Rotational-Linear Attack: A New Framework of Cryptanalysis on ARX ciphers with Applications to Chaskey

Yaqi Xu[1,2], Baofeng Wu[1,2], and Dongdai Lin[1,2]

[1] State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[2] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
{xuyaqi,wubaofeng,ddlin}@iie.ac.cn

Abstract. In this paper, we formulate a new framework of cryptanalysis called rotational-linear attack on ARX ciphers. We firstly build an efficient distinguisher for the cipher E consisted of the rotational attack and the linear attack together with some intermediate variables. Then a key recovery technique is introduced with which we can recover some bits of the last whitening key in the related-key scenario. To decrease data complexity of our attack, we also apply a new method, called bit flipping, in the rotational cryptanalysis for the first time and the effective partitioning technique to the key-recovery part.
Applying the new framework of attack to the MAC algorithm Chaskey, we build a full-round distinguisher over it. Besides, we have recovered 21 bits of information of the key in the related-key scenario, for keys belonging to a large weak-key class based on 6-round distinguisher. The data complexity is $2^{38.8}$ and the time complexity is $2^{46.8}$. Before our work, the rotational distinguisher can only be used to reveal key information by checking weak-key conditions. This is the first time it is applied in a last-rounds key-recovery attack. We build a 17-round rotational-linear distinguisher for ChaCha permutation as an improvement compared to single rotational cryptanalysis over it.

Keywords: Rotational-linear attack · ARX cipher · partitioning · key recovery· Chaskey · ChaCha permutation.

1 Introduction

Symmetric cryptographic algorithms have significant security-relevant applications and play influential roles in modern cryptography. Among various kinds of lightweight symmetric primitives, one class of design-structure called ARX has been adopted frequently because of its efficient software and hardware performance.

ARX ciphers. The ARX (Addition-Rotation-XOR) structure is an attractive candidate for designing lightweight cryptographic algorithms. In such a

structure, confusion and diffusion can be obtained with low consumption using three simple operations, namely, modular addition ($\boxplus$), rotation ($\lll$) and XOR ($\oplus$). ARX-based designs are not only applied to stream-ciphers, but also to the design of efficient block ciphers and message authentication codes (MAC). There are some stream-ciphers, e.g., Salsa20 [5] and ChaCha [3]. Unlike SPN ciphers, whose nonlinear part consists of S-boxes, the nonlinear operation within ARX ciphers is modular addition. The ARX-based lightweight symmetric primitives can also serve as alternatives of the SPN structure which are used in the design of many block ciphers such as the advanced encryption standard (AES). Besides, message authentication code algorithms can also use ARX-designs to achieve strong diffusion. For example, Chaskey [19] is an efficient MAC algorithm which processes a message m and a secret key K to generate a tag τ for micro controllers, using the ARX-design approach.

Due to their good confusion and diffusion properties, many classical cryptanalysis methods on symmetric ciphers are not efficient enough for ARX ciphers. For example, we generally cannot obtain differential [7] or linear [18] distinguishers with high probabilities for long enough rounds. To improve attacks on ARX ciphers, some combined cryptanalysis methods based on differential and linear attacks are often considered, such as the boomerang attack [20] and the differential-linear attack [15].

Differential-linear cryptanalysis. Differential-linear cryptanaysis was introduced by Langford and Hellman [15] for the first time. A cipher E is divided into two sub-ciphers E_1 and E_2, i.e., $E = E_2 \circ E_1$. The two parts are assumed to be independent. Then a differential distinguisher with probability p for E_1 and a linear distinguisher with correlation q for E_2 are combined to a differential-linear distinguisher with correlation pq^2. The assumption that E_1 and E_2 are independent may cause wrong estimates. To perform this attack, the cipher E is often divided into three parts, a differential part, a linear part, and a connective part. This attack has good performances in attacking ARX ciphers by carefully arranging rounds for these three parts, especially the ARX ciphers based on permutations like ChaCha [3] and Chaskey [14].

Last-rounds key recovery and partitioning technique. In the key recovery phase of an analysis of a symmetric cipher, Matsui's Algorithm 2 is a basic choice. It depends on a distinguisher on certain rounds of a cipher and a partial decryption on last few rounds by guessing some bits of sub-keys. After the partial decryption we can obtain the intermediate value to recover some bits of the key by detecting whether the distinguisher is satisfied or not. For ARX ciphers, applying partitioning technique is analogous to partial decryption during key recovery. The partitioning technique is proposed in [6] to improve linear cryptanalysis of ARX ciphers by finding a special relationship among two inputs and output on a modular addition, avoiding the impact of the carry information after selecting a set of inputs. Partitioning technique has also been applied to differential-linear attacks [16] to reduce the data complexity. In our work, we will apply partitioning technique in the key-recovery part based on the rotational-linear distinguisher.

Motivation of our work. Differential-linear attack is an important kind of combined attack and has been widely applied to many symmetric ciphers. However, the existing attacks still have some drawbacks, for example, the differential trail is limited by the connective part which increases the overall complexity obviously. Rotational attack is a specific kind of cryptanalysis which was applied to ARX ciphers primarily. This kind of distinguisher sometimes can hold with higher probability than the differential distinguisher for the same number of rounds for some ARX ciphers. However, key-recovery is usually infeasible and information of keys can only be revealed by checking certain weak-key conditions before. There is no article achieving key-recovery beyond weak keys under rotational distinguisher in prior of our work.

Motivated by the idea of differential-linear attack, we consider how to combine rotational attack and linear attack in this paper, aiming to obtain more efficient key recovery attacks on some ARX ciphers in addition to building a new kind of distinguisher. We call this attack a rotational-linear attack.

Our contribution. We build the framework of rotational-linear cryptanalysis and analyze the complexity to perform this attack. A cipher E to be attacked is also divided into three parts: the rotational part, the linear part and the connective part. Since the connectivity between rotational and linear part only effects the choice of input mask of linear part, which means it has no limit on the rotational part, and we can obtain more efficient distinguishers under our framework. We obtained a 12-round distinguisher for Chaskey with probability $2^{-60.38}$ in the related-key scenario, which is an improvement compared with the work in [14]. Besides, we show how to take advantage of the partitioning technique to recover partial bits of the key. In Table 1, we present different kinds of attacks applying to the MAC cipher Chaskey, including differential-linear attack with key recovery, rotational attacks and our rotational-linear attacks with corresponding data and time complexities. It turns out that the rotational-linear attack exhibits advantages compared to other kinds of attacks. Rotational-linear attack also gives guidelines when designing ARX ciphers, especially the key schedule.

Table 1. Review of different kinds of attacks applied to Chaskey.

Different kinds of attacks	Rounds	Time	Data	Ref.
Differential-linear attack with key recovery	7	2^{67}	2^{48}	[16]
Differential-linear attack with key recovery	7	$2^{51.21}$	$2^{40.21}$	[4]
Weak-key related-key rotational distinguishing attack	6		2^{42}	[14]
Weak-key related-key rotational attack, forge a valid tag	12		2^{86}	[14]
Related-key rotational-linear distinguishing attack	12		$2^{60.38}$	This paper
Related-key rotational-linear key-recovery attack	7	$2^{46.8}$	$2^{38.8}$	This paper

Relationship with the work in [17]. Recently, Liu et al. proposed the framework of rotational-XOR differential-linear (R-DL) attack in [17], which

degenerated to our rotational-linear attack when setting the rotational-XOR differences to be 0. We have to point out that our work is a parallel and independent one with [17] rather than a follow-up of it. Actually, combining the rotational/rotational-XOR distinguisher with the linear distinguisher is a natural idea, and the key problem lies in whether one can obtain successful attacks for specific ciphers.

In [17] the authors paid main attention to distinguishing attacks on some permutation based ciphers, while we are focused on key-recovery attacks in this paper. In addition, our rotational-linear distinguisher is different with Liu et al.'s. First, we add the connective part within the distinguisher experimentally. The hamming weight of two output masks for the distinguisher in [17] is set to 1 with rotational relationship. Because of the connective part, we can choose linear masks for the linear part without these limits in our work. Second, we use a method, that is bit flipping, in the rotational cryptanalysis for the first time to decrease the data complexity. Third, we combine the partitioning technique to the key recovery whereas the rotational/rotational-XOR cryptanalysis only applied to build distinguishers before our work. It is a natural problem to extended our key-recovery techniques to the R-DL framework, which will be left as a further work.

Organization. The rest of this paper is organized as follows. We give some relevant preliminaries in Sect. 2 and a review of the rotational attack in Sect. 3. The new framework of attack, i.e., the rotational-linear attack is presented in Sect. 4. Precise processes of our attack on Chaskey are given in Sect. 5. Conclusions are given in Sect. 6. Finally, an extended application of rotational-linear cryptanalysis to ChaCha permutation is presented in Appendix A.

2 Preliminaries

2.1 Notations

We denote an n-bit vector by $x = (x[n-1], ..., x[1], x[0])$, where $x[i]$ denotes the i-th bit of x. Let $x[i_l, \ldots, i_1]$ denote $\oplus_{j=1}^{l} x[i_j]$. The i-th unit vector is denoted by $[i] \in \mathbb{F}_2^n$. We denote the basic operations in ARX ciphers by $\boxplus$ (modular addition), $\oplus$ (XOR) and $\lll$ (rotation). A left rotation by the amount γ is denoted by $x \lll \gamma$. We also use $x = (x_0, x_1, \ldots, x_l)$ to represent an n-bit vector which splits into l m-bit sub-vectors, and the rotation of x is denoted by $\overleftarrow{x}^{\gamma} = (x_0 \lll \gamma, x_1 \lll \gamma, \ldots, x_l \lll \gamma)$. Let x, y be two n-bit vectors, $\langle x, y \rangle$ denotes the inner product of x and y. Given a set $\mathcal{S} \subseteq \mathbb{F}_2^n$ and a Boolean function $f : \mathbb{F}_2^n \to \mathbb{F}_2$, we define the correlation of f by $Cor_{x \in \mathcal{S}}[f(x)] := \frac{1}{|\mathcal{S}|} \sum_{x \in \mathcal{S}} (-1)^{f(x)}$.

2.2 Partitioning technique for modular additions

Partition is to find a special relationship between the two inputs and output of a modular addition avoiding the impact of carry information after selecting a set of the inputs. The following lemmas display the partitioning technique which

are applied in the key recovery. Assume there is a function $F : \mathbb{F}_2^{2m} \to \mathbb{F}_2^{2m}$ such that $F(a, b) = (s, b)$ where $s = a \boxplus b$. The partition could be used after selecting a set of the outputs as the following lemmas show from another point of view.

Lemma 1 ([4]). *Let* $i \geq 2$, $a, b \in \mathbb{F}_2^m$, $s = a \boxplus b$ *and* $\mathcal{S}_1 := \{(x_1, x_0) \in \mathbb{F}_2^{2m} \mid x_0[i-1] \neq x_1[i-1]\}$, $\mathcal{S}_2 := \{(x_1, x_0) \in \mathbb{F}_2^{2m} \mid x_0[i-1] = x_1[i-1]$ *and* $x_0[i-2] \neq x_1[i-2]\}$. *Then,*

$$a[i] = \begin{cases} b[i] \oplus s[i] \oplus b[i-1] \oplus 1 & \text{if } (s, b) \in \mathcal{S}_1, \\ b[i] \oplus s[i] \oplus b[i-2] \oplus 1 & \text{if } (s, b) \in \mathcal{S}_2. \end{cases}$$

Lemma 2 ([4]). *Let* $i \geq 2$, $a, b \in \mathbb{F}_2^m$, $s = a \boxplus b$ *and* $\mathcal{S}_3 := \{(x_1, x_0) \in \mathbb{F}_2^{2m} \mid x_0[i-1] = x_1[i-1]\}$, $\mathcal{S}_4 := \{(x_1, x_0) \in \mathbb{F}_2^{2m} \mid x_0[i-1] \neq x_1[i-1]$ *and* $x_0[i-2] \neq x_1[i-2]\}$. *Then,*

$$a[i] \oplus a[i-1] = \begin{cases} b[i] \oplus s[i] & \text{if } (s, b) \in \mathcal{S}_3, \\ b[i] \oplus s[i] \oplus b[i-1] \oplus b[i-2] \oplus 1 & \text{if } (s, b) \in \mathcal{S}_4. \end{cases}$$

2.3 Description of Chaskey

Chaskey is a lightweight MAC algorithm for 32-bit micro-controllers using an ARX structure in an Even-Mansour construction. A tag τ is extracted from the last state $K' \oplus F(M \oplus K)$. Chaskey splits a message m into l message blocks $m_1, m_2, \ldots, m_l$ of 128 bits each (after padding if needed). It employs a 12-round permutation π with 128-bit key K. If the message m is 128 bits, we can have $\tau = \pi(m \oplus K \oplus K_1) \oplus K_1$. Here the subkey K_1 is generated from the master key K. If $K[127]$ equals 0, then $K_1 = K \ll 1 \oplus 0^{128}$. If not, $K_1 = K \ll 1 \oplus 0^{120}10000111$. The permutation π is showed in Fig. 1.

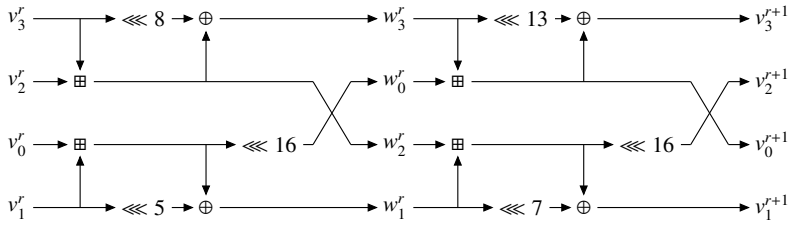

Fig. 1. The round function of Chaskey.

3 Rotational cryptanalysis

Rotational cryptanalysis, presented in [12] for the first time, takes advantage of the high probability rotational relation propagated through the ARX operations

and was applied to the reduced version of Threefish. In [12] the authors presented a method based on counting the number of additions in the scheme to get a universal upper bound on the probability of the distinguisher. However, it was showed in [13] that the rotational probabilities of ARX ciphers depends not only on the number of additions but also on their positions. An explicit formula to compute the rotational probability of chained modular additions is presented in [13]. In this section, we present the translation of rotational relations through different operations in ARX ciphers firstly.

Modular addition. The following proposition provides a general way to compute the propagation of a rotational relation through a single modular addition.

Proposition 1 ([10]) *For $x, y \in \mathbb{F}_2^m$ and $0 < \gamma < m$, we have*

$$\Pr[(x \boxplus y) \lll \gamma = (x \lll \gamma) \boxplus (y \lll \gamma)] = (1 + 2^{\gamma - m} + 2^{-\gamma} + 2^{-m})/4.$$

The probability is maximized to $2^{-1.415}$ when m is large and $\gamma = 1$. However, it was found in [13] this assumption is actually not true and the method to compute the probability is presented by the following lemma.

Lemma 3 ([13]). *Let $a_1, ..., a_l$ be m-bit vectors chosen at random and let γ be a positive integer with $0 < \gamma < m$. Then*

$$\begin{aligned}
&\Pr([(a_1 \boxplus a_2) \lll \gamma = (a_1 \lll \gamma) \boxplus (a_2 \lll \gamma)] \wedge \\
&... \\
&\wedge [(a_1 \boxplus \cdots \boxplus a_l) \lll \gamma = (a_1 \lll \gamma) \boxplus \cdots \boxplus (a_l \lll \gamma)]) \\
&= \frac{1}{2^{ml}} \binom{l + 2^\gamma - 1}{2^\gamma - 1} \binom{l + 2^{m-\gamma} - 1}{2^{m-\gamma} - 1}.
\end{aligned}$$

XOR operation. Assume that $x, y \in \mathbb{F}_2^m$ are random variables and $c \in \mathbb{F}_2^m$ is a constant. We can obtain the rotational relation $(x \oplus y) \lll \gamma = (x \lll \gamma) \oplus (y \lll \gamma)$ with probability 1 and $(x \lll \gamma) \oplus c = (x \oplus c) \lll \gamma$ if and only if $c = c \lll \gamma$.

Rotation. For the rotation operation, we have $\Pr[(x \lll a) \lll \gamma = (x \lll \gamma) \lll a] = 1$.

In the remaining part of this paper the rotational value γ will be set to 1 and we denote $x \lll 1$ by $\overleftarrow{x}$.For parallel applications of a cipher E, if the input pair (x, x') have the relation $x' = \overleftarrow{x}$, we say that (x, x') is a *rotational pair*. The input pairs together with the sub-key pairs (k_r, k'_r) XORed with intermediate values should be rotational pairs when building the rotational distinguisher. Obviously it is a kind of related-key attack. For this, we denote the transformation between master keys K and $\widetilde{K}$, such that their sub-key pairs (k_r, k'_r) are rotational pairs, by $\widetilde{K} = f(K)$.

4 Rotational-linear attacks on ARX ciphers

In this section, we will introduce the new framework of our attack combining the rotational and linear attacks, called rotational-linear attack. An entire cipher E is divided into two sub-ciphers E_1 and E_2 representing rotational and linear parts respectively and the intermediate states are showed in Fig. 2. For parallel applications of cipher E, we assume the input pairs are rotational pairs $(x, \overleftarrow{x})$. For two masks β^1 and β^0, we study correlation of linear approximations $Cor_{x\in\mathbb{F}_2^n}[\langle\beta^1, E(x)\rangle \oplus \langle\beta^0, E(\overleftarrow{x})\rangle]$. Making use of the distinguisher and the partition technique mentioned in [4, 16], we can recover partial bits of the key for last rounds.

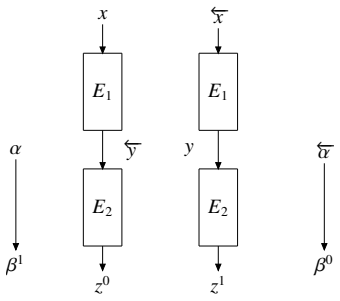

Fig. 2. Basic rotational-linear distinguisher with linear trails for linear part E_2.

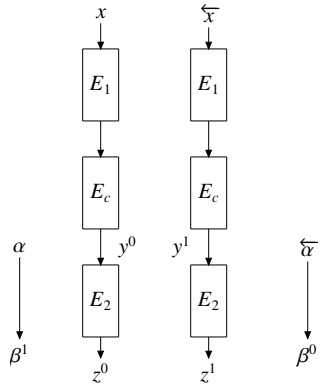

Fig. 3. The structure of rotational-linear distinguisher with a connective part E_c.

4.1 Correlation of linear approximations

After the rotational part E_1, we denote the rotational probability by $\Pr_{x\in\mathbb{F}_2^n}[\overleftarrow{E_1(x)} = E_1(\overleftarrow{x})] = p$, and we need to compute the correlation $Cor_{x\in\mathbb{F}_2^n}[\langle\beta^1, E(x)\rangle \oplus \langle\beta^0, E(\overleftarrow{x})\rangle]$. First we assume that E_1 is independent with E_2. Given a mask α, we have $\langle\alpha, E_1(x)\rangle = \langle\overleftarrow{\alpha}, E_1(\overleftarrow{x})\rangle$ if $\overleftarrow{E_1(x)} = E_1(\overleftarrow{x})$ after the rotational part. Then we need to find two linear trails for E_2 with the input masks α and $\overleftarrow{\alpha}$. The correlation of linear trails for the parallel are q_1 and q_0 respectively.

Now we can compute the correlation of the distinguisher as follows:

$$\begin{aligned}
&Cor_{x\in\mathbb{F}_2^n}[\langle\beta^1, E(x)\rangle \oplus \langle\beta^0, E(\overleftarrow{x})\rangle] \\
=\ &Cor_{x\in\mathbb{F}_2^n}[\langle\beta^1, E_2\circ E_1(x)\rangle \oplus \langle\beta^0, E_2\circ E_1(\overleftarrow{x})\rangle] \\
=\ &Cor_{y\in\mathbb{F}_2^n}[\langle\beta^1, E_2(y)\rangle \oplus \langle\beta^0, E_2(\overleftarrow{y})\rangle] \cdot Pr_{x\in\mathbb{F}_2^n}[\overleftarrow{E_1(x)} = E_1(\overleftarrow{x})].
\end{aligned}$$

Assume $\langle \beta^1, E_2(y)\rangle \oplus \langle \alpha, y\rangle$ and $\langle \beta^0, E_2(\overleftarrow{y})\rangle \oplus \langle \overleftarrow{\alpha}, \overleftarrow{y}\rangle$ are independent and $\langle \alpha, y\rangle = \langle \overleftarrow{\alpha}, \overleftarrow{y}\rangle$. By the Piling-up lemma we have the correlation of the distinguisher is pq_0q_1. Assume the data complexity of the rotational part is N_r, which is asymptotically $O(p^{-2})$. By preparing $N_r \cdot \delta(q_0q_1)^{-2}$ pairs of chosen plaintexts $(x, \overleftarrow{x})$, where $\delta \in \mathbb{N}$ is a small constant, one can distinguish the cipher from a pseudo random permutation.

4.2 The connective part

The assumption that E_1 and E_2 are independent might be a problem with wrong estimates for the correlation of linear approximations. The wrong estimate under differential-linear attacks has been testified in [2]. In rotational-linear cryptanalysis we also add a connective part as a simple solution (showed in Fig. 3). The correlation $p_c = Cor_{x\in\mathcal{S}}[\langle \alpha, E_c(x)\rangle \oplus \langle \overleftarrow{\alpha}, E_c(\overleftarrow{x})\rangle]$ can be evaluated experimentally where $\mathcal{S}$ denotes a set of random samples. Then the whole correlation of the linear approximation of the distinguisher is computed as $Cor_{x\in\mathbb{F}_2^n}[\langle \beta^1, E(x)\rangle \oplus \langle \beta^0, E(\overleftarrow{x})\rangle] = pp_cq_0q_1$.

4.3 Decrease the data complexity

Inspired by the work in [4], we want to construct many right pairs with probability (close to) 1 given a right pair satisfying rotational relationship. If we have a chained addition operation $s = a_0 \boxplus a_1 \boxplus a_2$ for $a_0, a_1, a_2 \in \mathbb{F}_2^m$, we denote the carry-bit vectors of $a_0 \boxplus a_1$ and $(a_0 \boxplus a_1) \boxplus a_2$ by c_1 and c_2 respectively. Note that $(a_0 \boxplus a_1 \boxplus a_2) \lll 1 = (a_0 \lll 1) \boxplus (a_1 \lll 1) \boxplus (a_2 \lll 1)$ and $(a_0 \boxplus a_1) \lll 1 = (a_0 \lll 1) \boxplus (a_1 \lll 1)$ if and only if the following conditions are satisfied:

$$\begin{aligned} &c_2[m-1] = 0,\ c_1[m-1] = 0, \\ &a_0[m-1] + a_1[m-1] + a_2[m-1] < 2. \end{aligned} \tag{1}$$

We flip a few bits of one input and assume it is a_0 without loss of generality. Write $s^r = a_0^r \boxplus a_1 \boxplus a_2$ where a_0^r is the flipped input. If the conditions in (1) can also be satisfied, then we obtain a new right pair for free. However, the conditions for a cipher is more complicated. Let us denote $\mathcal{R}$ as the set consisting of right data satisfying the rotational relation before and after an ARX permutation E_1, i.e., $\mathcal{R} = \{x \in \mathbb{F}_2^n \mid \overleftarrow{E_1(x)} = E_1(\overleftarrow{x})\}$. We assume that $\mathcal{R}$ contains an affine subspace $\mathcal{A} = a \oplus \mathcal{U}$. Instead of choosing random plaintexts from $\mathbb{F}_2^n$, we could generate a subset of $x \in \mathcal{A}$ to augment the correlation of the rotational-linear distinguisher. Now we can generate right data for the rotational part E_1 through a statistical model presented in Algorithm 1.

The number of flipping-bit candidates, namely, $|\mathcal{U}|$ getting from Algorithm 1, are not only related to the structure of E_1 but also to the number of rounds in E_1. The number of flipping-bit candidates needs to be sufficiently large to fulfill the data complexity of the remaining part of the attack. If the number of rounds in E_1 is beyond our expect, we can split E_1 into two parts $E_1 = E_{10} \circ E_{11}$ with rotational probabilities p_0 and p_1 respectively. Then the rotational probability of

Algorithm 1 Find the candidates of flipping-bits

Input: Permutation $E : \mathbb{F}_2^n \to \mathbb{F}_2^n$, a set $\mathcal{S} = \{x \mid x \in \mathbb{F}_2^n\}$ of N samples
Output: Probabilistic list of flipping-bit candidates $\mathcal{U}$, threshold ρ.

```
Let r = 0 and t_j = 0 for j ∈ {0, ..., n − 1}
for i = 0 to N do
    Pick one element x ∈ S, compute E(x) and E(←x) respectively
    if ←E(x) = E(←x) then
        r ← r + 1
        for j ∈ {0, ..., n − 1} do
            Prepare x' = x ⊕ 2^j, E(x') and E(←x')
            if ←E(x') = E(←x') then
                t_j ← t_j + 1
for j ∈ {0, ..., n − 1} do
    if t_j/r > ρ then
        Save j as a flipping-bit candidate
```

E_1 is $p = p_0 p_1$. After applying Algorithm 1 to E_{10}, we can get a list of flipping-bit candidates with average probability p_a. If the result meets requirements for the remaining part of our attack, we can decrease the data complexity N_r of E_1 from $\mathcal{O}((p_0 p_1)^{-2})$ to $\mathcal{O}(p_0^{-1}(p_a p_1)^{-2})$.

4.4 Key recovery

In this section, we present a method to recover part of the last whitening key k based on a rotational-linear distinguisher. The rotational pair $\{(x, \overleftarrow{x}) \mid x \in \mathbb{F}_2^n\}$ are inputs for the cipher E. Using the rotational-linear distinguisher built in Sect. 3 we have the correlation $Cor_{x \in \mathbb{F}_2^n}[\langle \beta^1, E(x) \rangle \oplus \langle \beta^0, E(\overleftarrow{x}) \rangle] = q$. The set of input pairs for the linear part is denoted by $\mathcal{D} := \{(y, \widetilde{y}) \mid y = E_c \circ E_1(x),\ \widetilde{y} = E_c \circ E_1(\overleftarrow{x})$ and $E_1(\overleftarrow{x}) = \overleftarrow{E_1(x)}$ for $x \in \mathbb{F}_2^n\}$. We denote the right pairs after cipher E by $\mathcal{D}_{out} := \{(z, \tilde{z}) \mid z = E_2(y),\ \tilde{z} = E_2(\tilde{y})$ for $(y, \tilde{y}) \in \mathcal{D}\}$ as the input pairs for F. In Matsui's last rounds attack in [18] one gathers some bits of key information with partial decryption of the last rounds.Because of the modular addition operation, we need to do some changes for the key-recovery work.

On the basis of correlation of the linear approximation $Cor_{(z,\tilde{z}) \in \mathcal{D}_{out}}[\langle \beta^1, z \rangle \oplus \langle \beta^0, \tilde{z} \rangle]$, we split β^ι into $\beta_0^\iota, \ldots, \beta_{l-1}^\iota$ for $\iota \in \{0, 1\}$ such that every β_i^ι is in the form of one or two consecutive active bits defined as a *partition point* [4] and the correlated linear approximation is $Cor_{(z,\tilde{z}) \in \mathcal{D}_{out}}[\langle \beta_i^1, z \rangle \oplus \langle \beta_j^0, \tilde{z} \rangle] = q_{i,j}$. The overall structure of key recovery is presented in Fig. 4. Applying the partition technique [4, 16] to the outputs of F, we can generate several linear trails $\beta_i^1 \to \mu_{i,i_1}^1$ and $\beta_j^0 \to \mu_{j,j_1}^0$ corresponding to special partition $\mathcal{P}$ with high correlations denoted by ϵ_{i,i_1}^1 and ϵ_{j,j_1}^0 for $i, j \in \{0, \ldots, l-1\}$ and $i_1, j_1 \in \{0, \ldots, s-1\}$. Then

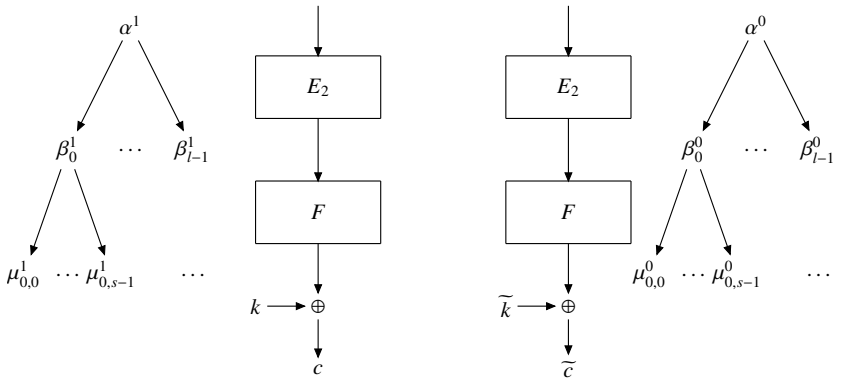

Fig. 4. To recover partial key, output mask of E_2, i.e., β^ι, is partitioned to *partition points* $\beta_0^\iota, \ldots, \beta_{l-1}^\iota$, $\iota \in \{0,1\}$. For every partition points we will find linear trails $\beta_i^\iota \to \mu_{i,j}^\iota$ after partition, $i \in \{0, \ldots, l-1\}$ and $j \in \{0, \ldots, s-1\}$.

we have

$$\begin{aligned} &Cor_{\substack{(z,\widetilde{z})\in\mathcal{D}_{out}\ \text{s.t.}\\ k\oplus c\in\mathcal{P},\ \widetilde{k}\oplus\widetilde{c}\in f(\mathcal{P})}}[\langle\mu_{i,i_1}^1, k\oplus c\rangle \oplus \langle\mu_{j,j_1}^0, \widetilde{k}\oplus\widetilde{c}\rangle] \\ =\ &Cor_{\substack{((z,\widetilde{z})\in\mathcal{D}_{out}\ \text{s.t.}\\ k\oplus c\in\mathcal{P},\ \widetilde{k}\oplus\widetilde{c}\in f(\mathcal{P})}}[\langle\beta_i^1, z\rangle \oplus \langle\beta_j^0, \widetilde{z}\rangle] = q_{i,j}\epsilon_{i,i_1}^1\epsilon_{j,j_1}^0. \end{aligned}$$

Partitioning the outputs of F is equivalent to partitioning the set $\{(k\oplus c, \widetilde{k}\oplus\widetilde{c}) \mid (z,\widetilde{z}) \in \mathcal{D}_{out}$ s.t. $k\oplus c = F(z),\ \widetilde{k}\oplus\widetilde{c} = F(\widetilde{z})\}$. We denote $\mathcal{P}$ as the set of bits relevant to partitioning technique. Moreover we represent k as $k_\mathcal{P} \oplus k_\mu \oplus k'$ where $k_\mathcal{P} \in \mathcal{P}$, k_μ is the set of bits related to the masks $\mu^1 \oplus \mu^0$ while k' is the remaining bits we need to guess. The whole process of the key-recovery is demonstrated in Algorithm 2.

Build the counter. Building a special counter is a significant speed-up in the key-recovery under the framework of differential-linear attack [4]. For our rotational-linear attack, we need to find the relations between active bits of β_j^0 and k before building a valid counter to recover the partial bits of the key, that is

$$\begin{aligned} &Cor_{\substack{(z,\widetilde{z})\in\mathcal{D}_{out}\ \text{s.t.}\\ k\oplus c\in\mathcal{P},\ \widetilde{k}\oplus\widetilde{c}\in f(\mathcal{P})}}[\langle\mu_{i,i_1}^1, c\oplus k\rangle \oplus \langle\mu_{j,j_1}^0, \widetilde{c}\oplus\widetilde{k}\rangle] \\ =\ &\sum_{\substack{(z,\widetilde{z})\in\mathcal{D}_{out}\ \text{s.t.}\\ k\oplus c\in\mathcal{P},\ \widetilde{k}\oplus\widetilde{c}\in f(\mathcal{P})}} (-1)^{\langle\mu_{i,i_1}^1,c\rangle\oplus\langle\mu_{j,j_1}^0,\widetilde{c}\rangle\oplus\langle\mu_{i,i_1}^1,k\rangle\oplus\langle f^{-1}(\mu_{j,j_1}^0),k\rangle} \\ =\ &(-1)^{\langle\mu_{i,i_1}^1\oplus f^{-1}(\mu_{j,j_1}^0),k\rangle} \sum_{\substack{(z,\widetilde{z})\in\mathcal{D}_{out}\ \text{s.t.}\\ k\oplus c\in\mathcal{P},\ \widetilde{k}\oplus\widetilde{c}\in f(\mathcal{P})}} (-1)^{\langle\mu_{i,i_1}^1,c\rangle\oplus\langle\mu_{j,j_1}^0,\widetilde{c}\rangle}. \end{aligned}$$

Algorithm 2 Key recovery

Require: Cipher E, the set $\mathcal{S}$ of N plaintext-ciphertext pairs, threshold Θ, the set of flipping-bits $\mathcal{U}$

Ensure: List of key candidates for $n_{\mathcal{P}} + \dim W$ bits of k.

for $(i,j) \in \{0,...,l-1\} \times \{0,...,l-1\}$ **do**
  for $k_{\mathcal{P}} \in \mathcal{P}$ **do**
    $\nu_{i,j}^{k_{\mathcal{P}}} \leftarrow 0$
for $a \in \mathcal{S}$ **do**
  $x \xleftarrow{\$} \mathcal{U} \oplus a$, $(c, \tilde{c}) \leftarrow (E_k(x), E_{f(k)}(x \lll 1))$
  for $(i,j) \in \{0,...,l-1\} \times \{0,...,l-1\}$ **do**
    for $k_{\mathcal{P}}$ **do**
      With $(c \oplus k_{\mathcal{P}}, \tilde{c} \oplus f(k_{\mathcal{P}}))$ get corresponding output masks μ_{i,i_1}^1 and μ_{j,j_1}^0
      $\nu_{i,j}^{k_{\mathcal{P}}} \leftarrow \nu_{i,j}^{k_{\mathcal{P}}} + \sum_{i_1=0}^{s-1} \sum_{j_1=0}^{s-1} (-1)^{\langle \mu_{i,i_1}^1, c \rangle \oplus \langle \mu_{j,j_1}^0, \tilde{c} \rangle}$
for $k_{\mathcal{P}} \in \mathcal{P}$ **do**
  Compute $\mathcal{C}(k_{\mathcal{P}}, k_\mu)$ using the Fast Walsh-Hadamard Transform
  if $\mathcal{C}(k_{\mathcal{P}}, k_\mu) > \Theta$ **then**
    Save $(k_{\mathcal{P}}, k_\mu)$ as a key candidate

The value of k only impacts the sign of the correlation of linear approximation. Following the previous work we define an intermediate variable $\alpha_{i,j}$ by

$$\alpha_{i,j} = (-1)^{sgn(\epsilon_{i,i_1}^1 \epsilon_{j,j_1}^0 q_{i,j})} \sum_{\substack{(z,\tilde{z}) \in \mathcal{D}_{out} \text{ s.t.} \\ k \oplus c \in \mathcal{P},\ \tilde{k} \oplus \tilde{c} \in f(\mathcal{P})}} (-1)^{\langle \mu_{i,i_1}^1, k \oplus c \rangle \oplus \langle \mu_{j,j_1}^0, \tilde{k} \oplus \tilde{c} \rangle}$$

$$= (-1)^{sgn(\epsilon_{i,i_1}^1 \epsilon_{j,j_1}^0 q_{i,j})} \sum_{\substack{(z,\tilde{z}) \in \mathcal{D}_{out} \text{ s.t.} \\ k \oplus c \in \mathcal{P},\ \tilde{k} \oplus \tilde{c} \in f(\mathcal{P}) \text{ and } \tilde{k} = f(k)}} (-1)^{\langle \mu_{i,i_1}^1, c \rangle \oplus \langle \mu_{j,j_1}^0, \tilde{c} \rangle \oplus \langle \mu_{j,j_1}^1, k \rangle \oplus \langle \mu_{j,j_1}^0, f(k) \rangle}$$

$$= (-1)^{\langle \mu_{i,i_1}^1 \oplus f^{-1}(\mu_{j,j_1}^0), k_\mu \rangle} \left| \sum_{\substack{(z,\tilde{z}) \in \mathcal{D}_{out} \text{ s.t.} \\ c \in k \oplus \mathcal{P},\ \tilde{c} \in f(k \oplus \mathcal{P})}} (-1)^{\langle \mu_{i,i_1}^1, c \rangle \oplus \langle \mu_{j,j_1}^0, \tilde{c} \rangle} \right|,$$

where $sgn(r) = \begin{cases} 0 & \text{if } r \geq 0 \\ 1 & \text{if } r < 0 \end{cases}$. We define $W = Span\{\mu_{i,i_1}^1 \oplus f^{-1}(\mu_{j,j_1}^0) \mid i,j \in \{0,\ldots,l-1\} \text{ and } i_1, j_1 \in \{0,\ldots,j-1\}\}$ and

$$\mathcal{C}(k_{\mathcal{P}}, k_\mu) := \sum_{\mu \in W} (-1)^{\langle \mu, k_\mu \rangle} \sum_{\substack{(i,j) \text{ s.t.} \\ \mu_{i,i_1}^1 \oplus f^{-1}(\mu_{j,j_1}^0) = \mu}} \nu_{i,j}.$$

We need to guess bits of k corresponding to active bits of output masks. Let $n_{\mathcal{P}}$ denote the number of elements in $\mathcal{P}$. Then the number of bits we can get in the key-recovery process is $n_{\mathcal{P}} + \dim W$. The fast Walsh-Hadamard transform [8] can be used during the key-recovery. Therefore, the whole running time of the key-recovery decreases to $2^{n_{\mathcal{P}}}(2N + \dim W \cdot 2^{\dim W})$ [4].

4.5 A simple toy example

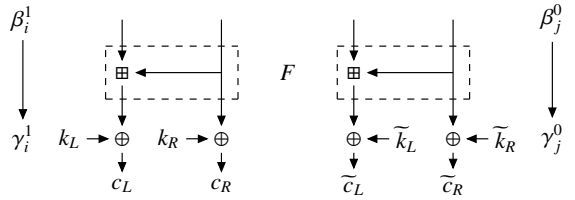

Fig. 5. A toy example of key recovery where the key-recovery part F only contains a modular addition.

In Fig. 5 there is a function $F : \mathbb{F}_2^{2m} \to \mathbb{F}_2^{2m}$ such that $F(a, b) = (s, b)$ where $s = a \boxplus b$. The input masks for F are denoted by β_i^1 and β_j^0. To simplify the key-recovery process, we suppose $\widetilde{k} = k \lll 1$ and the input masks for F are $(\beta^1, \beta^0) = ([i], [i+1, i])$ where β^0 is the rotation of β^1. If we denote $k = k^L || k^R = (k_{m-1}^L, \ldots, k_0^L) || (k_{m-1}^R, \ldots, k_0^R)$, then we have $\widetilde{k} = \widetilde{k}^L || \widetilde{k}^R = (k_{m-2}^L, \ldots, k_0^L, k_{m-1}^R) || (k_{m-2}^R, \ldots, k_0^R, k_{m-1}^L)$.

To apply the partitioning technique to find linear trails with high correlations, we need to obtain the values $k_{i-1}^L \oplus c_L[i-1] \oplus k_{i-1}^R \oplus c_R[i-1]$, $k_{i-2}^L \oplus c_L[i-2] \oplus k_{i-2}^R \oplus c_R[i-2]$, $\widetilde{k}_i^L \oplus \widetilde{c}_L[i] \oplus \widetilde{k}_i^R \oplus \widetilde{c}_R[i]$ and $\widetilde{k}_{i-1}^L \oplus \widetilde{c}_L[i-1] \oplus \widetilde{k}_{i-1}^R \oplus \widetilde{c}_R[i-1]$. If $i \geq 2$, we only need to guess two bits $k^L[i-1] \oplus k^R[i-1]$ and $k^L[i-2] \oplus k^R[i-2]$ before partitioning. Then we can find four trails, i.e.,

$$
\begin{aligned}
&\beta_0^1 = ([i], []),\ \mu_0^1 = ([i, i-1], [i]),\ \text{if } (k_L \oplus c_L, k_R \oplus c_R) \in \mathcal{S}_1,\\
&\beta_1^1 = ([i], []),\ \mu_1^1 = ([i, i-2], [i]),\ \text{if } (k_L \oplus c_L, k_R \oplus c_R) \in \mathcal{S}_2,\\
&\beta_0^0 = ([i+1, i], []),\ \mu_0^0 = ([i], [i]),\ \text{if } (\widetilde{k}_L \oplus \widetilde{c}_L, \widetilde{k}_R \oplus \widetilde{c}_R) \in \mathcal{S}_3,\\
&\beta_1^0 = ([i+1, i], []),\ \mu_1^0 = ([i, i-1, i-2], [i]),\ \text{if } (\widetilde{k}_L \oplus \widetilde{c}_L, \widetilde{k}_R \oplus \widetilde{c}_R) \in \mathcal{S}_4,
\end{aligned}
$$

all with correlations 1 or -1. Then we obtain $W = \{(\mu_i^1 \oplus \mu_j^0) \mid i, j \in \{0, 1\}\} = \{([], []), ([i-1], []), ([i-2], []), ([i-1, i-2], [])\}$ and could recover four bits of information of the key, namely, $k^L[i-1]$, $k^L[i-2]$, $k^L[i-1] \oplus k^R[i-1]$ and $k^L[i-2] \oplus k^R[i-2]$.

5 Application to Chaskey

5.1 Attack against 7-round Chaskey

The process of our key-recovery attack applying to Chaskey covers 7 rounds splitting into four parts, including 2.5-round rotational part E_1, 3-round connective part E_c and 0.5-round linear part E_2, with 1-round key-recovery part F.

Rotational part. In this part, we need to calculate the rotational probability of E_1 using Lemma 3. For Chaskey, we need that the input pairs $(m \oplus K \oplus K_1, \widetilde{m} \oplus \widetilde{K} \oplus \widetilde{K_1})$ have the relation $\widetilde{m} \oplus \widetilde{K} \oplus \widetilde{K_1} = \overleftarrow{m \oplus K \oplus K_1}$. For the chained modular additions we use Lemma 3 to calculate the rotational probability and if there are rotation or XOR operations between two modular additions we assume that they are independent, as used in [14] applying rotational cryptanalysis to Chaskey. For one chain of two modular additions $a_1 \boxplus a_2 \boxplus a_3$ with block size $n = 32$, the probability of rotation is $2^{-3.585}$.

Expected probability using Lemma 3 and the corresponding experimental probability of the rotational attack applied to chained modular additions presented in [12] are very close to each other. So the rotational probability for our E_1 part is reliable and the expected probability of 2.5-round E_1 is $2^{-17.17}$. We split the rotational part E_1 into E_{11} and E_{10}, where the E_{10} part includes the first 1.5 rounds and a modular addition $(w_1^1 \boxplus w_2^1)$. There are one modular addition and three double-addition so the rotational probability of E_{10} is $2^{-12.17}$. We apply Algorithm 1 to the E_{10} part with a set of random samples $\mathcal{S}$ which has 2^{32} elements. Then we collected 24 flipping bits with probability greater than 0.97 by experiments and the set of them is denoted by $\mathcal{U}$ from which we can generate 2^{24} right pairs $(a \oplus u, \overleftarrow{a \oplus u})$ for $u \in \mathcal{U}$ and a known right pair $(a, \overleftarrow{a})$. We can obtain the average probability with flipping-bits presented in Table 2, that is,

$$\Pr_{\substack{u \in \mathcal{U} \\ x \in \mathbb{F}_2^n}} [\overleftarrow{E_{10}(x \oplus u)} = E_{10}(\overleftarrow{x \oplus u}) \mid E_{10}(\overleftarrow{x}) = \overleftarrow{E_{10}(x)}] = 0.991.$$

This means if we have a right pair $(x, \overleftarrow{x})$, we can generate another right pair with probability 0.991 after E_{10}. These flipping bits can satisfy the data complexity for the later part and we can decrease the data complexity from $2^{34.34}$ to $2^{12.17+5\times2} \times (0.991)^{-2}$, that is $2^{22.195}$.

Table 2. Candidates of flipping-bit applying Algorithm 1 to E_{10}.

The vector to flip	Index	Probability
v_0^0	0, 1, 2, 3, 4, 5	> 0.97
v_1^0	0, 1, 2, 3	> 0.97
v_2^0	0, 1, 2, 3, 4, 5, 6, 7, 8	> 0.97
v_3^0	0, 1, 2, 3, 4	> 0.97

Middle part. After the rotational part E_1, we have the rotational relation $\overleftarrow{E_1(x)} = E_1(\overleftarrow{x})$, then we set the connective part E_c covering three rounds of Chaskey. We need to find input masks $(\alpha, \overleftarrow{\alpha})$ such that $\langle \alpha, E_c \circ E_1(x) \rangle \oplus \langle \overleftarrow{\alpha}, E_c \circ E_1(\overleftarrow{x}) \rangle$ has high correlation. On account of the limitation of computing capacity, we only search the masks α in the form of $[i]$. We observed for $\alpha = ([], [], [11], [])$, the correlation is $Cor_{x \in \mathcal{S}}[\langle \alpha, E_c(x) \rangle \oplus \langle \overleftarrow{\alpha}, E_c(\overleftarrow{x}) \rangle] = 2^{-1.73}$. Assume that the correlation obeys a normal distribution and the standard deviation of the normal

distribution is 2^{12}. Using a set $\mathcal{S}$ consisting 2^{24} random samples to estimate is enough.

Linear part and key recovery. Assume the input-mask pair for parallel E_2 is $(\alpha, \overleftarrow{\alpha}) = (\omega_1^5[11], \omega_1^5[12])$. After E_2 we have

$$\begin{aligned}\beta^1 =&\nu_0^6[24] \oplus \nu_0^6[11,10] \oplus \omega_2^6[0] \oplus \omega_3^6[0],\\ \beta^0 =&\nu_0^6[25,24] \oplus \nu_0^6[12] \oplus \omega_2^6[1,0] \oplus \omega_3^6[1,0],\end{aligned}$$

with the experimental correlation $2^{-3.187}$ over the set $\mathcal{S}$ consisting of 2^{26} random samples of ω^2. In the key-recovery part F, we need to partition the set of $(c \oplus K_1)$ and $(\widetilde{c} \oplus \widetilde{K}_1)$ to obtain the linear trails. Linear trails based on the partition technique are relevant to two chained modular additions. So we adopt the method in [4] and using the average of the absolute values of correlations of linear trails covering F, i.e., $2^{-0.83}$. Since it is difficult to evaluate the correlation $q_{i,j}$ after partition experimentally, we assume the correlations are equal for every partition, i.e., $q_{i,j} = 2^{-4.745}$ for all i and j.

In E_1 part the key $K \oplus K_1$ and $\widetilde{K} \oplus \widetilde{K_1}$ should satisfy $\widetilde{K} \oplus \widetilde{K_1} = \overleftarrow{K \oplus K_1}$. We denote values of the keys by

$$\begin{aligned}&K = ab*_0, \quad fg*_1, \quad kl*_2, \quad pq*_3; \qquad K_1 = b *_0 f, \quad g *_1 k, \quad l *_2 p, \quad (q *_3 a) \oplus \star;\\ &\widetilde{K} = b *_0 a, \quad g *_1 f, \quad l *_2 k, \quad q *_3 p; \quad \widetilde{K_1} = *_0 ag, \quad *_1 fl, \quad *_2 kq, \quad (*_3 pb) \oplus \star,\end{aligned}$$

where $*_i \in \{0,1\}^{30}$ for $i \in \{0,1,2,3\}$ and $\star \in \mathbb{F}_2^{32}$ denotes the vector 0^{32} or $0^{24}||10000110$ decided by the value of $K[127]$ and $\widetilde{K}[127]$. For the rotational relation, we need to set $a = b = 0$ to let $\star = 0^{32}$. However, to have $\widetilde{K_1} = \overleftarrow{K_1}$, we need to set $a = b = f = g = k = l = p = q$. Then we obtain the weak-key class containing 2^{120} keys.

β^1 is split to $\beta_0^1 = \nu_0^6[24]$, $\beta_1^1 = \nu_0^6[11,10]$ and $\beta_2^1 = \omega_2^6[0] \oplus \omega_3^6[0]$. β^0 is split to $\beta_0^0 = \nu_0^6[25,24]$, $\beta_1^0 = \nu_0^6[12]$ and $\beta_2^0 = \omega_2^6[1,0] \oplus \omega_3^6[1,0]$. After the partition of outputs of F, presented in Table 3, we obtain linear trails $\beta_i^\iota \to \mu_i^\iota$ for $\iota \in \{0,1\}$ and $i \in \{0,1,2\}$. Because of the weak-key class, $k_i[31] = k_i[0] = \widetilde{k_i}[1] = \widetilde{k_i}[0] = 0$ for $i \in \{0,1,2,3\}$. Furthermore, the corresponding partition to $(c \oplus K_1)$ and $(\widetilde{c} \oplus \widetilde{K}_1)$ is decided by $k_\mathcal{P}$ with $k_{\mathcal{P}_1} = k_1[15] \oplus k_0[25] \oplus k_0[30]||(k_0 \oplus k_3)[30]||(k_0 \oplus k_3)[26]||(k_0 \oplus k_3)[25]$ and $k_{\mathcal{P}_0} = k_1[28] \oplus k_0[11] \oplus k_0[6]||(k_0 \oplus k_3)[12]||(k_0 \oplus k_3)[11]||(k_0 \oplus k_3)[7]||(k_0 \oplus k_3)[6]$.

From the set $\{(\mu_{i,i_1}^1 \oplus \mu_{j,j_1}^0) \mid i,j \in \{0,\ldots,l-1\} \text{ and } i_1, j_1 \in \{0,\ldots,s-1\}\}$, we can see k_μ consists $k_0[26] \oplus k_0[25]$, $k_0[7] \oplus k_0[6]$, $k_0[12] \oplus k_0[11]$, $k_0[30]$, and $\dim(W) = 4$. Summing up the above analysis, we can recover $13 + 8$, that is 21 bits of the master key K.

Data and time complexities and success probability. The way to recover partial key bits is based on the correlation of linear approximations. Our method to build the counter is same as [4], so we use the same proposition used in it

Table 3. The outputs of F corresponding to partitioning technique. That is for every $\mathcal{P}$ in the table, we want to partition the set of ciphertexts to obtain the corresponding subset of it will belong to the $\{p \oplus k_{\mathcal{P}} \text{ for } \forall p \in \mathcal{P}\}$ after guessing partial bits of k.

input mask	$\mathcal{P}$	sub-key
β_1^1	$p_1[15] \oplus p_0[25] \oplus p_0[30] \| \| (p_0 \oplus p_3)[30] \| \| (p_0 \oplus p_3)[26] \| \| (p_0 \oplus p_3)[25]$	$k_{\mathcal{P}_1}$
β_1^0	$\widetilde{p}_1[16] \oplus \widetilde{p}_0[26] \oplus \widetilde{p}_0[31] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[31] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[27] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[26]$	
β_0^1	$p_1[28] \oplus p_0[11] \oplus p_0[6] \| \| (p_0 \oplus p_3)[12] \| \| (p_0 \oplus p_3)[11] \| \| (p_0 \oplus p_3)[7] \| \| (p_0 \oplus p_3)[6]$	$k_{\mathcal{P}_0}$
β_0^0	$\widetilde{p}_1[29] \oplus \widetilde{p}_0[12] \oplus \widetilde{p}_0[7] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[13] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[12] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[8] \| \| (\widetilde{p}_0 \oplus \widetilde{p}_3)[7]$	

which is summarized in Appendix. B to analyze the data and time complexity. The success probability is set to 0.978. Due to the E_1 part we need to run Algorithm 1 for $N_r = 2^{22.195}$ times to generate right pairs with expected probability $\frac{1}{2}$. The average correlation of connective part and linear part is $2^{-4.017}$. Use N data samples with the threshold defined as $\Theta = \sqrt{N^*} \times \Phi^{-1}\left(1 - \frac{2^{-22.195}}{2^{13}}\right)$ in Algorithm 2 where N^* is the data complexity corresponding to valid partitions.

We can compute that $N = \frac{4}{3}N^* = 2^{14.652}$. The data complexity is $2N_r \cdot N = 2^{22.195+14.652+1} = 2^{38.85}$. The major parts impacting on the time complexity are the ergodicity of key-guessing for $k_{\mathcal{P}}$, the collection of data samples and the Walsh-Hadamard transform during the computation of $\mathcal{C}(k_{\mathcal{P}}, k_\mu)$. The running time is estimated as $N_r \cdot 2^{n_{\mathcal{P}}}(2N + \dim W \cdot 2^{\dim W}) = 2^{22.195} \times 2^9 \times (2 \times 2^{14.652} + 4 \times 2^4) = 2^{46.8}$.

5.2 Distinguisher and experimental result

Under rotational-linear cryptanalysis with E_c and E_2 presented in Sect. 5.1, we implement toy versions of Chaskey with 4 and 5 rounds to have an additional experimental confirmation of the correctness of our framework and we build a full-round distinguisher with correlation $2^{-60.38}$.

To lunch the experiment we use a set of random samples $\mathcal{S} = \{x \mid x \in \mathbb{F}_2^{128}\}$ as input pairs $(x, \overleftarrow{x})$ and compute the correlation of $\langle \beta^1, E(x) \rangle \oplus \langle \beta^0, E(\overleftarrow{x}) \rangle$ with linear masks β_1 and β_0 presented in Sect. 5.1. The 4-round distinguisher is composed of 0.5-round E_1, 3-round E_c and 0.5-round E_2. There are two modular additions in E_1 with the rotational probability $2^{-2.83}$, E_c and E_2 are presented in Sect. 5.1 where the correlations are $2^{-1.73}$ and 2^{-2} respectively. So we can obtain that the expected probability of the 4-round distinguisher is $2^{-6.56}$. And the corresponding experimental probability is $2^{-6.4}$. The expected probability and experimental probability for 5-round distinguisher are $2^{-13.74}$ and $2^{-12.1}$.

6 Conclusion

In this paper, we build a new framework of attack, called the rotational-linear attack. As a new combination of different kinds of attacks, we establish the key-

recovery framework based on rotational cryptanalysis, which is mainly used in distinguishing attacks before. We also present a valid way to reduce the data complexity for the rotational part of our rotational-linear attack, called *bit flipping*. Our method is applied to Chaskey successfully. We built a full-round distinguisher for Chaskey and recovered partial bits of key under related-key attack, for keys belonging to a large weak-key class based on a 6-round distinguisher. It should be noted that although related-key attacks exists for any Even-Mansour ciphers [9], differentials of the whiting keys are considered before, while in our attack keys are related in a rotational manner.

The simple requirement that a cipher is suitable to rotational-linear attacks is that the rotational cryptanalysis can be applied to it. So rotational-linear attacks can be applied to many ARX ciphers, like Threefish and BLAKE2 [1]. And we hope that our framework of attack can be applied to other ARX ciphers to understand the constructing of rotational and linear cryptanalysis better. There still exist some possible improvements to our results. First, the connective part between rotational and linear parts is a little different from the differential-linear connectivity. Maybe one could construct a more exact connective part. Second, multidimensional linear cryptanalysis [11] maybe a way to improve the linear part of our attack.

References

1. Aumasson, J., Neves, S., Wilcox-O'Hearn, Z., Winnerlein, C.: BLAKE2: simpler, smaller, fast as MD5. In: Jr., M.J.J., Locasto, M.E., Mohassel, P., Safavi-Naini, R. (eds.) Applied Cryptography and Network Security - 11th International Conference, ACNS 2013, Banff, AB, Canada, June 25-28, 2013. Proceedings. Lecture Notes in Computer Science, vol. 7954, pp. 119–135. Springer (2013)
2. Bar-On, A., Dunkelman, O., Keller, N., Weizman, A.: Dlct: A new tool for differential-linear cryptanalysis. In: Ishai, Y., Rijmen, V. (eds.) Advances in Cryptology – EUROCRYPT 2019. pp. 313–342. Springer International Publishing, Cham (2019)
3. Barbero, S., Bellini, E., Makarim, R.H.: Rotational analysis of chacha permutation. CoRR **abs/2008.13406** (2020)
4. Beierle, C., Leander, G., Todo, Y.: Improved differential-linear attacks with applications to arx ciphers. In: Micciancio, D., Ristenpart, T. (eds.) Advances in Cryptology – CRYPTO 2020. pp. 329–358. Springer International Publishing, Cham (2020)
5. Bernstein, D.J.: The salsa20 family of stream ciphers. In: Robshaw, M.J.B., Billet, O. (eds.) New Stream Cipher Designs - The eSTREAM Finalists, Lecture Notes in Computer Science, vol. 4986, pp. 84–97. Springer (2008)
6. Biham, E., Carmeli, Y.: An improvement of linear cryptanalysis with addition operations with applications to FEAL-8X. In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. Lecture Notes in Computer Science, vol. 8781, pp. 59–76. Springer (2014)
7. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) Advances in Cryptology - CRYPTO '90, 10th

Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings. Lecture Notes in Computer Science, vol. 537, pp. 2–21. Springer (1990)
8. Carlet, C., Crama, Y., Hammer, P.L.: Boolean functions for cryptography and error-correcting codes. In: Crama, Y., Hammer, P.L. (eds.) Boolean Models and Methods in Mathematics, Computer Science, and Engineering, pp. 257–397. Cambridge University Press (2010)
9. Cogliati, B., Seurin, Y.: On the provable security of the iterated even-mansour cipher against related-key and chosen-key attacks. In: Oswald, E., Fischlin, M. (eds.) Advances in Cryptology – EUROCRYPT 2015. pp. 584–613. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
10. Daum, M.: Cryptanalysis of Hash functions of the MD4-family. Ph.D. thesis, Ruhr University Bochum (2005)
11. Hermelin, M., Cho, J.Y., Nyberg, K.: Multidimensional linear cryptanalysis. J. Cryptol. **32**(1), 1–34 (2019)
12. Khovratovich, D., Nikolić, I.: Rotational cryptanalysis of arx. In: Hong, S., Iwata, T. (eds.) Fast Software Encryption. pp. 333–346. Springer Berlin Heidelberg, Berlin, Heidelberg (2010)
13. Khovratovich, D., Nikolić, I., Pieprzyk, J., Sokołowski, P., Steinfeld, R.: Rotational cryptanalysis of arx revisited. In: Leander, G. (ed.) Fast Software Encryption. pp. 519–536. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
14. Kraleva, L., Ashur, T., Rijmen, V.: Rotational cryptanalysis on mac algorithm chaskey. In: Conti, M., Zhou, J., Casalicchio, E., Spognardi, A. (eds.) Applied Cryptography and Network Security. pp. 153–168. Springer International Publishing, Cham (2020)
15. Langford, S.K., Hellman, M.E.: Differential-linear cryptanalysis. In: Desmedt, Y. (ed.) Advances in Cryptology - CRYPTO '94, 14th Annual International Cryptology Conference, Santa Barbara, California, USA, August 21-25, 1994, Proceedings. Lecture Notes in Computer Science, vol. 839, pp. 17–25. Springer (1994). https://doi.org/10.1007/3-540-48658-5_3
16. Leurent, G.: Improved differential-linear cryptanalysis of 7-round chaskey with partitioning. In: Fischlin, M., Coron, J.S. (eds.) Advances in Cryptology – EUROCRYPT 2016. pp. 344–371. Springer Berlin Heidelberg, Berlin, Heidelberg (2016)
17. Liu, Y., Sun, S., Li, C.: Rotational cryptanalysis from a differential-linear perspective: Practical distinguishers for round-reduced friet, xoodoo, and alzette. Cryptology ePrint Archive, Report 2021/189 (2021), https://eprint.iacr.org/2021/189
18. Matsui, M.: Linear cryptanalysis method for des cipher. In: Helleseth, T. (ed.) Advances in Cryptology — EUROCRYPT '93. pp. 386–397. Springer Berlin Heidelberg, Berlin, Heidelberg (1994)
19. Mouha, N., Mennink, B., Herrewege, A.V., Watanabe, D., Preneel, B., Verbauwhede, I.: Chaskey: An efficient MAC algorithm for 32-bit microcontrollers. In: Joux, A., Youssef, A.M. (eds.) Selected Areas in Cryptography - SAC 2014 - 21st International Conference, Montreal, QC, Canada, August 14-15, 2014, Revised Selected Papers. Lecture Notes in Computer Science, vol. 8781, pp. 306–323. Springer (2014)
20. Wagner, D.A.: The boomerang attack. In: Knudsen, L.R. (ed.) Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1636, pp. 156–170. Springer (1999)

A Application to ChaCha permutation

The stream cipher ChaCha is an improvement of Salsa20 [5]. Each round of ChaCha uses 4 *Quarter Round Functions*, denoted by $QR(v_a^{(r)}, v_b^{(r)}, v_c^{(r)}, v_d^{(r)})$, to permute the 4×4 state matrix, denoted by $V^{(r)}$. Every word $v_i^{(r)}$ in $V^{(r)}$ is 32 bits, $i \in \{0, \ldots, 15\}$. For odd rounds, $V^{(r+1)}$ is calculated by selecting 4 columns, i.e., $(v_0^{(r)}, v_4^{(r)}, v_8^{(r)}, v_{12}^{(r)})$, $(v_1^{(r)}, v_5^{(r)}, v_9^{(r)}, v_{13}^{(r)})$, $(v_2^{(r)}, v_6^{(r)}, v_{10}^{(r)}, v_{14}^{(r)})$ and $(v_3^{(r)}, v_7^{(r)}, v_{11}^{(r)}, v_{15}^{(r)})$ as the inputs for QR functions. For even rounds, $V^{(r+1)}$ is computed by selecting 4 diagonals $(v_0^{(r)}, v_5^{(r)}, v_{10}^{(r)}, v_{15}^{(r)})$, $(v_1^{(r)}, v_6^{(r)}, v_{11}^{(r)}, v_{12}^{(r)})$, $(v_2^{(r)}, v_7^{(r)}, v_8^{(r)}, v_{13}^{(r)})$, $(v_3^{(r)}, v_4^{(r)}, v_9^{(r)}, v_{14}^{(r)})$ as the inputs for QR functions. The round function QR is presented in Fig. 6. In [3], the authors applied rotational

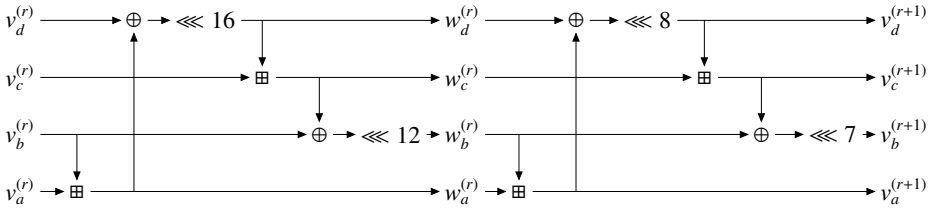

Fig. 6. The QR function of ChaCha.

cryptanalysis to the underlying permutation of ChaCha. They presented a rotational distinguisher for 17-round ChaCha permutation with probability greater than 2^{-488} whereas the probability of random permutation with same input size is 2^{-511}. It declares that the underlying permutation of ChaCha doesn't behave as a random permutation.

The extended application of rotational-linear cryptanalysis is presented as follows. We build a rotational-linear distinguisher for 17-round ChaCha permutation with 15-round rotational part E_1, 1-round connective part E_c and 1-round linear part E_2. The lower bound for the probability of E_1 is $2^{-430.2}$, given by [3]. The correlation of E_c is 2^{-2} with the masks $\alpha = \nu_{15}^{15}[0]$ and $\overleftarrow{\alpha} = \nu_{15}^{15}[1]$. For linear part E_2, the two output trails are $\beta_1 = (\nu_0^{16}[16, 0] \oplus \nu_5^{16}[7] \oplus \nu_{10}^{16}[0] \oplus \nu_{15}^{16}[24])$ and $\beta_0 = (\nu_0^{16}[17, 1, 0] \oplus \nu_5^{16}[8] \oplus \nu_{10}^{16}[1] \oplus \nu_{15}^{16}[25])$ with corresponding correlation 2^{-1}. In conclusion, the correlation of rotational-linear distinguisher for 17-round ChaCha permutation is greater than 2^{-433}. Compared to rotational cryptanalysis, the rotational-linear attack exhibits advantages.

B The proposition used when recovering partial key

Proposition 2 ([4]) *After running Algorithm 1 for N_r times, the probability that the correct key is among the key candidates is*

$$p_{success} \geq \frac{1}{2}\Pr(\mathcal{C}(k_\mu, k_\mathcal{P}) \geq \Theta) = \frac{1}{2}\left(1 - \Phi\left(\frac{\Theta - N \cdot cor}{\sqrt{N}}\right)\right).$$