Biclique Cryptanalysis of the PRESENT and LED Lightweight Ciphers

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

Bauhaus-University Weimar, Germany
{farzaneh.abed,christian.forler,eik.list,stefan.lucks,jakob.wenzel}@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^{44} chosen plaintexts, and have time complexities of $2^{79.46}$ and $2^{127.37}$ respectively. Our attacks on LED-64 and LED-128 need 2^{56} and 2^{64} chosen plaintexts and the time complexities are equivalent to $2^{63.34}$ and $2^{127.23}$ 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 [4,21,5,10,5,13,11], which are efficient in hardware and require only small memory resources. At the time, 64 bits of key material are considered adequate to ensure a sufficient short-term security for limited devices [22]. Therefore, usual lightweight ciphers support key lengths between 64 and 128 bits.

Biclique cryptanalysis is a generic technique that was introduced by Khovratovich *et al.* in 2011 [16] for preimage attacks on the hash functions Skein and SHA-2. The approach is an extension for the *splice-and-cut* framework by Aoki and Sasaki [2], which itself is a variant of meet-in-the-middle attacks.

While bicliques had been introduced for the analysis of block-cipher-based hash functions, it seemed obvious to adapt the general approach also for key-recovery attacks on block ciphers. In [3], 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 constructed the first attacks on the full versions of the AES in the single-key model. Since then, biclique attacks have been successfully applied on several ciphers, including SQUARE [17], ARIA-256 [7], Piccolo [24], TWINE [6], and HIGHT [12], and have become a well-understood generic principle. In this paper, we describe our cryptanalytic results of biclique attacks on the PRESENT and LED lightweight ciphers.

PRESENT is a lightweight cipher which was proposed by Bogdanov and Knudsen in 2007 [4], and supports the key lengths of 80 and 128 bits, which are denoted by PRESENT-80 and PRESENT-128. The cipher has become a reference design in the field of lightweight ciphers, because of its simple structure and its S-box characteristic. In 2008, Wang proposed a first differential attack with a time complexity of 2^{65} on a reduced version with 16 rounds [23]. In addition, Wang showed a differential and algebraic attack with time complexity of 2^{113} for 19 rounds. To the best of our knowledge, the most powerful attack on PRESENT is the work of Cho [8], which allows to distinguish 25 rounds out of 28 in PRESENT-80 (or out of 31 rounds in PRESENT-128) from a random permutation. Moreover, in the same work, an attack on 26 rounds was proposed which requires the entire codebook.

LED was designed by Jian *et al.* in 2011 [11] and supports key lengths between 64 and 128 bits. We denote the two most relevant versions by LED-64 and LED-128, corresponding to their key lengths. In their security analysis, Jian *et al.* proposed two related-key attacks from which they argued, 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 [14], 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.* analyzed differential properties of LED and published related-key attacks on up to half of the ciphers [18].
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 previous attacks on PRESENT and LED and compares their complexities with the attacks proposed in this work.

Primitive	Attack model	Rounds	Attack type	Comp. complexity	Data complexity	Reference
PRESENT						
PRESENT	SK	16	Differential	2^{65}	2^{64} CP	[23]
PRESENT	SK	19	Diff. + Algebraic	2^{113}	n/a	[1]
PRESENT	SK	24	Saturation	2^{57}	2^{57} CP	[9]
PRESENT	SK	24	Linear	n/a	$2^{63.5}$ KP	[19]
PRESENT	SK	25	Linear	2^{65}	$2^{62.4}$ KP	[8]
PRESENT	SK	26	Linear	2^{72}	2^{64} KP	[8]
PRESENT-80	SK	21	Biclique	$2^{79.03}$	2^{60} CP	This work
PRESENT-80	SK	31 (full)	Biclique	$2^{79.46}$	2^{60} CP	This work
PRESENT-128	SK	25	Biclique	$2^{126.99}$	2^{44} CP	This work
PRESENT-128	SK	31 (full)	Biclique	$2^{127.37}$	2^{44} CP	This work
LED						
LED-64	SK	8	Splice-and-cut	2^{56}	2^{8} CP	[14]
LED-64	CRK	15	Rebound	2^{16}	2^{118} CP	[11]
LED-64	RK	16	Differential	$2^{(n+\frac{1}{p})/2}$ (*)	2^{64} CP	[18]
LED-64	SK	27.5	Biclique	$2^{62.9}$	2^{56} CP	This work
LED-64	SK	31.5	Biclique	$2^{63.34}$	2^{56} CP	This work
LED-128	SK	16	Splice-and-cut	2^{112}	2^{16} CP	[14]
LED-128	CRK	27	Rebound	2^{16}	2^{118} CP	[11]
LED-128	RK	24	Differential	$2^{3n/2}$ (*)	2^{64} CP	[18]
LED-128	SK	44	Biclique	$2^{126.92}$	2^{64} CP	This work
LED-128	SK	48 (full)	Biclique	$2^{127.23}$	2^{64} CP	This work

Table 1: Attacks on PRESENT and LED (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.

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

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
- 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 bicliques, indexed by i
- ∇_j: Differences in the starting states of bicliques, indexed by j
- Δ_i^K: Differences in the keys for forward computation of bicliques, indexed by i
- ∇_j^K: Differences in the keys for backward computation of bicliques, 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 contain a detailed description of our attacks including the construction of bicliques, matchings and the resulting complexities. We conclude our paper in Section 8.

2 Preliminaries

This section contains brief descriptions of the ciphers PRESENT and LED to show how they work.

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. [4]).

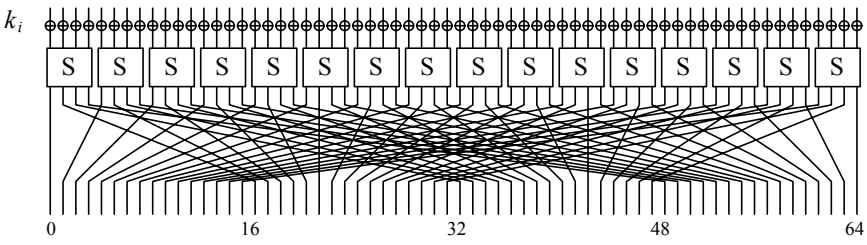


Fig. 1: Round structure of PRESENT with 80-bit key or 128-bit key.

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:

- $[k_{79}, k_{78}, \ldots, k_1, k_0] = [k_{18}, k_{17}, \ldots, k_1, k_0, k_{79}, k_{78}, \ldots, k_{20}, k_{19}]$
- $[k_{79}, k_{78}, k_{77}, k_{76}] = Sbox[k_{79}, k_{78}, k_{77}, k_{76}]$
- $[k_{19}, k_{18}, k_{17}, k_{16}, k_{15}] = [k_{19}, k_{18}, k_{17}, k_{16}, k_{15}] \oplus r$

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:

- $[k_{127}, k_{126}, \ldots, k_1, k_0] = [k_{66}, k_{65}, \ldots, k_1, k_0, k_{127}, k_{126}, \ldots k_{68}, k_{67}]$
- $[k_{127}, k_{126}, k_{125}, k_{124}] = Sbox[k_{127}, k_{126}, k_{125}, k_{124}]$
- $[k_{123}, k_{122}, k_{121}, k_{120}] = Sbox[k_{123}, k_{122}, k_{121}, k_{120}]$
- $[k_{66}, k_{65}, k_{64}, k_{63}, k_{62}] = [k_{66}, k_{65}, k_{64}, k_{63}, k_{62}] \oplus r$

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 [11].

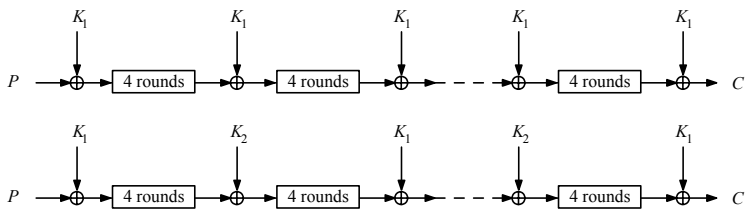


Fig. 2: Round structure of LED [11] with 64-bit key(top) or 128-bit key (bottom).

3 Biclique Cryptanalysis

In this section, we give an overview of biclique cryptanalysis following the descriptions by Bogdanov *et al.* [3].

3.1 Definition

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

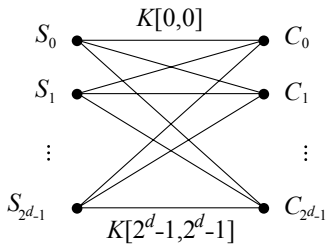


Fig. 3: Schematic view on bicliques.

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 length 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 an arbitrary part of the cipher and use a meet-in-the-middle or similar attack for the remaining parts.
[16,3] introduced different type of biclique attacks. This paper focuses on independent-biclique attacks which usually have a high dimension but are limited in the number of rounds they can cover.

3.2 Independent Bicliques

Independent bicliques allow the construction of bicliques over some subcipher $\mathcal{B}$ from two differentials. First, one chooses a base computation, *i.e.,* 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:

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

Then, one chooses 2^d forward differentials Δ_i which connect the state S_0 with ciphertexts C_i:

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

Similarly, one chooses 2^d backward differentials ∇_j, which connect the ciphertext C_0 with the states S_j:

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

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^{k-2d} computations of E in addition to the effort to construct the biclique and the effort for the matching. More positively, the adversary needs to have access to only either an encryption or a decryption oracle.
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, an alternative method called *matching with precomputations* can be used instead of a meet-in-the middle attack.

3.3 Matching with Precomputations

In [15,3], Khovratovich et al. introduced matching-with-precomputations as an efficient technique 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, one can reduce the computational effort significantly. A further reduction is possible by matching only in a part of the state at v, which is called *partial matching*.

3.4 Complexity Calculations

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 computational 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 Independent-Biclique Attack on Full 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 any 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 the eight bits (k_{64},k_{63},k_{62},k_{61},k_{60},k_{59}, k_{58}, k_{57}) in backward direction respectively to construct the bicliques. So, we have split the key space into 2^{64} groups.

$$\begin{aligned} K[0,0] &= (k_{79}, k_{78}, 0, \ldots, 0, k_{69}, \ldots, k_{13}, 0, \ldots, 0, k_4, k_0), \\ \Delta_i^K &= (0, \ldots, 0, k_{12}, k_{11}, k_{10}, k_9, k_8, k_7, k_6, k_5, 0, \ldots, 0), \\ \nabla_j^K &= (0, 0, k_{77}, k_{76}, k_{75}, k_{74}, k_{73}, k_{72}, k_{71}, k_{70}, 0, \ldots, 0). \end{aligned}$$

4.2 4-Round Biclique of Dimension 8

We can construct a biclique over the first four rounds. Due to the post-whitening after the final round, a biclique at the end of the cipher could cover only three rounds. So, our biclique transforms the plaintexts P_j to states S_i. The biclique construction is illustrated in Figure 4. For all key groups, we use the same value for P_0. As one can see in the biclique, the key differences in the nabla trails affect the plaintexts P_j in 15 out of 16 nibbles at most. So, we can say, that the data complexity will not exceed 2^{60} different values for all texts P_j.

4.3 Matching over 27 Rounds

We apply a matching with precomputations over the remaining rounds of the cipher. Figure 5 shows the matching part which covers the rounds 5-16 in forward direction and the rounds

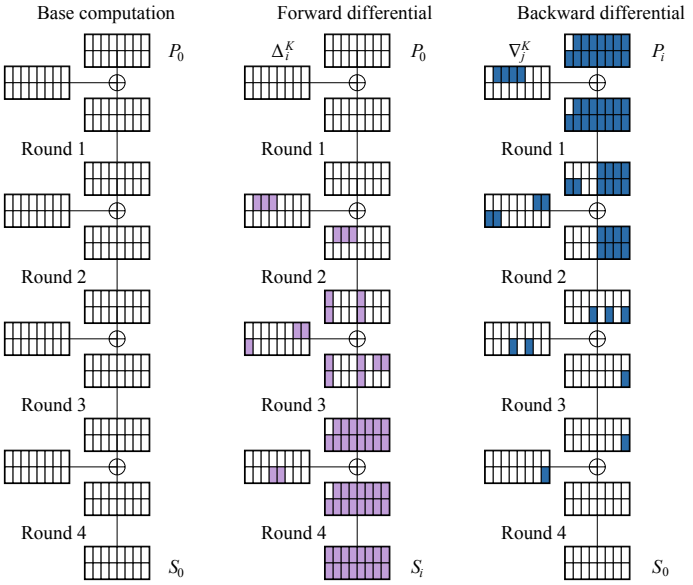


Fig. 4: Biclique for PRESENT-80 in rounds 1 - 4 with Δ_i- (forward) and ∇_j-differentials (backward).

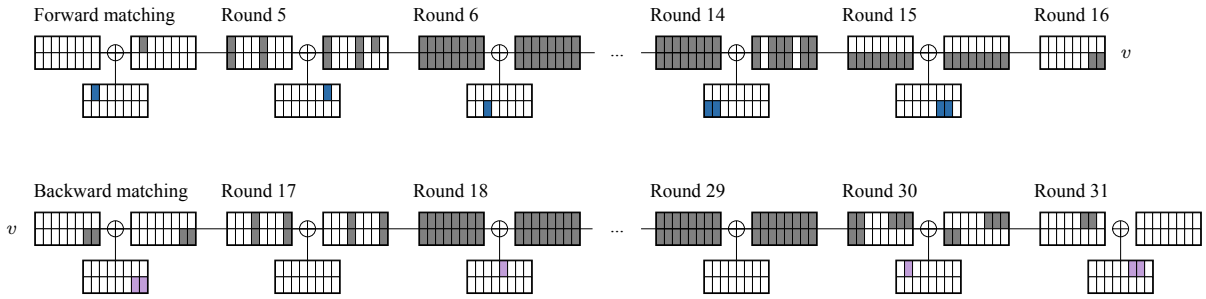


Fig. 5: Recomputations for PRESENT-80 in forward and backward direction.

17-31 in backward direction. We match in the bits $v_7, v_6, v_5, v_4, v_3, v_2, v_1, v_0$ of the state $v = (v_{63}, v_{62}, \ldots, v_1, v_0)$ after Round 16.
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 have a single number which refers best to the total effort, we consider PRESENT as a nibble-wise operating cipher and approximate the recomputation costs in the matching part by counting the number of S-box calls in the round transformation and the key schedule.
We are interested in those S-boxes, which need to be recomputed due to the injected key differences ∇_j^K. In the forward part from S_i to $\overleftarrow{v_{i,j}}$, there are $1 + 5 + 14 + 7 \cdot 16 + 12 + 8 = 152$ active S-boxes in the states. Similarly, in the backward part from C_j to $\overrightarrow{v_{i,j}}$, we need to recompute $2 + 4 + 12 \cdot 16 + 5 = 203$ active nibbles in backward direction.
Additionally, we need to consider the effort to recompute the round keys in the matching part. Again, we concentrate on the S-boxes which need to be recomputed. The effort here is quite low, since PRESENT-80 uses the S-box only for the leftmost four bits in each round key and the schedule has a very slow diffusion. In the key schedule, we need to recompute only the S-box in the round key for round 9. So, we need to recompute $152 + 203 + 1 = 356$ S-boxes in total.

4.4 The Complexity of the Attack

PRESENT-80 applies its S-box 16 times in each of its 31 rounds and uses it 62 times nibbles 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 $C_{recomp} = 2^{16} \cdot \frac{356}{527} \approx 2^{15.43}$ full encryptions. In all of our attacks, we match in eight bits in v. Thus, for every biclique, we can expect to have 2^8 false positives in average. So, the complexity to test them false positives $C_{falsepos}$ is 2^8 in all of our attacks.
For the biclique construction, one needs to compute $2 \cdot 2^8$ times the states of 4 out of 31 rounds. So, $C_{biclique}$ is equivalent to $2^{6.05}$ full encryptions. The precomputations costs $C_{precomp}$ are given by computing 2^8 times 12 rounds in forward direction P to v and 2^8 times 15 rounds in backward direction from S to v. So, one has to consider 2^8 times 27 out of 31 rounds, which is equal to $2^{7.8}$ encryptions. The full computational complexity is then given by

$$2^{64} \cdot (2^{6.05} + 2^{7.8} + 2^{15.43} + 2^8 + 2^8) = 2^{79.46}.$$

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

4.5 Independent-Biclique Attack on 21 Rounds of PRESENT-80

It is desirable to have an advantage of at least one power of two compared to exhaustive search. For PRESENT-80, we can mount an attack on a reduced version of the first 21 out of 31 rounds with the same biclique as before. There, we match in the eight bits $v_{23}, v_{22}, v_{22}, v_{20}, v_{19}, v_{18}, v_{17}$,-$v_{16}$ in the state v after Round 8, as shown in Figure 8 (see Appendix A).
One has to consider $1+5+8+8 = 22$ S-boxes in the forward, and $2+4+9 \cdot 16+4 = 154$ S-boxes in the backward part. Additionally, there is one active S-box in the key schedule for the round key of round 14. The recomputation effort is given by $22 + 154 + 1 = 177$ S-boxes. In the reduced PRESENT-80 cipher, there are $21 \cdot 16 + 21 = 357$ S-boxes in round transformation and key schedule. So, C_{recomp} is equal to $2^{16} \cdot \frac{177}{357} = 2^{14.99}$ encryptions. The construction of the biclique requires to compute $2 \cdot 2^8$ times 4 out of 21 rounds or $2^{6.61}$ encryptions; the precomputational costs make up 2^8 times encrypting 17 out of 21 rounds or $2^{7.70}$ encryptions. The total time complexity is then given by

$$2^{64} \cdot (2^{6.61} + 2^{7.70} + 2^{14.99} + 2^8 + 2^8) = 2^{79.03}$$

encryptions. The data and memory complexities remain the same as in the attack on full PRESENT-80.

5 Independent-Biclique Attack on Full PRESENT-128

This section describes an attack on the full version of 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 any 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 $(k_{101},k_{100},k_{99},k_{98},k_{97},k_{96},k_{95},k_{94})$ in forward direction, and other eight bits $(k_{36},k_{35},k_{34},k_{33},$-$k_{32},k_{31},k_{30},k_{29})$ in backward direction respectively to construct the bicliques. So, we have 2^{112} key groups in total.

$$\begin{aligned} K[0,0] &= (k_{127},\ldots,k_{102},0,\ldots,0,k_{93}\ldots,k_{37},0,\ldots,0,k_{28},\ldots,k_0),\\ \Delta_i^K &= (0,\ldots,0,k_{101},k_{100},k_{99},k_{98},k_{97},k_{96},k_{95},k_{94},0,\ldots,0),\\ \nabla_j^K &= (0,\ldots,0,k_{36},k_{35},k_{34},k_{33},k_{32},k_{31},k_{30},k_{29},0,\ldots,0). \end{aligned}$$

5.2 4-Round Biclique of Dimension 8

Similar to the attack on the 80-bit version, we construct a biclique at the beginning of the cipher. Otherwise, the key addition after the last round would cost one round. Figure 6 shows the biclique constrcution over the first four rounds. We fix P_0 for all key groups. As one can see from Figure 6, the plaintexts P_j are active in 11 out of 16 nibbles. Thus, the data complexity for all key groups does not exceed 2^{44} chosen plaintexts.

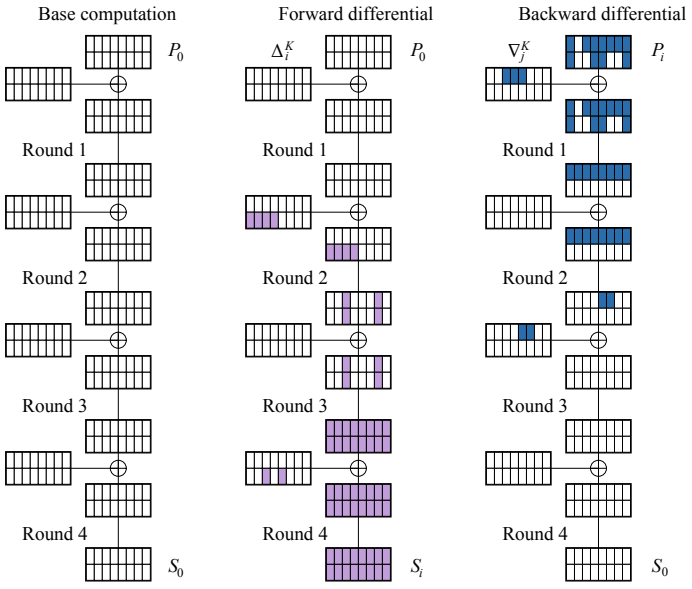


Fig. 6: Biclique for PRESENT-128 in rounds 1 - 4 with Δ_i- and ∇_j-differentials.

5.3 Matching over 27 Rounds

We match in the eight bits $v_{47}, v_{46}, v_{45}, v_{44}, v_{43}, v_{42}, v_{41}, v_{40}$ of the state v after Round 22, as shown in Figure 7. Again, we are interested in the S-boxes we need to recompute. In the forward part from round 5 to 22, there are $1 + 4 + 14 \cdot 16 + 8 + 8 = 245$ S-boxes. Similarly, one has to compute $2 + 4 + 6 \cdot 16 + 4 = 106$ S-boxes in the backward part from round 31 to 22. PRESENT-128 uses the S-box twice, this means for the leftmost eight bits of the key register. For our attack, we need to consider one additional nibble in the key schedule which is used as input of the S-box in the round key for round 18, and one more nibble in the round key of round 32 in the key schedule. In total, we need to recompute $245 + 106 + 2 = 353$ S-boxes in the attack.

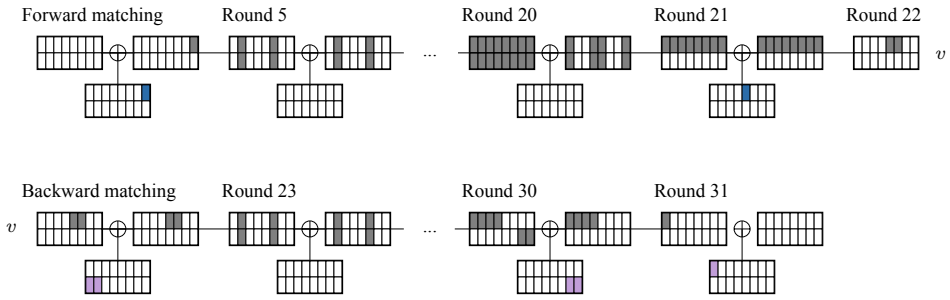


Fig. 7: Recomputations for PRESENT-128 in forward and backward direction.

5.4 The Complexity of the Attack

PRESENT-128 uses the S-box 16 times in each of the 31 rounds; in addition, the key schedule uses the S-box for the two leftmost nibbles of the key register for every for every round key. The total number of S-boxes in the cipher sum up to $31 \cdot 16 + 62 = 558$ S-boxes. Thus, the recomputation complexity C_{recomp} is equal to $2^{16} \cdot \frac{353}{558} = 2^{15.39}$. Since we match in eight bits of v, the complexity to test false positives $C_{falsepos}$ can be approximated by 2^8. Similar to the attack on PRESENT-80, $C_{biclique}$ is equivalent to $2^{6.05}$ full encryptions. The precomputations costs $C_{precomp}$ are given by computing 2^8 times 27 out of 31 rounds, which is equal to $2^{7.8}$ encryptions. The full computational complexity is equivalent to

$$2^{112} \cdot (2^{6.05} + 2^{7.8} + 2^{15.34} + 2^8 + 2^8) = 2^{127.37}.$$

The data complexity is 2^{44}, and we need to store 2^8 texts per group.

5.5 Independent-Biclique Attack on 24 Rounds of PRESENT-128

To obtain a better advantage, we can mount an attack on a version of PRESENT-128 reduced to the first 24 rounds. We use the same biclique structure as in the attack on full PRESENT-128 with a different matching procedure. We match in the eight bits $v_{63}, v_{62}, v_{61}, v_{60}, v_{59}, v_{58}, v_{57}, v_{56}$ in the state v after Round 15, as shown in Figure 9 (see Appendix B).
In the forward part from round 5 to 15, we have to recompute $1+4+7 \cdot 16+8+8 = 133$ S-boxes. In the backward part from round 24 to 15, there are $2 + 4 + 4 \cdot 16 + 6 = 76$ S-boxes. There are no active S-boxes in the key schedule. So, the effort sums up to $133 + 76 = 209$ S-boxes. Since the 24-round PRESENT-128 cipher uses $24 \cdot 16 + 24 \cdot 2 = 432$ S-boxes in round transformation and key schedule, C_{recomp} is equal to $2^{16} \cdot \frac{209}{432} = 2^{14.95}$. $C_{biclique}$ represents the costs to process

$2 \cdot 2^8$ times 4 out of 24 rounds or $2^{6.42}$ encryptions; the precomputational costs make up 2^8 times encrypting 20 out of 24 rounds or $2^{7.74}$ encryptions. The total time complexity is then given by

$$2^{112} \cdot (2^{6.42} + 2^{7.74} + 2^{14.95} + 2^8 + 2^8) = 2^{126.99}.$$

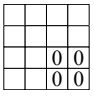
The data and memory complexities remain the same as in the attack on full PRESENT-128.

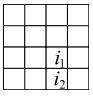
6 Independent-Biclique Attack on 31.5 Rounds of LED-64

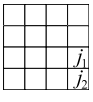
This part includes an explanation of our attack on 31.5 out of 32 rounds of LED-64. The attack has three steps: key partitioning, biclique construction and a matching part. The complexity of the attack comes at the end.

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 $K[0,0]$.

$K[0,0] =$ 

$\Delta_i^K(K_1) =$ 

$\nabla_j^K(K_1) =$ 

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 10 (see Appendix C) visualizes the biclique over seven rounds, from 25 to 32, without the final key addition.

6.3 Matching over 24 Rounds

In the matching phase, first, 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 11 (see Appendix C).
LED has a similar structure to the AES block cipher. 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 for the key and a constant addition. Therefore, the computational cost for this part is negligible. Also, there are no extra computational costs for the shift operations. Thus, we only need to concentrate on the number of MixColumnsSerial and SubCell operations which are affected by the key differences and are

required to recompute for the matching. 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 $2 + 8 + 4 = 14$ 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.

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 32 rounds of the cipher. So, for 2^{16} keys in one group, C_{recomp} is equivalent to approximately $2^{16} \cdot \frac{318}{512} \approx 15.31$ encryptions. The biclique construction costs $C_{biclique}$ are given by computing $2 \cdot 2^8$ times 8 out of 32 rounds, which is equal to 2^7 encryptions. The precomputation costs sum up to computing 2^8 times 24 out of 32 rounds or $2^{7.58}$ 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

$$2^{48} \cdot (2^7 + 2^{7.58} + 2^{15.31} + 2^8 + 2^8) = 2^{63.34}.$$

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 so 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

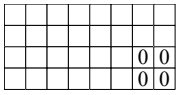
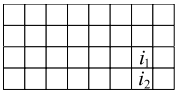
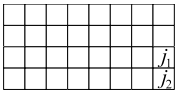
$$2^{48} \cdot (2^{6.81} + 2^{7.64} + 2^{14.86} + 2^8 + 2^8) = 2^{62.9}.$$

7 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 $(K_1 \| K_1)$ 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.

$K[0,0] =$  $\Delta_i^K(K_1\|K_2) =$  $\nabla_j^K(K_1\|K_2) =$ 

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, we can construct a biclique over 12 rounds, from rounds 37 - 48. Figure 12 (see Appendix D) 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. Because of the fast diffusion of LED, large parts of the cipher need to be recomputed in the matching phase. As we can see from Figure 13 (see Appendix D), 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 $45 \cdot 16 = 768$ S-boxes in the 48 rounds of the cipher, so, for 2^{16} keys in one group, C_{recomp} is equal to $2^{16} \cdot \frac{440}{768} \approx 2^{15.20}$ full encryptions. The complexity to test false positives $C_{falsepos}$ can be approximated by 2^8 and the decryption costs $C_{decrypt}$ are given by 2^8. $C_{biclique}$ is described by computing $2 \cdot 2^8$ times 12 out of 48 rounds or 2^7 full encryptions. In the matching step, one has to precompute 36 rounds 2^8 times, which is equal to $2^{7.59}$ encryptions. The full computational complexity is then

$$2^{112} \cdot (2^7 + 2^{7.59} + 2^{15.20} + 2^8 + 2^8) = 2^{127.23}.$$

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

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

8 Conclusion

In this paper, we showed the first full-round biclique attacks on PRESENT and LED in the single-key model, using independent bicliques. In our attack on PRESENT-80, the time is equivalent to $2^{79.46}$ full encryptions and the data complexity is 2^{60} chosen plaintexts. For the attack on PRESENT-128, we need time equivalent to $2^{127.37}$ full encryptions and at most 2^{44} chosen plaintexts. 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.
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. Kazumaro Aoki and Yu Sasaki. Preimage Attacks on One-Block MD4, 63-Step MD5 and More. In *Selected Areas in Cryptography'08*, pages 103–119, 2008.
3. Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. Biclique Cryptanalysis of the Full AES. Cryptology ePrint Archive, Report 2011/449, 2011. `http://eprint.iacr.org/`.
4. 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.
5. 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.
6. Mustafa Çoban, Ferhat Karakoç, and Özkan Boztaş. Biclique Cryptanalysis of TWINE. Cryptology ePrint Archive, Report 2012/422, 2012. `http://eprint.iacr.org/`.
7. Shaozhen Chen and Tianmin Xu. Biclique Attack of the Full ARIA-256. *IACR Cryptology ePrint Archive*, 2012:11, 2012.
8. 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.
9. 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.
10. 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.
11. Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED Block Cipher. In Preneel and Takagi [20], pages 326–341.
12. 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.
13. 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.
14. 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.
15. Dmitry Khovratovich and Christian Rechberger. A Splice-and-Cut Cryptanalysis of the AES. *IACR Cryptology ePrint Archive*, 2011:274, 2011. `http://eprint.iacr.org/2011/274`.
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. Hamid Mala. Biclique Cryptanalysis of the Block Cipher SQUARE. Cryptology ePrint Archive, Report 2011/500, 2011. `http://eprint.iacr.org/`.
18. Florian Mendel, Vincent Rijmen, Deniz Toz, and Kerem Varici. Differential Analysis of the LED Block Cipher. *IACR Cryptology ePrint Archive*, 2012:544, 2012.
19. 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.
20. 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.
21. Kyoji Shibutani, Takanori Isobe, Harunaga Hiwatari, Atsushi Mitsuda, Toru Akishita, and Taizo Shirai. Piccolo: An Ultra-Lightweight Blockcipher. In Preneel and Takagi [20], pages 342–357.
22. 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`.
23. 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.
24. 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.

A Independent-Biclique Attack on 21 Rounds of PRESENT-80

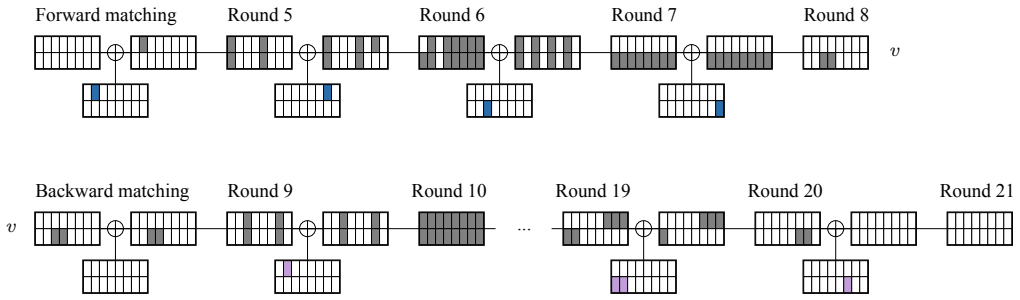


Fig. 8: Recomputations for reduced-round PRESENT-80 in forward and backward direction.

B Independent-Biclique Attack on 24 Rounds of PRESENT-128

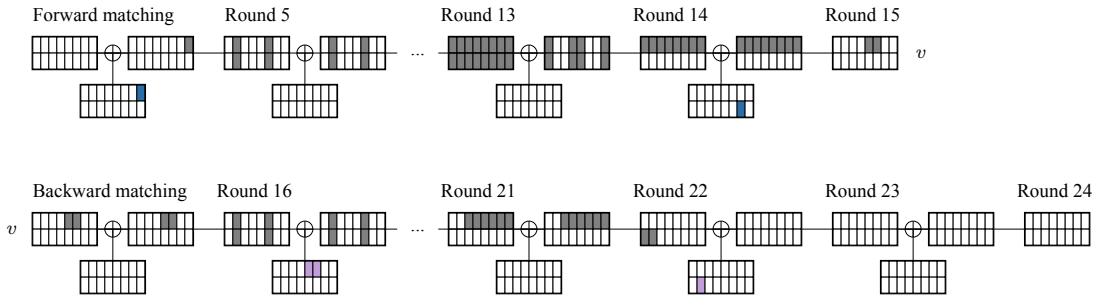


Fig. 9: Recomputations for reduced-round PRESENT-128 in forward and backward direction.

C Independent-Biclique Attack on 31.5 Rounds of LED-64

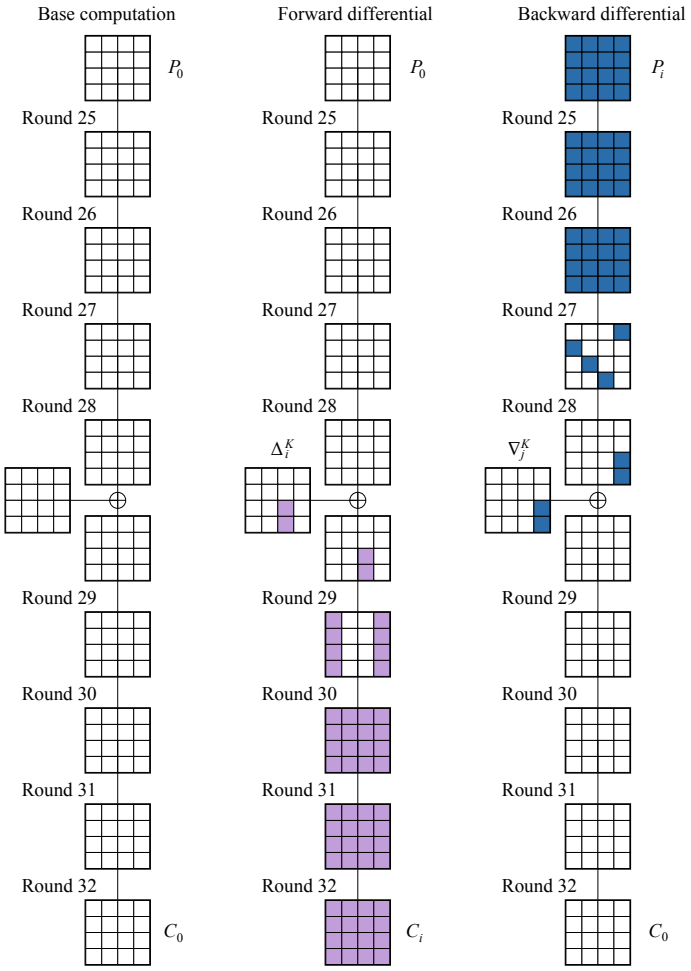


Fig. 10: Biclique for LED-64 in rounds 25 - 32 with Δ_i- and ∇_j-differentials.

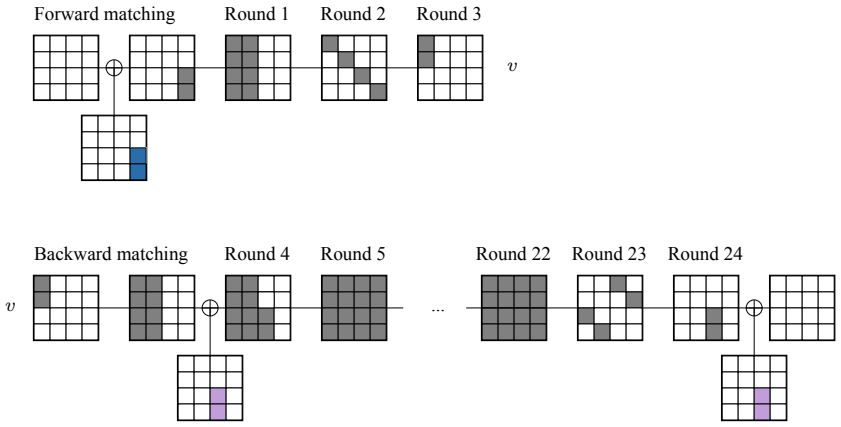


Fig. 11: Recomputations for LED-64 in forward and backward direction.

D Independent-Biclique Attack on Full LED-128

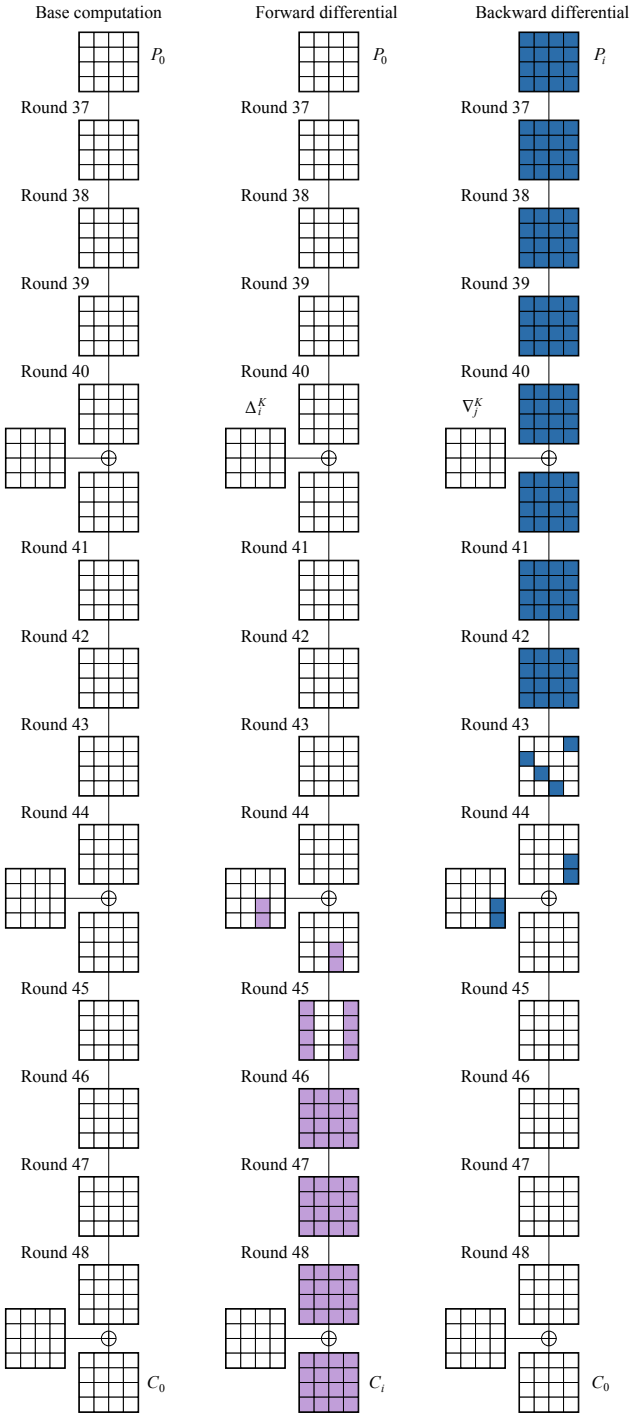


Fig. 12: Biclique for LED-128 in rounds 37 - 48 with Δ_i- and ∇_j-differentials.

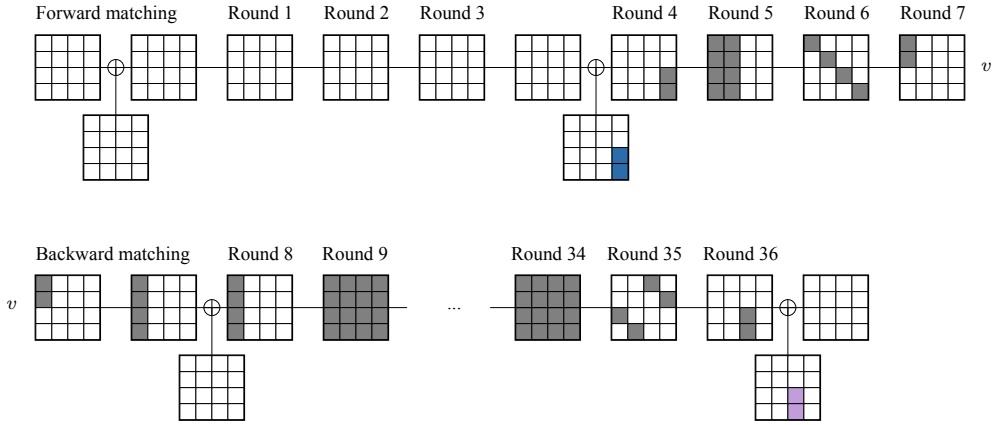


Fig. 13: Recomputations for LED-128 in forward and backward direction.