FFT Key Recovery for Integral Attack

Yosuke Todo and Kazumaro Aoki

NTT Secure Platform Laboratories, Tokyo, Japan

Abstract. An integral attack is one of the most powerful attacks against block ciphers. We propose a new technique for the integral attack called the Fast Fourier Transform (FFT) key recovery. When the integral distinguisher uses N chosen plaintexts and the guessed key is k bits, a straightforward key recovery requires the time complexity of $O(N2^k)$. However, the FFT key recovery method requires only the time complexity of $O(N + k2^k)$. As a previous result using FFT, at ICISC 2007, Collard *et al.* proposed that FFT can reduce the time complexity of a linear attack. We show that FFT can also reduce the complexity of the integral attack. Moreover, the estimation of the complexity is very simple. We first show the complexity of the FFT key recovery against three structures, the Even-Mansour scheme, a key-alternating cipher, and the Feistel cipher. As examples of these structures, we show integral attacks against PRØST, CLEFIA, and AES. As a result, 8-round PRØST $\tilde{P}_{128,K}$ can be attacked with about an approximate time complexity of 2^{80}. Moreover, a 6-round AES and 12-round CLEFIA can be attacked with approximate time complexities of $2^{52.6}$ and $2^{87.5}$, respectively.

Keywords: Block cipher, Integral attack, Fast Fourier Transform, Fast Walsh-Hadamard Transform, PRØST, CLEFIA, AES.

1 Introduction

An integral attack is one of the most powerful attacks against block ciphers. The integral attack was first proposed by Daemen *et al.* to evaluate the security of SQUARE [7], and then Knudsen and Wagner formalized this attack as an integral attack [14]. This attack uses N chosen plaintexts (CPs) and the corresponding ciphertexts. Generally, an integral attack consists of a distinguisher and key recovery. In the distinguisher, plaintexts are prepared in which the XOR of the R-th round output is 0. In the key recovery, R-th round outputs are recovered from ciphertexts by guessing round keys used in the last several rounds. If the guessed key is incorrect, the recovered texts are assumed to behave as random texts. On the other hand, if the guessed key is correct, the XOR of the recovered texts is 0.

We focus on the key recovery of the integral attack. Several techniques to improve the key recovery were proposed, *e.g.*, the partial-sum technique [11] and the meet-in-the-middle (MITM) technique [21]. The partial-sum technique was proposed by Ferguson *et al.* in 2000. When the integral attack uses N chosen

Table 1. Summary of FFT key recovery, where k, k_1, and k_2 are defined in Sect. 1.1

Target cipher	Time
Even-Mansour scheme	$O(k2^k)$
Key-alternating cipher	$O(k_2 2^k)$
Feistel cipher	$O(k_1 2^{k_1} + k_2 2^{k_2})$

Table 2. Comparison of attack results. Time column only includes the time complexity of the key recovery step, and it does not include the time complexity to count the frequency of the partial bit-string of ciphertexts corresponding to the chosen plaintexts (CPs).

Target cipher	#Round	Data (CP)	Time	Technique	Reference
PRØST $\tilde{P}_{128,K}$	8	2^{64}	2^{80}	FFT	Sect. 3
PRØST $\tilde{P}_{256,K}$	9	2×2^{64}	$2^{80.9}$	FFT	Appendix D
AES	6	6×2^{32}	6×2^{50}	Partial-sum	[11]
AES	6	6×2^{32}	6×2^{50}	FFT	Sect. 4
CLEFIA	12	13×2^{112}	13×2^{106}	MITM, Partial-sum	[21]
CLEFIA	12	5×2^{112}	$2^{87.5}$	MITM, FFT	Sect. 5

plaintexts and guesses a k-bit round key, a straightforward key recovery approach requires the time complexity of $O(N2^k)$. Therefore, if enormous number of chosen plaintexts is used, the complexity of the attack increases to a very high level. The partial-sum technique can reduce the complexity, where we partially compute the sum by guessing each key one after another and reuse the partial sums. Ferguson *et al.* applied this technique to AES [17], and showed that a 6-round AES can be attacked with 6×2^{50} S-box lookups. The MITM technique was proposed by Sasaki *et al.* in 2012. This technique can reduce the complexity of the integral attack against several Feistel ciphers. In the key recovery against several Feistel ciphers, $\bigoplus(x \oplus y) = 0$ is often evaluated, where x and y are calculated from ciphertexts by guessing keys. The MITM technique first calculates $\bigoplus x$ and $\bigoplus y$ independently, and then searches keys that satisfy $\bigoplus x = \bigoplus y$ by using analysis such as the MITM attack [8]. As a result, Sasaki *et al.* improved integral attacks against LBlock [23], HIGHT [12], and CLEFIA [22].

Several key recovery techniques using the Fast Fourier Transform (FFT) have recently been proposed. In 2007, Collard *et al.* first proposed a linear attack using the FFT [6], and then Nguyen *et al.* extended it to a multi-dimensional linear attack [18,19]. Moreover, Bogdanov *et al.* proposed a zero correlation attack using the FFT in 2013 [4]. We now point out that the FFT can also be applied to the integral attack.

1.1 Our Contribution

We propose a new improved technique for the integral attack called the FFT key recovery. Table 1 shows results of the FFT key recovery, and Table 2 summarizes results of integral attacks against specific ciphers.

The FFT key recovery is useful for an integral attack with an enormous number of chosen plaintexts because the time complexity of the FFT key recovery does not depend on the number of chosen plaintexts. Therefore, the motivation to introduce this technique is similar to the reason to introduce the partial-sum technique. However, the way that the two techniques are applied is a little different, and we discuss the differences between them in Sect. 6. Another important reason to introduce the FFT key recovery is that it enables easy estimation of the time complexity of the integral attack. The partial-sum technique effectively reduces the complexity, but the attack procedure is often complicated. On the other hand, the complexity of the FFT key recovery only depends on k, where k denotes the bit length of the keys that are required to call a distinguisher on the ciphertext side.

We focus on structures for block ciphers, and estimate the security against the integral attack. Here, we focus on three structures, the Even-Mansour scheme [10] in Sect. 3, the key-alternating cipher [5] in Sect. 4, and the (generalized) Feistel cipher in Sect. 5.

The Even-Mansour scheme is a famous scheme used to construct a block cipher from a permutation, and has recently been a popular discussion topic [9,5]. When FFT key recovery is used, the time complexity of the integral attack is estimated as $O(k2^k)$. As an example of the Even-Mansour scheme, we consider PRØST [13] and show integral attacks of the scheme. PRØST is an authenticated encryption scheme, which was submitted to the CAESAR competition. The core function is called the PRØST permutation, which is extended to a block cipher by the Even-Mansour scheme. Results show that 8-round PRØST $\tilde{P}_{128,K}$ and 9-round PRØST $\tilde{P}_{256,K}$ can be attacked with the time complexity of approximately 2^{80} and $2^{80.9}$, respectively.

The key-alternating cipher is a common type of block cipher, and AES is viewed as a 10-round key-alternating cipher [3]. The time complexity of the integral attack is at least $O(k2^k)$, but we can optimize it slightly. We assume that only k_2 bits of ciphertexts are required for the distinguisher, where k_2 is always less than or equal to k. In this case, the complexity is reduced to $O(k_2 2^k)$. As an example of the key-alternating cipher, we consider AES and show an integral attack against it. Results show that a 6-round AES can be attacked with a time complexity of approximately 6×2^{50}.

The Feistel cipher is also commonly used to construct a block cipher. The MITM technique is useful in reducing the time complexity, and works well in combination with the FFT key recovery. The MITM technique evaluates $\bigoplus x$ and $\bigoplus y$ instead of $\bigoplus(x \oplus y)$. We assume that k_1 and k_2 bits are required to evaluate $\bigoplus x$ and $\bigoplus y$, respectively. In this case, the complexity of the integral attack is $O(k_1 2^{k_1} + k_2 2^{k_2})$. As an example of the Feistel cipher, we show an integral attack against CLEFIA, which is a 4-branch generalized Feistel cipher and is adopted by the ISO/IEC standard [1]. Results show that a 12-round CLEFIA can be attacked with the time complexity of approximately $2^{87.5}$ [1].

[1] Since this attack uses 5×2^{112} chosen plaintexts, the dominant factor in determining the time complexity is the number of chosen plaintexts. The FFT key recovery can reduce the time complexity of the key recovery step.

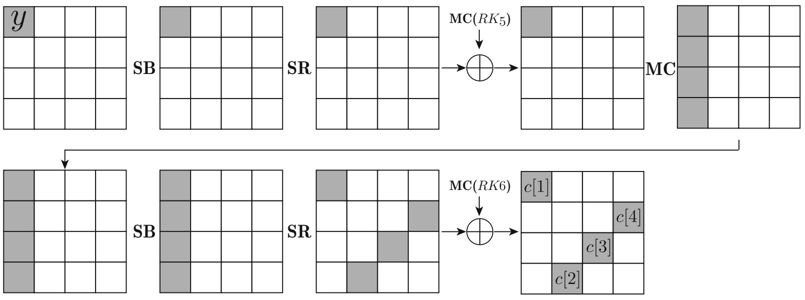

Fig. 1. Integral attack against 6-round AES

2 Related Work

2.1 Integral Attack

Integral Distinguisher. An integral distinguisher is constructed based on the following integral properties.

- All ($\mathcal{A}$) : All values appear and with exactly the same frequency in the (multi-)set of texts.
- Balance ($\mathcal{B}$) : The XOR of all texts in the set is 0.
- Constant ($\mathcal{C}$) : The bit-strings in a set are fixed to the same value.

For instance, we show the 4-round integral distinguisher of AES in Appendix A.

Key Recovery. In key recovery, the R-th round output is recovered from ciphertexts by guessing round keys used in the last several rounds. If the guessed key is incorrect, the recovered texts are expected to behave as random texts. On the other hand, if the guessed key is correct, the XOR of the recovered texts is always 0.

For instance, Fig. 1 shows the key recovery of the integral attack against a 6-round AES. Here we now have 2^{32} ciphertexts for the 6-round AES, and know that value y satisfies $\mathcal{B}$. Let $c[i]$ be bytes in the ciphertexts as shown in Fig. 1, and c_n denotes the n-th ciphertext. In this case, the XOR of y is calculated from 2^{32} ciphertexts as

$$\bigoplus_{n=1}^{2^{32}} S_5(S_1(c_n[1] \oplus K_1) \oplus S_2(c_n[2] \oplus K_2) \oplus S_3(c_n[3] \oplus K_3) \\ \oplus S_4(c_n[4] \oplus K_4) \oplus K_5) = 0, \tag{1}$$

where $S_1, S_2, \ldots, S_5$ are S-boxes, each of which consists of the inverse of the AES S-box and a multiplication by a field element from the inverse of the AES MDS matrix. Moreover, K_1, K_2, K_3, and K_4 are calculated from RK_6, and K_5 is calculated from RK_5. Therefore, the total bit length of the guessed keys is 40 bits. Analysis using a straightforward method incurs the approximate time complexity of $2^{32+40} = 2^{70}$. However, the partial-sum technique can reduce the complexity. Ferguson *et al.* showed that this analysis takes only 2^{50} S-box lookups [11].

2.2 FFT Key Recovery

Collard *et al.* showed a linear attack using FFT in 2007. The key recovery of a linear attack [16] uses N ciphertexts $c_1, c_2 \ldots, c_N$. Then, it guesses keys K and calculates

$$\sum_{n=1}^{N} f(c_n \oplus K). \tag{2}$$

It finally recovers the correct K to evaluate Eq. (2) for several possible Ks. Here, let $f : \{0,1\}^k \to \{0,1\}$ be a Boolean function, which is generated from the linear approximate equation. The evaluation of Eq. (2) requires the time complexity of $O(N2^k)$ using a straightforward method, and the size of N is generally enormous, *e.g.*, $N \approx 2^k$. Collard *et al.* showed that the evaluation of Eq. (2) requires the time complexity of approximately $O(k2^k)$. Nguyen *et al.* then noticed that the Fast Walsh-Hadamard Transform (FWHT) can be used instead of the FFT [18]. Hereinafter, we show the calculation method using the FWHT.

Two k-dimensional vectors v and w are first created, where v is generated from Boolean function f and w is generated from the set of ciphertexts as indicated below.

$$\begin{aligned} v_i &= f(i), \\ w_i &= \#\{1 \le n \le N | c_n = i\}. \end{aligned}$$

A k-dimensional vector, u, is calculated from v and w as

$$\begin{bmatrix} u_0 \\ u_1 \\ u_2 \\ \vdots \\ u_{2^k-1} \end{bmatrix} = \begin{bmatrix} v_0 & v_1 & v_2 & \cdots & v_{2^k-1} \\ v_1 & v_0 & v_3 & \cdots & v_{2^k-2} \\ v_2 & v_3 & v_0 & \cdots & v_{2^k-3} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ v_{2^k-1} & v_{2^k-2} & v_{2^k-3} & \cdots & v_0 \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_{2^k-1} \end{bmatrix}. \tag{3}$$

In this case, u_K is equal to the results of Eq. (2). Therefore, if Eq. (3) can be calculated quickly, the time complexity is reduced. Equation (3) is simply expressed as $u = \boldsymbol{V} \times w$. Here, matrix $\boldsymbol{V}$ consists of four 2^{k-1}-dimensional block matrices, $\boldsymbol{V}_1$ and $\boldsymbol{V}_2$, as

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}_1 & \boldsymbol{V}_2 \\ \boldsymbol{V}_2 & \boldsymbol{V}_1 \end{bmatrix}.$$

From the diagonalization of $\boldsymbol{V}$, we have

$$\boldsymbol{V} = \begin{bmatrix} \boldsymbol{V}_1 & \boldsymbol{V}_2 \\ \boldsymbol{V}_2 & \boldsymbol{V}_1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{I} \\ \boldsymbol{I} & -\boldsymbol{I} \end{bmatrix} \begin{bmatrix} \boldsymbol{V}_1 + \boldsymbol{V}_2 & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{V}_1 - \boldsymbol{V}_2 \end{bmatrix} \begin{bmatrix} \boldsymbol{I} & \boldsymbol{I} \\ \boldsymbol{I} & -\boldsymbol{I} \end{bmatrix},$$

where $\boldsymbol{I}$ is an identity matrix. Since $\boldsymbol{V}_1 + \boldsymbol{V}_2$ and $\boldsymbol{V}_1 - \boldsymbol{V}_2$ have the same structure as $\boldsymbol{V}$, we obtain

$$\boldsymbol{V} = \frac{1}{2^k} \times \boldsymbol{H}_{2^k} \times \mathrm{diag}(\boldsymbol{H}_{2^k} v) \times \boldsymbol{H}_{2^k},$$

where $\boldsymbol{H}_{2^k}$ is the 2^k-dimensional Walsh matrix[2], and $\mathrm{diag}(\boldsymbol{H}_{2^k} v)$ is a diagonal matrix whose element in the i-th row and i-th column is the i-th element of $\boldsymbol{H}_{2^k} v$. Therefore, Eq. (3) is expressed as

$$u = \boldsymbol{V} \times w = \frac{1}{2^k} \boldsymbol{H}_{2^k} \times \mathrm{diag}(\boldsymbol{H}_{2^k} v) \times \boldsymbol{H}_{2^k} w.$$

The procedure to calculate u is given below.

1. Let us calculate $\hat{v} = \boldsymbol{H}_{2^k} v$. Then, Eq. (3) is expressed as $u = \frac{1}{2^k} \boldsymbol{H}_{2^k} \times \mathrm{diag}(\hat{v}) \times \boldsymbol{H}_{2^k} w$.
2. Let us calculate $\hat{w} = \boldsymbol{H}_{2^k} w$. Then, Eq. (3) is expressed as $u = \frac{1}{2^k} \boldsymbol{H}_{2^k} \times \mathrm{diag}(\hat{v}) \hat{w}$.
3. Let us calculate $\hat{u}$ whose $\hat{u}_i$ is calculated from $\hat{v}_i \times \hat{w}_i$, and then calculate $u = \frac{1}{2^k} \boldsymbol{H}_{2^k} \hat{u}$.

In the first and second steps, we calculate the multiplication of the Walsh matrix using the FWHT, and time complexity for each is approximately the time of $k2^k$ additions. In the third step, we first calculate 2^k multiplications of the k-bit integers, where we regard that the complexity of one multiplication is equal to that for k additions. We next calculate the FWHT, and the time complexity is approximately the time of $k2^k$ additions. We finally calculate the division by 2^k, but the time complexity is negligible because it can be computed by a k-bit shift. Therefore, the time complexity of the third step is approximately $2k2^k$. Thus, the total time complexity is approximately the time of $4k2^k$ additions.

3 Integral Attack against Even-Mansour Scheme and Application to PRØST

3.1 Even-Mansour Scheme and FFT Key Recovery

The Even-Mansour scheme constructs an n-bit block cipher from an n-bit permutation, P, using two n-bit keys K_1 and K_2 as

$$c = K_2 \oplus P(p \oplus K_1),$$

where p and c denote a plaintext and a ciphertext, respectively [10]. The Even-Mansour scheme has recently been a popular topic of discussion [9,5]. When the FFT key recovery is used, we can easily evaluate the time complexity of the integral attack.

We first split permutation P into two permutations, P_1 and P_2, as $P = P_2 \circ P_1$ (see Fig. 2). We assume that P_1 has an integral distinguisher with N

[2] The Walsh matrix is defined as the following recursive formulae.

$$\boldsymbol{H}_{2^1} = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad \boldsymbol{H}_{2^k} = \begin{bmatrix} \boldsymbol{H}_{2^{k-1}} & \boldsymbol{H}_{2^{k-1}} \\ \boldsymbol{H}_{2^{k-1}} & -\boldsymbol{H}_{2^{k-1}} \end{bmatrix} \quad (k \geq 2).$$

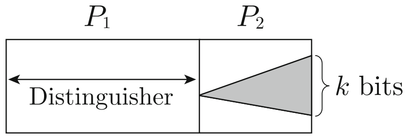

Fig. 2. Even-Mansour scheme and FFT key recovery

chosen plaintexts. Moreover, any one bit is diffused to k bits by P_2 at most. Let f be a Boolean function. Here, the input is k bits of the output of P_2, and the output is any one bit of the input of P_2. In this case, the key recovery can be expressed as

$$\bigoplus_{i=1}^{N} f(c'_i \oplus K'_2) = 0,$$

where c'_i and K'_2 are truncated to k bits from c_i and K_2, respectively. The FFT key recovery calculates the summation on the ring of integers, and we have

$$\sum_{i=1}^{N} f(c'_i \oplus K'_2) = 0 \bmod 2.$$

We can efficiently evaluate this equation using the FWHT, and the time complexity is $O(k2^k)$.

3.2 FFT Key Recovery against PRØST

PRØST is an authenticated encryption scheme, which was submitted to the CAESAR competition. The core function of PRØST $\tilde{P}_{n,K}(x)$ is the block cipher based on the single-key Even-Mansour scheme with key K, and it is defined as

$$\tilde{P}_{n,K}(x) := K \oplus P_n(x \oplus K) \quad \text{in } \{0,1\}^{2n},$$

where x and P_n denote the input and the PRØST permutation, respectively. We show the specification in Appendix B.

PRØST $\tilde{P}_{128,K}$ has a 6-round integral distinguisher with 2^{64} chosen plaintexts. We show the integral characteristics in Appendix C. Moreover, any one bit is diffused to 64 bits in 2 rounds. Let c'_i be the 64-bit truncation of ciphertexts c_i. Let f be a Boolean function that is generated from the last two-round permutation. The input is the 64-bit truncation of the output of the last 2-round permutation. The output is any one bit of the input of the last 2-round permutation. In this case, the key recovery can be expressed as

$$\bigoplus_{i=1}^{2^{64}} f(c'_i \oplus K') = 0,$$

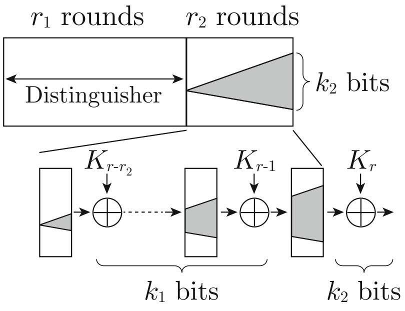

Fig. 3. Key-alternating cipher and FFT key recovery

where K' is truncated to 64 bits from K. The FFT key recovery calculates the summation on the ring of integers, and we calculate

$$\sum_{i=1}^{2^{64}} f(c'_i \oplus K').$$

From the description in Sect. 2.2, we evaluate this equation for all possible K' with $4 \times 64 \times 2^{64} = 2^{72}$ additions. The probability that this value is even for incorrect keys is expected to be 2^{-1}, but this value is always even for the correct key.

All the bits of the output of the 6-round integral characteristic satisfy $\mathcal{B}$. Therefore, we repeat this analysis for 256 bits. Since the probability that all 256 bits satisfy $\mathcal{B}$ for incorrect key K is 2^{-256}, we can recover the 256-bit key K. Thus, the total complexity is approximately $256 \times 2^{72} = 2^{80}$ additions.

We show an integral attack on 9-round PRØST $\tilde{P}_{256,K}$ in Appendix D. The time complexity is approximately $2^{80.9}$ additions.

4 Integral Attack against Key-Alternating Cipher and Application to AES

4.1 Key-Alternating Cipher and FFT Key Recovery

The key-alternating cipher [5] is one of the most popular block cipher structures. Let P_i be an n-bit permutation, and the key-alternating cipher is expressed as

$$c = K_r \oplus P_r(\cdots \oplus P_3(K_2 \oplus P_2(K_1 \oplus P_1(K_0 \oplus p)))),$$

where $K_0, K_1, \ldots, K_r$ are round keys that are calculated from the master key. Let p and c be a plaintext and a ciphertext, respectively. Similar to the case with the Even-Mansour scheme, the FFT key recovery is useful in evaluating the complexity on the integral attack.

We first split r rounds into r_1 and r_2 rounds as $r = r_1 + r_2$ (see Fig. 3). We assume that a key-alternating cipher has an r_1-round integral distinguisher with

N chosen plaintexts. Moreover, we need to guess a k-bit key which is required for this distinguisher for the ciphertext side, and any one bit of output from the r_1 rounds is diffused to k_2 bits by r_2 rounds. Let $F_{2,K'}$ be a function from k_2 bits to one bit, where K' is k_1-bit key that is calculated from $K_{r-r_2}, K_{r-r_2+1}, \ldots, K_{r-1}$. In this case, the key recovery can be expressed as

$$\bigoplus_{i=1}^{N} F_{2,K'}(c'_i \oplus K'_r) = 0,$$

where c'_i and K'_r are truncated to k_2 bits from c_i and K_r, respectively. To apply the FFT key recovery, we first guess correct K', and we have

$$\sum_{i=1}^{N} F_{2,K'}(c'_i \oplus K'_r) = 0 \bmod 2.$$

We can efficiently evaluate this equation using the FWHT. Thus, the time complexity is $O(2^{k_1} \times k_2 2^{k_2})$.

4.2 FFT Key Recovery against AES

We show the FFT key recovery for the integral attack against a 6-round AES (see Fig. 1). Since the FFT key recovery only calculates the summation on the ring of integers, we transform Eq. (1) to

$$\begin{aligned}&\sum_{n=1}^{2^{32}} S_5^{(i)}(S_1(c_n[1] \oplus K_1) \oplus S_2(c_n[2] \oplus K_2) \oplus S_3(c_n[3] \oplus K_3)\\ &\qquad\qquad\qquad\qquad \oplus S_4(c_n[4] \oplus K_4) \oplus K_5)\\ &= \sum_{n=1}^{2^{32}} f_{K_5}^{(i)}(F(c_n \oplus (K_1 \| K_2 \| K_3 \| K_4))), \qquad (4)\end{aligned}$$

where F is a function from $\{0,1\}^{32}$ to $\{0,1\}^{32}$. Moreover, $f_{K_5}^{(i)}$ is a Boolean function whose output is the i-th bit of the output of S_5. We first guess K_5, and then calculate this summation using the FWHT. According to the description in Sect. 2.2, the time complexity is $4 \times 32 \times 2^{32} = 2^{39}$ additions for every K_5. Moreover, since K_5 is 8 bits, the time complexity is $2^8 \times 2^{39} = 2^{47}$ additions. The probability that this value is even for incorrect keys is expected to be 2^{-1}, but this value is always even for the correct key.

We estimate the time complexity to recover 5 keys $K_1, K_2, \ldots, K_5$ because Ferguson *et al.* estimated it in [11]. Since the output of S_5 is 8 bits, we repeat this analysis using the 8 bits. The probability that all 8 bits satisfy $\mathcal{B}$ for incorrect key is 2^{-8}. Since the total bit length of $K_1, K_2, \ldots, K_5$ is 40 bits, we repeat the above attack using 6 different sets. Thus, the total complexity is approximately the time of $6 \times 8 \times 2^{47} = 6 \times 2^{50}$ additions. When we use the partial-sum technique, the total time complexity is approximately the time of 6×2^{50} S-box lookups. We discuss the differences between them in Sect. 6.

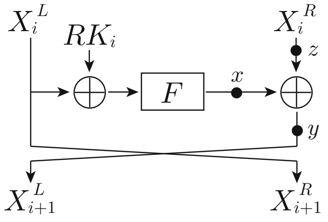

Fig. 4. Round function of Feistel cipher

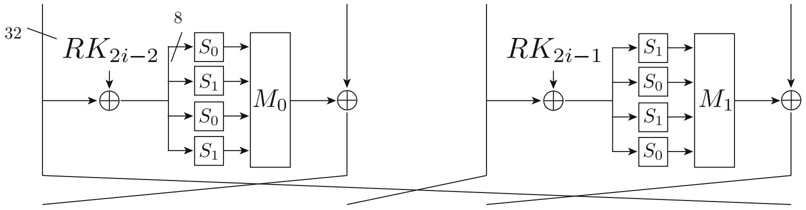

Fig. 5. Round function of CLEFIA

5 Integral Attack against Feistel Cipher and Application to CLEFIA

5.1 Feistel Cipher and FFT Key Recovery with MITM Technique

The Feistel cipher is commonly used to construct block ciphers. State X_i is separated into the left half, X_i^L, and the right half, X_i^R, and each half is updated as shown in Fig. 4. In 2012, Sasaki *et al.* proposed the MITM technique for the integral attack. Generally, the integral characteristics of the Feistel cipher satisfy $\mathcal{B}$ in the right half, namely $\bigoplus z$ becomes 0. In the MITM technique, we evaluate $\bigoplus x$ and $\bigoplus y$ independently instead of $\bigoplus z$. Then, we search for keys satisfying $\bigoplus x = \bigoplus y$ through analysis such as the MITM attack [8]. In [21], the partial-sum technique is used to evaluate $\bigoplus x$ and $\bigoplus y$, but the FFT can also be used to evaluate them.

We assume that we need to guess k_1 and k_2 bits to evaluate $\bigoplus x$ and $\bigoplus y$, respectively. If the round key is XORed with input from function F, the FFT key recovery can evaluate $\bigoplus x$ and $\bigoplus y$ with $O(k_1 2^{k_1})$ and $O(k_2 2^{k_2})$, respectively. Since the matching step of MITM analysis requires the time complexity of $O(\max\{2^{k_1}, 2^{k_2}\})$, the total time complexity of the integral attack is $O(k_1 2^{k_1} + k_2 2^{k_2})$.

5.2 CLEFIA

CLEFIA is a 128-bit block cipher, which was proposed by Shirai *et al.* in 2007. It has a 4-branch generalized Feistel network, and is adopted as an ISO/IEC standard. The round function is defined as in Fig. 5, and the i-th round output

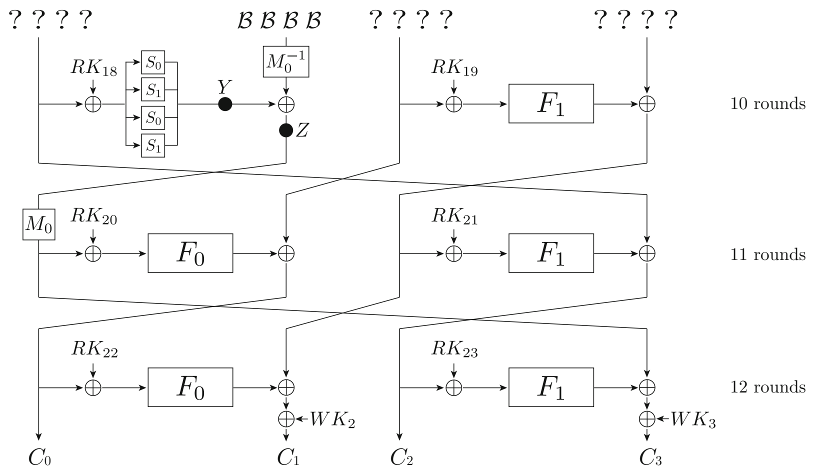

Fig. 6. Key recovery of 12-round CLEFIA

is calculated from the $(i-1)$-th round output, RK_{2i-2} and RK_{2i-1}. Moreover, the whitening keys, WK_0 and WK_1, are used in the first round, and WK_2 and WK_3 are used in the last round. For the 128-bit security version, the number of rounds is 18.

Shirai *et al.* showed that CLEFIA has an 8-round integral distinguisher in the proposal of CLEFIA [2], and then Li *et al.* showed that it has a 9-round integral distinguisher with 2^{112} chosen plaintexts [15]. Moreover, Sasaki and Wang showed that the complexity of the integral attack against a 12-round CLEFIA is 13×2^{106} S-box lookups using the MITM technique.

5.3 FFT Key Recovery against CLEFIA

We show the FFT key recovery against a 12-round CLEFIA. We first define some notations. Let C_1, C_2, C_3, and C_4 be ciphertexts and each value is 32 bits (see Fig. 6). Let X be any 32-bit value, and $X[i]$ denotes the i-th byte of X, namely $X = X[1]\|X[2]\|X[3]\|X[4]$. We define function $f_i : \{0,1\}^{32} \to \{0,1\}^8$ as

$$f_1(X)\|f_2(X)\|f_3(X)\|f_4(X) = F_1(X).$$

We first use the same method as the MITM technique [21], which applies the MITM attack [8] in the key recovery of the integral attack. It uses a 9-round integral distinguisher [15], where the second branch of the ninth round output satisfies $\mathcal{B}$. To optimize the MITM technique, we equivalently move the position of M_0 in the tenth round as shown in Fig. 6. As a result, we have $\bigoplus Y = \bigoplus Z$. Therefore, if $\bigoplus Y$ and $\bigoplus Z$ can be calculated with the guessed round keys, we can recover the secret key from the MITM technique. Let $\bigoplus Y$ and $\bigoplus Z$ be calculated as

$$\bigoplus Y = \bigoplus S(F_1(F_0(C_0 \oplus RK_{22}) \oplus C_1 \oplus RK_{21} \oplus WK_2) \oplus C_2 \oplus RK_{18}),$$
$$\bigoplus Z = \bigoplus M_0^{-1}(F_1(C_2 \oplus RK_{23}) \oplus C_3 \oplus WK_3),$$

where S denotes the concatenation of 4 S-boxes S_0, S_1, S_0, and S_1. In [21], each value is calculated using the partial-sum technique.

Hereinafter, we use the FFT key recovery. We need to guess 96 bits to evaluate $\bigoplus Y$. Since WK_3 does not affect $\bigoplus Z$, we need to guess 32 bits to evaluate $\bigoplus Z$. Clearly, the time complexity to evaluate $\bigoplus Z$ is negligible compared to that to evaluate $\bigoplus Y$. Therefore, we show the complexity required to evaluate $\bigoplus Y$. For instance, the first byte of $\bigoplus Y$ is calculated as

$$\begin{aligned}&\bigoplus Y[1]\\ &= \bigoplus S_0(f_1(F_0(C_0 \oplus RK_{22}) \oplus C_1 \oplus RK_{21} \oplus WK_2) \oplus C_2[1] \oplus RK_{18}[1]).\end{aligned}$$

To execute the FFT key recovery, we transform the above equation to

$$\begin{aligned}&\sum Y[1]^{(i)}\\ &= \sum S_0^{(i)}(f_1(F_0(C_0 \oplus RK_{22}) \oplus C_1 \oplus RK_{21} \oplus WK_2) \oplus C_2[1] \oplus RK_{18}[1]),\end{aligned}$$

where $Y[1]^{(i)}$ denotes the i-th bit of $Y[1]$ and the output of $S_0^{(i)}$ is the i-th bit of the output of S_0. Moreover, this equation is transformed by defining function f as

$$\sum Y[1]^{(i)} = \sum f\left((C_0 \| C_1 \| C_2[1]) \oplus (RK_{22} \| (RK_{21} \oplus WK_2) \| RK_{18}[1])\right).$$

From the description in Sect. 2.2, we can evaluate this equation for all possible $(RK_{22} \| (RK_{21} \oplus WK_2) \| RK_{18}[1])$ with $4 \times 72 \times 2^{72} \approx 2^{80.2}$ additions. Similarly, we evaluate $\sum Z[1]^{(i)}$ using the FFT key recovery, but the time complexity is negligible. Finally, we search for round keys satisfying $\sum Y[1]^{(i)} = \sum Z[1]^{(i)} \bmod 2$ using analysis such as the MITM attack. Since the complexity is approximately 2^{72}, it is also negligible.

Since the output of S_0 is 8 bits, we repeat this analysis for the eight bits. Moreover, we similarly calculate the second, third, and fourth bytes of $\bigoplus Y$ and $\bigoplus Z$. Therefore, the time complexity is approximately $4 \times 8 \times 2^{80.2} = 2^{85.2}$ additions. The probability that all 32 bits satisfy $\mathcal{B}$ for incorrect keys is expected to be 2^{-32}. Since the total bit length of RK_{18}, $RK_{21} \oplus WK_2$, RK_{22}, and RK_{23} is 128 bits, we repeat above analysis using 5 different sets. Thus, the total complexity is approximately $5 \times 2^{85.2} = 2^{87.5}$ additions.

6 Discussion

We compare the FFT key recovery and the partial-sum technique. We first compare them based on their units of complexity. The complexity of the partial-sum technique is estimated from the number of S-box lookups. On the other hand, that of the FFT key recovery is estimated from the number of additions. Since the two processing speeds depend on the environment, we cannot directly compare them. However, we can roughly compare them. In the partial-sum technique, we need at least the time complexity of $O(2^{k+\ell})$, where ℓ denotes the bit length of guessed key when we partially compute the sum, *e.g.*, $\ell = 8$ for AES and $\ell = 32$ for CLEFIA. We expect that the FFT key recovery is superior to the partial-sum technique when ℓ is greater than 8. Second, we compare them based on another aspect. We compare them based on memory access. The partial-sum technique randomly accesses memories. On the other hand, the FFT key recovery sequentially accesses memories. Generally, sequential access is more efficient than the random access.

We can further optimize the FFT key recovery against specific block ciphers. For instance, if we repeat the attack for different chosen plaintext sets, we do not need to calculate $\hat{v}$ every time. We can use the same $\hat{v}$ several times. Moreover, if we use the same set of ciphertexts, we do not need to calculate $\hat{w}$ every time. Namely, we can use the same $\hat{w}$ several times. Thus, the complexity of the FFT key recovery can be reduced using these properties.

We have an open problem regarding the FFT key recovery. Since round keys of block ciphers are calculated from the secret key, some bits of round keys are automatically recovered if some bits of the secret key are recovered. The partial-sum technique can utilize this property and efficiently reduce the complexity. For instance, the integral attack against a 22-round LBlock utilizes this property [20]. However, in the FFT key recovery, we do not yet know how to utilize this property.

7 Conclusion

We proposed a new technique for the integral attack called the FFT key recovery. This technique is useful in an integral attack with an enormous number of chosen texts. Moreover, the time complexity only depends on the bit length of keys that are required for a distinguisher from the ciphertext side. Therefore, we can easily estimate the time complexity. We focus on three structures, the Even-Mansour scheme where the block size is k bits; the key-alternating cipher where the block size is k bits, and k_2 bits are required to evaluate the integral distinguisher; and the Feistel cipher where k_1 and k_2 bits are used to evaluate the MITM integral key recovery. The time complexity is $O(k2^k)$, $O(k_2 2^k)$, and $O(k_1 2^{k_1} + k_2 2^{k_2})$, respectively. As applications of the three structures, we show that 8-round PRØST $\tilde{P}_{128,K}$, a 6-round AES, and a 12-round CLEFIA can be attacked with 2^{80}, $2^{52.6}$, and $2^{87.5}$ additions, respectively.

References

1. ISO/IEC 29192-2. Information technology - Security techniques - Lightweight cryptography - Part 2: Block ciphers, http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=56552

2. The 128-bit Blockcipher CLEFIA Security and Performance Evaluations. Sony Corporation (2007)
3. Andreeva, E., Bogdanov, A., Dodis, Y., Mennink, B., Steinberger, J.P.: On the Indifferentiability of Key-Alternating Ciphers. In: Canetti, R., Garay, J.A. (eds.) CRYPTO 2013, Part I. LNCS, vol. 8042, pp. 531–550. Springer, Heidelberg (2013)
4. Bogdanov, A., Geng, H., Wang, M., Wen, L., Collard, B.: Zero-Correlation Linear Cryptanalysis with FFT and Improved Attacks on ISO Standards Camellia and CLEFIA. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 306–323. Springer, Heidelberg (2013)
5. Bogdanov, A., Knudsen, L.R., Leander, G., Standaert, F.-X., Steinberger, J., Tischhauser, E.: Key-Alternating Ciphers in a Provable Setting: Encryption Using a Small Number of Public Permutations. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 45–62. Springer, Heidelberg (2012)
6. Collard, B., Standaert, F.-X., Quisquater, J.-J.: Improving the Time Complexity of Matsui's Linear Cryptanalysis. In: Nam, K.-H., Rhee, G. (eds.) ICISC 2007. LNCS, vol. 4817, pp. 77–88. Springer, Heidelberg (2007)
7. Daemen, J., Knudsen, L.R., Rijmen, V.: The Block Cipher SQUARE. In: Biham, E. (ed.) FSE 1997. LNCS, vol. 1267, pp. 149–165. Springer, Heidelberg (1997)
8. Diffie, W., Hellman, M.E.: Exhaustive cryptanalysis of the NBS Data Encryption Standard. Computer 10, 74–84 (1977)
9. Dunkelman, O., Keller, N., Shamir, A.: Minimalism in Cryptography: The Even-Mansour Scheme Revisited. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 336–354. Springer, Heidelberg (2012)
10. Even, S., Mansour, Y.: A Construction of a Cipher from a Single Pseudorandom Permutation. J. Cryptology 10(3), 151–162 (1997)
11. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: Improved Cryptanalysis of Rijndael. In: Schneier, B. (ed.) FSE 2000. LNCS, vol. 1978, pp. 213–230. Springer, Heidelberg (2001)
12. Hong, D., et al.: HIGHT: A New Block Cipher Suitable for Low-Resource Device. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 46–59. Springer, Heidelberg (2006)
13. Kavun, E.B., Lauridsen, M.M., Leander, G., Rechberger, C., Schwabe, P., Yalçin, T.: PRØST v1 (2014), Submission to CAESAR competition
14. Knudsen, L.R., Wagner, D.: Integral Cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Heidelberg (2002)
15. Li, Y., Wu, W., Zhang, L.: Improved Integral Attacks on Reduced-Round CLEFIA Block Cipher. In: Jung, S., Yung, M. (eds.) WISA 2011. LNCS, vol. 7115, pp. 28–39. Springer, Heidelberg (2012)
16. Matsui, M.: Linear Cryptanalysis Method for DES Cipher. In: Helleseth, T. (ed.) EUROCRYPT 1993. LNCS, vol. 765, pp. 386–397. Springer, Heidelberg (1994)
17. National Institute of Standards and Technology: Specification for the ADVANCED ENCRYPTION STANDARD (AES). Federal Information Processing Standards Publication 197 (2001)
18. Nguyen, P.H., Wei, L., Wang, H., Ling, S.: On Multidimensional Linear Cryptanalysis. In: Steinfeld, R., Hawkes, P. (eds.) ACISP 2010. LNCS, vol. 6168, pp. 37–52. Springer, Heidelberg (2010)
19. Nguyen, P.H., Wu, H., Wang, H.: Improving the Algorithm 2 in Multidimensional Linear Cryptanalysis. In: Parampalli, U., Hawkes, P. (eds.) ACISP 2011. LNCS, vol. 6812, pp. 61–74. Springer, Heidelberg (2011)

20. Sasaki, Y., Wang, L.: Comprehensive Study of Integral Analysis on 22-Round LBlock. In: Kwon, T., Lee, M.-K., Kwon, D. (eds.) ICISC 2012. LNCS, vol. 7839, pp. 156–169. Springer, Heidelberg (2013)
21. Sasaki, Y., Wang, L.: Meet-in-the-Middle Technique for Integral Attacks against Feistel Ciphers. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 234–251. Springer, Heidelberg (2013)
22. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-Bit Blockcipher CLEFIA (Extended Abstract). In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 181–195. Springer, Heidelberg (2007)
23. Wu, W., Zhang, L.: LBlock: A Lightweight Block Cipher. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 327–344. Springer, Heidelberg (2011)

Appendix A: 4-round Integral Distinguisher of AES

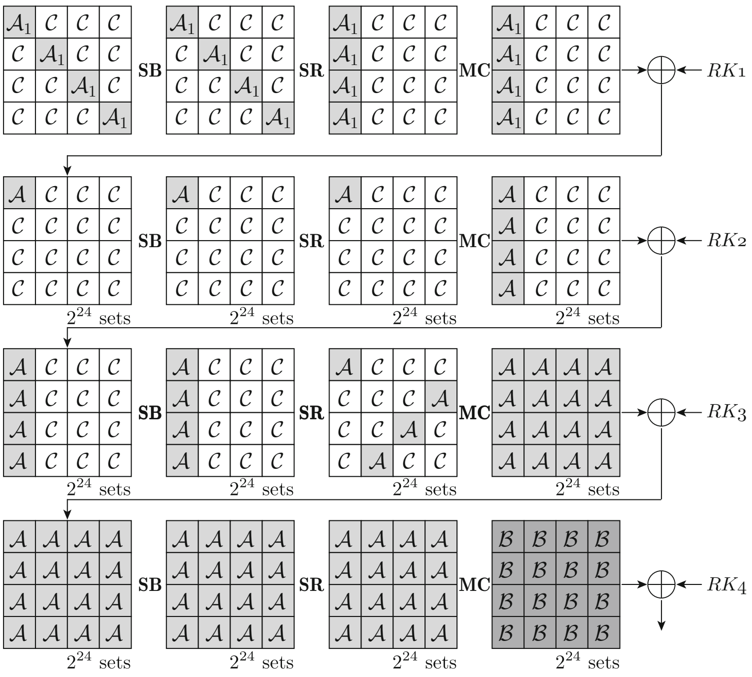

Fig. 7. 4-round integral distinguisher of AES

Figure 7 shows the 4-round integral distinguisher of AES. In the first round, 4 values satisfy $\mathcal{A}_1$, where the concatenation of their values also satisfies $\mathcal{A}$. This distinguisher uses 2^{32} chosen plaintexts, and each byte after encrypting 4 rounds satisfies $\mathcal{B}$.

Appendix B: Specification of PRØST

PRØST is an authenticated encryption scheme, which was submitted to the CAESAR competition. Refer to the original specification [13] and the reference implementation[3] for details.

PRØST permutation P_n ($n = 128$ or 256) adopts a substitution-permutation network, and inputs $2n$ bits and outputs $2n$ bits. We call a $2n$-bit string a state, and a 4-bit string a nibble. A state is represented as $4 \times d$ nibbles, where $d = 16$ and 32 for $n = 128$ and 256, respectively. We also refer to a four-nibble column as a slice. Figure 8 shows the state of PRØST, where the figure on the left shows the state of PRØST in [13] and the figure on the right shows our 2-dimensional representation, whose top-left square marked in light-gray is the origin of the columns and rows.

PRØST permutation consists of T rounds, where $T = 16$ and 18 for $n = 128$ and 256, respectively. The i-th round function is defined as

$$R_i := \texttt{AddConstant} \circ \texttt{ShiftPlanes} \circ \texttt{SubSlices} \qquad \text{for } i = 1, 2, \ldots, T$$

`SubSlices` substitutes each slice using a super S-box. A super S-box replaces 16 bits, and it consists of 4 S-boxes and a multiplication by 16×16-bit matrix M. `ShiftPlanes` cyclically shifts the j-th row by $\pi_{2-(i \bmod 2)}(j)$ nibbles to the left for the i-th round, where π_i is defined in Table 3.

`AddConstant` XORs a round constant.

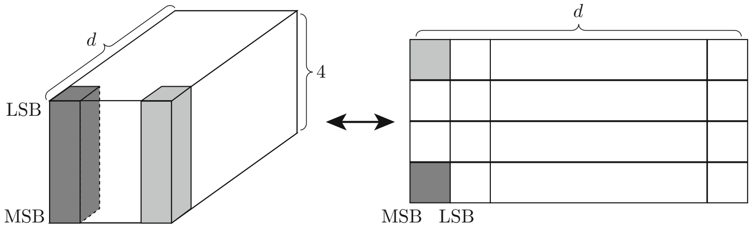

Fig. 8. Two different representations for the state of PRØST

Table 3. Definition of π_1 and π_2

	$n = 128$				$n = 256$			
	1	2	3	4	1	2	3	4
π_1	0	2	4	6	0	4	12	26
π_2	0	1	8	9	1	24	26	31

[3] The reference implementation can be obtained from the `crypto_aead/proest*` directory in the SUPERCOP package which is available at `http://bench.cr.yp.to/supercop.html`.

Appendix C: Integral Distinguisher of PRØST

We experimentally search for integral distinguishers of PRØST. We set a slice of the second round input as $\mathcal{A}$ and observe the sixth round output. The XOR of the output depends on the value of the constant nibbles of the second round input. However, we can expect that bit positions whose XOR values are always zero are $\mathcal{B}$ by changing the value of the constant nibbles of the second round input. We try 1024 randomly chosen values for the constant nibbles, and the number of trials is sufficient to determine that the output bits satisfy $\mathcal{B}$ by assuming that the non-$\mathcal{B}$ bits uniformly take 0 and 1.

Through the experiment, we obtain the 5-round integral distinguisher of $\tilde{P}_{128,K}(x)$ with 2^{16} chosen plaintexts (see Fig. 9). This distinguisher is extended to the 6-round distinguisher as shown in Fig. 9, and it uses 2^{64} chosen plaintexts.

Similarly, we show the integral distinguisher of $\tilde{P}_{256,K}(x)$. We first prepare chosen plaintexts where each slice satisfies $\mathcal{A}$, and this distinguisher is extended to the 7-round distinguisher with 2^{64} chosen plaintexts. When the first slice (16 bits) satisfies $\mathcal{A}$, the seventh round output satisfy the following integral characteristic.

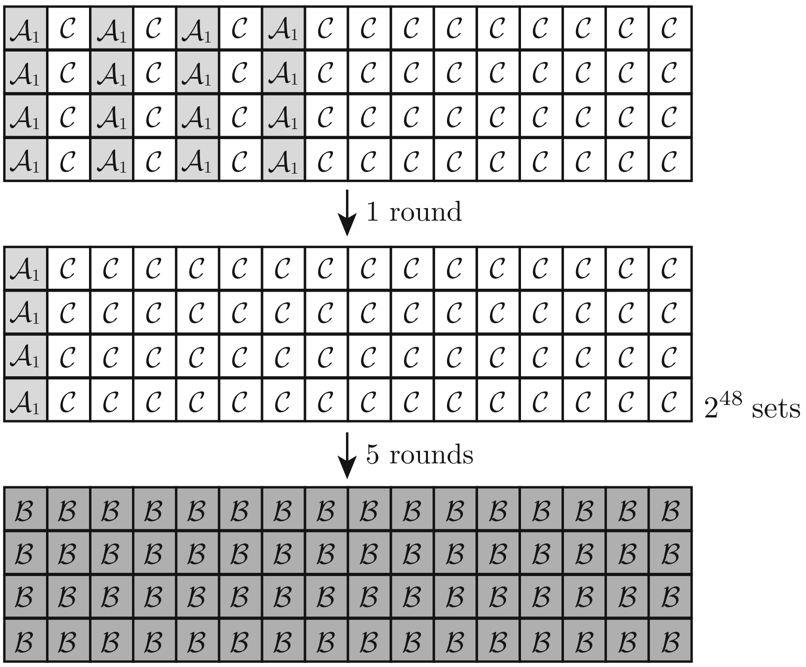

Fig. 9. 6-round integral distinguisher of $\tilde{P}_{128,K}$

```
0x30f0c00d3dc930cd090f0d0000d09000
0xd00c9fd390dc030d0f0000c0300090d0
0xd0930c0f0d000030c000d0d09003df9c
0x90d000c0f03009cd3dc0390d0d0f0000
```

Here, if the hexadecimal value in the i-th row and j-th slice is `0x3`, the upper two bits satisfy $\mathcal{B}$ and the lower two bits do not satisfy $\mathcal{B}$. As another example, when the seventeenth slice (16 bits) satisfies $\mathcal{A}$, the seventh round output satisfy the following integral characteristic.

```
0x090f0d0000d0900030f0c00d3dc930cd
0x0f0000c0300090d0d00c9fd390dc030d
0xc000d0d09003df9cd0930c0f0d000030
0x3dc0390d0d0f000090d000c0f03009cd
```

In both integral characteristics, 348 bits of the seventh round output satisfy $\mathcal{B}$.

Appendix D: FFT Key Recovery against PRØST $\tilde{P}_{256,K}$

In Appendix C, we show two 7-round integral distinguishers with 2^{64} chosen plaintexts. In each distinguisher, only 348 bits of the seventh round output satisfy $\mathcal{B}$. We show the FFT key recovery using these two distinguishers. The sum of the number of bits that the two distinguishers do not satisfy $\mathcal{B}$ is 24 bits. Therefore, 488 bits of the output of the 7-round integral characteristic satisfy $\mathcal{B}$. We repeat the FFT key recovery 488 times. The time complexity is the time of $488 \times 2^{72} = 2^{80.9}$ additions. Since the probability that all 488 bits satisfy $\mathcal{B}$ for incorrect key K is 2^{-488} at most, and 2^{24} incorrect keys are expected to remain. Therefore, we exhaustively search for the remaining keys. The time complexity is 2^{24}, and it is negligible compared to the time complexity for the two FFT key recoveries.