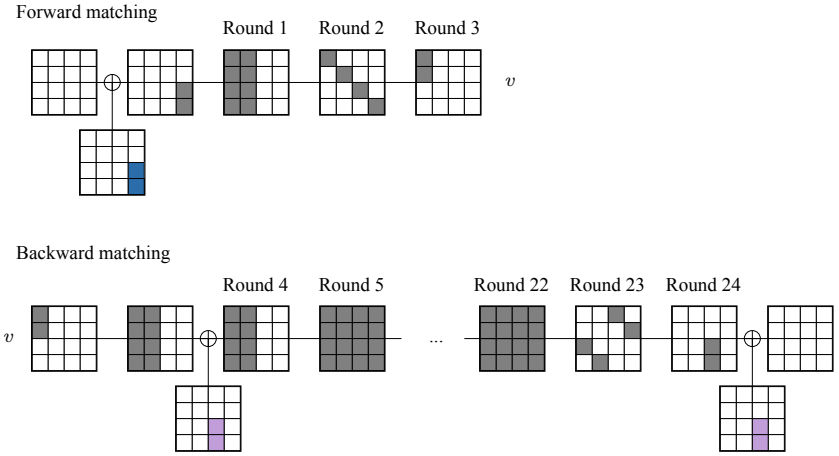

Fig. 10: Recomputations for LED-64 in forward and backward direction.

6.4 The Complexity of the Attack

The structure of LED is quite similar to that of the AES. In its round transformation, LED uses XORs with round constants, XORs with the key, S-boxes, row shifts and column-wise multiplications. In our attacks, we have only a negligible number of XORs in the `AddRoundKey` or `AddConstants` operations. Further, there are no extra computational costs for the `ShiftRow` operations. Thus, we only need to concentrate on the number of `MixColumnsSerial` and `SubCell` operations which are affected by the key differences and required to recompute for the matching part. In our attacks, the number of S-box calls is the larger summand in the total complexity compared to the mixing operations. Thus, we only count the number of S-boxes which need to be recomputed in the matching part as an approximation for the total effort.

There are $32 \cdot 16 = 512$ S-boxes in the full 32 rounds of the cipher, so, for 2^{16} keys in one group, C_{recomp} is equivalent to approximately $2^{16} \cdot \frac{318}{512} \approx 15.37$ encryptions. $C_{biclique}$ and $C_{precomp}$ sum up to 2^8 encryptions. Since we match in eight bits in the state v, we expect $C_{falsepos}$ to be 2^8 in average. $C_{decrypt}$ requires 2^8 decryptions. The full computational complexity is then given by

$$2^{48} \cdot (2^8 + 2^{15.37} + 2^8 + 2^8) = 2^{63.40}.$$

In the Δ_i-differentials, the end states C_i are fully active. Yet, we only create 2^8 ciphertexts for every of our 2^{48} bicliques. Thus, the data complexity is upper bounded by 2^{56} ciphertexts and the memory complexity by 2^8.

We can decrease the time complexity if we mount an attack on a round-reduced version of the cipher. Using the same biclique structure, an attack on a version of LED-64 with 27.5 rounds, reduced to the rounds 5-32 without the final key addition, has a time complexity of

$$2^{48} \cdot (2^{6.81} + 2^{7.64} + 2^{14.86} + 2^8 + 2^8) = 2^{62.9}.$$

7 New Independent-Biclique Attack on Full LED-128

In this part, we describe our results for an independent-biclique attack on the full 128-bit version of LED.

7.1 Key Space Partitioning

We divide the key space into 2^{112} groups. The base keys $K[0,0]$ are all 2^{112} values $(K_1\|K_2)$ with four nibbles fixed to 0 and all other nibbles can take any other possible values. With respect to $K[0,0]$, the keys in a group $\{K[i,j]\}$ are enumerated by all possible differences $i = (i_1\|i_2)$ and $j = (j_1\|j_2)$. Therefore, the key space is divided into 2^{112} groups of 2^{16} keys each, as shown in Figure 11.

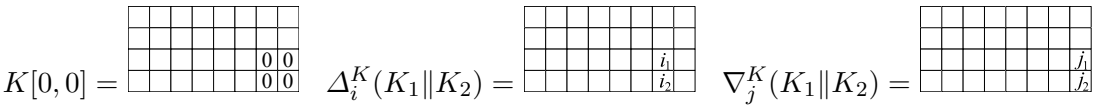

Fig. 11: Key differences used in the biclique for LED-128.

7.2 12-Round-Biclique of Dimension 8

Applying two key words alternately is a positive aspect in the 128-bit version of LED. Therefore, we can extend the length of our biclique by injecting differences only in one 64-bit word of the key, for instance, in K_2. Then, since K_1 does not inject any difference, and we can construct a biclique over 12 rounds, from rounds 37 - 48. Figure 12 visualizes the construction of the biclique.

7.3 Matching over 36 Rounds

We locate the matching state v after Round 7 and then, match on two nibbles. Figure 13 depicts these nibbles. Because of the fast diffusion of LED, large parts of the cipher need to be recomputed in the matching phase. Since the first four rounds are not affected by K_2, we have to recompute only $2 + 8 + 4 = 14$ S-boxes in forward direction and $2 + 4 + 26 \cdot 16 + 4 = 426$ S-boxes in backward direction, which sum up to 440 S-boxes in the 36 matching rounds.

7.4 The Complexity of the Attack

There are $48 \cdot 16 = 768$ S-boxes in the full 48 rounds of the cipher, so, for 2^{16} keys in one group, C_{recomp} is equal to $2^{16} \cdot \frac{440}{768} \approx 2^{15.22}$ full encryptions. $C_{biclique}$ and $C_{precomp}$ sum up to an equivalent of 2^8 full encryptions, and since we match in eight bits at v, the average effort for $C_{falsepos}$ is approximated by 2^8. $C_{decrypt}$ is equivalent to 2^8 decryptions. The full computational complexity is given by

$$2^{112} \cdot (2^8 + 2^{15.22} + 2^8 + 2^8) = 2^{127.25}.$$

The data complexity is 2^{64} and the memory requirement is 2^8.

To obtain a higher advantage, we can regard a version of LED-128 with 44 rounds, reduced to the rounds 5-48. Using the same biclique structure, the time complexity is then given by

$$2^{112} \cdot (2^7 + 2^{7.58} + 2^{14.88} + 2^8 + 2^8) = 2^{126.92}.$$

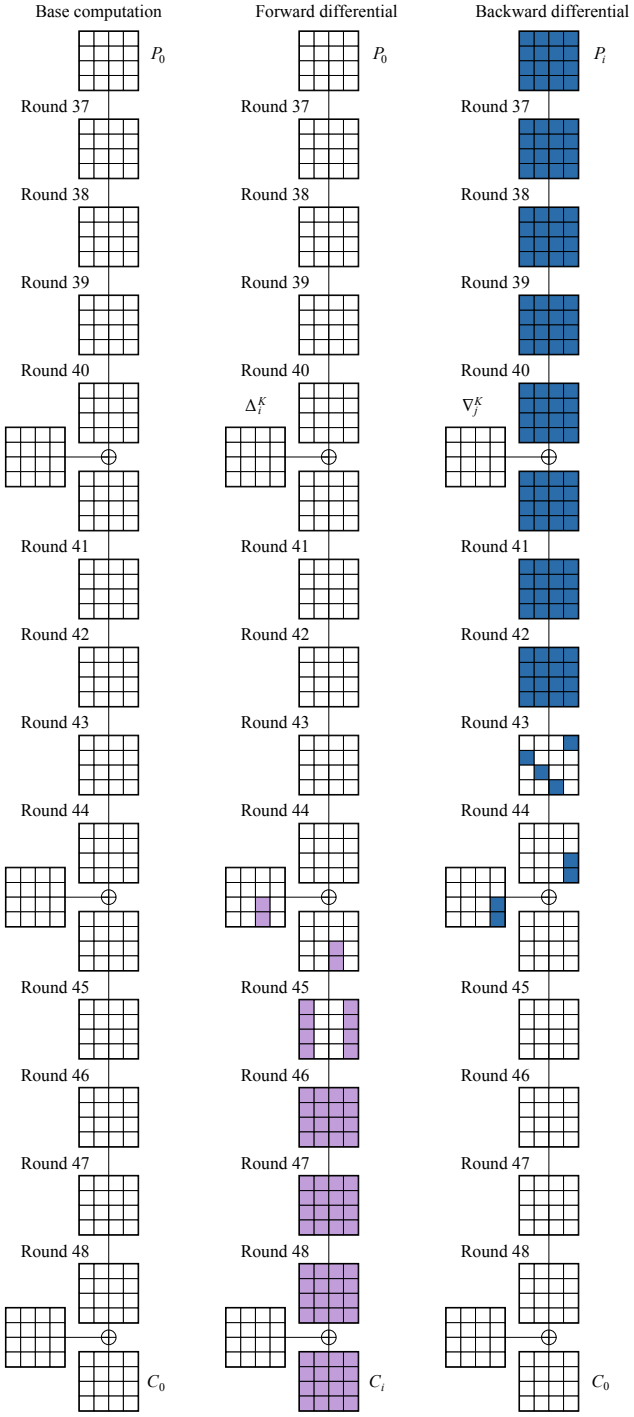

Fig. 12: Biclique for LED-128 in rounds 37 - 48 with Δ_i- and ∇_j-differentials.

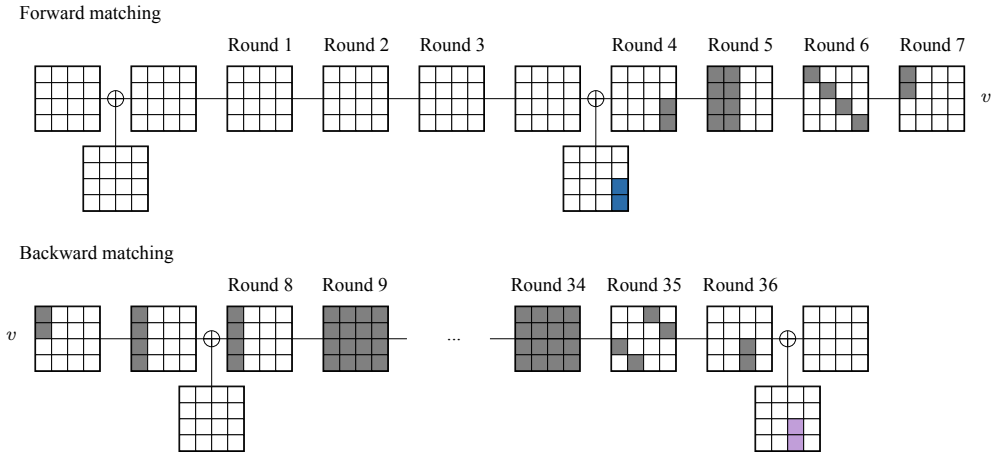

Fig. 13: Recomputations for LED-128 in forward and backward direction.

8 Conclusion

In this paper, we showed the first full-round biclique attacks on PRESENT and LED in the single-key model, using the approach of independent-bicliques. In our attack on PRESENT-80, the time is equivalent to $2^{79.54}$ full encryptions and the data complexity is 2^{60} chosen plaintexts. For the attack on PRESENT-128, we need time complexity of $2^{127.42}$ and at most 2^{56} chosen plaintexts for the data. For LED-64, the time and data complexities are $2^{63.40}$ and 2^{56}, respectively. The attack on LED-128 has the time complexity of $2^{127.25}$ and the data complexity of 2^{64}. In all attacks, we have used bicliques of dimension 8. Thus, the memory complexity is always given by 2^8.

As we can see from our work, biclique attacks have the potential to significantly reduce the computational complexity of an exhaustive key search. Moreover, biclique cryptanalysis is a powerful generic technique, which can be applied to a variety of ciphers in a comparable way.

References

1. Martin Albrecht and Carlos Cid. Algebraic Techniques in Differential Cryptanalysis. Cryptology ePrint Archive, Report 2008/177, 2008. http://eprint.iacr.org/.
2. Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. Cryptology ePrint Archive, Report 2011/449, 2011. http://eprint.iacr.org/.
3. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: An Ultra-Lightweight Block Cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *CHES*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.
4. Christophe De Cannière, Orr Dunkelman, and Miroslav Knezevic. KATAN and KTANTAN - A Family of Small and Efficient Hardware-Oriented Block Ciphers. In Christophe Clavier and Kris Gaj, editors, *CHES*, volume 5747 of *Lecture Notes in Computer Science*, pages 272–288. Springer, 2009.
5. Mustafa Çoban, Ferhat Karakoç, and Özkan Boztaş. Biclique Cryptanalysis of TWINE. Cryptology ePrint Archive, Report 2012/422, 2012. http://eprint.iacr.org/.
6. Shaozhen Chen and Tianmin Xu. Biclique Attack of the Full ARIA-256. *IACR Cryptology ePrint Archive*, 2012:11, 2012.
7. Joo Yeon Cho. Linear Cryptanalysis of Reduced-Round PRESENT. In Josef Pieprzyk, editor, *CT-RSA*, volume 5985 of *Lecture Notes in Computer Science*, pages 302–317. Springer, 2010.

8. Baudoin Collard and François-Xavier Standaert. A Statistical Saturation Attack against the Block Cipher PRESENT. In Marc Fischlin, editor, *CT-RSA*, volume 5473 of *Lecture Notes in Computer Science*, pages 195–210. Springer, 2009.
9. Zheng Gong, Svetla Nikova, and Yee Wei Law. KLEIN: A New Family of Lightweight Block Ciphers. In Ari Juels and Christof Paar, editors, *RFIDSec*, volume 7055 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.
10. Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In Preneel and Takagi [21], pages 326–341.
11. Deukjo Hong, Bonwook Koo, and Daesung Kwon. Biclique Attack on the Full HIGHT. In Howon Kim, editor, *ICISC*, volume 7259 of *Lecture Notes in Computer Science*, pages 365–374. Springer, 2011.
12. Deukjo Hong, Jaechul Sung, Seokhie Hong, Jongin Lim, Sangjin Lee, Bonseok Koo, Changhoon Lee, Donghoon Chang, Jaesang Lee, Kitae Jeong, Hyun Kim, Jongsung Kim, and Seongtaek Chee. Hight: A new block cipher suitable for low-resource device. In Louis Goubin and Mitsuru Matsui, editors, *CHES*, volume 4249 of *Lecture Notes in Computer Science*, pages 46–59. Springer, 2006.
13. Takanori Isobe and Kyoji Shibutani. Security Analysis of the Lightweight Block Ciphers XTEA, LED and Piccolo. In Willy Susilo, Yi Mu, and Jennifer Seberry, editors, *ACISP*, volume 7372 of *Lecture Notes in Computer Science*, pages 71–86. Springer, 2012.
14. Dmitry Khovratovich. Bicliques for Permutations: Collision and Preimage Attacks in Stronger Settings. Cryptology ePrint Archive, Report 2012/141, 2012. http://eprint.iacr.org/.
15. Dmitry Khovratovich, Gaëtan Leurent, and Christian Rechberger. Narrow-Bicliques: Cryptanalysis of Full IDEA. In *EUROCRYPT*, pages 392–410, 2012.
16. Dmitry Khovratovich, Christian Rechberger, and Alexandra Savelieva. Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 Family. Cryptology ePrint Archive, Report 2011/286, 2011. http://eprint.iacr.org/.
17. Manoj Kumar, Pratibha Yadav, and Meena Kumari. Flaws in differential cryptanalysis of reduced round present. Cryptology ePrint Archive, Report 2010/407, 2010. http://eprint.iacr.org/.
18. Hamid Mala. Biclique Cryptanalysis of the Block Cipher SQUARE. Cryptology ePrint Archive, Report 2011/500, 2011. http://eprint.iacr.org/.
19. Florian Mendel, Vincent Rijmen, Deniz Toz, and Kerem Varici. Differential Analysis of the LED Block Cipher. *IACR Cryptology ePrint Archive*, 2012:544, 2012.
20. Kenji Ohkuma. Weak Keys of Reduced-Round PRESENT for Linear Cryptanalysis. In Michael J. Jacobson Jr., Vincent Rijmen, and Reihaneh Safavi-Naini, editors, *Selected Areas in Cryptography*, volume 5867 of *Lecture Notes in Computer Science*, pages 249–265. Springer, 2009.
21. Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, volume 6917 of *Lecture Notes in Computer Science*. Springer, 2011.
22. Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In Preneel and Takagi [21], pages 342–357.
23. Nigel Smart. ECRYPT II Yearly Report on Algorithms and Keysizes (2010-2011), D. SPA.17 Rev. 1.0, ICT-2007-216676 (2011). Technical report, European Network of Excellence in Cryptology II, June 2011. www.ecrypt.eu.org/documents/D.SPA.17.pdf.
24. Meiqin Wang. Differential Cryptanalysis of Reduced-Round PRESENT. In Serge Vaudenay, editor, *AFRICACRYPT*, volume 5023 of *Lecture Notes in Computer Science*, pages 40–49. Springer, 2008.
25. Yanfeng Wang, Wenling Wu, and Xiaoli Yu. Biclique Cryptanalysis of Reduced-Round Piccolo Block Cipher. In Mark Dermot Ryan, Ben Smyth, and Guilin Wang, editors, *ISPEC*, volume 7232 of *Lecture Notes in Computer Science*, pages 337–352. Springer, 2012.