Noname manuscript No.
(will be inserted by the editor)

Generalized Related-Key Rectangle Attacks on Block Ciphers with Linear Key Schedule

Applications to SKINNY and GIFT

**Boxin Zhao · Xiaoyang Dong · Willi Meier ·
Keting Jia · Gaoli Wang**

Received: date / Accepted: date

Abstract This paper gives a new generalized key-recovery model of related-key rectangle attacks on block ciphers with linear key schedules. The model is quite optimized and applicable to various block ciphers with linear key schedule. As a proof of work, we apply the new model to two very important block ciphers, i.e. SKINNY and GIFT, which are basic modules of many candidates of the Lightweight Cryptography (LWC) standardization project by NIST.

For SKINNY, we reduce the complexity of the best previous 27-round related-tweakey rectangle attack on SKINNY-128-384 from 2^{331} to 2^{294}. In addition, the first 28-round related-tweakey rectangle attack on SKINNY-128-384 is given, which gains one more round than before. For the candidate LWC SKINNY AEAD M1, we conduct a 24-round related-tweakey rectangle attack with a time complexity of 2^{123} and a data complexity of 2^{123} chosen plaintexts. For the case of GIFT-64, we give the first 24-round related-key rectangle attack with a time complexity $2^{91.58}$, while the best previous attack on GIFT-64 only reaches 23 rounds at most.

Keywords MSC (2000): 94A60; Key Recovery, Rectangle Attack, SKINNY, SKINNY AEAD, GIFT, Related-Key

Mathematics Subject Classification (2000) 94A60

1 Introduction

The boomerang attack [46], proposed by Wagner, is a variant of differential cryptanalysis [17]. It combines two short differentials with high probabilities to get a long distinguisher. Refinements on the boomerang attack have been published, namely, the amplified boomerang attack [31], and thereafter the rectangle attack [7]. At ASIACRYPT 2009, Biryukov *et al.* [13] introduced the concept of boomerang switch to further increase the probability of the boomerang distinguisher. Another improvement was made by Dunkelman *et al.* [23], which is called sandwich attack. At Eurocrypt 2018, Cid *et al.* [20] proposed a novel technique named Boomerang connectivity table (BCT), which solved the problem of incompatibility in boomerang distinguishers noted by Murphy [36]. Later, the BCT effect in multiple rounds of boomerang switch was studied by Wang and Peyrin [47] and Song *et al.* [42].

Boomerang and rectangle attacks in a related-key setting [9] are quite powerful, which break various important block ciphers, including the key-recovery attacks on KASUMI [10, 23] and AES [13]. Recently, many (tweakable) block ciphers adopt linear key schedules, such as MANTIS [11], LED [25], MIDORI [4], GIFT [16], Simon [18], CRAFT [15], and the popular TWEAKEY [29] framwork based ciphers, including SKINNY [11], Deoxys-BC [30], QARMA [3], and Joltik-BC [29]. Notably, in the CAESAR competition [21] for secure authenticated encryption, Deoxys-II [30], which is based on Deoxys-BC, has been selected as one of the winners.

Boxin Zhao is with Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, School of Mathematics, Shandong University, Jinan 250100, China. E-mail: boxinzhao@mail.sdu.edu.cn
Xiaoyang Dong is with Institute for Advanced Study, Tsinghua University, Beijing 100084, China.
E-mail: xiaoyangdong@tsinghua.edu.cn
Willi Meier is with FHNW, Institute ISE, Windisch, Aargau Switzerland. E-mail: willi.meier@fhnw.ch
Keting Jia is with Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China.
E-mail: ktjia@tsinghua.edu.cn
Gaoli Wang is with Shanghai Key Lab of Trustworthy Computing, East China Normal University, Shanghai 200062, China.
E-mail: glwang@sei.ecnu.edu.cn
Xiaoyang Dong is the corresponding author.

With the significant spread of the Internet of Things (IoT) in recent years, lightweight cryptography is urgently needed in numerous devices with little computing power. Therefore, NIST launched the Lightweight Cryptography (LWC) standardization project [37] to select lightweight authenticated encryption with associated data (AEAD) and hashing function. In 2019, about 56 candidates were included in the Round 1 of the project [37]. Among the candidates, many are based on block ciphers with linear key schedule to become lightweight, such as SKINNY-AEAD and SKINNY-Hash [12], SUNDAE-GIFT [5], TGIF [26], GIFT-COFB [6], Remus [27], Romulus [28] and Saturnin [19], etc. The study of boomerang and rectangle attacks on block ciphers with linear key schedule becomes relevant. At ToSC 2017, Liu *et al.* [33] introduced a generalized key-recovery model for the related-key rectangle attack on block ciphers with linear key schedule. Then, they applied their model to the attacks on reduced-round SKINNY [11].

Our Contributions.

In this paper, we find that Liu *et al.*'s model [33] can be significantly improved in the phase of generating quartets. Therefore, we construct a new key-recovery model for the related-key rectangle attacks on block ciphers with linear key schedules. In order to show the effectiveness of model, we apply it to the two important block ciphers, i.e. SKINNY [11] and GIFT [16]. Note that, in the LWC standardization project by NIST [37], many candidates such as SKINNY-AEAD and SKINNY-Hash [12], SUNDAE-GIFT [5], TGIF [26], GIFT-COFB [6], Remus [27] and Romulus [28] are based on SKINNY or GIFT. To study SKINNY and GIFT is very important for the security evaluation of these candidates.

For the sake of a clear comparison between our model and the previous one by Liu *et al.* [33], we utilize the same 23-round boomerang distinguishers of SKINNY-128-384 [11] as proposed by Liu *et al.* [33] and the same 19-round boomerang distinguishers on GIFT-64 [16] as proposed by Chen *et al.* [22] to launch our key-recovery attacks.

- For SKINNY-128-384, we improve the time complexity of the best previous 27-round attack by a factor of 2^{37}. Moreover, we present the first key-recovery attack on 28-round SKINNY-128-384 with a time complexity of $2^{315.25}$ and 2^{122} chosen plaintexts.
- In addition, we give a related-tweakey rectangle attack on 24-round SKINNY-128-384 with time and data complexities of 2^{123}, which is successfully applied to the SKINNY AEAD member M1 [12] (one of the 56 candidates in the NIST Lightweight Cryptography selection process). To our knowledge, this is the first attack on round-reduced SKINNY AEAD M1.
- For GIFT-64, we conduct a 24-round attack [1], which gains one more round than the best previous attacks, and the time complexity is $2^{91.58}$.

The cryptanalysis results on SKINNY-128-384, SKINNY AEAD M1 scheme and GIFT-64 are listed in Table 1.

2 The Related-key Rectangle Attack

The boomerang attack, proposed by Wagner [46], is an extension of the differential attack using adaptive chosen plaintexts and ciphertexts to analyze block ciphers. It attempts to generate a quartet structure at an intermediate value halfway through the cipher. When the adversaries can not find a long differential characteristic with probability higher than for a random permutation, they can decompose the cipher in two shorter ciphers as $E = E_1 \circ E_0$ and connect two short differential trails to conduct the attack. For E_0, the differential characteristic is $\alpha \to \beta$ with probability p, and the differential characteristic for E_1 is $\delta \to \gamma$ with probability q.

Then a right quartet can be obtained by a boomerang distinguisher which is the connection of the two shorter differential characteristics as in the following steps:

1. Randomly choose a plaintext pair (P_1, P_2) with difference $P_1 \oplus P_2 = \alpha$, and make queries over E to get the ciphertext pair (C_1, C_2), where $C_1 = E(P_1), C_2 = E(P_2)$.
2. Generate another ciphertext pair (C_3, C_4) by $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$, then make queries to the decryption oracle to obtain their plaintexts (P_3, P_4) with two adaptive chosen-ciphertext queries.
3. Check whether the difference of (P_3, P_4) equals to α or not.

The adversary can get a right quartet with a probability of p^2q^2, thus the probability of the distinguisher has to satisfy $pq > 2^{-n/2}$.

When the values of α and δ are fixed and don't restrain the values of β and γ as long as $\beta \neq \gamma$, the boomerang attack is developed into the amplified boomerang attack [31] or rectangle attack [7], which are

[1] Note that the authors of GIFT [16] do not give any security claim in the related-key setting, but as shown by Liu *et al.* [48] and Chen *et al.* [22], it is still theoretically meaningfull to understand its security margin in this setting.

Table 1: Summary of analysis results of SKINNY-128-384, GIFT-64 and SKINNY AEAD M1. (I)DC stands for (impossible) differential cryptanalysis; IC stands for integral cryptanalysis; MITM stands for meet-in-the-middle attack; SK stands for single-key; RK stands for related-key.

SKINNY-128-384

Rounds	Approach	Setting	Time	Data	Memory	Size set up	Ref.
22	IDC	SK	$2^{373.48}$	$2^{92.22}$	$2^{147.22}$	$k = 384$	[45]
22	MITM	SK	$2^{382.46}$	2^{96}	$2^{330.99}$	$k = 384$	[43]
27	rectangle	RK	2^{351}	2^{127}	2^{160}	$k = 384$	[33]
	rectangle	RK	2^{331}	2^{123}	2^{155}	$k = 384$	[33]
	rectangle	RK	2^{294}	2^{122}	$2^{122.32}$	$k = 384$	Sect. 4.4
28	rectangle	RK	$2^{315.25}$	2^{122}	$2^{122.32}$	$t < 68, k > 316$	Sect. 4.5

GIFT-64

Rounds	Approach	Setting	Time	Data	Memory	Size set up	Ref.
14	IC	SK	2^{97}	2^{63}	–	$k = 128$	[16]
15	MITM	SK	2^{120}	2^{64}	–	$k = 128$	[16]
15	MITM	SK	2^{112}	–	–	$k = 128$	[38]
19	DC	SK	2^{112}	2^{63}	–	$k = 128$	[48]
20	DC	SK	$2^{112.68}$	2^{62}	2^{112}	$k = 128$	[49]
21	DC	SK	$2^{107.61}$	2^{64}	2^{96}	$k = 128$	[49]
23	boomerang	RK	$2^{126.6}$	$2^{63.3}$	–	$k = 128$	[34]
23	rectangle	RK	2^{107}	2^{60}	2^{60}	$k = 128$	[22]
24	rectangle	RK	$2^{91.58}$	2^{60}	$2^{60.32}$	$k = 128$	Sect. 5.3

SKINNY AEAD M1

	Rounds	key size	Time	Data	Memory	Approach	Ref.
SKINNY M1	24	128	2^{123}	2^{123}	2^{121}	rectangle	Sect. 4.7

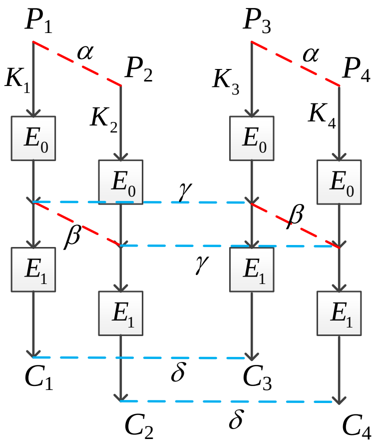

Fig. 1: Related-key rectangle distinguisher framework

chosen-plaintext attacks. The probability of getting a right quartet is $2^{-n}\hat{p}^2\hat{q}^2$, where

$$\hat{p} = \sqrt{\sum_i Pr^2(\alpha \to \beta_i)} \ \ and \ \ \hat{q} = \sqrt{\sum_j Pr^2(\gamma_j \to \delta)}.$$

If the four plaintexts in a quartet are encrypted under different master keys K_1, K_2, K_3 and K_4 respectively, the attack is developed into a related-key rectangle attack [9], where $C_1 = E_{K_1}(P_1)$, $C_2 = E_{K_2}(P_2)$, $C_3 = E_{K_3}(P_3)$, and $C_4 = E_{K_4}(P_4)$. Assume one has a related-key differential $\alpha \to \beta$ over E_0 under a key difference ΔK with a probability p and another related-key differential $\gamma \to \delta$ over E_1 under a key difference ∇K with probability q, and $K_1 \oplus K_2 = \Delta K$, $K_3 \oplus K_4 = \Delta K$, $K_1 \oplus K_3 = \nabla K$. Then a right quartet can be obtained as follows:

1. Randomly choose two plaintext pairs (P_1, P_2) and (P_3, P_4) with difference $P_1 \oplus P_2 = \alpha$ and $P_3 \oplus P_4 = \alpha$, and encrypt them with E to get the ciphertext pairs (C_1, C_2) and (C_3, C_4) under four master keys, where $K_1 \oplus K_2 = \Delta K$, $K_3 \oplus K_4 = \Delta K$ and $K_1 \oplus K_3 = \nabla K$.

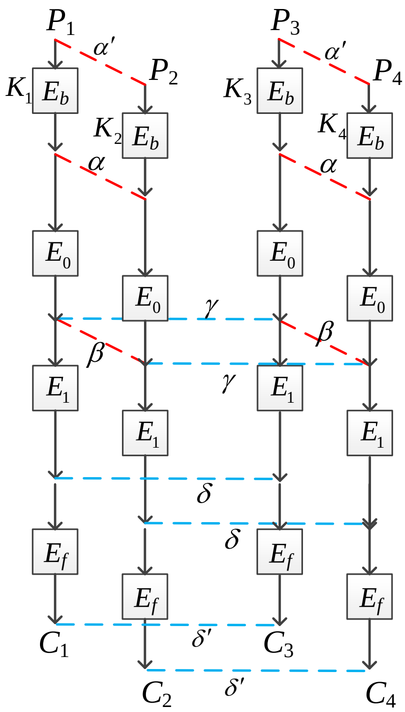

Fig. 2: Related-key rectangle attack framework

2. Check whether the differences satify $C_1 \oplus C_3 = \delta$ and $C_2 \oplus C_4 = \delta$ or not. If yes, a right quartet is obtained, otherwise return to step 1.

In the key recovery process, adversaries only need to recover one of the four master keys, since the values of $\varDelta K$ and ∇K are known and the other three master keys can be computed by the recovered key.

For clarity, the related-key rectangle framework is illustrated in Figure 1.

3 New Model of Related-key Rectangle Attack

Under a related-key rectangle distinguisher, our key recovery algorithm is adapted from Biham *et al.*'s algorithm [8] which is a single-key rectangle attack and Liu *et al.*'s [33] algorithm which is a related-key rectangle attack. In the key recovery algorithm, we follow the notations in [33].

We decompose the cipher algorithm E into three components as $E = E_f \circ E' \circ E_b$, where E' is determined by the related-key rectangle distinguisher and E_b and E_f are the rounds extended backward from the start and forward from the end of the distinguisher, respectively. Let c denote the size of a cell (the unit goes through Sbox), k denote the size of the master key and n be the size of the state in the block cipher. After extending the rectangle distinguisher backward for several rounds (*i.e.* E_b) under the related-key difference $\varDelta K$, we denote the number of unknown bits in the difference of plaintexts as r_b. Let m_b be the number of involved subkey bits that affect the plaintext difference while encrypting plaintext pairs to the position of the known difference under E_b. Similarly, when extending several rounds for the difference of the distinguisher δ under E_f by the related-key difference ∇K, we define r_f and m_f for E_f. The related-key rectangle framework is illustrated in Figure 2. The algorithm being composed of data collection and key recovery is as follows:

1. Construct structures including 2^{r_b} plaintexts, which traverse all the possible values for the r_b/c active cells while assigning suitable constants to the other cells that hold zero or known differences. If s denotes the expected number of right quartets, attackers need to prepare $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q}$ different structures.
2. Query the corresponding ciphertexts for the 2^{r_b} plaintexts in each structure under the four related keys K_1, K_2, K_3 and K_4 and get four plaintext-ciphertext sets L_1, L_2, L_3 and L_4, where K_1 is the secret key and $K_2 = K_1 \oplus \varDelta K$, $K_3 = K_1 \oplus \nabla K$ and $K_4 = K_1 \oplus \varDelta K \oplus \nabla K$. Insert L_2 and L_4 into hash tables H_1 and H_2 indexed by the r_b bits of plaintexts.
3. Guess the 2^{m_b} possible m_b bits of subkey involved in E_b:
   (a) Initialize a list of 2^{m_f} counters, each of which corresponds to a m_f-bit subkey guess.

(b) For each set L_1 of every structure, partially encrypt plaintext $P_1 \in L_1$ under E_b by the guessed subkeys of K_1, and partially decrypt it under the subkey of $K_2 = K_1 \oplus \Delta K$ to the plaintext P_2 after xoring the known difference α, *i.e.* $P_2 = D_{b_{K_2}}(E_{b_{K_1}}(P_1) \oplus \alpha)$ where $D_{b_{K_2}}$ is the partial decryption process D_b using K_2. Then check H_1 to find the corresponding plaintext-ciphertext pair indexed by the r_b bits of P_2. Proceed with a similar process for sets L_3 and L_4 and obtain two sets as

$$S_1 = \{(P_1, C_1, P_2, C_2) : (P_1, C_1) \in L_1, (P_2, C_2) \in L_2, E_{b_{K_1}}(P_1) \oplus E_{b_{K_2}}(P_2) = \alpha\}$$

and

$$S_2 = \{(P_3, C_3, P_4, C_4) : (P_3, C_3) \in L_3, (P_4, C_4) \in L_4, E_{b_{K_3}}(P_3) \oplus E_{b_{K_4}}(C_4) = \alpha\}.$$

(c) There are $M = y \cdot 2^{r_b}$ chosen plaintexts under each key, and the sizes of S_1 and S_2 are all $y \cdot 2^{r_b}$ due to y structures. Denote δ' being the truncated form propagated from δ under E_f with probability 1. There are $n - r_f$ bits whose differences are 0 in δ'. Insert S_1 into hash table H_3 indexed by the $n - r_f$ bits of C_1 and $n - r_f$ bits of C_2 that are 0 in δ'. Then for each element of S_2, we check the hash table H_3 to find (P_1, C_1, P_2, C_2) so that (C_1, C_3) and (C_2, C_4) collide in the $n - r_f$ bits. There will be about $M^2 \cdot 2^{-2(n-r_f)}$ quartets remaining.

(d) With the quartets obtained in step (c), we conduct the key recovery process for the subkeys involved in E_f. Instead of guessing all the m_f-bit subkeys at once, we firstly determine whether a candidate quartet is useful by guessing only a small fraction of the unknown involved subkey bits, which is just a guess and filter procedure. We denote the time complexity in this step as ε.

(e) Select the top $2^{m_f - h}$ hits in the counter to be the candidates, which delivers a h-bit or higher advantage.

(f) Guess the remaining $k - m_b - m_f$ unknown key bits exploiting the key schedule algorithm, and exhaustively search over them to recover the correct master key. If the guessed m_b-bit keys are not right, go to step 3 with another guess.

Since the key recovery attack is a related-key attack, the data complexity is $D = 4M = 4y \cdot 2^{r_b}$ chosen plaintexts.

In the quartets collection and key recovery process, we need $2^{m_b} \cdot 2M$ table look-ups in step 3(b) and $2^{m_b} \cdot M$ table look-ups in step 3(c) resulting in $2^{m_b} \cdot 3M$ to prepare the quartets. $M^2 \cdot 2^{-2(n-r_f)} \cdot \varepsilon$ encryptions in step 3(d) and 2^{k-h} encryptions in step 3(f) are needed to recover the master key. Thus the total time complexity, which is composed of data collection and key recovery, is $4M + 2^{m_b} \cdot M^2 \cdot 2^{-2(n-r_f)} \cdot \varepsilon + 2^{k-h}$.

The memory complexity is $4M + M + 2^{m_f} = 5M + 2^{m_f}$.

Success Probability. We use the method by Selçuk [39] to compute the success probability:

$$P_s = \Phi\left(\frac{\sqrt{sS_N} - \Phi^{-1}(1 - 2^{-h})}{\sqrt{S_N + 1}}\right), \tag{1}$$

where S_N is the signal-to-noise ratio and $S_N = \hat{p}^2 \hat{q}^2 / 2^{-n}$.

Compared to the related-key rectangle attack in [33], the main improvements in time complexity are summarized as follows:

1. In the data collection program to prepare quartets, paper [33] constructs quartets first, and then checks whether the quartet can be encrypted to the difference of the start of the distinguisher one by one to filter all the useless quartets. However, we guess the key bits involved in the E_b and make partial encryption and decryption to construct plaintext pairs that can be encrypted to the difference of the start of the distinguisher. Therefore, the quartets we obtained don't need to be filtered. It makes the time complexity be highly reduced.
2. In the key recovery process, we don't guess all the key bits involved in E_f but utilize a process that guess partial key bits one time and filter the useless quartets step by step. It makes the time complexity reduce further.

4 Application to SKINNY

The SKINNY family [11] provides 64-bit and 128-bit block versions and denotes n as the block size. Several candidates of the Lightweight Cryptography (LWC) standardization project by NIST [37] are based on the SKINNY block cipher, such as SKINNY-AEAD and SKINNY-Hash [12], Remus [27] and Romulus [28]. The family of lightweight block ciphers SKINNY has three tweakey size versions SKINNY$-n-t$, where $t = n$, $t = 2n$ and $t = 3n$, and denotes the tweakey state by $TK1$ when $t = n$, by $TK1$ and $TK2$ when $t = 2n$, and finally by $TK1$, $TK2$ and $TK3$ when $t = 3n$.

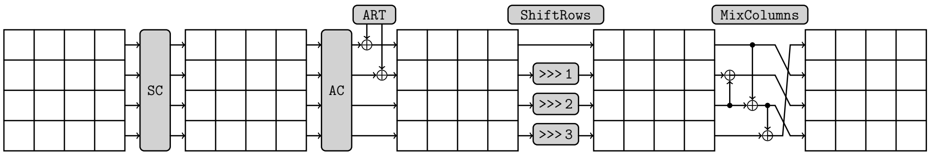

Fig. 3: The SKINNY round function

Since SKINNY was proposed, there has been a number of third-party cryptanalysis from all over the world. Tolba *et al.* [45] applied impossible differential attacks to 18-, 20- and 22-round SKINNY-n-n, SKINNY-n-$2n$ and SKINNY-n-$3n$, respectively, in the single-key model at AFRICACRYPT 2017. At ToSC 2017, Liu *et al.* [33] searched related-tweakey impossible differentials and related-tweakey rectangle distinguishers and applied them to analyze up to 19-, 23- and 27-round SKINNY-n-n, SKINNY-n-$2n$ and SKINNY-n-$3n$ respectively. In [2], Abdelkhalek *et al.* proposed a method to model the DDT of large Sboxes and verified that no differential characteristic with probability higher than 2^{-128} for 14-round SKINNY-128 exists. In [41], Sadeghi *et al.* presented zero correlation attacks on SKINNY-64-64/128 and gave a related-tweakey impossible differential attack on SKINNY-128-256 up to 23 rounds. At ASIACRYPT 2018, Shi *et al.* [43] analyzed 22-round SKINNY-128-384 by the Demirci-Selcuk meet-in-the-middle attack. At ToSC 2019, Song *et al.* [42] revisited the Boomerang Connectivity Table [20] and recalculated the probabilities of some related-tweakey boomerang distinguishers proposed in [33]. Besides, there are analyses on SKINNY-64 in [1, 40, 44].

4.1 Specification of SKINNY and SKINNY AEAD

The block cipher SKINNY [11] is an SPN cipher that uses a compact Sbox, a sparse diffusion layer and a light key schedule. SKINNY follows the TWEAKEY framework [29], thus except for a plaintext P and a master key K it takes a tweak T as the third input, and different ciphertexts can be obtained under the same plaintext and master key due to the different tweaks. Inspired by the TWEAKEY framework [29], SKINNY provides a unified view for key and tweak by tweakey.

For all versions of SKINNY, the tweak size and the key size can vary according to the users but the key size should be at least as large as the block size. Both the intermediate state and tweakey state are viewed as a 4×4 square array of cells indexed by

$$\begin{bmatrix} 0 & 1 & 2 & 3 \\ 4 & 5 & 6 & 7 \\ 8 & 9 & 10 & 11 \\ 12 & 13 & 14 & 15 \end{bmatrix}.$$

Note that SKINNY adopts a row-wise form rather than column-wise fashion as AES [24], as it is more hardware-friendly as pointed out in [35].

The Round Function. One encryption round of SKINNY consists of five operations in the following order: `SubCells` (SC), `AddConstants` (AC), `AddRoundTweakey` (ART), `ShiftRows` (SR) and `MixColumns` (MC), which is illustrated in Figure 3.

`SubCells.` Apply a 4-bit Sbox in case of $n = 64$ and an 8-bit Sbox in case of $n = 128$ to the 16 cells of the state. It is the non-linear operation in the round function.

`AddConstants.` The round constants are generated by a 6-bit affine LFSR. Three round constants are xor'ed to the first cell of the first three rows, respectively.

`AddRoundTweakey.` Only the first two rows of the round tweakey are xor'ed to the first two rows of the internal state. The round tweakey tk_i in round i is defined as:

- $t = n$: $tk_i = (TK1)_i$,
- $t = 2n$: $tk_i = (TK1)_i \oplus (TK2)_i$,
- $t = 3n$: $tk_i = (TK1)_i \oplus (TK2)_i \oplus (TK3)_i$,

where $(TK1)_i$, $(TK2)_i$ and $(TK3)_i$ are generated by the tweakey schedule algorithm that is introduced below.

Table 2: The two LFSRs used in SKINNY tweakey schedule

$LFSR_2$	$(x_7\|\|x_6\|\|x_5\|\|x_4\|\|x_3\|\|x_2\|\|x_1\|\|x_0) \rightarrow (x_6\|\|x_5\|\|x_4\|\|x_3\|\|x_2\|\|x_1\|\|x_0\|\|x_7 \oplus x_5)$
$LFSR_3$	$(x_7\|\|x_6\|\|x_5\|\|x_4\|\|x_3\|\|x_2\|\|x_1\|\|x_0) \rightarrow (x_0 \oplus x_6\|\|x_7\|\|x_6\|\|x_5\|\|x_4\|\|x_3\|\|x_2\|\|x_1)$

`ShiftRows`. Rotate the 4 cells of the $j-$th row right by $\rho[j]$ positions, where $\rho = (0, 1, 2, 3)$.

`MixColumns`. Pre-multiply the internal state by a 4×4 binary constant matrix $\mathbf{M}$ to update the state. The matrix $\mathbf{M}$ and its inverse matrix $\mathbf{M}^{-1}$ are represented as follows:

$$\mathbf{M} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{bmatrix}, \quad \mathbf{M}^{-1} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{bmatrix}.$$

Definition of the round tweakey (subtweakey). The tweakey schedule algorithm of SKINNY is a linear transformation. The t-bit tweakey input is firstly divided into $z = t/n$ n-bit blocks, and located in $TK1$ with $t = n$, or $TK1,\ TK2$ with $t = 2n$ or $TK1,\ TK2,\ TK3$ with $t = 3n$.

First, a permutation P_T is applied to the cells of all tweakey arrays as $TK_{z_i} \leftarrow TK_{z_{P_T[i]}}$ for all $0 \leq i \leq 15$ with

$$P_T = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7],$$

for $z \in \{1, 2\}$ and $z \in \{1, 2, 3\}$. This is a position permutation with the value unchanged.

Then, each cell in the first two rows of $TK2$ (or $TK2$ and $TK3$) are updated by one (or two) LFSRs to get $(TKm)_i$ $(m = 1, 2, 3)$. The LFSRs are listed in Table 2, we only give the LFSRs used for the 8-bit cells.

The SKINNY AEAD modes.

The SKINNY AEAD modes are proposed by Beierle *et al.* [12] and included in the Round 1 candidates of the NIST lightweight cryptography competition. The authenticated encryption scheme follows the Θ`CB3` mode [32] and uses either SKINNY-128-384 or SKINNY-128-256 as its internal tweakable block cipher. Totally, there are six AEAD modes proposed and the SKINNY AEAD M1 based on SKINNY-128-384 is their primary member. M1 is a nonce-based AEAD and assumed to be nonce-respecting for the adversary. Here, we give a simple description for the AEAD M1.

The tweakey size of SKINNY AEAD M1 is 384 bits, the last 256 bits of the tweakey is just the concatenation of the 128-bit nonce N and the 128-bit key K, but the first 128 bits are different in different blocks in a long message. They can be updated in a series of blocks in the following way.

The first 128 bits of tweakey store eight bytes that come from a 64-bit LFSR, followed by seven bytes of zeros and a single byte for the domain separation (d_0 or d_1 whether the block is padded). The LFSR is initialized to $\mathrm{LFSR}_0 = 0^{63}||1$ and updated by upd_{64} that is defined as

$\mathrm{upd}_{64} : x_{63}||x_{62}|| \cdots ||x_1||x_0 \ \rightarrow \ y_{63}||y_{62}|| \cdots ||y_1||y_0$

with:

$$\begin{aligned} &y_i \leftarrow x_{i-1} \quad for \quad i \in \{63, 62, ..., 1\} \backslash \{4, 3, 1\}, \\ &y_4 \leftarrow x_3 \oplus x_{63}, \\ &y_3 \leftarrow x_2 \oplus x_{63}, \\ &y_1 \leftarrow x_0 \oplus x_{63}, \\ &y_0 \leftarrow x_{63}. \end{aligned}$$

Before the bytes of the LFSR are loaded in the tweakey input, the order of them is reversed, *i.e.* $\mathrm{rev}_{64}(LFSR)||0^{56}||d_0$ (d_0 will be replaced by d_1 for the padded block), where rev_{64} is defined as

$\mathrm{rev}_{64} : x_7||x_6||x_5||x_4||x_3||x_2||x_1||x_0 \longmapsto$
$x_0||x_1||x_2||x_3||x_4||x_5||x_6||x_7 \ (\forall i : |x_i| = 8)$

In encryption for each block, the 384-bit tweakey is set to be $\mathrm{rev}_{64}(LFSR)||0^{56}||d_0||N||K$. In fact, as described in [12], the 64-bit LFSR plays the same role as a block counter. The encryption part of SKINNY AEAD M1 is illustrated in Figure 4. For more details, we refer to [12].

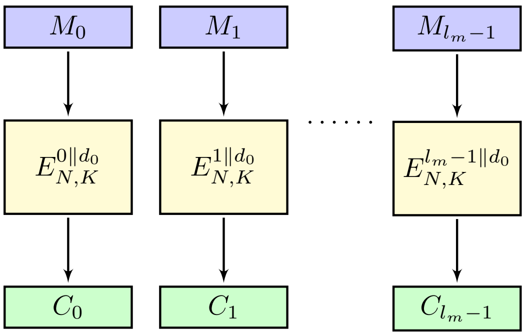

Fig. 4: The encryption part of SKINNY AEAD M1

4.2 Notations and Definitions of SKINNY

In this section, the notations are defined as follows:

X_i	:	state before SC and AC operation in Round i, $0 \le i \le r-1$,
Y_i	:	state after SC and AC operation in Round i, $0 \le i \le r-1$,
Z_i	:	state after ART and SR operation in Round i,$0 \le i \le r-1$,
		The details of the i-th round $(0 \le i \le r-1)$ are as follows:

$$X_i \xrightarrow[\texttt{AC}]{\texttt{SC}} Y_i \xrightarrow[STK_i]{\texttt{ART, SR}} Z_i \xrightarrow{\texttt{MC}} X_{i+1}.$$

ΔX	:	difference of the state X,
$X_i[j \cdots k]$	:	j^{th} byte, $\cdots$, k^{th} byte of X_i, where $0 \le j, k \le 15$,
$Y_i[j \cdots k]$	:	j^{th} byte, $\cdots$, k^{th} byte of Y_i, where $0 \le j, k \le 15$,
$Z_i[j \cdots k]$	:	j^{th} byte, $\cdots$, k^{th} byte of Z_i, where $0 \le j, k \le 15$.

4.3 Properties of SKINNY

Here, we introduce several properties and a lemma on SKINNY that will be used in the related-tweakey rectangle attack.

1. The matrix used in the `MixColumns` operation is not an MDS matrix. Therefore, extra values of some cells may need to be guessed except the active cells in both input and output of the MC operation, which leads to more subtweakey bytes that need to be guessed. We use the same example as in [33] which is illustrated in Figure 5 to explain the property.
   When we backtrack the trail from ΔX_{17} to round 14 and guarantee that only $\Delta X_{14}[8]$ is active, it is necessary to check whether the differences in $\Delta X_{15}[2, 10, 14]$ lead to a single active cell in $\Delta Z_{14}[8]$. Therefore, the differences of $\Delta X_{15}[2, 10, 14]$ are needed to indicate the values as well as differences at $Y_{15}[2, 10, 14]$. To compute the value at $Y_{15}[10]$, the value of $X_{16}[4, 12]$ is required, thus an additional cell $STK_{16}[4]$ needs to be guessed.
2. Since the `AddRoundTweakey` operation follows the `SubCells` operation, and `ShiftRows` and `MixColumns` operations are all linear transformations, we can xor an equivalent subtweakey to the internal state after the MC operation, *i.e.* $STK^{eq} = MC \circ SR(STK)$ as can be seen in Figure 6.

Lemma 1. [33] For any non-zero input-output difference pair $(\delta_{in}, \delta_{out})$ for the SKINNY Sbox S, there is one solution x satisfying $S(x) \oplus S(x \oplus \delta_{in}) = \delta_{out}$ on average.

Note that `MixColumns` operation is not omitted in the last round of SKINNY, but it is well known that `MixColumns` is a linear operation which does not impact the differential cryptanalysis. To simplify the discussion, we omit the `ShiftRows` operation and `MixColumns` operation in the last round, and denote $SR \circ MC^{-1}(C)$ (*i.e.* state Z_{r-1} in r-round attack) by C in the last round, where C is the ciphertext.

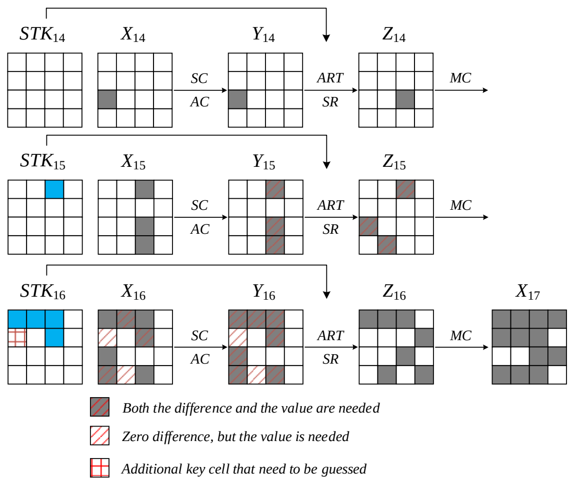

Fig. 5: Property of MC operation of SKINNY

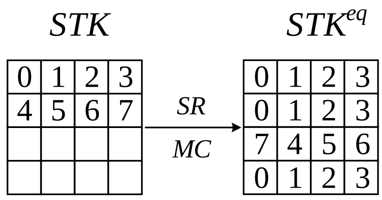

Fig. 6: The equivalent subtweakey after SR and MC operations

4.4 Related-tweakey Rectangle Attack on 27-round SKINNY-128-384

We use the same related-tweakey differential trail as in [33] which is listed in Table 3. As described in [33], the probability of the 11-round related-tweakey differential trail is 2^{-21}, and there are two trails holding with the same probability with the input and output difference unchanged leading to $\hat{p} = 2^{-20}$. A 12-round differential trail with a probability of 2^{-37} can be obtained by extending one round at the start of the 11-round trail, and connecting the 11-round and 12-round trails to construct a 23-round rectangle distinguisher. When using the boomerang switch technique in [13], four Sboxes can be saved leading to $\hat{q} = 2^{-36}$. Thus the probability of the 23-round rectangle distinguisher is $2^{-n} \cdot \hat{p}^2\hat{q}^2 = 2^{-240}$.

We prefix two rounds at the beginning of the 23-round distinguisher and append two rounds at the end to conduct a related-tweakey rectangle attact on 27-round SKINNY-128-384, which is illustrated in Figure 7.

In data collection, since $\Delta Y_1 = ART^{-1} \circ SR^{-1} \circ MC^{-1}(\alpha)$, where α is the difference in the start of the rectangle distinguisher that is known, we only need to guess the 8 bytes of STK_0 to construct sets S_1 and S_2. There are two bytes of 0 differences in the difference of plaintexts, $r_b = 14c$, $m_b = 8c$, $r_f = 13c$ and $m_f = 12c$ where $c = 8$. Totally, there are about $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^{176}$ quartets as (C_1, C_2, C_3, C_4) remaining. We give the details of the key recovery process for the m_f bit subtweakeys involved in E_f as follows (we restate that we treat Z_{26} to be the ciphertext):

1. In the second column of the ciphertext pair (C_1, C_3), the value of $Z_{26}[13]$ is known and the value of $X_{26}[13]$ can be deduced since there is no subtweaky involved. According to the inverse of the `MixColumns` operation, $\Delta Z_{25}[13] = \Delta X_{26}[13] \oplus \Delta X_{26}[1]$ and $\Delta Z_{25}[13] = 0$. With $\Delta X_{26}[1] = \Delta X_{26}[13]$, $\Delta Y_{26}[1]$ and as the value as well as the difference at $Z_{26}[1]$ can be obtained from the ciphertext pair, there is one solution for $STK_{26}[1]$ on average. Similarly, $\Delta X_{26}[5] = \Delta X_{26}[9] \oplus \Delta X_{26}[13]$ since $\Delta Z_{25}[5] = 0$, and there is one solution for $STK_{26}[5]$ on average.
2. Partially decrypt the second column of ciphertext pair (C_1, C_3) for one round to get the values as well as differences at $Z_{25}[1, 9]$. Since the input difference of the Sbox at $X_{25}[1]$ can be obtained from the rectangle distinguisher and the output difference is just $\Delta Z_{25}[1]$, there is one solution for $STK_{25}[1]$ on

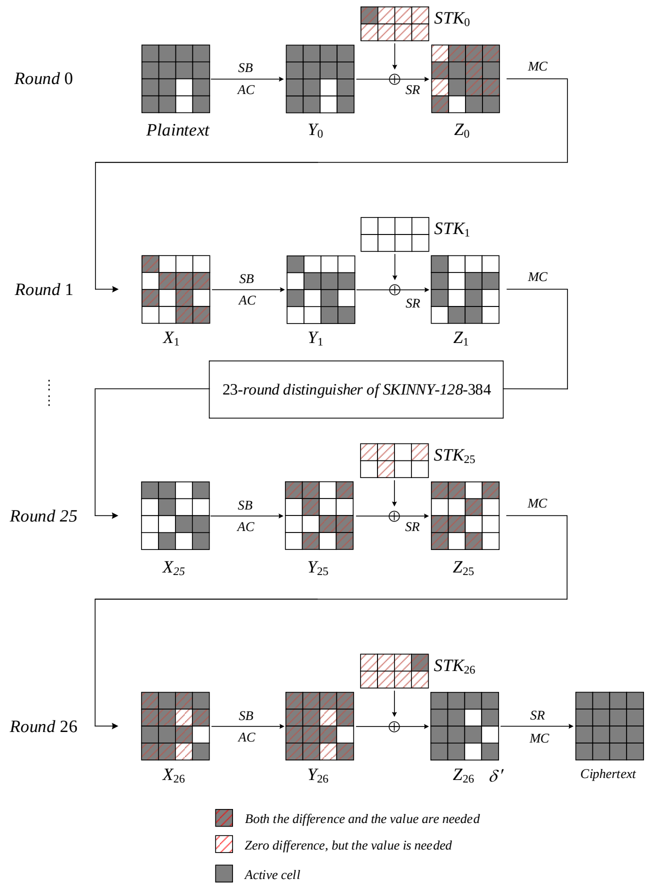

Fig. 7: Key-recovery attack against 27-round SKINNY-128-384

average. But the difference $\Delta Z_{25}[9]$ can be mapped to the known difference $\Delta X_{25}[11]$ with a probability of 2^{-8}. There are about $y^2 \cdot 2^{176} \cdot 2^{-8} = y^2 \cdot 2^{168}$ quartets remaining.

3. Partially decrypt the second column of ciphertext pair (C_2, C_4) to compute the values and differences at $Z_{25}[1, 5, 9, 13]$. The probability that $\Delta Z_{25}[5, 13] = 0$ is 2^{-16}, the probability that $Z_{25}[1]$ of (C_2, C_4) can be mapped to the known difference $\Delta X_{25}[1]$ under the obtained subtweakey is 2^{-8}, and the probability that $Z_{25}[9]$ of (C_2, C_4) can be mapped to the known difference $\Delta X_{25}[11]$ with no subtweakey involved is 2^{-8}. Totally, there are about $y^2 \cdot 2^{168} \cdot 2^{-32} = y^2 \cdot 2^{136}$ quartets remaining.
4. Similarly, for the first column of ciphertext pair (C_1, C_3), $\Delta X_{26}[4] = \Delta X_{26}[8] \oplus \Delta X_{26}[12]$ can be deduced since $\Delta Z_{25}[4] = 0$ and $\Delta X_{26}[8, 12]$ can be computed by the ciphertext pair. We can get one value of $STK_{26}[4]$ on average. Then guess the value of $STK_{26}[0]$ and compute the values as well as the differences at $Z_{25}[0, 4, 8, 12]$, deduce the value of $STK_{25}[0]$, check whether the known differences $\Delta X_{25}[10, 13]$ can be obtained by the values at $Z_{25}[8, 12]$. There are about $y^2 \cdot 2^{136} \cdot 2^8 \cdot 2^{-16} = y^2 \cdot 2^{128}$ combinations of the remaining quartets associated with the guessed keys, *i.e.* there remains 2^8 candidate values of $STK_{26}[0], STK_{25}[0]$ with about 2^{120} quartets each. Then partially decrypt (C_2, C_4) with the obtained subtweakey involved and check whether $\Delta Z_{25}[4] = 0$, $\Delta X_{25}[0] = 0x80$, $\Delta X_{25}[10] = 0x83$, $\Delta X_{25}[13] = 0x80$. There are about $y^2 \cdot 2^{128} \cdot 2^{-32} = y^2 \cdot 2^{96}$ combinations of the quartets associated with the guessed keys remaining.
5. Utilizing a similar process with step 1 to step 3 to recover $STK_{26}[3, 7]$ and $STK_{25}[3]$, there are about $y^2 \cdot 2^{96} \cdot 2^{-24} = y^2 \cdot 2^{72}$ combinations of the quartets associated with the guessed keys remaining.
6. Since $\Delta X_{26}[6] = \Delta Z_{26}[6] = 0$ and $\Delta X_{26}[2]$ is unknown, we must guess the values of $STK_{26}[2, 6]$ to compute the values as well as the differences at $Z_{25}[6, 14]$. Utilizing the filtration at $X_{25}[5, 15]$ and $Z_{25}[6, 14]$, there are about $y^2 \cdot 2^{72} \cdot 2^{16} \cdot 2^{-24} = y^2 \cdot 2^{64}$ combinations of the quartets associated with the guessed keys remaining to count for the 96-bit subtweakeys involved in E_f.
7. Output the top $2^{m_f - h}$ counters for the candidates and exhaustively search the other $k - m_b - m_f$ bit keys to check whether the guessed key is correct.

When the expected number of right quartets $s = 1$, $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q} = 2^8$, the data complexity is $D = 4M = 4 \cdot y \cdot 2^{r_b} = 2^{122}$ chosen plaintexts. And $2^{m_b} \cdot 3M = 2^{185.58}$ table look-ups are needed. In each guessed m_b-bit subtweakey, $M^2 \cdot 2^{-2(n-r_f)}$ one-round decryptions are conducted which are equal to $M^2 \cdot 2^{-2(n-r_f)} \cdot 1/27 = 2^{187.25}$ encryptions, thus the time complexity is $4M + 2^{m_b} \cdot M^2 \cdot 2^{-2(n-r_f)} \cdot \varepsilon + 2^{k-h} \approx 2^{294}$ when the size of the master key is $k = 384$, and the success probability is 84.39% when $h = 90$. The memory complexity is $5M + 2^{m_f} \approx 2^{122.32}$.

When the expected number of right quartets equals 2, the data complexity is $2^{122.5}$ chosen plaintexts, time complexity is 2^{294} and the memory complexity is $2^{122.82}$. And the success probability is 92.56% when $h = 90$.

4.5 Related-tweakey Rectangle Attack on 28-round SKINNY-128-384

Extending one more round backward from the 27-round attack in Subsection 4.4, all the bytes of difference in the plaintext are active. However, the ART operation can be conducted after the MC operation by xoring an equivalent subtweakey as it is described in Sec.4.3. Therefore, the 28-round rectangle attack only needs to guess extra 64-bit subtweakeys compared to the 27-round attack. We have $r_b = 14c$, $m_b = 16c$, $r_f = 13c$ and $m_f = 12c$ where $c = 8$. The key recovery process is identical to that in the 27-round attack on SKINNY-128-384.

If the expected number of right quartets $s = 1$, a 28-round related-tweakey rectangle attack on SKINNY-128-384 can be conducted with a data complexity of 2^{122} chosen plaintexts, a time complexity of $2^{315.25} + 2^{304} \approx 2^{315.25}$ encryptions when $h = 80$, a memory complexity of $2^{122.32}$ and a success probability is 83.15%.

4.6 Related-tweakey Rectangle Attack on 24-round SKINNY-128-384

We construct a 24-round related-tweakey rectangle attack by extending the 23-round distinguisher one round forward, *i.e.* we delete the first two rounds and the last one round from the 27-round attack for SKINNY-128-384 which is illustrated in Figure 7, and we treat the state Z_{25} to be the ciphertext. Since there is no round extended from the start of the distinguisher, no involved subtweakey needs to be guessed. There are four active bytes in the start of the distinguisher, eight active bytes in ΔZ_{25} and four bytes of subtweakey involved in the last round. Thus $r_b = 4c$, $m_b = 0$, $r_f = 8c$ and $m_f = 4c$ where $c = 8$.

We choose y structures of 2^{32} plaintexts each, and $M^2 \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^{64} \cdot 2^{-2(128-64)} = y^2 \cdot 2^{-64}$ quartets. The key recovery process can be conducted as follows:

1. Since the difference at $X_{25}[0]$ is known, the difference at $Y_{25}[0]$ and the value at $Z_{25}[0]$ can be obtained from the ciphertext pair (C_1, C_3), one solution of $STK_{25}[0]$ can be computed on average. Then by verifying the obtained keys by the ciphertext pair (C_2, C_4), about $y^2 \cdot 2^{-72}$ quartets are remaining. Since no byte of STK_{25} is involved in the computation of $Z_{25}[8, 12]$, the values of $Z_{25}[8, 12]$ in both(C_1, C_3) and (C_2, C_4) can be computed to the known differences $\Delta X_{25}[10, 13]$ with a probability of 2^{-32}, and about $y^2 \cdot 2^{-104}$ quartets are remaining.
2. Conducting a process similar to that in step 1 to the other three columns of Z_{25}, about $y^2 \cdot 2^{-160}$ quartets are remaining to count for the 32-bit subtweakey.

When the expected number of right quartets $s = 4$ and advantage $h = 20$, $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q} = 2^{89}$, a 24-round related-tweakey rectangle attack on SKINNY-128-384 can be conducted with a data complexity of 2^{123} chosen plaintexts, a time complexity of $2^{123} + 2^{114}/24 + 2^{100} \approx 2^{123}$ encryptions, and the success probability is 97.6%. Since no subtweakey needs to be guessed in the upper part, we don't need to store the chosen plaintexts, and the memory complexity is 2^{121}.

4.7 Related-tweakey Rectangle Attack on SKINNY AEAD

We have analyzed the tweakable block cipher SKINNY-128-384 by a related-tweakey rectangle attack in Section 4, where there is no constraint to the value and difference of the tweak. However, the SKINNY AEAD member M1 adopts SKINNY-128-384 as its internal primitive, but M1 initializes the tweakey bytes in a more complex process which leads to more constraints appearing when the adversary conducts a related-tweakey attack on it. Here, we summarize the constraints as follows.

The first and most important, as depicted by the designers, the SKINNY AEAD member M1 employs a 384-bit tweakey input but a 128-bit master key. And for all versions of SKINNY AEAD, they claim full 128-bit security for key recovery, confidentiality and integrity in the *nonce-respecting* mode. Moreover, when using the recommended parameters given in [12], the total size of the message does not exceed 2^{64}

blocks and the maximum number of messages that can be handled under the same key is 2^{128} in SKINNY AEAD member M1. Therefore, only the attack on SKINNY-128-384 with time complexity $\leq 2^{128}$ can be applied to SKINNY AEAD M1.

Secondly, a restriction to the difference of $TK1$ appears due to the specific initialization of the first 128-bit tweakey. The first 128-bit tweakey is composed of a 64-bit number that is updated by a linear transformation rev_{64} and a LFSR, 7 bytes of zeros and a single byte for the domain separation that is a constant. The other 256-bit tweakey is the concatenation of nonce N and key K. Thus, in a related-tweakey attack, the last 64 bits of $TK1$ can not contain any difference. Fortunately, both ΔK used in the upper trail and ∇K used in the lower trail don't contain any difference in the 64 bits in the 23-round related-tweakey rectangle distinguisher.

Thirdly, for a AEAD scheme, no decryptions are proceeded when a *tag* is invalid and only a null character is returned. This implies that the adversary can only make queries to the encryption oracle, which prevents any chosen ciphertext attack. But it is not problematic to the rectangle attack where only chosen plaintext is needed.

Finally, the nonce input of the AEAD mode may be a problem in data collection. SKINNY AEAD M1 is a nonce-respecting scheme, the adversary can only query a nonce once under the same key, but a nonce can be queried several times in different keys *i.e.* in the related-key setting. In the case of SKINNY AEAD M1, the nonce N is used in tweakey input together with the master key and some other string, which implies that the tweakey input is controlled for the adversary. Thus the adversary can make queries in advance and conduct a related-tweakey rectangle attack on its internal primitive SKINNY-128-384.

Explanation for the data collection. As described in [12], the 384-bit tweakey of the SKINNY AEAD M1 consist of a 128-bit $\text{rev}_{64}(LFSR)||0^{56}||d_0$, a 128-bit nonce N and a 128-bit secret key K, where $\text{rev}_{64}(LFSR)$ play the same role as a block counter that traverses from 0 to $2^{64}-1$. The attacker can make queries to the encryption oracle as follows:

1. The attacker can choose an arbitrary nonce N_1 and query a plaintext chain with a size of 2^{64} blocks under the secret key K_1, the block counter denoted by l_1 will traverse from 0 to $2^{64}-1$. Note that there is no constraint to the plaintexts, the attacker can set arbitrary value to each plaintext block.
2. The attacker choose another nonce N_2 and query a plaintext chain with a size of 2^{64} blocks under the secret key K_2, the block counter denoted by l_2 will also traverse from 0 to $2^{64}-1$. But the nonce N_2 and secret key K_2 must satisfy that $N_1 \oplus N_2$ and $K_1 \oplus K_2$ are equal to the differences in ΔK. For each value of block counter l_1 and the corresponding plaintext block P_1, there exists a value of block counter l_2 that $l_1 \oplus l_2$ satisfies the differences in ΔK, so the value of the plaintext block P_2 corresponding to block counter l_2 must satisfies that $P_1 \oplus P_2$ is equal to the difference in the start of the attack. Totally, the attacker can obtain 2^{64} plaintext pairs in the two steps to proceed the attack.
3. Utilizing a way similar to steps 1 and 2, the attacker can make queries under nonces N_3, N_4 and secrect keys K_3, K_4. The values of plaintext blocks under N_3 and K_3 can be arbitrary, but the differences $N_1 \oplus N_3$ and $K_1 \oplus K_3$ must be equal to the differences in ∇K. The values of plaintext blocks under N_4 and K_4 will be set according to the value of block counter following the same way as in step 2, and the differences $N_3 \oplus N_4$ and $K_3 \oplus K_4$ need to be equal to the differences in ΔK.

Note that each query made by the attacker is used to construct plaintext pairs without waste. Therefore, the attack on SKINNY AEAD M1 does not need a higher data complexity.

The related-tweakey rectangle attack on 24-round SKINNY-128-384 has a data complexity of 2^{123} chosen plaintexts and a time complexity 2^{123}, which is applicable to the SKINNY AEAD M1.

5 Application to GIFT-64

The GIFT block cipher, proposed by Banik *et al.* at CHES 2017 [16], is an improved version of PRESENT [14]. In the NIST Lightweight Cryptography Standardization process [37], the candidates SUNDAE-GIFT [5], TGIF [26] and GIFT-COFB [6] are based on the GIFT block cipher. GIFT has two versions, GIFT-64 and GIFT-128, according to the block size, while both versions support the 128-bit key size. At IWSEC 2018, Sasaki [38] introduced a MitM attack on 15-round GIFT-64 with a time complexity 2^{112}. At CT-RSA 2019, Zhu *et al.* [48] analyzed the 19-round GIFT-64 with a 12-round differential characteristic under the single-key mode, and give a 22-round differential attack for GIFT-128. At ACISP 2019, Liu and Sasaki [34] explored the BCT effect on GIFT-64 and GIFT-128 by a SAT-based method, and gave a 23-round key-recovery attack on GIFT-64. Concurrently, Chen *et al.* [22] also gave a 23-round key-recovery attack based on the generalized model of related-key rectangle attack by Liu *et al.* [33]. In this paper, we use the same distinguisher given by Chen *et al.* [22] to launch a new 24-round key-recovery attack based on our new generalized model of related-key rectangle attack.

Table 3: 11-round trail for SKINNY-128-384 in [33]. The 23-round distinguisher uses the 11-round trail for the upper part and in the lower part the 12-round trail which is extended backward for one round from the 11-round one where $0x7b$ is used instead. In each round, the rows represent input/output differences of the Sbox layer and the round tweakey difference.

ΔK	0,`aa`,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,`e6`,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,`cf`,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
R1	0,20,0,0, 10(7b),0,0,0, 0,0,0,10, 0,0,10,0 0,83,0,0, 40,0,0,0, 0,0,0,40, 0,0,40,0 0,83,0,0, 0,0,0,0
R2	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,40,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,04,0,0 0,0,0,0, 0,0,0,0
R3	04,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 01,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 01,0,0,0, 0,0,0,0
R4	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0
R5	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0
R6	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0
R7	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0
R8	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0
R9	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,01,0
R10	0,0,0,0, 0,0,0,0, 0,0,0,01, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,20, 0,0,0,0 0,0,0,0, 0,0,0,0
R11	0,20,0,0, 0,0,0,0, 0,20,0,0, 0,20,0,0 0,80,0,0, 0,0,0,0, 0,80,0,0, 0,80,0,0 0,0,0,0, 0,83,0,0

5.1 Specification of GIFT

GIFT [16], proposed by Banik *et al.* in 2017, is an SPN cipher. There are two versions for GIFT according to the block size *i.e.* GIFT-64 and GIFT-128. Both versions have a key length of 128-bit and the number of rounds is 28 and 40 for GIFT-64 and GIFT-128, respectively.

The round function is composed of three subfunctions named `SubCells`, `PermBits` and `AddRoundKey`, which are defined as follows:

1. `SubCells` : Apply the 4-bit Sbox to every nibble of the internal state, where the Sbox is defined as Table 4.
2. `PermBits` : Update the internal state by a linear bit permutation as $b_{P(i)} \leftarrow b_i,\ \forall i \in \{0, 1, ...n-1\}$, where the $P(i)$s are expressed as

$$P_{64}(i) = 4\lfloor \frac{i}{16} \rfloor + 16((3\lfloor \frac{i \bmod 16}{4} \rfloor + (i \bmod 4) \bmod 4) + (i \bmod 4),$$
$$P_{128}(i) = 4\lfloor \frac{i}{16} \rfloor + 32((3\lfloor \frac{i \bmod 16}{4} \rfloor + (i \bmod 4) \bmod 4) + (i \bmod 4),$$

   for GIFT-64 and GIFT-128, respectively.
3. `AddRoundKey` : An $n/2$-bit round key RK is extracted from the key state and is further partitioned into 2 s-bit words $RK = U||V = u_{s-1}...u_0||v_{s-1}v_0$, where $s = n/4$.
   For GIFT-64, the round key is XORed to the state as

$$b_{4i+1} \leftarrow b_{4i+1} \oplus u_i,\ b_{4i} \leftarrow b_{4i} \oplus v_i,\ \forall i \in \{0, ..., 15\}.$$

   For GIFT-128, the round key is XORed to the state as

$$b_{4i+2} \leftarrow b_{4i+2} \oplus u_i,\ b_{4i+1} \leftarrow b_{4i+1} \oplus v_i,\ \forall i \in \{0, ..., 31\}.$$

Table 4: The Sbox of GIFT

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$GS(x)$	1	a	4	c	6	f	3	9	2	d	b	7	5	0	8	e

Table 5: Differential paths of 19-round GIFT-64 [22], where "*" denotes the probability of the rounds that are evaluated for the ladder switch.

Round	Differentce	Δk_i	Δk_{i+1}	Probability
$1r$	0000 00a0 0000 6000	(4, 0, 0, 0)	(0, 1, 0, 0)	2^{-4}
$2r$	0000 0000 0000 0000	(0, 0, 0, 0)	(0, 0, 0, 0)	1
$3r$	0000 0000 0000 0000	(0, 0, 0, 2)	(0, 0, 0, 0)	1
$4r$	0000 0000 0000 0010	(0, 0, 0, 0)	(0, 0, 0, 0)	2^{-3}
$5r$	0000 0008 0000 0000	(0, 0, 0, 4)	(0, 0, 4, 0)	2^{-2}
$6r$	0000 0000 0000 0000	(0, 0, 0, 0)	(0, 0, 0, 0)	1
$7r$	0000 0000 0000 0000	(0, 0, 2, 0)	(0, 0, 0, 0)	1
$8r$	0000 0000 0010 0000	(0, 0, 0, 0)	(0, 0, 0, 0)	2^{-3}
$9r$	0000 0080 0000 0000	(0, 0, 4, 0)	(0, 0, 1, 0)	2^{-2}
$10r$	0100 0000 0102 0200	(0, 0, 0, 0)	(0, 0, 0, 0)	1^*
$11r$	00a2 0000 8020 0044	(0, 2, 0, 0)	(0, 0, 0, 0)	1^*
$10r$	0000 0e03 0000 0073	(0, 0, 0, 1)	(0, 0, 0, 0)	1^*
$11r$	0000 050c 0a00 0000	(0, 2, 0, 0)	(0, 0, 0, 0)	1^*
$12r$	0a00 0000 0000 0000	(0, 8, 0, 0)	(0, 0, 0, 0)	2^{-2}
$13r$	0000 0000 0000 0000	(0, 0, 0, 0)	(0, 0, 0, 0)	1
$14r$	0000 0000 0000 0000	(0, 0, 1, 0)	(0, 0, 0, 0)	1
$15r$	0000 0000 0001 0000	(2, 0, 0, 0)	(0, 0, 0, 0)	2^{-3}
$16r$	0090 0000 0000 0000	(8, 0, 0, 0)	(0, 0, 0, 0)	2^{-3}
$17r$	0000 0000 0000 0000	(0, 0, 0, 0)	(0, 0, 0, 0)	1
$18r$	0000 0000 0000 0000	(0, 1, 0, 0)	(0, 0, 0, 0)	1
$19r$	0000 0001 0000 0000	(0, 0, 2, 0)	(0, 0, 0, 0)	2^{-3}

For both versions, a single bit "1" and a 6-bit constant C are XORed into the internal state at positions $n-1$, 23, 19, 15, 11, 7 and 3 respectively.

The key schedule for GIFT is very simple. The 128-bit master key is initialized as $K = k_7||k_6||...||k_0$, where $|k_i| = 32$. For GIFT-64, the round key RK is $RK = U||V = k_1||k_0$. For GIFT-128, the round key RK is $RK = U||V = k_5||k_4||k_1||k_0$. And for both versions, the key state is updated as follows,

$$k_7||k_6||...||k_0 \leftarrow k_1 >>> 2||k_0 >>> 12||...||k_3||k_2,$$

where $\ggg i$ is an i-bit right rotation within a 16-bit word. For more details of GIFT, we refer to [16].

5.2 Notations and Definitions of GIFT

In this section, the notations are defined as follows:

ΔP	:	the difference in plaintext
ΔX_S^i	:	the difference after `SubCells` operation in Round i, $0 \leq i \leq r-1$
ΔX_P^i	:	the difference after `PermBits` operation in Round i, $0 \leq i \leq r-1$
ΔX_K^i	:	the difference after `AddRoundKey` operation in Round i, $0 \leq i \leq r-1$
"?"	:	represent an unknown difference
$\Delta X_S^i[j \cdots k]$	:	j^{th} byte, $\cdots$, k^{th} byte of ΔX_S^i

5.3 24-Round Attack on GIFT-64

We use the same 19-round related-key rectangle distinguisher of GIFT-64 listed in Table 5 by Chen *et al.* [22] to give the first 24-round key-recovery attack on GIFT-64. We append three rounds backward and two rounds forward to the distinguisher to conduct a 24-round related-key rectangle attack. The propagation of the differentials is illustrated in Table 6.

Note that, there is no whitening key xored to the plaintext, we collect data in ΔX_P^0, which is similar to the previous works [22, 34, 48]. There are 46 unknown bits in ΔX_P^0 denoted by "?" which affect 12 Sboxes in Round 1 and four Sboxes in Round 2, thus $r_b = 46$ and the number of key bits needed to be guessed in the upper part is $m_b = 24$. Similarly, we have $r_f = 20$ and $m_f = 12$ for the lower part. Totally, there are $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^4$ quartets remaining for each guessed m_b-bit key. The key recovery part is very similar to that in Section 4.4, we give the brief description of it as follows (For generalization, we treat the "1" in ΔX_S^{22} and ΔX_K^{22} as "?"):

Table 6: Related-key Rectangle Attack of 24-round GIFT-64

ΔP	????	????	????	????	????	????	????	????	????	????	????	????	????	????	????	????
ΔX_S^0	0???	?0??	1?0?	?1?0	0???	?0??	??0?	???0	0???	?0??	??0?	???0	0???	?0??	??0?	???0
ΔX_P^0	????	????	????	????	11??	????	????	????	????	????	????	????	0000	0000	0000	0000
ΔX_K^0	????	????	????	????	11??	????	????	????	????	????	????	????	0000	0000	0000	0000
ΔX_S^1	000?	?000	0?00	00?0	0100	00?0	000?	1000	0?0?	?0?0	0?0?	?0?0	0000	0000	0000	0000
ΔX_P^1	0000	11??	????	0000	0000	0000	0000	0000	????	0000	????	0000	0000	0000	0000	0000
ΔX_K^1	0000	11??	????	0000	0000	0000	0000	0000	????	0000	????	0000	0000	0000	0000	0000
ΔX_S^2	0000	0100	0010	0000	0000	0000	0000	0000	0010	0000	1000	0000	0000	0000	0000	0000
ΔX_P^2	0000	0000	0000	0000	0000	0000	1010	0000	0000	0000	0000	0000	0110	0000	0000	0000
ΔX_K^2	0000	0000	0000	0000	0000	0000	1010	0000	0000	0000	0000	0000	0110	0000	0000	0000
$\vdots$	Distinguisher of 19-round GIFT-64															
ΔX_K^{21}	0000	1000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
ΔX_S^{22}	0000	??11	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000	0000
ΔX_P^{22}	0010	0000	0000	0000	0001	0000	0000	0000	?000	0000	0000	0000	0?00	0000	0000	0000
ΔX_K^{22}	0010	0000	0000	0001	0001	0000	0000	0000	?000	0000	0000	0000	0?00	0000	0000	0000
ΔX_S^{23}	????	0000	0000	????	????	0000	0000	0000	????	0000	0000	0000	????	0000	0000	0000
ΔX_P^{23}	??00	0?00	0?00	0?00	0??0	00?0	00?0	00?0	00??	000?	000?	000?	?00?	?000	?000	?000
ΔX_K^{23}	??00	0?00	0?00	0?00	0??0	00?0	00?0	00?0	00??	000?	000?	000?	?00?	?000	?000	?000

1. The difference of $\Delta X_S^{23}[60,61,62,63]$ can be computed by the cipertext pair (C_1, C_3) and the difference of $\Delta X_K^{22}[60,62,63] = 0$ is known. Thus we guess the 2^2 possible values of involved key bits in this Sbox and partially decrypt cipertext pairs (C_1, C_3) and (C_2, C_4) and check whether the difference of $\Delta X_K^{22}[60,62,63]$ is 0 or not. If yes, we keep the guessed key and the quartet, otherwise discard it. There are about $y^2 \cdot 2^4 \cdot 2^2 \cdot 2^{-6} = y^2$ combinations of the remaining quartets associated with the guessed 2-bit keys, *i.e.* there remains about $y^2 \cdot 2^{-2}$ quartets with 2^2 candidate values of the 2-bit involved keys each.
2. Conducting a similar process to all the active Sboxes in Round 23, there are about $y^2 \cdot 2^{-4\times 4} = y^2 \cdot 2^{-16}$ combinations of the remaining quartets associated with the guessed keys,.
3. Partially decrypt all the remaining quartets with the obtained key bits in steps 1 and 2. The difference of $\Delta X_K^{21}[56,57,58,59]$ can be obtained from the end of the distinguisher, thus guess the 2^2 possible values of the key bits involved in this Sbox. For each guess, only 2^{-8} of the quartets remain *i.e.* $y^2 \cdot 2^{-16} \cdot 2^2 \cdot 2^{-8} = y^2 \cdot 2^{-22}$. Utilize the remaining quartets to count the $m_f = 12$ key bits.

When the expected number of right quartets $s = 4$, we need to choose $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q} = 2^{12}$ structures of 2^{46} plaintexts each, and the data complexity is $4M = 2^{60}$ chosen plaintexts. $2^{m_b} \cdot 3M = 2^{91.58}$ table lookups are needed to prepare quartets. For each guessed m_b-bit key, $y^2 \cdot 2^4 \cdot 2^2$ one-round encryptions are conducted which are equal to $y^2 \cdot 2^4 \cdot 2^2/24 \approx 2^{25.42}$ encryptions. If we choose the advantage $h = 40$, $2^{m_b} \cdot 2^{25.42} + 2^{128-48} \approx 2^{88}$ encryptions are needed to recover all the key bits, and the success probability is 97.41%. Thus the time complexity is bounded by the $2^{m_b} \cdot 3M = 2^{91.58}$ table lookups. The memory complexity is $5M + 2^{m_f} \approx 2^{60.32}$.

6 Conclusion

In this paper, we give a new model of the generalized related-key rectangle attack. Based on the new model, we give improved attacks on both, round-reduced SKINNY-128-384 and GIFT-64. We also give the first third party cryptanalysis on SKINNY AEAD M1, which is a candidate of the NIST Lightweight Cryptography project.

- As one open problem, our model may be also applicable to more SKINNY-based or GIFT-based authenticated encryption candidates of the ongoing NIST Lightweight Cryptography project, such as SUNDAE-GIFT [5], TGIF [26], GIFT-COFB [6], Remus [27] and Romulus [28].
- Another open problem is to apply our model to evaluate the security of other block ciphers with linear key schedules, such as Saturnin [19], Simon [18].
- For LOTUS-AEAD and LOCUS-AEAD [50], a Round 2 candidate of the NIST LWC, the designers state that "*the keys are computed by a predictable way in the mode and used with a fixed tweak. This implies that related-key security of TweGIFT-64 matters in the related-key security of the entire construction.*". Hence, it is relevant to study GIFT-64 against related-key attack. The attacks in our paper do not cover the concrete impact on LOTUS-AEAD and LOCUS-AEAD. We would like to leave it as an open problem.

Acknowledgments

This work is supported by the National Key Research and Development Program of China (No. 2017YFA0303903), the National Natural Science Foundation of China (No. 61902207), the National Cryptography Development Fund (No. MMJJ20180101, MMJJ20170121).

References

1. Ralph Ankele, Subhadeep Banik, Avik Chakraborti, Eik List, Florian Mendel, Siang Meng Sim, and Gaoli Wang. Related-key impossible-differential attack on reduced-round skinny. In *Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings*, pages 208–228, 2017.
2. Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M. Youssef. MILP modeling for (large) s-boxes to optimize probability of differential characteristics. *IACR Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.
3. Roberto Avanzi. The QARMA block cipher family. almost MDS matrices over rings with zero divisors, nearly symmetric even-mansour constructions with non-involutory central rounds, and search heuristics for low-latency s-boxes. *IACR Trans. Symmetric Cryptol.*, 2017(1):4–44, 2017.
4. Subhadeep Banik, Andrey Bogdanov, Takanori Isobe, Kyoji Shibutani, Harunaga Hiwatari, Toru Akishita, and Francesco Regazzoni. Midori: A block cipher for low energy. In *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, pages 411–436, 2015.
5. Subhadeep Banik, Andrey Bogdanov, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, Elmar Tischhauser, and Yosuke Todo. SUNDAE-GIFT. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
6. Subhadeep Banik, Avik Chakraborti, Tetsu Iwata, Kazuhiko Minematsu, Mridul Nandi, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT-COFB. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
7. Eli Biham, Orr Dunkelman, and Nathan Keller. The rectangle attack - rectangling the serpent. In *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, pages 340–357, 2001.
8. Eli Biham, Orr Dunkelman, and Nathan Keller. New results on boomerang and rectangle attacks. In *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, pages 1–16, 2002.
9. Eli Biham, Orr Dunkelman, and Nathan Keller. Related-key boomerang and rectangle attacks. In *Advances in Cryptology - EUROCRYPT 2005, 24th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings*, pages 507–525, 2005.
10. Eli Biham, Orr Dunkelman, and Nathan Keller. A related-key rectangle attack on the full KASUMI. In *Advances in Cryptology - ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings*, pages 443–461, 2005.
11. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 123–153, 2016.
12. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. SKINNY-AEAD and SKINNY-Hash v1.0. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
13. Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In *Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings*, pages 1–18, 2009.
14. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, pages 450–466, 2007.
15. Christof Beierle, Gregor Leander, Amir Moradi, and Shahram Rasoolzadeh. CRAFT: lightweight tweakable block cipher with efficient protection against DFA attacks. *IACR Trans. Symmetric Cryptol.*, 2019(1):5–45, 2019.
16. Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, pages 321–345, 2017.
17. Eli Biham and Adi Shamir. Differential Cryptanalysis of DES-like Cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO 90*, volume 537 of *Lecture Notes in Computer Science*, pages 2–21. Springer, 1991.
18. Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK families of lightweight block ciphers. *IACR Cryptology ePrint Archive*, 2013:404, 2013.
19. Anne Canteaut, Sébastien Duval, Gaëtan Leurent, Marıa Naya-Plasencia, Léo Perrin, Thomas Pornin, and André Schrottenloher. Saturnin v1: a suite of lightweight symmetric algorithms for post-quantum security. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
20. Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II*, pages 683–714, 2018.
21. The CAESAR Committee. CAESAR: Competition for authenticated encryption: Security, applicability, and robustness, 2014.
22. Lele Chen, Gaoli Wang, and Guoyan Zhang. MILP-based related-key rectangle attack and its application to GIFT, Khudra, MIBS. *Accepted by The Computer Journal.*
23. Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. In *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, pages 393–410, 2010.
24. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard.* Information Security and Cryptography. Springer, 2002.
25. Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED block cipher. In *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, pages 326–341, 2011.

26. Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Ling Sun. Thank Goodness It's Friday (TGIF). *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
27. Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Remus v1. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
28. Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Romulus v1. *Submission to Round 1 of the NIST Lightweight Cryptography Standardization process (2019).*
29. Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In *Advances in Cryptology - ASIACRYPT 2014 - 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, R.O.C., December 7-11, 2014, Proceedings, Part II*, pages 274–288, 2014.
30. Jérémy Jean, Ivica Nikolić, Thomas Peyrin, and Yannick Seurin. Submission to caesar : Deoxys v1.41, October 2016.
31. John Kelsey, Tadayoshi Kohno, and Bruce Schneier. Amplified boomerang attacks against reduced-round MARS and serpent. In *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, pages 75–93, 2000.
32. Ted Krovetz and Phillip Rogaway. The software performance of authenticated-encryption modes. In *Fast Software Encryption - 18th International Workshop, FSE 2011, Lyngby, Denmark, February 13-16, 2011, Revised Selected Papers*, pages 306–327, 2011.
33. Guozhen Liu, Mohona Ghosh, and Ling Song. Security analysis of SKINNY under related-tweakey settings (long paper). *IACR Trans. Symmetric Cryptol.*, 2017(3):37–72, 2017.
34. Yunwen Liu and Yu Sasaki. Related-key boomerang attacks on GIFT with automated trail search including bct effect. Cryptology ePrint Archive, Report 2019/669, 2019.
35. Amir Moradi, Axel Poschmann, San Ling, Christof Paar, and Huaxiong Wang. Pushing the limits: A very compact and a threshold implementation of AES. In *Advances in Cryptology - EUROCRYPT 2011 - 30th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Tallinn, Estonia, May 15-19, 2011. Proceedings*, pages 69–88, 2011.
36. Sean Murphy. The return of the cryptographic boomerang. *IEEE Trans. Information Theory*, 57(4):2517–2521, 2011.
37. National Institute of Standards and Technology (NIST). Lightweight cryptography (LWC) standardization process, 2019. https://csrc.nist.gov/Projects/Lightweight-Cryptography/Round-1-Candidates.
38. Yu Sasaki. Integer linear programming for three-subset meet-in-the-middle attacks: Application to GIFT. In *Advances in Information and Computer Security - 13th International Workshop on Security, IWSEC 2018, Sendai, Japan, September 3-5, 2018, Proceedings*, pages 227–243, 2018.
39. Ali Aydin Selçuk. On probability of success in linear and differential cryptanalysis. *J. Cryptology*, 21(1):131–147, 2008.
40. Siwei Sun, David Gerault, Pascal Lafourcade, Qianqian Yang, Yosuke Todo, Kexin Qiao, and Lei Hu. Analysis of AES, SKINNY, and others with constraint programming. *IACR Trans. Symmetric Cryptol.*, 2017(1):281–306, 2017.
41. Sadegh Sadeghi, Tahereh Mohammadi, and Nasour Bagheri. Cryptanalysis of reduced round SKINNY block cipher. *IACR Trans. Symmetric Cryptol.*, 2018(3):124–162, 2018.
42. Ling Song, Xianrui Qin, and Lei Hu. Boomerang connectivity table revisited. application to SKINNY and AES. *IACR Trans. Symmetric Cryptol.*, 2019(1):118–141, 2019.
43. Danping Shi, Siwei Sun, Patrick Derbez, Yosuke Todo, Bing Sun, and Lei Hu. Programming the demirci-selçuk meet-in-the-middle attack with constraints. In *Advances in Cryptology - ASIACRYPT 2018 - 24th International Conference on the Theory and Application of Cryptology and Information Security, Brisbane, QLD, Australia, December 2-6, 2018, Proceedings, Part II*, pages 3–34, 2018.
44. Yu Sasaki and Yosuke Todo. New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, pages 185–215, 2017.
45. Mohamed Tolba, Ahmed Abdelkhalek, and Amr M. Youssef. Impossible differential cryptanalysis of reduced-round SKINNY. In *Progress in Cryptology - AFRICACRYPT 2017 - 9th International Conference on Cryptology in Africa, Dakar, Senegal, May 24-26, 2017, Proceedings*, pages 117–134, 2017.
46. David A. Wagner. The boomerang attack. In *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, pages 156–170, 1999.
47. Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. application to AES variants and Deoxys. *IACR Trans. Symmetric Cryptol.*, 2019(1):142–169, 2019.
48. Baoyu Zhu, Xiaoyang Dong, and Hongbo Yu. MILP-based differential attack on round-reduced GIFT. In *Topics in Cryptology - CT-RSA 2019 - The Cryptographers' Track at the RSA Conference 2019, San Francisco, CA, USA, March 4-8, 2019, Proceedings*, pages 372–390, 2019.
49. Huaifeng Chen, Rui Zong, and Xiaoyang Dong. Improved Differential Attacks on GIFT-64. To appear in *ICICS 2019.*
50. Avik Chakraborti, Nilanjan Datta, Ashwin Jha, Cuauhtemoc Mancillas-López, Mridul Nandi and Yu Sasaki. INT-RUP Secure Lightweight Parallel AE Modes. *IACR Trans. Symmetric Cryptol.* 2019(4):81–118, 2019.