Hindawi
Security and Communication Networks
Volume 2022, Article ID 4186874, 13 pages
https://doi.org/10.1155/2022/4186874

Research Article

Rotational Differential-Linear Attack on Chaskey

Yuan Qin,[1,2] **Gaoli Wang**,[1,2] **and Guoyan Zhang**[3,4,5]

[1]*Shanghai Key Laboratory of Trustworthy Computing, Software Engineering Institute, East China Normal University, Shanghai, China*
[2]*State Key Laboratory of Cryptology, P.O. Box 5159, Beijing, China*
[3]*Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Jinan, China*
[4]*School of Cyber Science and Technology, Shandong University, Jinan, China*
[5]*Shandong Institute of Blockchain, Jinan, China*

Correspondence should be addressed to Gaoli Wang; glwang@sei.ecnu.edu.cn

Received 18 March 2022; Revised 13 June 2022; Accepted 18 July 2022; Published 25 August 2022

Academic Editor: Yong Yu

Rotational/Rotational-XOR differential-linear attack is a kind of powerful attack on ARX primitives. In this paper, we analyse the message authentication code algorithm—Chaskey—by using rotational differential-linear attack and using partitioning technique for key recovery. We presentthe results of distinguishing attack under weak-key and related-key settingswith correlation $2^{-55.636}$ for 9-round Chaskey. This is for the first time rotational differential-linear distinguisher has been used to analyse Chaskey. It is concluded that Chaskey can resist against rotational differential-linear attack.

1. Introduction

MAC (Message Authentication Code) is a small piece of information generated by a specific algorithm, which can check the integrity of a certain piece of message and is used for message authentication. It can be used to check whether the content of the message has been changed during message delivery, regardless of whether the reason for the change is accidental or deliberate. In addition, it can be used as the identity verification to confirm the source of message. Specifically, when given a message m, the MAC algorithm produces a tag τ by processing message m and secret key k. Then, one can send the combination of message and tag as (m, τ). It should be difficult for an attacker to forge a tag without knowing the key.

MAC algorithm can be constructed from other cryptography primitives. Chaksey[1] is a MAC algorithm, the permutation of which is based on ARX primitives. Since ARX primitives only consist of modular additions, bit rotations, and XORs, they are lightweight and can be implemented easily. The designer of Chaksey thought that it is lightweight and performs well in software as well as hardware. It is natural to think that the security level of MACs mainly depends on its permutation. Therefore, we analyse the permutation of Chaskey, which is presented in Section 4.

Rotational cryptanalysis [2–4] was first introduced by Dmitry Khovratovich et al. It is a probabilistic technique which mainly focuses on rotational pair $(m, m \lll l)$ through the rounds, where m is plaintext and $m \lll l$ is a rotation of m to the left with l bits. This attack is powerful to ARX primitives because the property of rotational pair is preserved by XOR and rotational operations, and also preserved by modular additions with high probability. The only prerequisite for successfully applying a rotation attack is that all constants used in the function maintain their values after rotation. Differential-linear attack [5] combines two most powerful techniques for symmetric-key cryptanalysis, proposed by Langford and Hellman at CRYPTO 1994. At EUROCRYPT 2021, Liu et al. proposed a new attack called rotational/rotational-XOR differential-linear attack [6], which combined the advantages of rotational attack and differential-linear attack.

1.1. Our Contributions. Rotational differential-linear attack combines the advantage of rotational attack and differential-linear attack, and it is powerful to analyse ARX

primitives. This attack can be divided into two parts. The first part is rotational attack. If there are no constant-xor or there is only rotational-invariable constant in function, the probability of rotational attack is only related to the number of modular additions. However, the conclusion is met when the cipher satisfies Markov cipher assumption, which results in a Markov chain assumption. Since there are some consecutive modular additions in the permutation of Chaksey, the Markov cipher assumption may not hold. In order to get precise results, we should consider the effects of consecutive modular additions. The second part is the linear attack. In this part, we transform the search for linear traces of high correlation into boolean satisfiability problem and use an automatic tool to search the best linear approximations.

In order to recover the whitening key, we consider some techniques in differential-linear attack. One of them is the partitioning technique proposed by Beierle et al. [8] at CRYPTO 2020. This technique mainly focuses on multiple linear characteristics rather than one to increase the correlation of linear approximations. We will explain it in detail in Section 3.

As a consequence, we get the distinguishing attack under the related-key setting with correlation $2^{-56.636}$ over 9 rounds of the permutation. Our results with the corresponding complexities are summarized in Tables 1 and 2, together with a comparison to the best attacks published so far. It is the first time to attack Chaskey by using rotational differential-linear attack with partitioning technique.

The structure of the remaining paper is as follows: Section 2 briefly describes some cryptanalysis methods and techniques for understanding content in this paper. Section 3 gives some examples to introduce the partitioning technique explicitly. Section 4 elaborates on how to attack Chaskey using all the techniques mentioned in the previous section and finally Section 5 is a short conclusion.

Table 1: Distinguishing attack on Chaskey.

Rounds	Probability	Attack type	Ref.
12	2^{-86}	Weak-key related-key forgery attack Using rotational attack	[7]
9	$2^{-111.272}$	Weak-key related-key distinguisher Using rotational differential-linear attack	Section 4

Table 2: Key recovery attack on Chaskey.

Rounds	Data	Time	Attack type	Ref
7	$2^{40.21}$	$2^{51.21}$	Differential-linear key recovery attack With partitioning technique	[8]
8	$2^{113.544}$	$2^{124.944}$	Weak-key related-key key recovery attack Using rotational differential-linear attack With partitioning technique	Section 4
12	2^{120}		Weak-key related-key key recovery attack Using rotational attack	[7]

2. Preliminaries

We use $\mathrm{rot}(x)$ to represent the binary vector x to rotate 1 bit to the left and $\mathrm{rot}^{-1}(x)$ to represent the binary vector x to rotate 1 bit to the right, respectively. For $\gamma, x \in F_2^n$, $\gamma \cdot x$ means the inner product of γ and x. $x[i]$ is used for denoting the i-th bit of x and $\overline{x}$ denotes the bitwise complement of x. We denote n dimensional binary vector that all bits are 0 by , the i-th unit vector by $[i]$ and the xor sum of vectors $[i_1], [i_2] \ldots [i_s]$ by $[i_1, i_2, \ldots, i_s]$ where n is the block size of a cipher. Considering a set $S \subseteq F_2^n$ and a Boolean function f: $F_2^n \longrightarrow F_2^n$, we define

$$Cor_{x \in S}[f(x)] = \frac{1}{|S|} \sum_{x \in S} (-1)^{f(x)}. \tag{1}$$

In most situations, the correlation of $f(x)$ is estimated by using a subset of F_2^n due to the limitation of computation resources.

2.1. Markov Ciphers and Differential Cryptanalysis. Differential cryptanalysis [9] is a probabilistic chosen-plaintext attack introduced by Biham and Shamir. We choose plaintext pair $(x, x\prime)$ with difference Δ and study how it propagates throughout the function. If an input difference Δ can yield an output difference δ whatever the middle differences are, (Δ, δ) is a differential characteristic. The differential probability (Δ, δ) is the number of pairs for which plaintext pair $(x, x\prime)$ with difference Δ and ciphertext pair $(y, y\prime)$ with difference δ over the total number of pairs with input difference Δ. However, in practice, it is hard to calculate differential probability. Since the differential characteristics can be regarded as a Markov chain, the differential probability is estimated by the product of intermediate probabilities [10].

Theorem 1 (see [7]). *A sequence of discrete random variables $(v_0, \ldots, v_r)$ is a Markov chain, for $0 < i < r$,*

$$\begin{aligned} &\Pr\left(v_{i+1} = \beta_{i+1} | v_i = \beta_i, v_{(i-1)} = \beta_{(i-1)}, \ldots, v_0 = \beta_0\right) \\ &\quad = \Pr\left(v_{i+1} = \beta_{i+1} | v_i = \beta_i\right). \end{aligned} \tag{2}$$

2.2. Boolean Satisfiability Problem and Automatic Search. In recent years, a lot of work that focuses on searching differential characteristic and linear characteristic automatically have come out. There are mainly two types of automation tools. One is Mixed Integer Linear Programming (MILP) and the other is boolean satisfiability problem. The aim of MILP is to optimize an objective function under certain constraints. MILP is usually used for searching differential characteristic and linear characteristic of a cipher whose substitution layer is S-box structure. This idea was first proposed by Sun et al. [11] and many methods which

can reduce search time or model for large S-box have been proposed recently [12–15].

Three or more SAT is NP-complete. But the solution can be found in a reasonable time for most practical problems. There are a large number of heuristic SAT solvers, which output satisfiable or unsatisfiable depending on whether problem can be solved. More specifically, SAT solvers assign initial values to variables, then compute the number of conflicting clauses, and decides the next step to search until a valid or no solution is found.

2.3. Rotational Attack and Differential-Linear Attack. Rotational cryptanalysis was first introduced by Dmitry Khovratovich et al. We use $x \lll r$ and $x \ggg r$ to represent the variable rotation to the left and right by r bits, respectively. A rotational pair can be denoted by $(x, x \lll r)$. It is easy to see that a rotational pair is preserved throughout bitwise XOR and any rotational operations. However, it is not always preserved by modular addition. Therefore, the probability of this attack only depends on the rotational probability of modular additions.

For large n and small r, Table 3 is obtained. It is clear to see when r is equal to 1, the probability is the highest, that is why we always choose 1 to be the rotational offset.

In Ref. [3], they claimed that when rotation amount r is fixed, the probability of rotational attack depends on the number of modular addition. But the prerequisite of the lemma is that variables in and round can form a Markov chain. When variables are not random and independent, the lemma will not give a precise answer. Due to some consecutive modular addition in the permutation of Chaskey, we realize that the probability of rotational attack may not only rely on the number of modular addition. In Ref. [7], they explored and proposed a lemma, which calculates the rotational probability of $k-1$ consecutive modular additions precisely. The result shows that when consecutive modular exits, the probability of rotational attack will be much lower.

Differential-linear attack [5, 16] was first introduced by Langford and Hellman. As shown in Figure 1, when considering a cipher E, it can be divided into two subparts E_1 and E_2, which are covered by differential characteristics and linear characteristics, respectively. Assume that input differences δ and output differences Δ for E_1 holds with probability

$$\Pr\left(x \oplus x' = \delta \xrightarrow{F} y \oplus y' = \Delta\right) = p. \tag{3}$$

Assume that input mask γ and output mask Γ for E_2 holds with bias

$$\epsilon_{\gamma,\Gamma} = \Pr\left[\gamma \cdot y \oplus \Gamma \cdot E_1(y) = 0\right] - \frac{1}{2}. \tag{4}$$

Assume that the variables in E_1 and E_2 are random and independent, the overall bias of differential-linear distinguisher is

$$\epsilon_{\delta,\Gamma} = \Pr\left[\Gamma \cdot \left(E(x) \oplus E(x \oplus \delta)\right) = 0\right] - \frac{1}{2} = (-1)^{\gamma \cdot \Delta} \cdot 2p\epsilon_{\gamma,\Gamma}^2. \tag{5}$$

Table 3: Rotational offset r and corresponding probability.

r	P_r	$\log_2(p_r)$
1	0.375	−1.1415
2	0.313	−1.676
3	0.281	−1.831

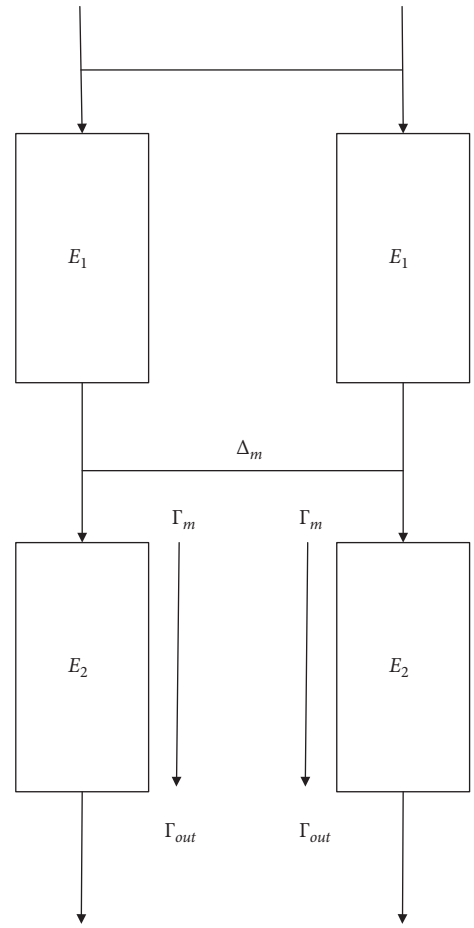

Figure 1: Differential-linear distinguisher.

The data complexity of differential-linear distinguisher is $O(p^{-2}\epsilon_{\gamma,\Gamma}^{-4})$.

In almost all practical ciphers, E_1 and E_2 are not independent. In recent years, many significant researches have come out to pursue more accurate results. For example, Bar-On et al. proposed Differential-Linear Connectivity Table (DLCT) [17] to deal with the dependencies between the differential and linear parts. Here, cipher E is divided into three parts: $E = E_2 \circ E_m \circ E_1$, and the intermediate part E_m is evaluated experimentally.

2.4. Rotational Differential-Linear Attack. Rotational differential-linear attack is an attack method that combines the advantage of rotational attack and differential-linear attack, the difference between rotational differential-linear distinguisher and differential-linear distinguisher is the differential part, which is replaced by a rotational differential

trace. Given a cipher E, it can be described as a cascade of two subparts such that $E = E_2 \circ E_1$. E_1 is covered by rotational differential with

$$Pr\Big(x \oplus (x \lll t) = \delta \xrightarrow{F} E_1(x) \oplus \big(E_1(x) \lll t\big) = \Delta\Big) = p, \quad (6)$$

and E_2 is covered by a linear approximation

$$\begin{cases} \epsilon_{\gamma,\Gamma} = \Pr\left[\gamma \cdot y \oplus \Gamma \cdot E_2(y) = 0\right] - \dfrac{1}{2} \\ \epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)} = \Pr\left[rot^{-1}(\gamma) \cdot y \oplus rot^{-1}(\Gamma) \cdot E_2(y) = 0\right] - \dfrac{1}{2}. \end{cases} \quad (7)$$

Then, the bias of a rotational differential-linear distinguisher can be represented as follows [6]:

$$\begin{aligned} \epsilon_{\delta,\Gamma} &= \Pr[\Gamma \cdot (rotE(x)) \oplus E(rot(x)) = 0] - \frac{1}{2} \\ &= (-1)^{\gamma \cdot \Delta} \cdot 2p\epsilon_{\gamma,\Gamma}\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}. \end{aligned} \quad (8)$$

Note that if rotational offset is 0, the rotational differential-linear distinguisher will be degenerated to differential-linear distinguisher. It is clear to see that the bias of rotational differential-linear attack depends on p, $\epsilon_{\gamma,\Gamma}$ and $\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}$. $p$is related to the first part, which is corresponding to rotational differential attack. $\epsilon_{\gamma,\Gamma}$ and $\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}$ are the correlations of linear attack. We will explain how to get a higher correlation of the distinguisher in Section 4.

3. Partitioning Technique

In this section, we use some examples in Ref. [8] to explain how to utilize partitioning technique briefly.

3.1. An Introduction of Partitioning Technique. Partitioning technique can improve the correlation of differential-linear attack powerfully by yielding linear equations between cipher and key, which was first proposed by Leurent at EUROCRYPT 2016 [18]. For ARX primitives, modular addition is the only nonlinear component. Assume $a, b \in F_2^n$, $s = a + b$, where + means modular addition. For $i = 0$, $s[i] = a[i] \oplus b[i]$. But for $i > 0 s[i]$ is not always equal to $a[i] \oplus b[i]$. In order to derive certain linear equation, the following lemma is given.

Lemma 1 (see [18]). *Let $a, b \in F_2^n$ and $s = a + b$. For $i \geq 2$, we have*

$$s[i] = \begin{cases} a[i] \oplus b[i] \oplus a[i-1] \text{if} a[i-1] = b[i-1], \\ a[i] \oplus b[i] \oplus a[i-2] \text{if} a[i)-1] \neq b[i)-1] \text{and} [i-2] = b[i-2]. \end{cases} \quad (9)$$

Lemma 1 shows the way that $s[i]$ can be calculated by a linear equation if some certain conditions are met with (a, b). As shown in Figure 2, we want to get specific equations on $z_0[i]$; the above will not work because the relation between z_0 and (y_0, y_1) are not the relation of modular addition but modular subtraction. Formally, it is expressed by $z_0 = y_0 - y_1$, here $-$ means modular subtraction. For this case, the following lemma can help solve the problem.

Lemma 2 (see [8]). *Let $i \geq 2 and let S_1 := \{(x_1, x_0) \in F_2^{2m} | x_0[i-1] \neq x_1[i-1]\} and S_2 := \{(x_1, x_0) \in F_2^{2m} | x_0[i-1] = x_1[i-1] and x_0[i-2] \neq x_1[i-2]\}$. Then,*

$$z_0[i] = \begin{cases} y_0[i] \oplus y_1[i] \oplus y_0[i-1] \oplus 1 \text{if} (y_1, y_0) \in S_1, \\ y_0[i] \oplus y_1[i] \oplus y_0[i-2] \oplus 1 \text{if} (y_1, y_0) \in S_2. \end{cases} \quad (10)$$

In the later section, we not only care about $z_0[i]$ but also $z_0[i, i-1]$. It is clear that Lemma 2 can be applied to $z_0[i]$ and $z_0[i-1]$ separately. Then $z_0[i, i-1]$ is evaluated by the equations which are yielded by applying Lemma 2 on $z_0[i]$ and $z_0[i-1]$. However, this method requires knowledge of three bits. The following lemma states that $z_0[i, i-1]$ is evaluated by knowing only two bits. Since $y_0 = c_0 \oplus k_0$ and $y_1 = c_1 \oplus k_1$, the following lemma helps us guess less bits to get linear equations.

Lemma 3 (see [8]). *Let $i \geq 2 and let S_3 := \{(x_1, x_0) \in F_2^{2m} | x_0[i-1] = x_1[i-1]\} and S_4 := \{(x_1, x_0) \in F_2^{2m} | x_0[i-1] \neq x_1[i-1] and x_0[i-2] \neq x_1[i-2]\}$. Then,*

$$z_0[i] \oplus z_0[i-1] = \begin{cases} y_0[i] \oplus y_1[i] \text{if} (y_1, y_0) \in S_3, \\ y_0[i] \oplus y_1[i] \oplus y_0[i-1] \oplus y_0[i-2] \oplus 1 \text{if} (y_1, y_0) \in S_4. \end{cases} \quad (11)$$

Next, we will explain partitioning technique explicitly. The structure of rotational differential-linear attack is shown in Figure 3. Note that E_1 is the rotational differential part, E_2 is the linear part, and F is the part that we use to recover the whitening key. The key idea of partitioning technique is that we want to use multiple linear approximations $\Gamma_{\text{out}}^{p_i} \cdot z = rot(\Gamma_{\text{out}}^{p_j}) \cdot \tilde{z}$ to recover some bits of key. The space of ciphertext F_2^n can be split into a direct sum $R \oplus P$. We denote the dimension of

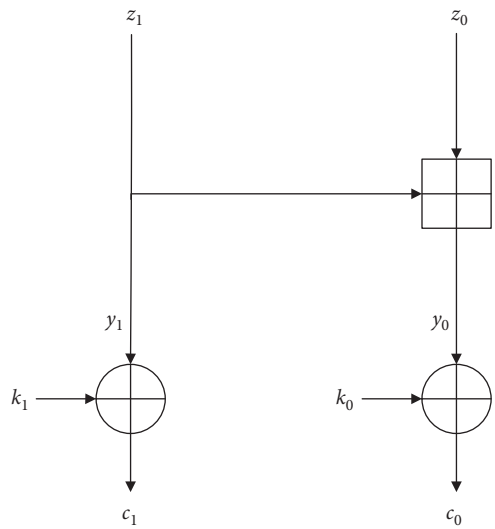

Figure 2: Sample for partitioning technique.

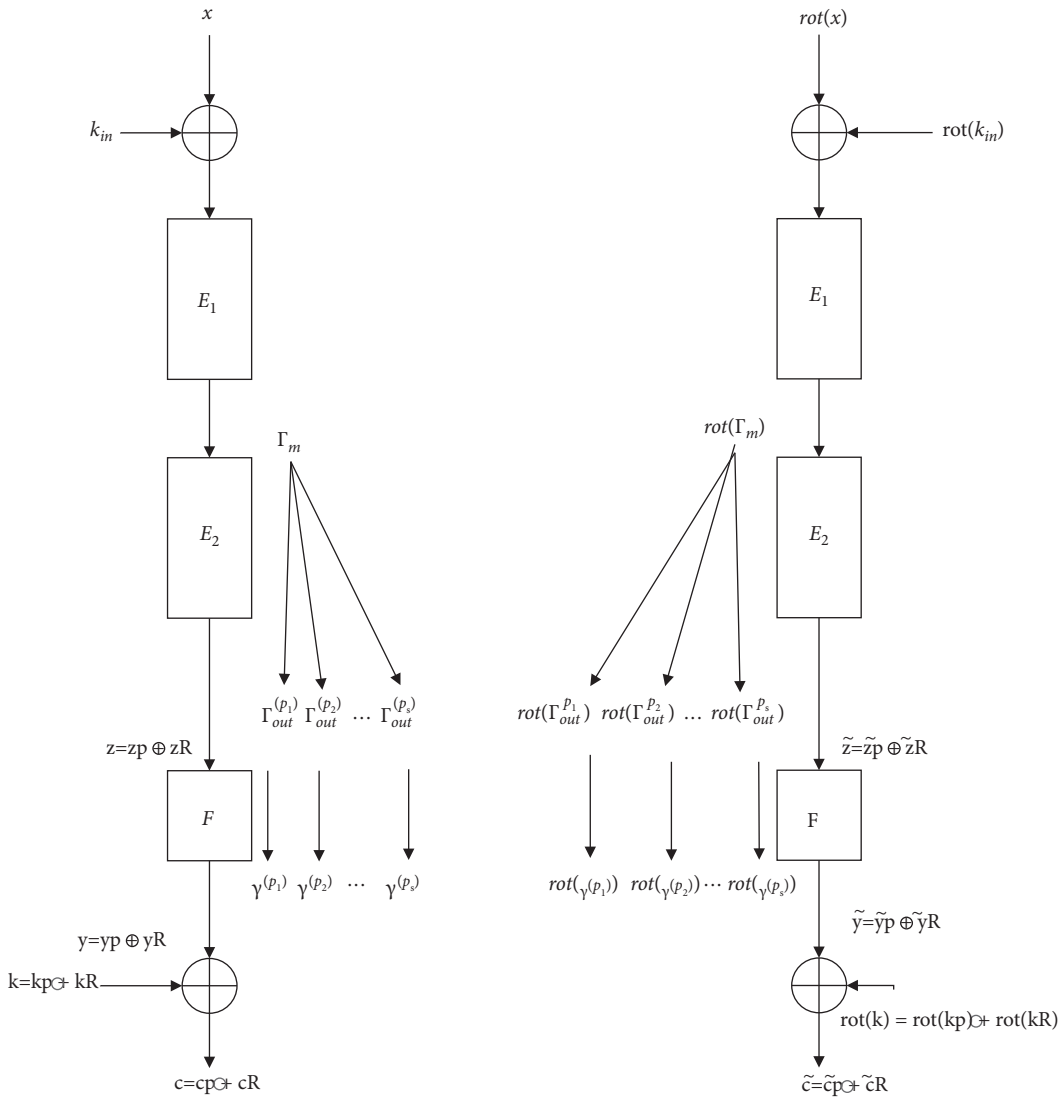

Figure 3: The structure of overall rotational differential-linear attack with partitioning technique.

P by n_P and the dimension of R by n_R. The specific definition of R and P depends on the attack. Considering some tuples $(\tau_{(p_i)}, \Gamma_{\text{out}}^{(p_i)}, \gamma^{(p_i)})$, where $\tau_{p_i} = R \oplus p_i$ is a coset of $R \subseteq F_2^n$, $\Gamma_{\text{out}}^{(p_i)} \in F_2^n$ and $\gamma^{(p_i)} \in R$. If we can get a high correlation,

$$\epsilon_i = Cor_{y \in \tau_{p_i}} \left[\Gamma_{\text{out}}^{p_i} \cdot z \oplus \gamma^{p_i} \cdot y\right] = 1. \tag{12}$$

The above equation reveals that if the ciphertext belongs to some specific subset (defined by $y \in \tau_{p_i}$), we yield some linear equation on k because $\gamma^{p_i} \cdot y = \gamma^{p_i} \cdot (c \oplus k)$ and c is the ciphertext that can be observed. If the ϵ_i is not equal to 1 but close to 1, we still obtain the equation on k with high correlation. Since $y = c \oplus k$, we can guess the n_P bits of k and split ciphertext into corresponding τ_{p_i}. After some equations on k are obtained with high correlation, bits on k_R will be recovered. The way to recover some bits on k_R is introduced as follows. We define

$$q_{ij} = Cor_{\substack{x \in \chi \text{suchthat} \\ (c,\tilde{c}) \in \tau_{p_i} \times rot\left(\tau_{p_j}\right) \oplus (k_P, rot(k_P))}} \left[\Gamma_{\text{out}}^{p_i} \cdot z \oplus \text{rot}^{-1}\left(\text{rot}\left(\Gamma_{\text{out}}^{p_j}\right) \cdot \tilde{z}\right)\right]. \tag{13}$$

We obtain

$$\begin{aligned} &Cor_{\substack{x \in \chi \text{suchthat} \\ (c,\tilde{c}) \in \tau_{p_i} \times \text{rot}\left(\tau_{p_j}\right) \oplus (k_P, \text{rot}(k_P))}} \left[\gamma^{(p_i)} \cdot c \oplus \text{rot}^{-1}\left(\text{rot}\left(\gamma^{(p_j)}\right) \cdot \tilde{c}\right) \oplus \left(\gamma^{(p_i)} \cdot k\right) \oplus \left(k \cdot \gamma^{(p_j)}\right)\right] \\ &= Cor_{\substack{x \in \chi \text{suchthat} \\ (c,\tilde{c}) \in \tau_{p_i} \times \text{rot}\left(\tau_{p_j}\right) \oplus (k_P, \text{rot}(k_P))}} \left[\gamma^{(p_i)} \cdot y \oplus \text{rot}^{-1}\left(\text{rot}\left(\gamma^{(p_j)}\right) \cdot \tilde{y}\right)\right] = \epsilon_i \epsilon_j q_{ij}. \end{aligned} \tag{14}$$

For $r \in \mathbf{R}$, let us define $\text{sgn}(r) = \begin{cases} 0 \text{if} r \geq 0, \\ 1 \text{if} r < 0. \end{cases}$; we define

$$h_{ij} = (-1)^{\text{sgn}\left(\epsilon_i \epsilon_j q_{ij}\right)} Cor_{\substack{x \in \chi \text{suchthat} \\ (c,\tilde{c}) \in \tau_{p_i} \times \text{rot}\left(\tau_{p_j}\right) \oplus (k_P, \text{rot}(k_P))}} \left[\gamma^{(p_i)} \cdot c \oplus \text{rot}^{-1}\left(\text{rot}\left(\gamma^{(p_j)}\right) \cdot \tilde{c}\right)\right], \tag{15}$$

We have $h_{ij} = (-1)^{(\gamma^{(p_i)} \cdot k \oplus \gamma^{(p_j)} \cdot k)} |\epsilon_i \epsilon_j q_{ij}|$. Let us define,

$$\left\{x \in \chi | (c, \tilde{c}) \in \tau_{p_i} \times rot\left(\tau_{p_j}\right) \oplus (k_P, rot(k_P))\right\}, \tag{16}$$

which is of equal size σ for all (i, j) and consider the scaled version of h_{ij}, i.e.,

$$\alpha_{ij} = \sigma \cdot h_{ij} = (-1)^{\text{sgn}\left(\epsilon_i \epsilon_j q_{ij}\right)} \sum_{x \in \chi \text{suchthat} (c,\tilde{c}) \in \tau_{p_i} \times \text{rot}\left(\tau_{p_j}\right) \oplus (k_P, \text{rot}(k_P))} (-1)^{\left(\gamma^{(p_i)} \cdot c\right) \oplus \text{rot}^{-1}\left(\text{rot}\left(\gamma^{(p_j)}\right) \cdot \tilde{c}\right)}. \tag{17}$$

For each $\gamma \in W = \text{Span}\left\{\gamma^{(p_i)} \oplus \gamma^{(p_j)} | i, j \in \{1, \ldots, s\}\right\}$, we define

$$\beta(\gamma) = \sum_{\substack{(i,j) \text{suchthat} \\ \gamma^{(p_i)} \oplus \gamma^{(p_j)} = \gamma}} \alpha_{(ij)}. \tag{18}$$

We can use this function β to recover W bits of information on k_R. k_R can be expressed as $k_L \oplus k_R'$, where k_L is the key that can be obtained from β. By using Fast Walsh–Hadamard Transform on β. Fast Walsh–Hadamard Transform reduces the time complexity from $O(2^{2n})$ to $O(n2^n)$.

We compute a cumulative counter.

$$C(k_P, k_L) = \sum_{\gamma \in W} (-1)^{\gamma \cdot k_L} \beta(\gamma), \tag{19}$$

for each tuple (k_P, k_L).

Note that when counter C is larger than some threshold Θ, we store the tuple (k_P, k_L) in the list of key candidates.

3.2. A Simple Example for Partition Point. As shown in Figure 2, we define partition point $\xi = (z_1[i], z_1[i] \oplus z_1[i-1])$. For $i \geq 2$, in order to compute $z_1[i]$ or $z_1[i] \oplus z_1[i-1]$ by using Lemmas 2 and 3, we need to know which condition some bits in the ciphertext meets. Considering some definitions in Section 2, a cipher space is split into the direct sum $P \oplus R$, P is complement space of the space

$$R = \left\{(x_1, x_0) \in F_2^{2m} | x_0[i-1] \oplus x_1[i-1] = 0 \text{and} x_0[i-2] \oplus x_1[i-2] = 0\right\}. \tag{20}$$

Therefore, the dimension of P is 2. For each element $p \in P$, it can be rewritten as $b_1 b_0$ where b_0 indicates the value of $\overline{x}_0[i-1] \oplus x_1[i-1]$ and b_1 indicates the value of $\overline{x}_0[i-2] \oplus x_1[i-2]$. To enumerate all possible values of p, we can yield the following four tuples $(\tau_{b_0 b_1}, \Gamma_{\text{out}}^{b_0 b_1}, \gamma^{b_0 b_1})$ and corresponding $\epsilon_{b_0 b_1}$ according to Lemmas 2 and 3:

$$\begin{aligned}
&\tau_{00} = R \oplus 00 = S_4 \quad \Gamma_{\text{out}}^{00} = (, [i, i-1]) \quad \gamma^{00} = ([i], [i, i-1, i-2]) \quad \epsilon_{00} = -1 \\
&\tau_{01} = R \oplus 01 = \frac{S_1}{S_4} \quad \Gamma_{\text{out}}^{01} = (, [i]) \quad \gamma^{01} = ([i], [i, i-1]) \quad \epsilon_{01} = -1 \\
&\tau_{10} = R \oplus 10 = S_2 \quad \Gamma_{\text{out}}^{10} = (, [i]) \quad \gamma^{10} = ([i], [i, i-2]) \quad \epsilon_{10} = -1 \\
&\tau_{11} = R \oplus 11 = \frac{S_3}{S_2} \quad \Gamma_{\text{out}}^{11} = (, [i, i-1]) \quad \gamma^{11} = ([i], [i]) \quad \epsilon_{11} = 1
\end{aligned} \tag{21}$$

We take the second tuple to illustrate the principle. When it is observed that $y \in \tau_{01}$, the linear masks $(\Gamma_{\text{out}}^{01}, \gamma^{01})$ are determined. Therefore, the correlation $\epsilon_{01} = Cor_{y \in \tau_{01}}[\Gamma_{\text{out}}^{01} \cdot z \oplus \gamma^{01} \cdot y]$ can be deducted by using Lemma 2. Then, we have $W = \text{Span}\ \{\gamma^a \oplus \gamma^b | a, b \in F_2^n\} = \{(,), (, [i-1]), (, [i]), (, [i, i-1])\}$. Two bits $k_0[i]$ and $k_0[i-1]$ will be recovered by using Fast Walsh–Hadamard Transform.

3.3. Two Consecutive Modular Additions for Partition Point. In this Section, we introduce some cases of consecutive modular additions. As shown in Figure 4, we still define partition point $\xi = (z_1[i], z_1[i] \oplus z_1[i-1])$. Note that $z_1[i]$ and $z_1[i] \oplus z_1[i-1]$ is related to some bits of c_0, c_1 and c_2. Formally,

$$z_1 = (((y_1 - y_0) \ggg b) - ((y_1 - y_0) \ggg c) \oplus (y_2 \ggg a)), \tag{22}$$

where means modular subtraction and $c = a + b$. Therefore, we get a 5-dimensional subspace P, which is a complement subspace of the subspace:

$$R = \left\{ \begin{array}{c} (x_2, x_1, x_0) \in F_2^{3m} | x_2[i_a - 1] \oplus x_1[i_b - 2] \oplus x_1[i_c - 2] = 0 x_0[i_b - 1] \oplus x_1[i_b - 1] = 0, \\ x_0[i_b - 2] \oplus x_1[i_b - 2] = 0, x_0[i_c - 1] \oplus x_1[i_c - 1] = 0, x_0[i_c - 2] \oplus x_1[i_c - 2] = 0 \end{array} \right\}, \tag{23}$$

where $i_a = i + a, i_b = i + b$ and $i_c = i + a + b$. For each element $p \in P$, it can be rewritten as $b_4 b_3 b_2 b_1 b_0$, where $b_4 b_3 b_2 b_1 b_0 = (y_2[i_a - 1] \oplus y_1[i_b - 2] \oplus y_1[i_c - 2], s[i_b - 1], s[i_b - 2], s[i_c - 1], s[i_c - 2])$ with $s = \overline{y}_0 \oplus y_1$. To enumerate all possible values of p, we can yield 2^5 tuples $(\tau_{(p_i)}, \Gamma_{\text{out}}^{(p_i)}, \gamma^{(p_i)})$ and the corresponding $\epsilon_{(p_i)}$. In these tuples, $\Gamma_{\text{out}}^{(p_i)} \in \{(, [i],), (, [i, i-1],)\}$, and the corresponding linear mask $\gamma^{(p_i)}$ involves these bits:

$$y_2[i_a], y_0[i_b], y_1[i_b - 1], y_1[i_b - 2], y_0[i_c], y_1[i_c], y_1[i_c - 1], y_1[i_c - 2]. \tag{24}$$

For the 2^5 tuples given above, 4 of them correspond to correlation $\epsilon = \pm 1$, 8 of them correspond to correlation $\epsilon = \pm 2^{-1}$, and 12 of them correspond to correlation $\epsilon = \pm 2^{-0.263}$. Therefore, there are 24 tuples with high correlation in total. The tuples with the corresponding correlations are listed in the appendix.

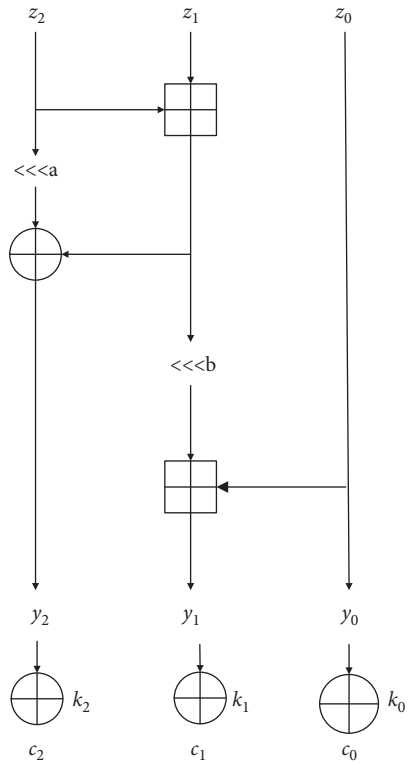

Figure 4: Case of two consecutive modular additions.

4. Application to Chaskey

4.1. Description of Chaskey. Chaskey, proposed by Nicky Mouha et al., is a very efficient Message Authentication Code (MAC) algorithm for 32-bit microcontrollers. Chaskey takes a 128-bit key k and processes a message m in 128-bit blocks using a 128-bit permutation π. Designers claimed that Chaskey performs well both in software and hardware, and it is an Even-Mansour construction. A message m is processed by n-bit key K into a tag τ. The message m is split into l blocks $m_1, m_2, \ldots, m_l$ of 128-bit each, except for the last block m_l which may be incomplete. If the last block m_l is not complete, a periodical binary string 10^* is pushed back until the size of m_l is 128-bit. This permutation π is based on the Addition–Rotation–XOR (ARX) design methodology. In practical analysis, we care about the permutation which is shown in Figure 5. When we want to tag a message m with one block of 128 bits, the tag would be $\tau = \pi(K \oplus K_1 \oplus m) \oplus K_1$. K_1 is generated by K, i.e., $K_1 = xK \bmod g(x)$, where $g(x) = x^{128} + x^7 + x^2 + x + 1$, as shown in Figure 6.

4.2. Rotational Differential-Linear Distinguishing Attack on Chaskey. In this section, we recall the weak-key class which was introduced in Ref. [7] at first. Then a 9-round rotational differential-linear distinguisher is represented. Finally, an 8-round key recovery attack of Chaskey with partitioning techinique is explained in detail.

4.2.1. Weak Key Class of Rotational Attack. Because of the consecutive modular additions in Chaskey, we need to

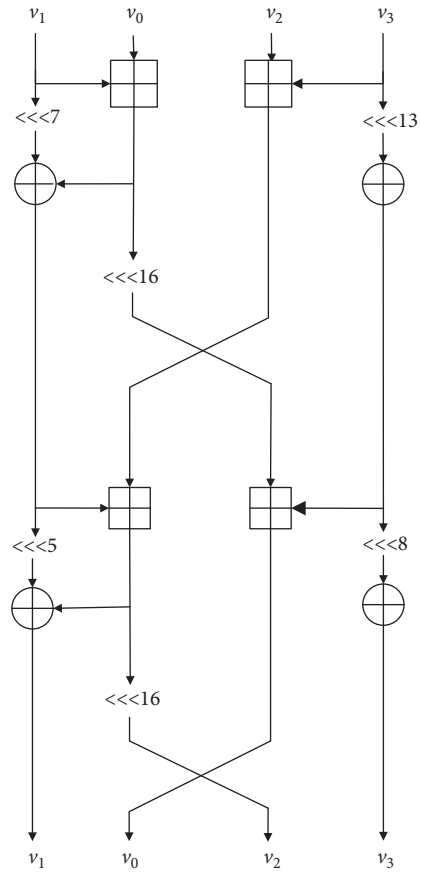

Figure 5: Permutation of Chaskey.

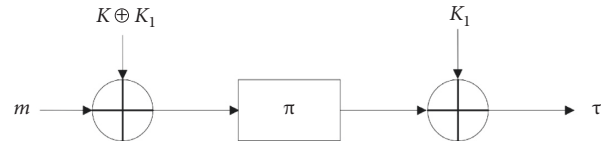

Figure 6: Mode of operation for message with single 128 bits block of Chaksey.

Table 4: Table with the expected rotational probabilities for any number of rounds of Chaskey.

Rounds	Number of chains modular additions	Expected probability
1	1	−6.436
2	3	−13.636
3	5	−20.836
4	7	−28.036
5	9	−35.236
6	11	−42.436
7	13	−49.636
8	15	−56.836
9	17	−64.036
10	19	−71.236
11	21	−78.436
12	23	−86.636

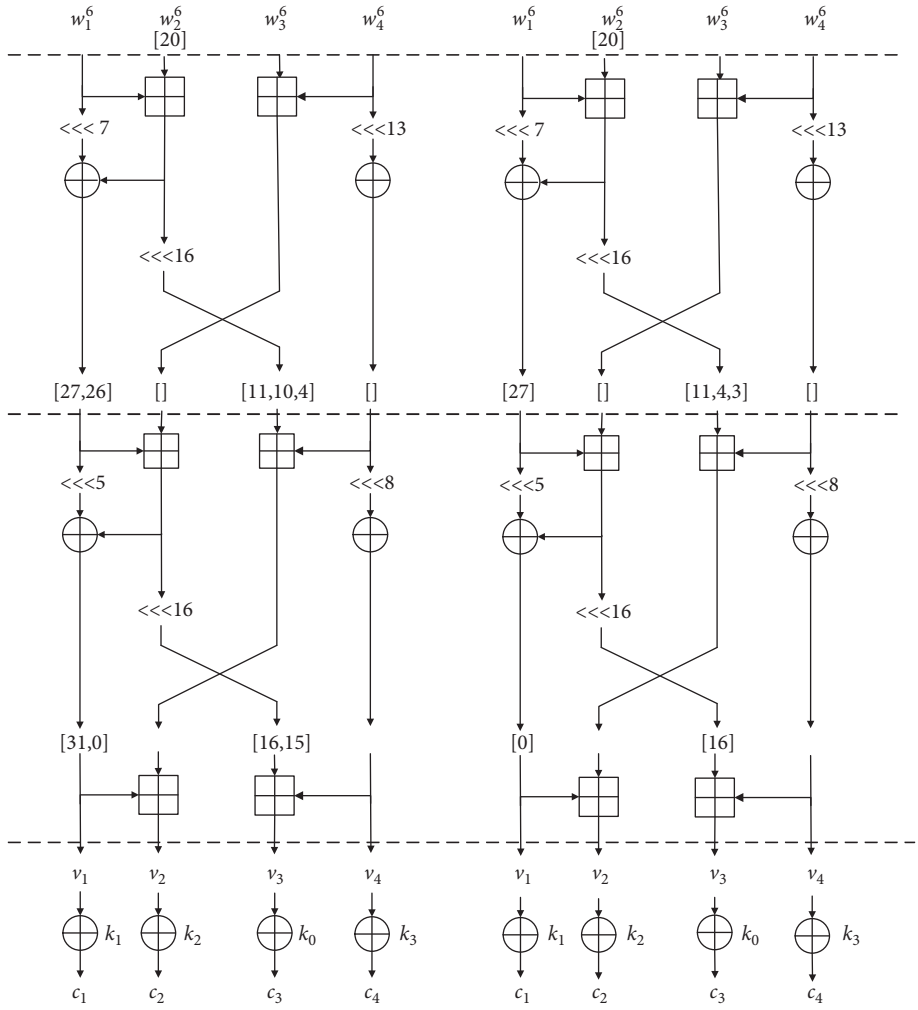

Figure 7: Partition points.

explore the properties in consecutive modular additions to get more accurate formulas. This work has been done in Ref. [7]. Note that in order to preserve the rotational property, we have to consider the pair $\pi(K \oplus K_1 \oplus m) \oplus K_1$ and $\pi(rot(K) \oplus K_1' \oplus rot(m)) \oplus K_1'$, where K_1' is a key generated from $rot(K)$, that is, $K_1' = xrot(K) \bmod g(x)$. To ensure K_1' is the word rotation of K_1, the first 2 bits of every word is assumed to be zero. The key K must be in the following form with the aim of attacking successfully.

$$K = 00 * 00 * 00 * 00 *, K_1 = 0 * 00 * 00 * 00 * 0,$$
$$rot(K) = 00 * 00 * 00 * 00 *, K_1' = * 00 * 00 * 00 * 00, \quad (25)$$

where $*$ is 30 bits that we do not care. The set of keys that satisfy such a condition is called the weak key class. It is clear that the size of the weak key class is 2^{120}. The result that we use to calculate the probability precisely is described in Table 4. Note that there are 2 single modular additions in the every round of permutation.

A 9-round distinguisher is divided into two parts. The first part is covered by rotational attack and the second part is covered by linear attack. We discuss them separately and the detail of attack is represented as follows.

4.2.2. Rotational Differential Part. Here we set 7 rounds for rotational part and the expected probability of this part is $2^{-49.636}$, which can be seen in Table 4. Mentioned in Section 2, the probability of rotational attack only depends on the number of (consecutive) modular additions. Since our work on key recovery is interested in rotational attack of 6.5 rounds Chaskey, it is easy to calculate the number of (consecutive) modular additions and the probability is $2^{-45.272}$.

4.2.3. Linear Part. Since $\epsilon_{\gamma,\Gamma}$ is close to $\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}$, we cannot use two high-correlation linear characteristics without specific relations directly. Sometimes $\epsilon_{\gamma,\Gamma}$ is the best linear characteristic, but $\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}$ has very low correlation. In order to get $\epsilon_{\gamma,\Gamma}\epsilon_{rot^{-1}(\gamma),rot^{-1}(\Gamma)}$ as high as possible, we should search two linear characteristics simultaneously. This problem can be transformed to a SAT problem [19, 20] and a SAT solver can be used to search. Because more binary variables are needed to add constraints, we may search less

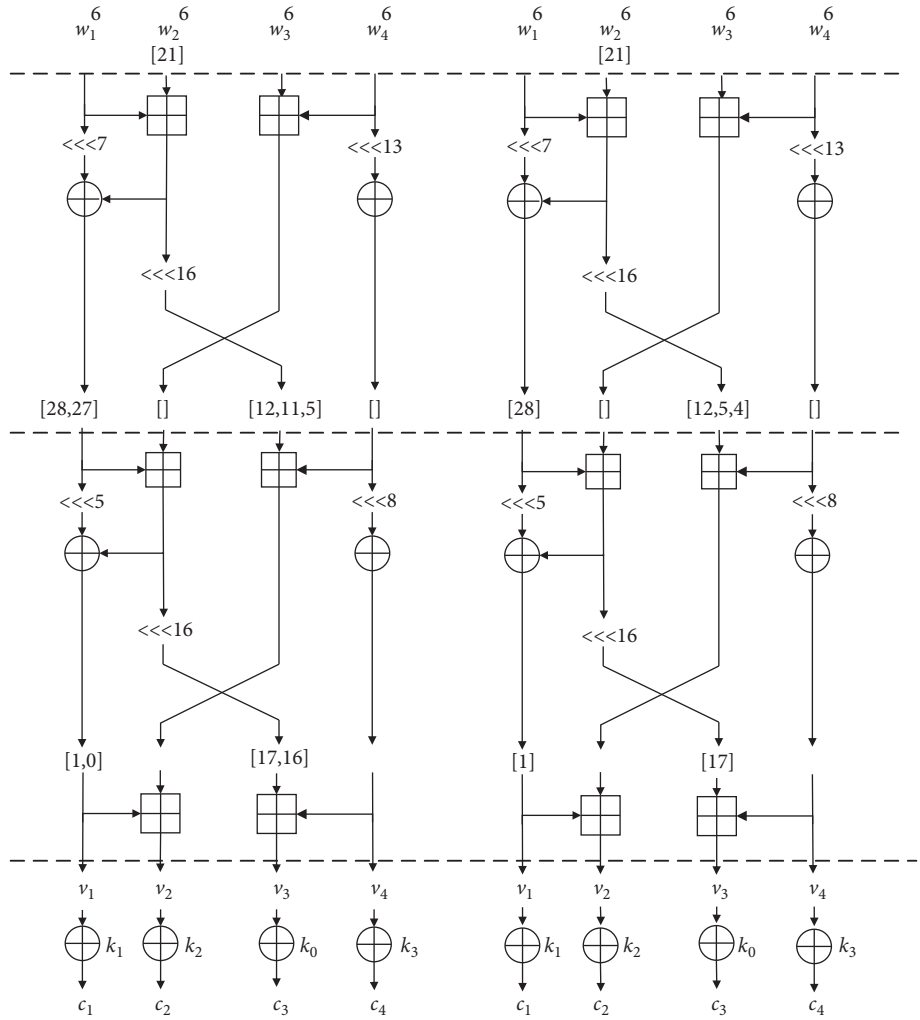

Figure 8: Rotational partition points.

round than a single linear characteristic. Therefore, we set 2 rounds for linear part and the correlation of linear part is 2^{-7}. The total correlation is $2^{-49.636} \times 2^{-7} \times 2 = 2^{-55.636}$ and the complexity is $(2^{-55.636})^{-2} = 2^{111.272}$.

4.2.4. Experimental Verification. In order to verify the correctness of formula in Section 2.4, we use 2^{20} random data to check the correlation of 3-round Chaskey with 1 round for rotational part and 2 rounds for linear part. The expected correlation of 3-round Chaskey is $2^{-6.436} \times 2^{-7} \times 2 = 2^{-12.436}$. The experimental result shows the correlation is $2^{-11.729}$. We also provide 3 code files, the first one is a python script to generate a smt2 file, which can be solved by the SAT solver. The second one is smt2 file mentioned above. The third one is a c++ file which is used to verify the correlation. The code URL is https://github.com/yuanyuan1024/Chaskey_code.

4.3. Partioning Technique for Permutation of Chaskey. Ref. [8] improves the results of differential-linear attack on permutation of Chaskey by using the partitioning technique. Inspired by this, we do a similar research on the rotational differential-linear distinguisher. We want to recover the last whitening key of Chaksey. Since the process of key recovery needs some time, a 9-round distinguisher can no longer be used. Therefore, we set 6.5 rounds for the rotational part, 0.5 rounds for linear part, and 1 round for key recovery. For linear part E_2, we consider one trail for mask $\Gamma_m = ((), ([20]), (), ())$, that is,

$$\begin{aligned} \psi^{(1)} &= v_2^6[11, 10, 4] \oplus w_1^6[31, 0] \oplus w_0^6[16, 15], \\ \psi^{(0)} &= v_2^6[11, 4, 3] \oplus w_1^6[0] \oplus w_0^6[16], \end{aligned} \tag{26}$$

and the other trail for mask $rot(\Gamma_m) = ((), ([21]), (), ())$, that is,

$$\begin{aligned} rot\left(\psi^{(1)}\right) &= v_2^6[12, 11, 5] \oplus w_1^6[1, 0] \oplus w_0^6[17, 16], \\ rot\left(\psi^{(0)}\right) &= v_2^6[12, 5, 4] \oplus w_1^6[1] \oplus w_0^6[17], \end{aligned} \tag{27}$$

Partition point and rotational partition point are shown in Figures 7 and 8, respectively. Then we begin to compute the correlation. Note that $\Gamma_{\text{out}}^{p_i} \cdot z$ can be computed by either $\psi^{(1)}$ or $\psi^{(0)}$ and rot $(\Gamma_{\text{out}}^{p_j}) \cdot \tilde{z}$ can be computed by either

$rot(\psi^{(1)})$ or $rot(\psi^{(0)})$; the attacker uses one of the highest correlation partition. For example, we assume that $\psi^{(1)}$ has a higher correlation than $\psi^{(0)}$ and rot $(\psi^{(1)})$ has a higher correlation than rot $(\psi^{(0)})$.

The total correlation can be computed in two steps. The first step is computing the correlation of $\Gamma_{\text{out}}^{P_i} \cdot z = rot^{-1}((rot(\Gamma_{\text{out}}^{p_j}) \cdot \tilde{z}))$. We experimentally evaluated the correlations of any combination, that is:

$$\begin{aligned}&\left(\psi^{(0)}, rot\left(\psi^{(0)}\right)\right), \left(\psi^{(1)}, rot\left(\psi^{(1)}\right)\right), \left(\psi^{(1)}, rot\left(\psi^{(0)}\right)\right),\\ &\left(\psi^{(0)}, rot\left(\psi^{(1)}\right)\right),\end{aligned} \tag{28}$$

which is the correlation of 2×2 rotational differential-linear distinguishers. We use 2^{30} random data to calculate the empirical correlations. Subsequently, the correlation is $2^{-6.42}$. Then, we focus on some partition bits. Partition Γ_{p_i} is defined as the following 11 bits:

$$\begin{pmatrix} s^R[15], s^R[14], v_3[18] \oplus v_2[9,17], s^L[10], s^L[9], s^L[18], s^L[17], \\ v_3[11] \oplus v_2[2,10], s^L[3], s^L[2], s^L[11] \end{pmatrix}, \tag{29}$$

and partition $rot(\Gamma_{p_j})$ is defined as the following 11 bits:

$$\begin{pmatrix} s^R[16], s^R[15], v_3[19] \oplus v_2[10,18], s^L[11], s^L[10], \\ s^L[19], s^L[18], v_3[12] \oplus v_2[3,11], s^L[4], s^L[3], s^L[12] \end{pmatrix}, \tag{30}$$

where $s^L = \overline{v}_1 \oplus v_2$ and $s^R = \overline{v}_3 \oplus v_0$. It is difficult to evaluate the actual correlations of all $q_{ij}, i, j \in \{1, \ldots, 2^{11}\}$ experimentally with a high significance. Therefore, we simply assume that these correlations are the same for each partition, which is $q_{ij} = 2^{-6.42}$ for all i and j. The second step is computing the correlation of each partition. We enumerate all possible values of 11 bits for each partition according to Table 5 in the appendix and the results are shown in Table 6.

In this table, we can see that 1472 partitions have high correlation and Table 5 average of the absolute value of those correlations is $2^{-0.779}$. Therefore, in this part, we can get the total correlation $h = 2^{-6.42-0.779\times 2} = 2^{-7.98}$. The process of key recovery is the same as Ref. [6]; we will introduce it briefly. First, we choose rotational pair $(x, x \lll 1)$ and encrypt them to get $(c, \tilde{c})$. For every cipher pair we get, we guess all possible values of $k_P' \in P$ and identify $\Gamma_i \times \Gamma_j$ for $(c \oplus k_P', \tilde{c} \oplus rot(k_P'))$ and get corresponding $\gamma_{(p_i)}$ and $rot(\gamma_{(p_j)})$, and meanwhile set $\alpha_{ij}^{k_P'} = \alpha_{ij}^{k_P'} + (-1)^{\gamma^{(p_i)} \cdot c \oplus rot^{-1}(rot(\gamma^{(p_j)}) \cdot \tilde{c})}$. Second, for $k_p' \in P$, we compute $C(k_P', k_L)$ by using Fast Walsh–Hadamard Transform. The symbol Θ is used to denote the threshold that we set. If $C(k_P', k_L) > \Theta$, we save (k_P', k_L) as a key candidate.

The successful probability of key recovery can be computed by following proposition.

Proposition 1 (see [8]). *After running the algorithm of key recovery p^{-2} times, the probability that the correct key is among the key candidates is*

$$p_{\text{success}} \geq \frac{1}{2}\Pr\left(C\left(k_P, k_L\right) \geq \Theta\right) = \frac{1}{2}\left(1 - \Phi\left(\frac{\Theta - N^* h}{\sqrt{N^*}}\right)\right). \tag{31}$$

where h is the correlation that we compute in linear part and N^* is available pairs, and Φ is the cumulative distribution function of the standard normal distribution.

4.3.1. Data and Time Complexities and Success Probability. In order to get the right pair, we repeat the process of key recovery $p^{-2} = 2^{90.544}$ times. For each iteration, we use

Table 5: Partition for two consecutive modular additions.

p	$z_1[i]$		$z_1[i, i-1]$	
	Mask γ	Correlation ϵ	Mask γ	Correlation ϵ
00000	—	0	1111101110	1
00001	111111110	-2^{-1}	1111101110	2^{-1}
00010	111111101	-1	—	2^{-1}
00011	111111101	-2^{-1}	111101101	-2^{-1}
00100	111111110	$2^{-0.263}$	111101110	$2^{-0.263}$
00101	—	0	111101110	$2^{-0.263}$
00110	1111111101	$-2^{-0.263}$	111101101	$2^{-0.263}$
00111	1111111101	$-2^{-0.263}$	—	0
01000	111001110	1	—	0
01001	111001110	2^{-1}	111011110	2^{-1}
01010	—	0	111011101	2^{-1}
01011	111001101	-2^{-1}	111011101	2^{-1}
01100	111001110	$2^{-0.263}$	111011110	$-2^{-0.263}$
01101	111001110	$2^{-0.263}$	—	0
01110	111001101	$2^{-0.263}$	111011101	$2^{-0.263}$
01111	—	0	111011101	$2^{-0.263}$
10000	111111110	-1	—	0
10001	111111110	-2^{-1}	111101110	2^{-1}
10010	—	0	111101101	1
10011	111111101	2^{-1}	111101101	2^{-1}
10100	11111110	$2^{-0.263}$	111101110	$2^{-0.263}$
10101	—	0	111101110	$2^{-0.263}$
10110	1111111101	$2^{-0.263}$	111101101	$2^{-0.263}$
10111	1111111101	$2^{-0.263}$	—	0
11000	—	0	111011110	1
11001	111001110	2^{-1}	111011110	2^{-1}
11010	111001101	1	—	0
11011	111001101	2^{-1}	111011101	-2^{-1}
11100	111001110	$2^{-0.263}$	111011110	$2^{-0.263}$
11101	111001110	$2^{-0.263}$	—	0
11110	111001101	$2^{-0.263}$	111011101	$-2^{-0.263}$
11111	—	0	111011101	$-2^{-0.263}$

Table 6: The distribution of all 2^{11} linear characteristics and its correlations.

Correlations	Number of linear characteristics
2^0	48
$2^{0.263}$	288
$2^{0.526}$	368
2^1	192
$2^{1.263}$	448
2^2	128
0	576

$N = 2^{22}$ pairs and available pairs $N^* = N \times 1472/2^{11} = 2^{21.047}$. By using the threshold $\Theta = \sqrt{N^*} \times \Phi^{-1}(1 - p^2/2^n)$, we can get a success probability of 0.972 under the condition that the right pair is successfully obtained during $2^{90.544}$ iterations. On this success probability, the data complexity is $2^{1+22+90.544} = 2^{113.544}$ and the time complexity is $2^{90.544} \times 2^{11} \times (2 \times 2^{22} + 10 \times 2^{10}) = 2^{124.944}$.

5. Conclusion

In this paper, we study the rotational differential-linear distinguisher of Chaskey's permutation and partitioning technique for the linear part [21]. For the linear part, we use an automatic tool to search for the linear characteristic to get the highest correlation. As a consequence, we give a distinguishing attack over 9 rounds of Chaskey with complexity $2^{111.272}$, and a key recovery attack over 8 rounds of Chaskey with complexity $2^{124.944}$.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work is supported by National Natural Science Foundation of China (Nos. 62072181, 62132005), NSFC-ISF Joint Scientific Research Program (No. 61961146004), Shanghai Trusted Industry Internet Software Collaborative Innovation Center.

References

[1] N. Mouha, B. Mennink, A. Van Herrewege, D. Watanabe, B. Preneel, and I. Verbauwhede, "Chaskey: an efficient MAC algorithm for 32-bit microcontrollers," in *Proceedings of the Cryptography - SAC 2014 - 21st International Conference*, J. Antoine and A. M. Youssef, Eds., Springer, Montreal, QC, Canada, pp. 306–323, August, 2014.

[2] T. Ashur and Y. Liu, "Rotational cryptanalysis in the presence of constants," vol. 2016, pp. 57–70, 2016.

[3] D. Khovratovich and I. Nikolic, "Rotational cryptanalysis of ARX," in *Proceedings of the FSE 2010 17th International Workshop*, S. Hong and T. Iwata, Eds., Springer, Seoul, Korea, pp. 333–346, February, 2010.

[4] D. Khovratovich, I. Nikolic, J. Pieprzyk, P. Sokolowski, and R. Steinfeld, "Rotational cryptanalysis of ARX revisited," in *Proceedings of the FSE 2015 22nd International Workshop*, G. Leander, Ed., Springer, Istanbul, Turkey, pp. 519–536, March 2015.

[5] S. K. Langford and M. E. Hellman, "Differential-linear cryptanalysis," in *Proceedings of the CRYPTO '94, 14th Annual International Cryptology Conference*, Y. Desmedt, Ed., Springer, Santa Barbara, California, USA, pp. 17–25, August 1994.

[6] Y. Liu, S. Sun, and C. Li, "Rotational cryptanalysis from a differential-linear perspective - practical distinguishers for round-reduced friet, xoodoo, and alzette," in *Proceedings of the EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, A. Canteaut and F. X. Standaert, Eds., Springer, Zagreb, Croatia, pp. 741–770, October 2021.

[7] L. Kraleva, T. Ashur, and R. Vincent, "Rotational cryptanalysis on MAC algorithm chaskey," in *Proceedings of the 18th International Conference, ACNS 2020 Applied Cryptography and Network Security*, M. Conti, J. Zhou, E. Casalicchio, and A. Spognardi, Eds., Springer, Rome, Italy, pp. 153–168, October 2020.

[8] C. Beierle, G. Leander, and Y. Todo, "Improved differential-linear attacks with applications to ARX ciphers," in *Proceedings of the CRYPTO 2020 - 40th Annual International Cryptology Conference*, D. Micciancio and T. Ristenpart, Eds., Springer, Santa Barbara, CA, USA, pp. 329–358, August 2020.

[9] E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," *Journal of Cryptology*, vol. 4, no. 1, pp. 3–72, 1991.

[10] X. Lai, J. L. Massey, and S. Murphy, "Markov ciphers and differential cryptanalysis," in *Proceedings of the EUROCRYPT '91, Workshop on the Theory and Application of of Cryptographic Techniques*, D. W. Davies, Ed., Springer, Brighton, UK, pp. 17–38, April, 1991.

[11] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic security evaluation and (related-key) differential characteristic search: application to simon, present, lblock, DES (L) and other bit-oriented block ciphers," in *Proceedings of the ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security*, pp. 158–178, Springer, Kaoshiung, Taiwan, December, 2014.

[12] A. Abdelkhalek, Yu Sasaki, Y. Todo, M. Tolba, and A. M. Youssef, "MILP modeling for (large) s-boxes to optimize probability of differential characteristics," *IACR Transactions on Symmetric Cryptology*, vol. 2017, no. 4, pp. 99–129, 2017.

[13] L. Li, W. Wu, L. Zhang, and Y. Zheng, "New method to describe the differential distribution table for large s-boxes in MILP and its application," *IET Information Security*, vol. 13, no. 5, pp. 479–485, 2019.

[14] Yu Sasaki and Y. Todo, "New algorithm for modeling s-box in MILP based differential and division trail search," in *Proceedings of the 10th International Conference, SecITC 2017 Innovative Security Solutions for Information Technology and Communications*, P. Farshim and E. Simion, Eds., Springer, Bucharest, Romania, pp. 150–165, June, 2017.

[15] C. Zhou, W. Zhang, T. Ding, and Z. Xiang, "Improving the milp-based security evaluation algorithm against differential/linear cryptanalysis using A divide-and-conquer approach," *IACR Transactions on Symmetric Cryptology*, vol. 2019, no. 4, pp. 438–469, 2020.

[16] M. Matsui, "Linear cryptanalysis method for DES cipher," in *Proceedings of the EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques*, T. Helleseth, Ed., Springer, Lofthus, Norway, pp. 386–397, May, 1993.

[17] A. Bar-On, D. Orr, N. Keller, and A. Weizman, "DLCT: a new tool for differential-linear cryptanalysis," in *Proceedings of the EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques*, Yuval Ishai and R. Vincent, Eds., Springer, Darmstadt, Germany, pp. 313–342, May, 2019.

[18] G. Leurent, "Improved differential-linear cryptanalysis of 7-round chaskey with partitioning," in *Proceedings of the*

EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, M. Fischlin and J. S. Coron, Eds., Springer, Vienna, Austria, pp. 344–371, May, 2016.

[19] Y. Liu, Q. Wang, and R. Vincent, "Automatic search of linear trails in ARX with applications to SPECK and chaskey," in *Proceedings of the 14th International Conference, ACNS 2016 Applied Cryptography and Network Security*, M. Manulis, A.-R. Sadeghi, and S. A. Schneider, Eds., Springer, Guildford, UK, pp. 485–499, June, 2016.

[20] L. Song, Z. Huang, and Q. Yang, "Automatic differential analysis of ARX block ciphers with application to SPECK and LEA," in *Proceedings of the 21st Australasian Conference, ACISP 2016 Information Security and Privacy*, J. K. Liu and R. Steinfeld, Eds., Springer, Melbourne, VIC, Australia, pp. 379–394, July, 2016.

[21] M. Daum, *Cryptanalysis of Hash Functions of the MD4-family*, Ruhr University Bochum, PhD Thesis, 2005.