IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 0, No. 0, pp. 0–0. DOI:10.13154/tosc.v0.i0.0-0

Automated Search Oriented to Key Recovery on Ciphers with Linear Key Schedule

Applications to Boomerangs in SKINNY and ForkSkinny

Lingyue Qin[1], Xiaoyang Dong[1] ✉, Xiaoyun Wang[1,3] ✉, Keting Jia[2] and Yunwen Liu[4,5]

[1] Institute for Advanced Study, BNRist, Tsinghua University, Beijing, China. {qinly,xiaoyangdong,xiaoyunwang}@tsinghua.edu.cn
[2] Institute for Network Sciences and Cyberspace, BNRist, Tsinghua University, Beijing, China. ktjia@tsinghua.edu.cn
[3] Key Laboratory of Cryptologic Technology and Information Security, Ministry of Education, Shandong University, Qingdao, China.
[4] College of Liberal arts and Science, National University of Defense Technology, Changsha, China.
[5] Hunan Engineering Research Center of Commercial Cryptography Theory and Technology Innovation, Changsha, China. univerlyw@hotmail.com

Abstract. Automatic modelling to search distinguishers with high probability covering as many rounds as possible, such as MILP, SAT/SMT, CP models, has become a very popular cryptanalysis topic today. In those models, the optimizing objective is usually the probability or the number of rounds of the distinguishers. If we want to recover the secret key for a round-reduced block cipher, there are usually two phases, i.e., finding an efficient distinguisher and performing key-recovery attack by extending several rounds before and after the distinguisher. The totally attacked number of rounds is not only related to the chosen distinguisher, but also to the extended rounds before and after the distinguisher. In this paper, we try to combine the two phases in a uniform automatic model.
Concretely, we apply this idea to automate the related-key rectangle attacks on SKINNY and ForkSkinny. We propose some new distinguishers with advantage to perform key-recovery attacks. Our key-recovery attacks on a few versions of round-reduced SKINNY and ForkSkinny cover 1 to 2 more rounds than the best previous attacks.

Keywords: Key recovery · SKINNY · ForkSkinny · Differential attack · Rectangle attack

1 Introduction

Differential cryptanalysis [BS91], proposed by Biham and Shamir, is one of the most successful cryptanalysis techniques. To launch a differential key-recovery attack on block ciphers, the first step is to find a differential with probability larger than a random case. Based on the found differential, the second step is to append several extra rounds before and after the differential distinguisher to recover the keys.

In the perspective of a *distinguishing attack*, good differential distinguishers are those with a relatively high probability or those covering a larger number of rounds. Based on the target cipher, one could utilise a dedicated search strategy to obtain differential-like distinguishers, such strategies include Matsui's branch and bound method [Mat93], MILP-based automatic search [MWGP11, SHW+14] and SAT/SMT-based tools [KLT15].

Whereas in a *key-recovery attack*, the goal is to attack as many rounds as possible with a relatively low (data, time and memory) complexity. Therefore, a good distinguisher for recovering the key is expected to balance the factors in determining the data/time/memory complexities and the number of attacked rounds.

As a consequence, the strategy of searching good distinguishers for a key-recovery attack may differ from that of a distinguisher attack. In other words, an optimal differential in distinguishing attack is not necessarily the optimal one for recovering the key. For instance, in order to extend more rounds before and after a differential distinguisher, one aims at minimizing the number of active bits from the input/output differences of the differential distinguisher, such that a data collection to filter the wrong pairs can be efficiently performed.

In order to search for differential distinguishers targeting at an improvement on the number of covered rounds by a key-recovery attack, one has to take into account multiple factors and their interactive influences, including the probability and the length of the differential distinguisher, the number of inactive bits in the differences after the forward and backward extension, and the number of guessed key bits for a partial decryption. It is interesting to study an automatic search model to analyze the trade-offs amongst various factors such that the constructed distinguisher is optimized for an efficient key-recovery attack. Till now, there are a few works covering related domains. Derbez et al. [DF16], Shi et al. [SSD+18] and Chen et al. [CSSH19] introduced automatic tools on Demirci-Selçuk Meet-in-the-Middle attack that take the distinguisher and the key-recovery phases as a uniform searching model. Zong et al. [ZDC+21] studied the key-recovery-attack friendly differential and linear distinguishers on `GIFT-128` [BPP+17].

With the emergence of a large number of highly-constrained devices in burgeoning fields such as the Internet of things (IoTs), sensor networks, etc., lightweight cryptography becomes an active research domain in symmetric-key research groups. For applications in lightweight block ciphers where the adversary may have more power under some advanced attacking models, more interactive factors should be taken into consideration in the search of distinguishers. For instance, an attacker may have access to or control over the keys or tweaks. A related-key attack scenario where the encryption oracle is queried under a pair of keys with certain known relation is practical for some lightweight ciphers.

At CRYPTO 2016, Beierle *et al.* proposed a new lightweight block cipher family - `SKINNY` [BJK+16], which has comparable hardware/software performances with `SIMON` [BSS+13] and also has much stronger security guarantees. Also in 2016, NIST started the Lightweight Cryptography (LWC) standardization project [oSN20] to solicit lightweight cryptographic algorithms that are suitable for constrained devices. Among the Round 2 candidates of the LWC project, three algorithms are based on `SKINNY`, that is, `SKINNY-AEAD` and `SKINNY-Hash` [BJK+20], `ForkAE` [ALP+19a] and `Romulus` [IKMP19]. And `Romulus` is one of the finalists in the LWC project. So the security analysis of `SKINNY` is of great importance, which also affects the security evaluation of these candidates.

There are several cryptanalysis results on `SKINNY` under single-tweakey and related-tweakey settings using different techniques, such as impossible differential attack [TAY17, LGS17, SMB18, YQC17, ABC+17, DHLP20], rectangle attack [LGS17, ZDM+20], zero-correlation attack [SMB18, ADG+19], Demirci-Selçuk Meet-in-the-Middle attack [SSD+18], etc. Among these cryptanalysis results, the rectangle attacks in related-tweakey setting can cover more rounds for most versions. Besides, there are many tools [BN10, LS19, CHP+17, LGS17, HBS20] to search related-key rectangle distinguishers. Based on these works, we build a new MILP model combining the key-recovery process and the distinguisher search process together.

1.1 Our contributions

In this paper, we focus on an automatic model to search for distinguishers that directly improve the cryptanalytic results. Firstly, we analyze the detailed factors that restrict the generic differential key-recovery attack, and specifically a generalized rectangle attack model given by Zhao *et al.* [ZDM+20, ZDJ19]. With data complexity and time complexity lower than that of the exhaustive search, we try to maximize the attacked rounds for a block cipher. The constraints that we take into consideration include the probability of the distinguisher, the number of differential inactive bits of the input-output of the attacked cipher, and the number of key bits needed to be guessed in the extended rounds. Therefore, we propose a new automatic search MILP model of related-key rectangle attacks on `SKINNY`, where the probability of a distinguisher and the dominating factors of the key-recovery phase are systematically processed by the constraints. So the key-recovery attacks may be improved in the number of covered rounds and/or the attack complexity. The uniform MILP model is built mostly based on the works [BJK+16, HBS20] that takes all the above constraints in searching a good distinguisher. We are able to find new good properties in the distinguishers, which can be used to perform key-recovery attacks on `SKINNY` and `ForkSkinny` covering more rounds than previous results. The cryptanalytic results are summarized in Table 1.

2 Tradeoff Between Distinguisher and Key-recovery Attack

2.1 The tradeoff in differential cryptanalysis

Denote the N-round cipher E as $E = E_f \circ E' \circ E_b$. E' is the N_d-round differential distinguisher $(\alpha \mapsto \beta)$ with probability of p. The N_b-round E_b and N_f-round E_f are the rounds added before and after the distinguisher, respectively. Denote the block size as n, the number of active bits of the input difference of E_b as r_b bits, and the number of key bits needed to be guessed in E_b as m_b bits. Similarly, we define r_f and m_f for E_f. After appending E_b to the rectangle distinguisher E', there are still inactive bits in the input of E_b, i.e. $r_b < n$. Let k denote the master key size.

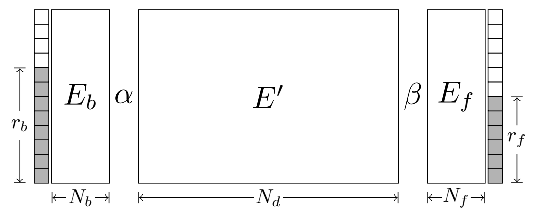


Figure 1: Differential key-recovery attack on block cipher E

In differential cryptanalysis, an attacker's goal is to either distinguish E from a random function, or to recover the master key based on the differential and partial decryption technique.

When searching for a differential distinguisher, the attacker aims at differentials that cover the highest number of rounds or with the maximized probability to distinguish the primitive from a random function. Meanwhile, for performing a key recovery attack, there are two metrics to be optimized, *i.e.*, on top of maximizing the number of attacked rounds, minimizing the attack complexity. Intuitively, using a differential distinguisher with highest probability of longest rounds, it is more likely to launch a better key recovery

Table 1: Summary of cryptanalytic results on `SKINNY` and `ForkSkinny`, where ID, ZC and DS-MITM denote impossible differential, zero correlation and Demirci-Selçuk Meet-in-the-Middle cryptanalysis. SK and RK denote single-key and related-key settings, respectively.

`SKINNY`

Version	Rounds	Data	Time	Memory	Approach	Setting	Ref.
64-128	18	$2^{62.68}$	2^{126}	2^{64}	ZC	SK	[SMB18]
	18	2^{60}	2^{116}	2^{112}	ID	SK	[DHLP20]
	19	2^{62}	$2^{119.8}$	2^{110}	ID	SK	[YQC17]
	20	$2^{47.69}$	$2^{121.08}$	$2^{74.69}$	ID	SK	[TAY17]
	20	$2^{68.4}$	$2^{97.5}$	2^{82}	ZC/Integral	SK	[ADG+19]
	22	$2^{63.5}$	$2^{110.9}$	$2^{63.5}$	Rectangle	RK	[LGS17]
	23	$2^{62.47}$	$2^{125.91}$	2^{124}	ID	RK	[LGS17]
	23	$2^{62.47}$	2^{124}	$2^{77.47}$	ID	RK	[SMB18]
	23	$2^{71.4}$	2^{79}	$2^{64.0}$	ID	RK	[ABC+17]
	23	$2^{60.54}$	$2^{120.7}$	$2^{60.9}$	Rectangle	RK	[HBS20]
	24	$2^{61.67}$	$2^{96.83}$	2^{84}	Rectangle	RK	Sect. 5.2
64-192	21	2^{62}	$2^{180.5}$	2^{170}	ID	SK	[YQC17]
	22	$2^{47.84}$	$2^{183.97}$	$2^{74.84}$	ID	SK	[TAY17]
	23	$2^{73.2}$	$2^{155.6}$	2^{138}	ZC/Integral	SK	[ADG+19]
	27	$2^{63.5}$	$2^{165.5}$	2^{80}	Rectangle	RK	[LGS17]
	29	$2^{62.92}$	$2^{181.7}$	2^{80}	Rectangle	RK	[HBS20]
	30	$2^{62.87}$	$2^{163.11}$	$2^{68.05}$	Rectangle	RK	Sect. 5.1
128-256	19	2^{123}	$2^{241.8}$	2^{221}	ID	SK	[YQC17]
	20	$2^{92.1}$	$2^{245.72}$	$2^{147.1}$	ID	SK	[TAY17]
	22	2^{127}	$2^{235.6}$	2^{127}	Rectangle	RK	[LGS17]
	23	$2^{124.47}$	$2^{251.47}$	2^{248}	ID	RK	[LGS17]
	23	$2^{124.41}$	$2^{243.41}$	$2^{155.41}$	ID	RK	[SMB18]
	24	$2^{125.21}$	$2^{209.85}$	$2^{125.54}$	Rectangle	RK	[HBS20]
	25	$2^{124.48}$	$2^{226.38}$	2^{168}	Rectangle	RK	Sect. 5.4
128-384	21	2^{123}	$2^{353.6}$	2^{341}	ID	SK	[YQC17]
	22	2^{96}	$2^{382.46}$	$2^{330.99}$	DS-MITM	SK	[SSD+18]
	22	$2^{92.22}$	$2^{373.48}$	$2^{147.22}$	ID	SK	[TAY17]
	27	2^{123}	2^{331}	2^{155}	Rectangle	RK	[LGS17]
	28	2^{122}	$2^{315.25}$	$2^{122.32}$	Rectangle	RK	[ZDM+20]
	30	$2^{125.29}$	$2^{361.68}$	$2^{125.8}$	Rectangle	RK	[HBS20]
	30	2^{122}	$2^{341.1}$	$2^{128.02}$	Rectangle	RK	Sect. 5.3

`ForkSkinny`

Version	Rounds ($R_{\texttt{init}}/R_{\text{I}}/R_{\text{II}}$)	Data	Time	Memory	Approach	Setting	Ref.
128-256 (128-bit key)	24(7/27/17)	$2^{122.5}$	$2^{124.5}$	$2^{97.5}$	ID	RK	[BDL20]
	25(18/27/7)	$2^{118.88}$	$2^{118.88}$	$2^{119.2}$	Rectangle	RK	Sect. 6.2
128-256 (256-bit key)	26(7/27/19)	2^{125}	$2^{254.6}$	2^{160}	ID	RK	[BDL20]
	26(7/27/19)	2^{127}	$2^{250.3}$	2^{160}	ID	RK	[BDL20]
	28(20/27/8)	$2^{118.88}$	$2^{246.98}$	2^{136}	Rectangle	RK	Sect. 6.1

attack. However, in practice, a good key recovery attack often requires a comprehensive trade-off between the key recovery phase and the differential distinguisher.

Differential key-recovery attack. The general procedures of the key-recovery attack based on a differential $\alpha \mapsto \beta$ (see Fig. 1):

1. Data collection: Collect $y = 2 \times 2^{-r_b} \cdot s/p$ structures of 2^{r_b} plaintexts each, where s is the expected number of right pairs.
2. Filter the wrong pairs using inactive bits of the ciphertext and there are $y \cdot 2^{2r_b-1}/2^{n-r_f}$ pairs left.
3. Initialize a list of $2^{m_b+m_f}$ empty counters.
4. For all $y \cdot 2^{2r_b-1}/2^{n-r_f}$ pairs, perform a guess and filter procedure to determine candidate keys (denote the time complexity of the guess and filter procedure as ε) and increase the corresponding counters.

The data complexity is about $y \cdot 2^{r_b} = 2 \cdot s/p$. The time complexity to generate the key rank counters is about $y \cdot 2^{2r_b-1}/2^{n-r_f} \cdot \varepsilon \approx 2 \cdot s/p \cdot 2^{r_b+r_f-n} \cdot \varepsilon$. In order to lower the time complexity of the key-recovery attack, we have to not only increase the probability of $\alpha \mapsto \beta$, but also decrease $2^{r_b+r_f-n}$. Hence, there may exist various trade-offs between the differentials, the attacked rounds, the data complexity and the time complexity. A differential with lower probability may lead to better time-data tradeoff, or even lead to longer attacked rounds, due to the potentially marginal term $2^{r_b+r_f}$ in such a differential.

In this paper, we maximize the number of attacked rounds. Hence, the following constraints are necessary in our optimization strategy.

$$\begin{cases} s/p < 2^n \\ s/p \cdot 2^{r_b+r_f-n} \cdot \varepsilon < 2^k \end{cases} \tag{1}$$

Maximize:

$$N_b + N_d + N_f \tag{2}$$

2.2 The tradeoff in rectangle attack on ciphers with linear key-schedule

Boomerang attack is a statistical cryptanalysis proposed by Wagner in 1999 [Wag99]. The original boomerang distinguisher is constructed by splitting the encryption function into two parts $E' = E_1 \circ E_0$, where two differentials $\alpha \xrightarrow{E_0} \beta$ and $\gamma \xrightarrow{E_1} \delta$ are combined to a boomerang. The probability of a boomerang is estimated by p^2q^2, where p, q are the probability of the differentials. A number of studies have shown advanced techniques for a better evaluation of the boomerang probability, including sandwich attack, boomerang switch, etc., in both single-key and related-key models [Mur11, BK09, BDK05]. In 2018, previous observations on the boomerang switch are unified in the framework of boomerang connectivity table (BCT) by Cid *et al.* [CHP+18]. As a result, the probability of the middle round in a boomerang can be precisely evaluated. Based on the proposal of the BCT table, Liu *et al.* studied the automatic search model tailored for boomerang distinguishers with the BCT table [LGS17]. Recently, Song *et al.* [SQH19] and Wang *et al.* [WP19] presented new techniques to consider the middle layer consisting of multiple rounds. Thanks to the latest progress on computing the middle part [CHP+18, BC18, SQH19, WP19], one can compute the probability of the middle part theoretically for a few rounds.

A boomerang distinguisher requires a chosen-plaintext chosen-ciphertext model, and it can be converted into a chosen-plaintext attack that is known as the rectangle attack [BDK01] or amplified boomerang attack [KKS00]. The probability of the rectangle distinguisher is $2^{-n}p^2q^2$. In the attack, only α and δ are fixed and the internal differences β and γ can be arbitrary values as long as $\beta \neq \gamma$. Hence, the probability would be increased to $2^{-n}\hat{p}^2\hat{q}^2$, where

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

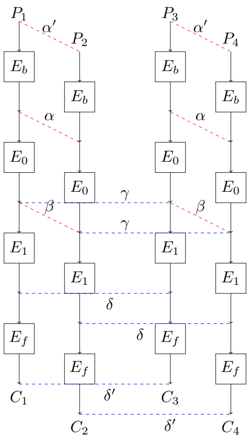


Figure 2: Rectangle attack on block cipher E

For block ciphers with linear key-schedule, Zhao *et al.* [ZDM+20, ZDJ19] proposed a new generalized related-key rectangle attack. We briefly recall the procedures with similar symbols as shown in Fig. 1. The steps of Zhao *et al.*'s framework in Fig. 2 are given below:

1. Construct $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q}$ structures of 2^{r_b} plaintexts each, where s is the expected number of right quartets.

2. For each structure, query the 2^{r_b} plaintexts by the encryption oracle under K_1, K_2, K_3 and K_4 and obtain four plaintext-ciphertext sets denoted by L_1, L_2, L_3 and L_4, where K_1 is the secret key and $K_2 = K_1 \oplus \Delta K$, $K_3 = K_1 \oplus \nabla K$ and $K_4 = K_1 \oplus \Delta K \oplus \nabla K$. 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 m_b subkey bits 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 structure, partially encrypt plaintext $P_1 \in L_1$ to the position of α by the guessed subkey bits, and partially decrypt it to the plaintext P_2 after xoring the known difference α. Then we look up H_1 to find the plaintext-ciphertext indexed by the r_b bits. Do the same operation with P_3 and P_4. We get two sets

$$S_1 = \{(P_1,C_1,P_2,C_2)|(P_1,C_1) \in L_1,(P_2,C_2) \in L_2, E_{b_{K_1}}(P_1) \oplus E_{b_{K_2}}(P_2) = \alpha\}, \quad (3)$$
$$S_2 = \{(P_3,C_3,P_4,C_4)|(P_3,C_3) \in L_3,(P_4,C_4) \in L_4, E_{b_{K_3}}(P_3) \oplus E_{b_{K_4}}(P_4) = \alpha\}. \quad (4)$$

   (c) The size of S_1, as well as S_2, is $y \cdot 2^{r_b}$ with y structures. Insert S_1 into a hash table H_3 indexed by the $(n - r_f)$ bits of C_1 and $(n - r_f)$ bits of C_2 that set to 0 in the output difference through E_f from δ. Then for each element of S_2, we find the corresponding (P_1, C_1, P_2, C_2) satisfying $C_1 \oplus C_3 = 0$ and $C_2 \oplus C_4 = 0$ in the $(n - r_f)$ bits. In total we obtain $y^2 \cdot 2^{2r_b-2(n-r_f)}$ quartets.

(d) We use all the quartets obtained in step (c) to determine the key candidates involved in E_f and increase the corresponding counters. This phase is just a guess and filter procedure. We denote the time complexity in this step as ε.

The data complexity is $4 \cdot y \cdot 2^{r_b} = 4 \cdot \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q}$ chosen plaintexts. The time complexity to generate the counters is about $2^{m_b} \cdot y^2 \cdot 2^{2r_b-2(n-r_f)} \cdot \varepsilon \approx 2^{m_b+2r_f-n} \cdot s/\hat{p}^2\hat{q}^2 \cdot \varepsilon$. Hence, in order to attack more rounds, we need

$$\begin{cases} 4 \cdot \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q} < 2^n, \\ 2^{m_b+2r_f-n} \cdot s/\hat{p}^2\hat{q}^2 \cdot \varepsilon < 2^k. \end{cases} \tag{5}$$

Maximize:

$$N_b + N_d + N_f. \tag{6}$$

3 Specification of SKINNY

The lightweight block cipher SKINNY was proposed by Beierle *et al.* [BJK+16]. Let n denote the block size, t denote the tweakey size and c denote the cell size, the family of SKINNY has six main versions SKINNY-n-t: for each $n \in \{64, 128\}$, there are three tweakey size versions $t = n$, $t = 2n$ and $t = 3n$. The internal state is viewed as a 4×4 square array of cells and the tweakey is viewed as a set of z 4×4 square arrays of cells, where $z = t/n \in \{1, 2, 3\}$. The set of tweakey arrays are denoted as $(TK1)$ when $z = 1$, $(TK1, TK2)$ when $z = 2$, and $(TK1, TK2, TK3)$ when $z = 3$. SKINNY follows an SPN structure and a TWEAKEY framework [JNP14]. In each round of SKINNY, the state is updated with 5 operations: SubCells (SC), AddConstants (AC), AddRoundTweakey (ART), ShiftRows (SR) and MixColumns (MC), which is illustrated in Fig. 3.

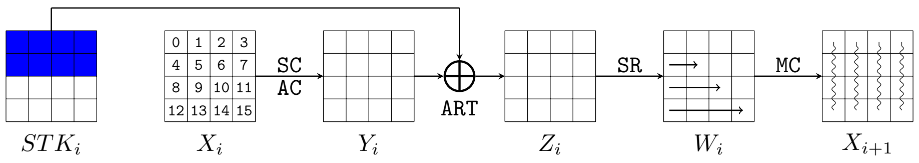


Figure 3: The i^{th} round of SKINNY

The subtweakey STK_i is only xored to the first two rows. Refer to Appendix A.1 for more details of the tweakey schedule. ΔX_i is the difference of state X in round i. $X_i[j, \dots, k]$ denote the cells of the state with index $\{j, j+1, \cdots, k\}$, where $0 \le j, k \le 15$. We denote the equivalent subtweakey in round i by ETK_i, where $ETK_i = \mathtt{MC} \circ \mathtt{SR}(STK_i)$ as Fig. 4.

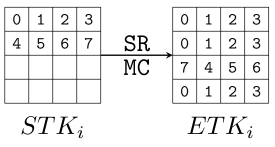


Figure 4: The relations between the cells of STK_i and ETK_i

The MC operation adopts non-MDS binary matrix M. The matrix M and its inverse matrix M^{-1} are as follows:

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}, \quad M^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}. \tag{7}$$

For the MDS matrix of AES, when 4 out of 8 input-output bytes are fixed, other bytes are determined. However, the situation is different for SKINNY's non-MDS matrix. For instance, when the input bytes are $(1, 1, 1, ?)$, the output bytes are $(?, 1, 1, 1)$, where "1" labels a known value and "?" unknown. Let $M \cdot (a, b, c, d)^{\mathtt{T}} = (\alpha, \beta, \gamma, \delta)^{\mathtt{T}}$ and $M^{-1} \cdot (\alpha, \beta, \gamma, \delta)^{\mathtt{T}} = (a, b, c, d)^{\mathtt{T}}$, we have

$$\begin{cases} a \oplus c \oplus d = \alpha, \\ a = \beta, \\ b \oplus c = \gamma, \\ a \oplus c = \delta. \end{cases} \tag{8} \qquad \begin{cases} \beta = a, \\ \beta \oplus \gamma \oplus \delta = b, \\ \beta \oplus \delta = c, \\ \alpha \oplus \delta = d. \end{cases} \tag{9}$$

Lemma 1. *[BDL20] For any given* SKINNY *S-box S and any two non-zero differences δ_{in} and δ_{out}, the equation $S_i(y) \oplus S_i(y \oplus \delta_{in}) = \delta_{out}$ has one solution on average.*

4 Automated Search Oriented to Key Recovery

4.1 Previous automatic modelling of searching boomerang distinguishers on SKINNY

The designers of SKINNY [BJK+16] first gave the Mixed-Integer Linear Programming (MILP) model to search truncated differentials of SKINNY. Later, Liu *et al.* [LGS17] tweaked the model to search boomerang distinguishers. However, the probability $\hat{p}^2\hat{q}^2$ of the boomerang distinguisher is highly inaccurate without considering the dependence between the two differential trails. In [SQH19], Song *et al.* revisited the Boomerang Connectivity Table (BCT) proposed by Cid *et al.* in [CHP+18], and proposed a generalized framework of BCT to systematically calculate the probability of a boomerang distinguisher considering the dependence. They re-evaluated the probabilities of the boomerang distinguishers given in [LGS17], where the probabilities are much higher than before.

As in Dunkelman *et al.*'s (related-key) sandwich attack framework [DKS10], the N_d-round cipher E' is considered as $\tilde{E_1} \circ E_m \circ \tilde{E_0}$, where $\tilde{E_0}$, E_m, $\tilde{E_1}$ contain r_0, r_m, r_1 rounds, respectively. Let $\tilde{p}$ and $\tilde{q}$ be the probabilities of the upper differential used for $\tilde{E_0}$ and the lower differential used for $\tilde{E_1}$. The middle part E_m specifically handles the dependence and contains a small number of rounds. If the probability of generating a right quartet for E_m is t, the probability of the whole N_d-round boomerang distinguisher is $\tilde{p}^2\tilde{q}^2t$.

Recently, [HBS20] introduced a heuristic approach to search a boomerang distinguisher using MILP/SAT models. They introduced some new tables for S-boxes to model the dependency between the upper and lower differentials in boomerang distinguishers. We briefly introduce their searching approach by following steps:

1. Firstly, search truncated differentials with the minimum number of active S-boxes with a word-oriented MILP model considering the switching effect in multiple rounds, which is based on the idea of [CHP+17]'s model. In their MILP model, they searched for a $(r_0 + r_m)$-round upper truncated differential and a $(r_1 + r_m)$-round lower truncated differential. The objective function is the number of active S-boxes in the distinguisher. Considering the dependency, for the r_m-round middle part, only the S-boxes which are active in both upper and lower truncated differentials are involved in the objective function.

2. Then, use the MILP/SAT models to get the actual differential characteristics for the upper and lower truncated differential separately. If there is no actual differential characteristic, go back to step 1.

3. Evaluate the probability of the middle part experimentally. If the probability is equal to 0, go back to step 1.

4. Based on the algorithm proposed in [SQH19], evaluate the probability of the middle part mathematically.

Almost at the same time, [DDV20] proposed a new automatic tool to search boomerang distinguishers and provided their source code to facilitate follow-up works. Similar with [HBS20], they also introduced a set of tables which help to calculate the probability of the boomerang distinguisher. With the tables to help roughly evaluate the probability, they use an MILP model to search for the upper and lower trails throughout all rounds by automatically handling the middle rounds. Then a CP model is applied to search for the best possible instantiations.

4.2 Our model to determine a distinguisher

According to Sect. 2, to launch a boomerang attack covering more rounds with a boomerang distinguisher, the interaction between the N_d-round distinguisher and the N_b and N_f extended rounds need to be considered simultaneously. Given the time complexity $2^{m_b+2r_f-n} \cdot s/\hat{p}^2\hat{q}^2 \cdot \varepsilon$ in Sect. 2, it is necessary to set additional constraints over the active r_f bits in the output state of the lower trail and m_b-bit subtweakey involved in the extended N_b rounds. So we present an extended model for searching the entire $(N_b + N_d + N_f)$ rounds of a boomerang attack. The aim is to find new boomerang distinguishers in a related-tweakey setting that result in key-recovery attacks with more rounds. In Sect. 2.2, the target is to maximize $N_b + N_d + N_f$. However, in practical programming, we take N_b, $N_d = r_0 + r_m + r_1$ and N_f as parameters to input the model, and the target is the time complexity $2^{m_b+2r_f-n} \cdot s \cdot (\tilde{p}^2\tilde{q}^2 t)^{-1} \cdot \varepsilon$ as in Sect. 2. By feeding different values of N_b, N_d and N_f into the model, we try to find the maximal value for $N_b + N_d + N_f$.

Our new model tweaks the model of Hadipour *et al.* [HBS20] to search for a $(N_b + r_0 + r_m)$-round upper truncated differential and a $(r_m + r_1 + N_f)$-round lower truncated differential.

Let X^u_r and X^l_r denote the internal state before `SubCells` in round r of the upper and lower truncated differentials. For i-th cell of the internal state X^u_r of the upper differential, we define a binary variable $\texttt{DXU}[r][i]$ $(0 \le r \le N_b + r_0 + r_m, 0 \le i \le 15)$, where $\texttt{DXU}[r][i] = 1$ indicates that the corresponding cell is active. Then a binary variable $\texttt{DSTKU}[r][i]$ $(0 \le r \le N_b + r_0 + r_m, 0 \le i \le 15)$ is defined for the i-th cell of the subtweakey STK_r of the upper differential, to indicate whether the subtweakey cell is active or not. Similarly, there are $\texttt{DXL}[r][i]$ and $\texttt{DSTKL}[r][i]$ $(0 \le r \le r_m + r_1 + N_f, 0 \le i \le 15)$ defined for the lower differential. The modelling strategies of the $(r_0 + r_m)$-round and $(r_m + r_1)$-round are the same to the model of Hadipour *et al.* [HBS20]. The constraints for the tweakey schedule of `SKINNY` are also the same to the previous ones [BJK+16, HBS20, LGS17]. Here, we only list the differences in our model.

Modelling the active cells propagation in the N_f rounds after the lower differential. Starting from the $(r_m + r_1)$-round internal state $X^l_{r_m+r_1}$ of the lower differential, the truncated difference is propagated forwards with probability 1. Hence, the constraints on `DXL` and `DSTKL` is different from those in the $(r_m + r_1)$ rounds. As introduced in [BJK+16], the linear diffusion layer `SR` and `MC` of `SKINNY` can be seen as a binary 16×16 matrix L. Let $L_x(i)$ (resp. $L^{-1}_x(i)$) be the set of the indexes j such that the coefficient $L_{i,j} = 1$ in matrix L (resp. $L^{-1}_{i,j} = 1$ in L^{-1}, see in Appendix A.2). In the related-tweakey attacks, the subtweakey differences will affect the differences of the internal state. Take the key-recovery attack on SKINNY-64-192 as example (see Fig.7). In Round 27, the non-zero difference in $\Delta STK_{27}[3]$ leads to differences in $\Delta X_{28}[7, 15]$, and further affects the following internal states. Note that the subtweakeys are only xored to the first two rows of the state. ETK_r is produced by STK_r by applying the matrix L (Eq.(16)). As shown in Fig. 4, each cell of ETK_r only depends on one cell of STK_r, since there is only one entry of $L_{i,j} = 1$ with $0 \le j \le 7$ for a given i in the linear matrix L (Eq.(16)). For each i, we denote such j that makes $L_{i,j} = 1$ $(0 \le j \le 7)$ as $L_k(i)$. Therefore, the constraints on the impact of DSTKL to the internal state and the propagation of the active cells in N_f rounds are given below: $\forall\ 0 \le r \le N_f - 1, 0 \le i \le 15$,

$$\texttt{DXL}[r_m + r_1 + r + 1][i] = (\bigvee_{j \in L_x(i)} \texttt{DXL}[r_m + r_1 + r][j]) \ \vee (\texttt{DSTKL}[r_1 + r_m + r][L_k(i)]). \tag{10}$$

In addition, we require that there are some inactive cells in the output state of the lower differential, which can help us filter quartets to reduce the time complexity in the key-recovery attack [1].

$$\sum_{0 \le i \le 15} \texttt{DXL}[r_m + r_1 + N_f - 1][i] \le 15.$$

Modelling the active cells propagation in the N_b rounds before the upper differential. For the $(N_b + r_0 + r_m)$-round upper differential, starting from the N_b-round internal state $X^u_{N_b}$ of the upper differential, the truncated difference is propagated backwards with probability 1.

Note that different from the key-recovery attack, in the searching model, we use the original style of representation of SKINNY, i.e., we do not use the equivalent key ETK_0 to replace STK_0. The constraints describing the impact of DSTKU to the internal state and the active cells propagation backwards from the N_b-round are as follows (note that the subtweakeys are only xored to the first two rows of the state):

$$\begin{aligned} \texttt{DXU}[r][i] =& (\bigvee_{j \in \mathtt{L_x^{-1}(i)}} \texttt{DXU}[r+1][i]) \ \vee \texttt{DSTKU}[r][i], \ \forall\ 0 \le r \le N_b - 2, 0 \le i \le 7, \\ \texttt{DXU}[r][i] =& \bigvee_{j \in \mathtt{L_x^{-1}(i)}} \texttt{DXU}[r+1][i], \ \forall\ 0 \le r \le N_b - 2, 8 \le i \le 15. \end{aligned} \tag{11}$$

where, $\texttt{DXU}[N_b - 1][i]$ is determined by the input difference of the $(r_0 + r_m)$-round upper differential. Taking Fig. 5 as an example, $N_b = 4$, the $\texttt{DXU}[3][i]$, i.e., the activeness of state X_3 is determined by Y_3, where the constraints of Y_3 to W_3 are included by the program of $(r_0 + r_m)$ rounds truncated differential.

To avoid all cells active in the plaintext, we require that at least one cell of the internal state X_1 of the upper differential is inactive[2]:

$$\sum_{0 \le i \le 15} \texttt{DXU}[1][i] \le 15.$$

[1]Note that, in the key-recovery attack, we use the active cells of $X^l_{r_m+r_0+N_f-1}$ to filter quartets instead of $X^l_{r_m+r_0+N_f}$.

[2]In the key-recovery attack, we use the active cells of X^u_1 to collect data instead of X^u_0.

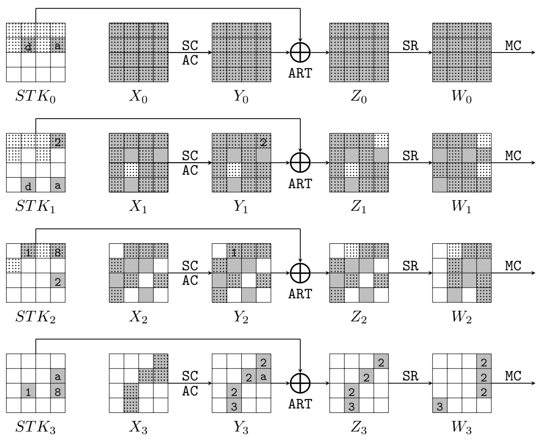


Figure 5: Phase (1): The N_b-round of the attack on `SKINNY-64-192`

Modeling the subtweakey cells involved in the N_b rounds before the upper differential. There are many papers considering the dependence of keys in key-recovery, such as [DFJ13, FN20, DF16]. As shown in Eq.(1), the time complexity of the rectangle attack is also highly related to the number of keys involved in N_b, i.e., m_b. In fact, we need to guess m_b-bit subtweakey first to deduce P_2 from P_1 by partial encryption and decryption in the first N_b rounds. Please recall the details in Step 3 (b) of the rectangle attack model in Sect. 2.2.

We hope that the smaller m_b, the better. Since the matrix M in the `MC` operation is not an MDS matrix, the subtweakeys involved in the partial encryption and decryption are different. So we model the subtweakey cells from two aspects.

Taking the key-recovery attack on `SKINNY-64-192` as example (see Fig. 7), we decompose the whole process of deducing P_2 from P_1 into two phases:

(1) Partially encrypt P_1 to Y_3 (only active cells) as shown in Fig. 5.

(2) Partially decrypt $\bar{Y_3} = Y_3 \oplus \Delta Y_3$ to get P_2 as shown in Fig. 6.

In Phase (1) shown in Fig. 5, in order to compute $Y_3[3,6,7,9,13]$ from P_1, we need the values of the cells marked by ▦ in X_3. With the details of `MC` and `SR`, the values of the cells marked by ⬚ and ▦ in Z_2 and corresponding STK_2 are needed. The cell positions in X_2 which need to be known are the same as those in Z_2, which are marked by ⬚ and ▦ . Similarly deducing for other rounds, the cells need to be known are all determined for round 0-3, which are marked by ⬚ and ▦ .

We define a binary variable `KnownEnc`$[r][i]$ to identify whether the ith ($0 \leq i \leq 15$) cells of the internal state Z_r ($0 \leq r \leq N_b - 1$) should be known in Phase (1). A binary 16×16 matrix L^E (see in Appendix A.2) is introduced to describe the linear diffusion (combination of `SR` and `MC`) determining the cells need to be known in Z_r (same for X_r) from X_{r+1}. Let $L^E_z(i)$ be the set of the indexes j such that the coefficient $L^E_{i,j} = 1$ in the matrix L^E. Initially in round $N_b - 1$ (round 3 in Fig.5), we have `KnownEnc`$[N_b - 1][i] =$ `DXU`$[N_b - 1][i] (0 \leq i \leq 15)$. For round $N_b - 2$ to 0, we have

$$\texttt{KnownEnc}[r][i] = \bigvee_{j \in L^E_z(i)} \texttt{KnownEnc}[r+1][j],\ \forall\ 0 \leq r \leq N_b - 2,\ 0 \leq i \leq 15. \tag{12}$$

In addition, for $0 \leq r \leq N_b - 2$ (round 0-2 in Fig. 5), whether $STK_r[i]$ should be guessed is identified by `KnownEnc`$[r][i]$, where $0 \leq i \leq 7$.

In Phase (2) as shown in Fig. 6, with $Y_3[3,6,7,9,13]$ computed in Phase (1), we compute $\bar{Y_3}[3,6,7,9,13] = Y_3 \oplus \alpha[3,6,7,9,13]$. So the differences of the active cells in

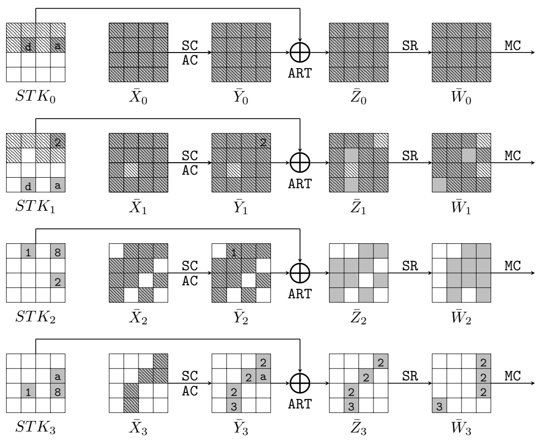


Figure 6: Phase (2): The N_b-round of the attack on SKINNY-64-192

ΔX_3 are determined. Therefore, we compute the differences of active cells in ΔY_2 from ΔX_3 through the linear operations SR and MC. In order to compute backwards further, we have to know the values of the differential active cells of Y_2 from P_1. The calculation from P_1 to Y_2 is similar to the calculation from P_1 to Y_3 in the encryption of Phase (1). To compute the values of the active cells of Y_2, all the cells in Z_1 marked by north west lines have to be known. Moreover, similar to Y_2, all active cells of Y_1 are also needed to be known. Hence, combining the cells to compute Y_2 and the active cells in Y_1, all the cells marked by north west lines in Y_1 are needed to be known.

Suppose we have known such cells in Y_2 marked by ▨ , then we are able to compute the active cells marked by ▨ of $\bar{Y}_2 = Y_2 \oplus \Delta Y_2$ and the difference of active cells of ΔX_2. Similarly, with ΔY_1 deduced from ΔX_2 and active cells known in Y_1, we compute ΔX_1. Then P_2 is determined. Totally, we have to compute cells marked by north west lines of Y_2 and Y_1.

In our programming of the model, we integrate the above two phases. For example, in the round 2, X_2 marked by dots in Phase (1) and by north west lines in Phase (2) are all needed to be known. Hence, we can take the union of these marked cells and compute backwards further. We define a binary variable Known $(1 \leq r \leq N_b - 1,\ 0 \leq i \leq 15)$ for each cell of each internal state X_r (same for Y_r) to indicate whether the value is needed either in Phase (1) or Phase (2). Known $= 1$ is marked by north east lines in Fig. 7. Then the binary variable KnownEnc $(0 \leq r \leq N_b - 2,\ 0 \leq i \leq 15)$ for each cell of Z_r indicate whether the value is needed to be known to compute the needed cells of X_{r+1}. Whether $STK_r[i]$ should be guessed is also identified by KnownEnc$[r][i]$, where $0 \leq r \leq N_b - 2$ and $0 \leq i \leq 7$. KnownEnc $= 1$ is also marked by north east lines in Fig. 7. For the round $N_b - 1$, only active cells (i.e., state X_3 in Fig. 7) in the internal state need to be known and the subtweakey of this round does not need to be guessed:

$$\texttt{Known}[N_b - 1][i] = \texttt{DXU}[N_b - 1][i],\ \forall\ 0 \leq i \leq 15.$$

From round $N_b - 2$ to round 0, we give the constraints over the linear diffusion (SR and MC) determining which cells to be known in Z_r from X_{r+1} as in Phase (1) :

$$\texttt{KnownEnc}[r][i] = \bigvee_{j \in L_z^E(i)} \texttt{Known}[r+1][j],\ \forall\ 0 \leq r \leq N_b - 2,\ 0 \leq i \leq 15.$$

In round $N_b - 2$ to round 1, the cells in X_r need to be known involve two types: the active cells need to be known in Phase (2), and cells need to be known in Z_r, which are computed from the needed cells of X_{r+1} in last round :

$$\texttt{Known}[r][i] = \texttt{KnownEnc}[r][i] \vee \texttt{DXU}[r][i],\ \forall\ 1 \le r \le N_b - 2,\ 0 \le i \le 15.$$

The objective function. As in Sect. 2, the time complexity is $2^{m_b+2r_f-n} \cdot s \cdot (\tilde{p}^2\tilde{q}^2t)^{-1} \cdot \varepsilon$. For the $(r_0+r_m+r_1)$-round part, the target to be optimized is the same to Hadipour *et al.* [HBS20]. First we add the variables `DXU` of r_0-round upper differential (on behalf of the $\tilde{p}$) with weight w_0 to the objective function. Similarly add the variables `DXL` of r_1-round lower differential (on behalf of the $\hat{q}$) with weight w_1. Then, considering the switching effects, we add the variables `DXU` and `DXL` in r_m-round middle part (on behalf of the t) with weight w_m to the objective function. In order to find the distinguishers whose probabilities are likely larger than random case, we set an upper bound on the number of active Sbox in the $(r_0+r_m+r_1)$-round part:

$$\begin{aligned}&\sum_{0\le r\le r_0-1,\ 0\le i\le 15} w_0 \cdot \texttt{DXU}[N_b+r][i] + \sum_{0\le r\le r_1-1,\ 0\le i\le 15} w_1 \cdot \texttt{DXL}[r_m+r][i] + \\ &\sum_{0\le r\le r_m-1,\ 0\le i\le 15} w_m \cdot (\texttt{DXU}[N_b+r_0+r][i] \wedge \texttt{DXL}[r][i]) \ \le\ \texttt{BOUND},\end{aligned} \tag{13}$$

where `BOUND` is selected experimentally. Namely, suppose minimum of Eq. (13) is `MIN`, we set `BOUND`= `MIN` + 10.

In addition, we add the variables `KnownEnc` (on behalf of the m_b) and `DXL` (on behalf of the r_f) with different weight w_u and w_l to get a uniformed objective:

$$\begin{aligned}obj = &\sum_{0\le r\le r_0-1,\ 0\le i\le 15} w_0 \cdot \texttt{DXU}[N_b+r][i] + \sum_{0\le r\le r_1-1,\ 0\le i\le 15} w_1 \cdot \texttt{DXL}[r_m+r][i] + \\ &\sum_{0\le r\le r_m-1,\ 0\le i\le 15} w_m \cdot (\texttt{DXU}[N_b+r_0+r][i] \wedge \texttt{DXL}[r][i]) + \\ &\sum_{0\le r\le N_b-2,\ 0\le i\le 7} w_u \cdot \texttt{KnownEnc}[r][i] + \sum_{0\le i\le 15} w_l \cdot \texttt{DXL}[r_m+r_1+N_f-1][i].\end{aligned} \tag{14}$$

Because different parameters have different coefficients in the formula of the time complexity, we give them different weights to model the objective more accurately. For example, in Eq. (5), with $m_b + 2 \cdot r_f$, the complexity is more sensitive with r_f than m_b. So we set the the coefficients as $w_l = 2 \cdot w_u$.

Then considering the probabilities in the DDT tables of the S-boxes and the switching effects similar to [HBS20], we adjust the weight $w_l = 2w_u = 2w_0 = 2w_1 = 4w_m = 4$. We use different N_b, N_d and N_f to run our model. N_b and N_f are chosen from 1 to 4. N_d is chosen based on experience, for example, when the best previous distinguisher have y rounds, we will choose N_d with $y-3 \le N_d \le y$. As pointed out in [SQH19], the dependencies of the upper and lower trails could affect up to 6 rounds. So we choose $r_m = 6$, and then r_0 and r_1 vary with N_d.

With our new model, we search for more proper truncated upper and lower differentials for applying the key-recovery attack. And then, for r_0-round $\tilde{E}_0$ of the upper differential and r_1-round $\tilde{E}_1$ of the lower differential, we use the CP model to get a instantiation for the truncated differentials, as [SGL⁺17]. We also calculate the probability $\tilde{p}$ and $\tilde{q}$ considering the clustering effect. We experimentally calculate the probability of $r_m = 6$-round middle part of the distinguisher. Note that the probability of the middle part should be high to be verified with a small computer in reasonable time. We use one computer equipped with one RTX 2080 Ti to experimentally compute the probability of the middle part and the results of our experiments are listed in Table 2. Note that t may be zero. If so, we need to find a new instantiation for the truncated differentials or even need to search new

Table 2: Experiments on the middle part of the boomerang distinguishers for SKINNY

Version	r_m	Probability t	Complexity	Time
64-128	6	$2^{-23.96}$	2^{32}	29.51s
64-192	6	$2^{-18.73}$	2^{29}	5.48s
128-256	6	$2^{-32.39}$	2^{38}	2835.37s
128-384	6	$2^{-27.03}$	2^{34}	226.64s

truncated differentials. When $t > 0$ and $\tilde{p}^2\tilde{q}^2t > 2^{-n}$, we successfully get a boomerang distinguisher with probability $\tilde{p}^2\tilde{q}^2t$.

The details of the boomerang distinguishers we obtained are listed in Table 4, 5, 6 and 7. For more details, we refer to Table 16, 17, 18 and 19 in Appendix. B. In addition, we summarize the previous boomerang distinguishers for SKINNY in Table 3.

Table 3: Summary of related-tweakey boomerang distinguishers for SKINNY. N_d is the round number of distinguishers; $N_b + N_d + N_f$ is the total attacked number of rounds.

Version	N_d	Probability $\tilde{p}^2\tilde{q}^2t$	$N_b + N_d + N_f$	Ref.
64-128	17	$2^{-29.78}$	-	[SQH19]
	17	$2^{-48.72}$	21	[LGS17]
	19	$2^{-51.08}$	23	[HBS20]
	19	$2^{-54.36}$	-	[DDV20]
	18	$2^{-55.34}$	24	Ours
64-192	22	$2^{-42.98}$	-	[SQH19]
	22	$2^{-54.94}$	26	[LGS17]
	23	$2^{-55.85}$	29	[HBS20]
	23	$2^{-57.93}$	-	[DDV20]
	22	$2^{-57.73}$	30	Ours
128-256	18	$2^{-77.83}$	-	[SQH19]
	18	$2^{-103.84}$	22	[LGS17]
	20	$2^{-85.77}$	-	[DDV20]
	21	$2^{-116.43}$	24	[HBS20]
	19	$2^{-116.97}$	25	Ours
128-384	22	$2^{-48.30}$	-	[SQH19]
	23	2^{-112}	27	[LGS17]
	23	2^{-112}	28	[ZDM+20]
	24	$2^{-86.09}$	-	[DDV20]
	25	$2^{-116.59}$	30	[HBS20]
	22	$2^{-101.49}$	30	Ours

Remarks. We compare our distinguishers with the other recent distinguishers, in particular with those provided by Hadipour *et al.* in [HBS20]. Our distinguishers have the same $r_m = 6$ with [HBS20], as pointed out in [SQH19] that the upper and lower differentials can be dependent up to 6 rounds. Then for the r_0-round upper differentials and r_1-round lower differentials, there are some differences, which are listed as follows:

- For the upper differentials, there are more than one active cell in the tweakey. So benefiting from the linear key schedule, the state differences propagation can be controlled by the subkey differences. Although our r_0 is larger or equal to that in [HBS20], there are fewer active cells in the input state. So when we extend same N_b rounds before the distinguisher, the number of subkey bits involved in E_b is smaller than the attack in [HBS20], e.g., for the attack on Skinny-64-128, by adding 2 rounds before the distinguisher, there are $m_b = 3c$ in our attack (see Sect. 5.2) and $m_b = 8c$ in [HBS20]. In some cases, we can extend more rounds, e.g., for the attack on `SKINNY-64-192`, there are $N_b = 4$ and $m_b = 19c$ in our attack (see Sect. 5.1) and $N_b = 3$ and $m_b = 16c$ in [HBS20].

- For the lower differentials, our distinguishers for `Skinny` have smaller r_1 than [HBS20] and fewer active cells in the output state. So we can extend more rounds (N_f) with fewer active cells (r_f) in the ciphertext, and filter more quartets with inactive cells of the ciphertexts before the tweakey recovery process to reduce the time complexity. Taking the attack on `SKINNY-64-192` as an example, when we add 4 rounds after our distinguisher, there are 12 active cells in the ciphertexts ($N_f = 4, r_f = 12$, see Sect. 5.1); in [HBS20], all the cells in the ciphertexts are active when 3 rounds are added ($N_f = 3, r_f = 16$).

So considering all the parameters affecting the key recovery attack, i.e. the number m_b of guessed subkeys in E_b, the number r_f of active cells in the ciphertext, the number $(N_b+r_0+r_m+r_1+N_f)$ of rounds of the whole attack and the probability of the $(r_0+r_m+r_1)$-round distinguisher, the numbers of attacked rounds in our paper and [HBS20] are different. From the point of number of attacked rounds, our new distinguishers perform better.

Table 4: The 18-round related-tweakey boomerang distinguisher for `SKINNY-64-128`

$r_0 = 8$, $r_m = 6$, $r_1 = 4$, $\tilde{p} = 2^{-15.69}$, $t = 2^{-23.96}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-55.34}$
$\Delta TK1 =$ 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, a, 0, 0, 0
$\Delta TK2 =$ 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 3, 0, 0, 0
$\Delta X_0 =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2
$\nabla TK1 =$ 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0
$\nabla TK2 =$ 0, 0, 0, 0, 0, 0, 0, 0, b, 0, 0, 0, 0, 0, 0, 0
$\nabla X_{18} =$ 0, 0, d, 0, 0, 0, d, 0, 0, 0, 0, 0, 0, 0, d, 0

Table 5: The 22-round related-tweakey boomerang distinguisher for `SKINNY-64-192`

$r_0 = 10$, $r_m = 6$, $r_1 = 6$, $\tilde{p} = 2^{-19.5}$, $t = 2^{-18.73}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-57.73}$
$\Delta TK1 =$ 8, 0, 0, 0, 0, 0, 0, c, 0, 0, 0, 0, 0, 0, 0, 1
$\Delta TK2 =$ b, 0, 0, 0, 0, 0, 0, 5, 0, 0, 0, 0, 0, 0, 0, 7
$\Delta TK3 =$ e, 0, 0, 0, 0, 0, 0, f, 0, 0, 0, 0, 0, 0, 0, c
$\Delta X_0 =$ 3, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla TK1 =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
$\nabla TK2 =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0, 0
$\nabla TK3 =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, d, 0, 0, 0, 0, 0
$\nabla X_{22} =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, a, 0, 0, 0, 0, 0

Table 6: The 19-round related-tweakey boomerang distinguisher for `SKINNY-128-256`

$r_0 = 9$, $r_m = 6$, $r_1 = 4$, $\tilde{p} = 2^{-42.29}$, $t = 2^{-32.39}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-116.97}$
$\Delta TK1 =$ 00, 00, 00, 00, a0, 00, 00, 00, ef, 00, 00, 00, 00, 00, 00, 00
$\Delta TK2 =$ 00, 00, 00, 00, 54, 00, 00, 00, 9d, 00, 00, 00, 00, 00, 00, 00
$\Delta X_0 =$ 00, 00, 00, 00, 13, 00, 10, 00, 00, 10, 00, 00, 10, 00, 00, 00
$\nabla TK1 =$ 00, 00, 00, 00, 00, 00, 00, ff, 00, 00, 00, 00, 00, 00, 00, 00
$\nabla TK2 =$ 00, 00, 00, 00, 00, 00, 00, cc, 00, 00, 00, 00, 00, 00, 00, 00
$\nabla X_{19} =$ 00, 01, 00, 00, 00, 01, 00, 00, 00, 00, 00, 00, 00, 01, 00, 00

Table 7: The 22-round related-tweakey boomerang distinguisher for SKINNY-128-384

$r_0 = 10$, $r_m = 6$, $r_1 = 6$, $\tilde{p} = 2^{-37.23}$, $t = 2^{-27.03}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-101.49}$
$\Delta TK1 =$ 14, 00, 00, 00, 00, 00, 00, 1b, 00, 00, 00, 00, 00, 00, 00, 56
$\Delta TK2 =$ 06, 00, 00, 00, 00, 00, 00, 84, 00, 00, 00, 00, 00, 00, 00, 66
$\Delta TK3 =$ 30, 00, 00, 00, 00, 00, 00, 24, 00, 00, 00, 00, 00, 00, 00, 37
$\Delta X_0 =$ 81, 00, 00, 00, 00, 00, 00, 01, 00, 00, 00, 00, 00, 00, 00, 00
$\nabla TK1 =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 8a, 00, 00, 00, 00, 00, 00
$\nabla TK2 =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 0c, 00, 00, 00, 00, 00, 00
$\nabla TK3 =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 7f, 00, 00, 00, 00, 00, 00
$\nabla X_{22} =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 50, 00, 00, 00, 00, 00, 00

5 Related-tweakey Rectangle Attacks on Round-reduced SKINNY

This section gives related-tweakey rectangle attacks on SKINNY-64-192, SKINNY-64-128, SKINNY-128-384 and SKINNY-128-256 using the new distinguishers obtained in Sect. 4.2.

5.1 The Key-recovery attack on 30-round SKINNY-64-192

We use the 22-round related-tweakey rectangle distinguisher with probability of $2^{-n} \cdot \tilde{p}^2\tilde{q}^2t = 2^{-64-57.73} = 2^{-121.73}$ for SKINNY-64-192 given in Table 5, where

$$\begin{aligned} \alpha &= (3,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0), \\ \delta &= (0,0,0,0,0,0,0,0,0,0,\mathtt{a},0,0,0,0,0). \end{aligned}$$

By extending 4 rounds before and 4 rounds after the 22-round distinguisher, we attack the 30-round SKINNY-64-192 as illustrated in Fig. 7.

In the first round, we first apply SR and MC operations, and then apply the ART operation with the equivalent subtweakey ETK_0 instead of the subtweakey STK_0. So there is no subtweakey involved in the first round, and we can build our structures at W_0'. Treating W_0' as the plaintext and Z_{29} as the ciphertext, we can get following parameters: $r_b = 15c$, $m_b = 19c$, $r_f = 12c$, $m_f = 17c$, where $c = 4$. (The notations are introduced in Sect. 2).

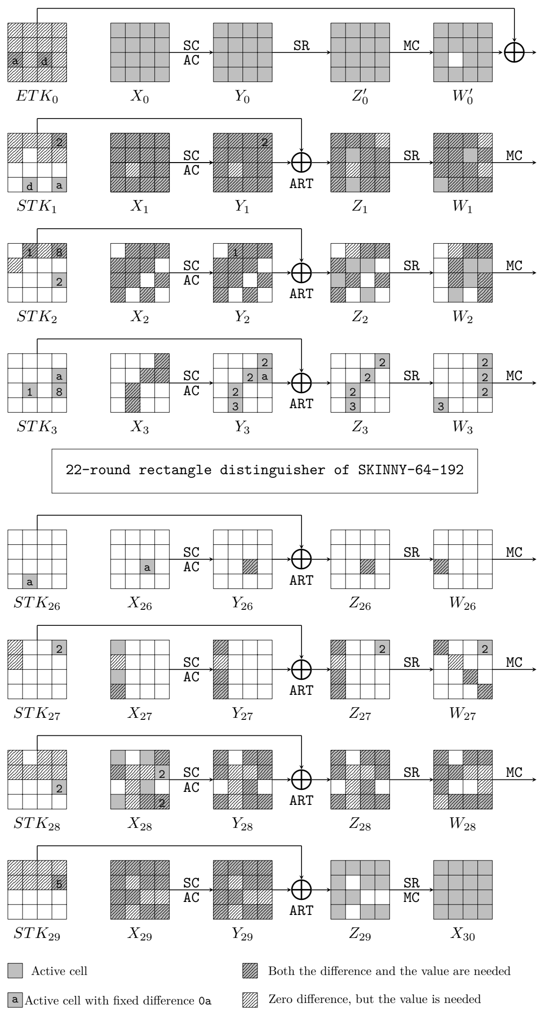
Based on Zhao *et al.*'s key recovery algorithm [ZDM+20], the details of our attack are as follows:

Data collection.

1. Construct $y = \sqrt{s} \cdot 2^{n/2-r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1}$ structures, where s is the expected number of right quartets. Each structure includes 2^{r_b} plaintexts by traversing all the possible values of $r_b/c = 15$ cells $W_0'[0-8, 10-15]$. Let $M = y \cdot 2^{r_b} = y \cdot 2^{60}$ denote the number of chosen plaintexts under each key.

2. Let K_1 be the master secret key, $K_2 = K_1 \oplus \Delta K$, $K_3 = K_1 \oplus \nabla K$ and $K_4 = K_1 \oplus \Delta K \oplus \nabla K$. For each structure, query the corresponding ciphertexts for the 2^{r_b} plaintexts under four related keys K_1, K_2, K_3 and K_4, which are named as four plaintext-ciphertext sets L_1, L_2, L_3 and L_4. Then insert L_2 and L_4 into hash tables H_1 and H_2 indexed by the 15 cells $W_0'[0-8, 10-15]$.

3. We have to guess $ETK_0[0-15]$, $STK_1[0-4, 6, 7]$ and $STK_2[1-4]$ marked by north west lines. According to Fig. 4, $ETK_0[i] = ETK_0[i+4] = ETK_0[i+12] = STK_0[i]$ for $0 \leq i \leq 3$. Therefore, there are only $2^{m_b} = 2^{19\times 4} = 2^{76}$ possible values of subtweakeys need to guessed in E_b:



SC
AC
SR
MC
ART
ETK_0
X_0
Y_0
Z'_0
W'_0
STK_1
STK_2
STK_3
22-round rectangle distinguisher of SKINNY-64-192
STK_26
STK_27
STK_28
STK_29
X_30
Active cell
Active cell with fixed difference 0a
Both the difference and the value are needed
Zero difference, but the value is needed

Figure 7: The 30-round attack against `SKINNY-64-192`

(a) Initialize a list of $2^{m_f} = 2^{68}$ counters, where each corresponds to a 68-bit subtweakey guess involved in E_f.

(b) For each structure, we need to deduce P_2 from P_1. Due to that the values of $W'_0[0-15]$ are known from P_1 and $ETK_0[0-15]$ are guessed, we can deduce the values of $Y_1[0-15]$. Then using the guessed $STK_1[0-4,6,7]$, we deduce the values of $Y_2[1-6,8,9,11,12,14]$. With the guessed $STK_2[1-4]$, the values of $Y_3[3,6,7,9,13]$ are known. Because $\bar{Y}_3[3,6,7,9,13] = Y_3[3,6,7,9,13] \oplus \Delta Y_3[3,6,7,9,13]$, we compute the differences in $\Delta X_3[3,6,7,9,13]$ and propagate to the differences in $\Delta Y_2[1-6,8,9,11,12,14]$. Since the values of $Y_2[1-6,8,9,11,12,14]$ are known, we can compute the differences in $\Delta X_2[1-6,8,9,11,12,14]$ and propagate the differences to $\Delta Y_1[0-8,10-15]$. Similarly, we can compute the differences in $\Delta W'_0[0-8,10-15]$ using the known $Y_1[0-8,10-15]$. So we can deduce $\bar{W}'_0[0-8,10-15] = W'_0[0-8,10-15] \oplus \Delta W'_0[0-8,10-15]$. Query the hash table H_1 with $\bar{W}'_0[0-8,10-15]$ of P_2 to get the corresponding plaintext-ciphertext pair, and get S_1 as Eq. 3.

For L_2 and L_4, use the similar method to get set S_2 as Eq. 4. The sizes of S_1 and S_2 are both $y \cdot 2^{r_b} = y \cdot 2^{60}$.

(c) Insert S_1 into hash table H_3 indexed by the 4 inactive cells $Z_{29}[5, 10, 11, 13]$ of C_1 and 4 inactive cells of C_2. For each element of S_2, we check the hash table H_3 to find the element in S_1 where (C_1, C_3) and (C_2, C_4) collide in the 8 inactive cells. So there are $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^{88}$ quartets as (C_1, C_2, C_3, C_4), which can be used to conduct the tweakey recovery for the 68-bit subtweakey involved in E_f. Call the tweakey recovery process to check whether the guessed tweakey is correct.

Tweakey recovery.

1. **In round 29:** for the 4th column of X_{29} of (C_1, C_3), we obtain $\Delta W_{28}[11] = \Delta X_{29}[7] \oplus \Delta X_{29}[15] = 0$ according to Eq.(9). Since STK_{29} is only xored to the first two rows of the internal state, $X_{29}[15]$ is determined by the ciphertext. Thereafter, the input difference $\Delta X_{29}[7]$ and output difference $\Delta Y_{29}[7]$ of the `SC` operation are known. By Lemma 1, we can deduce $STK_{29}[7]$, which has one solution on average. Similarly, we also deduce $STK'_{29}[7]$ for (C_2, C_4). Since the difference $\Delta STK_{29}[7]$ is fixed, we get a 4-bit filter. $y^2 \cdot 2^{88} \cdot 2^{-4} = y^2 \cdot 2^{84}$ quartets remain.

2. For the 1st column of X_{29} of (C_1, C_3), we obtain $\Delta W_{28}[12] = \Delta X_{29}[0] \oplus \Delta X_{29}[12] = 0$ according to Eq.(9). With known $\Delta X_{29}[0]$ and $\Delta Y_{29}[0]$, we deduce $STK_{29}[0]$ by Lemma 1. Similarly, we deduce $STK'_{29}[0]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{84} \cdot 2^{-4} = y^2 \cdot 2^{80}$ quartets remain.

3. For the 3rd column of X_{29} of (C_1, C_3), we obtain $\Delta W_{28}[10] = \Delta X_{29}[6] \oplus \Delta X_{29}[14] = 0$ from Eq.(9). With known $\Delta X_{29}[6]$ and $\Delta Y_{29}[6]$, we deduce $STK_{29}[6]$ by Lemma 1. Similarly, we deduce $STK'_{29}[6]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{80} \cdot 2^{-4} = y^2 \cdot 2^{76}$ quartets remain.

4. Now we have deduced $STK_{29}[0, 6, 7]$. Guessing $STK_{29}[2]$, we compute $W_{28}[14]$ and $X_{28}[15]$. $\Delta X_{28}[15] = \texttt{0x2}$ is a 4-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{76} \cdot 2^{-8} = y^2 \cdot 2^{68}$ quartets remain.

5. Guessing $STK_{29}[4]$, we compute $W_{28}[4]$. Since $\Delta X_{28}[7] = \texttt{0x2}$ and $\Delta Y_{28}[7]$ is determined by $W_{28}[4]$, we deduce $STK_{28}[7]$ for (C_1, C_3) and $STK'_{28}[7]$ for (C_2, C_4) by Lemma 1, which acts as a 4-bit filter. $y^2 \cdot 2^{68} \cdot 2^{-4} = y^2 \cdot 2^{64}$ quartets remain.

6. Guess $STK_{29}[1, 3, 5]$ and compute Z_{28} to peel off round 29.

7. **In round 28:** for the 1st column of X_{28} of (C_1, C_3), we obtain $\Delta X_{28}[0] = \Delta X_{28}[4] = \Delta X_{28}[12]$ from Eq.(9). With known $\Delta X_{28}[0, 4]$ and $\Delta Y_{28}[0, 4]$, we deduce $STK_{28}[0, 4]$ by Lemma 1. Similarly, we deduce $STK'_{28}[0, 4]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{64} \cdot 2^{-8} = y^2 \cdot 2^{56}$ quartets remain.

8. For the 3rd column of X_{28} of (C_1, C_3), we obtain $\Delta X_{28}[2] = \Delta X_{28}[10] = \Delta X_{28}[14]$ from Eq.(9). With known $\Delta X_{28}[2]$ and $\Delta Y_{28}[2]$, we deduce $STK_{28}[2]$. Similarly, we deduce $STK'_{28}[2]$ for (C_2, C_4), which acts as a 4-bit filter. In addition, $\Delta X_{28}[10] = \Delta X_{28}[14]$ is a 4-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{56} \cdot 2^{-4} \cdot 2^{-8} = y^2 \cdot 2^{44}$ quartets remain.

9. Now we have deduced $STK_{28}[0, 2, 4, 7]$. We guess $STK_{28}[3, 5, 6]$ and compute Z_{27} to peel off round 28.

10. **In round 27:** for the 1st column of X_{27} of (C_1, C_3), we obtain $\Delta X_{27}[0] = \Delta X_{27}[8] = \Delta X_{27}[12]$ from Eq.(9). With known $\Delta X_{27}[0]$ and $\Delta Y_{27}[0]$, we deduce $STK_{27}[0]$. Similarly, we deduce $STK'_{27}[0]$ for (C_2, C_4), which acts as a 4-bit filter. In addition, $\Delta X_{27}[8] = \Delta X_{27}[8]$ is a 4-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{44} \cdot 2^{-4} \cdot 2^{-8} = y^2 \cdot 2^{32}$ quartets remain.

11. **In round 26:** Guessing $STK_{27}[4]$, we compute Z_{27} to X_{26}. $\Delta X_{26} = \texttt{0xa}$ is a 4-bit filter for both (C_1, C_3) and (C_2, C_4). Therefore, $y^2 \cdot 2^{32} \cdot 2^{-8} = y^2 \cdot 2^{24}$ quartets remain.
12. Output the top 2^{68-h} counters for the candidates and exhaustively search the other $(k - m_b - m_f) = 48$ unknown key bits to check whether the guessed key is correct.

Table 8: Time complexity and guessed subtweakeys of each step in the tweakey recovery process for 30-round `SKINNY-64-192`. `D:` deduced key cells; `G:` guessed key cells.

Step i	Time complexity T_i	Involved subtweakeys
1	$y^2 \cdot 2^{88} \cdot \frac{4}{30} = y^2 \cdot 2^{85.1}$	`D:` $STK_{29}[7]$
2	$y^2 \cdot 2^{84} \cdot \frac{4}{30} = y^2 \cdot 2^{81.1}$	`D:` $STK_{29}[0]$
3	$y^2 \cdot 2^{80} \cdot \frac{4}{30} = y^2 \cdot 2^{77.1}$	`D:` $STK_{29}[6]$
4	$y^2 \cdot 2^{76} \cdot 2^4 \cdot \frac{4}{30} = y^2 \cdot 2^{77.1}$	`G:` $STK_{29}[2]$
5	$y^2 \cdot 2^{68} \cdot 2^4 \cdot 2^4 \cdot \frac{4}{30} = y^2 \cdot 2^{73.1}$	`G:` $STK_{29}[4]$, `D:` $STK_{28}[7]$
6	$y^2 \cdot 2^{64} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot \frac{4}{30} = y^2 \cdot 2^{81.1}$	`G:` $STK_{29}[1,3,5]$
7	$y^2 \cdot 2^{64} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot \frac{4}{30} = y^2 \cdot 2^{81.1}$	`D:` $STK_{28}[0,4]$
8	$y^2 \cdot 2^{56} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot \frac{4}{30} = y^2 \cdot 2^{73.1}$	`D:` $STK_{28}[2]$
9	$y^2 \cdot 2^{44} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot 2^{12} \cdot \frac{4}{30} = y^2 \cdot 2^{73.1}$	`G:` $STK_{28}[3,5,6]$
10	$y^2 \cdot 2^{44} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot 2^{12} \cdot \frac{4}{30} = y^2 \cdot 2^{73.1}$	`D:` $STK_{27}[0]$
11	$y^2 \cdot 2^{32} \cdot 2^4 \cdot 2^4 \cdot 2^{12} \cdot 2^{12} \cdot 2^4 \cdot \frac{4}{30} = y^2 \cdot 2^{65.1}$	`G:` $STK_{27}[4]$
	$\sum_i T_i \approx y^2 \cdot 2^{85.36}$	

In the above steps, we totally guess 9 cells of key and $y^2 \cdot 2^{24}$ remain. Therefore, we obtain totally $y^2 \cdot 2^{24} \cdot 2^{36} = y^2 \cdot 2^{60}$ key counters. Since there are totally 68-bit key involved in E_f, the counter under each 68-bit key is about $y^2 \cdot 2^{60}/2^{68} = y^2 \cdot 2^{-8}$ on average.

Complexity. The data complexity is $D = 4M = 4 \cdot y \cdot 2^{r_b} = y \cdot 2^{62}$ chosen plaintexts. The memory complexity is $5M + 2^{m_f} = y \cdot 2^{62.32} + 2^{68}$.

In the `Data collection` process, one needs $4M = y \cdot 2^{62}$ encryptions in step 2 and $2^{m_b} \cdot 3M = y \cdot 2^{137.58}$ table look-ups in step 3(b) and 3(c). In the `Tweakey recovery` process, one needs about $2^{m_b} \cdot y^2 \cdot 2^{85.36} + 2^{192-h} = y^2 \cdot 2^{161.36} + 2^{192-h}$ encryptions according to Table 8. Totally, the time complexity is $y \cdot 2^{62} + y \cdot 2^{137.58} + y^2 \cdot 2^{161.36} + 2^{192-h}$.

Set the expected number of right quartets $s = 1$ and $y = \sqrt{s} \cdot 2^{n/2 - r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1} = 2^{0.87}$. The data complexity is $2^{62.87}$ and the memory complexity is $2^{68.05}$. Set the advantage $h = 36$ and the time complexity is about $2^{163.11}$. Let the signal-to-noise be $S_N = 2^{-n} \cdot \tilde{p}^2\tilde{q}^2 t / 2^{-2n}$, based on the theoretical analysis by Selcuk [SB02], the success probability is about $P_s = \Phi(\frac{\sqrt{sS_N} - \Phi^{-1}(1-2^{-h})}{\sqrt{S_N + 1}}) = 62.3\%$.

5.2 The Key-recovery attack on 24-round `SKINNY-64-128`

We give a 24-round key-recovery rectangle attack on `SKINNY-64-128` by adding 2 rounds before and 4 rounds after the 18-round distinguisher, where

$$\begin{aligned} \alpha &= (\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{2}), \\ \delta &= (\texttt{0},\texttt{0},\texttt{d},\texttt{0},\texttt{0},\texttt{0},\texttt{d},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{0},\texttt{d},\texttt{0}). \end{aligned}$$

The probability of the distinguisher is $2^{-n} \cdot \tilde{p}^2\tilde{q}^2 t = 2^{-64-55.34} = 2^{-119.34}$. The data collection phase is similar to Sect. 5.1. As shown in Fig. 8, We construct $y = \sqrt{s} \cdot 2^{n/2-r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1}$ structures by traversing $r_b = 4c = 16$ inactive bits in W_0' [3]. $ETK_0[4, 6, 9, 12]$ marked by north west lines are involved in E_b. Due to $ETK_0[4] = ETK_0[12] = STK_0[0]$ according to Fig. 4, there has $m_b = 3c = 12$. In addition, we know $r_f = 12c = 48$ and $m_f = 21c = 84$ where $c = 4$.

[3]Note that we do not traverse the cell with fixed difference $W_0'[0]$ while building structures.

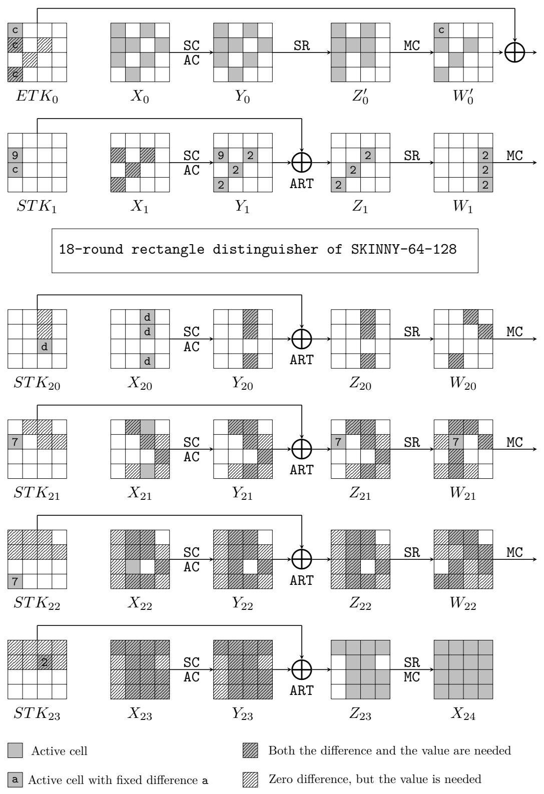


Figure 8: The 24-round attack against SKINNY-64-128

According to Zhao *et al.*'s attack procedures, for each guessing of $2^{m_b} = 2^{12}$ possible values in E_b, there are about $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2$ quartets (C_1, C_2, C_3, C_4) remaining. We give the detailed processes to recover $m_f = 84$ key bits in E_f. For each quartet remaining, do (for briefness, take Z_{23} as the ciphertext C):

1. **In round 23:** for the 3rd column of X_{23} of (C_1, C_3), we obtain $\Delta X_{23}[2] = \Delta X_{23}[6] = \Delta X_{23}[14]$ due to Eq.(9). Since STK_{23} is only xored to the first two rows of the internal state, $X_{23}[14]$ is determined by the ciphertext. With known $\Delta X_{23}[2,6]$ and $\Delta Y_{23}[2,6]$, we deduce $STK_{23}[2,6]$ with Lemma 1. Similarly, we deduce $STK'_{23}[2,6]$ for (C_2, C_4). Since the differences $\Delta STK_{23}[2,6]$ are fixed, we get an 8-bit filter. $y^2 \cdot 2^{-8}$ quartets remain.

2. For the 2nd column of X_{23} of (C_1, C_3), we obtain $\Delta X_{23}[5] = \Delta X_{23}[9] \oplus \Delta X_{23}[13]$. With known $\Delta X_{23}[5]$ and $\Delta Y_{23}[5]$, we deduce $STK_{23}[5]$. Similarly, we deduce $STK'_{23}[5]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{-8} \cdot 2^{-4} = y^2 \cdot 2^{-12}$ quartets remain.

3. For the 4th column of X_{23} of (C_1, C_3), we obtain $\Delta X_{23}[3] = \Delta X_{23}[15]$. With known $\Delta X_{23}[3]$ and $\Delta Y_{23}[3]$, we deduce $STK_{23}[3]$. Similarly, we deduce $STK'_{23}[3]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{-12} \cdot 2^{-4} = y^2 \cdot 2^{-16}$ quartets remain.

4. Guessing $STK_{23}[1,7]$, we compute the 2nd, 3rd and 4th columns of W_{22}. For the 3rd column of X_{22} of (C_1, C_3), we obtain $\Delta X_{22}[2] = \Delta X_{22}[6] = \Delta X_{22}[14]$. Thereafter, we compute $\Delta X_{22}[14]$ from $W_{22}[13]$ and $\Delta Y_{22}[2,6]$ from $W_{22}[2,7]$ and deduce $STK_{22}[2,6]$. Similarly, we deduce $STK'_{22}[2,6]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{-16} \cdot 2^{-8} = y^2 \cdot 2^{-24}$ quartets remain.

5. Guess $STK_{23}[0,4]$ and compute W_{22} to peel off round 23. For the 2nd column of X_{22} of (C_1, C_3), we obtain $\Delta X_{22}[5] = \Delta X_{22}[9] \oplus \Delta X_{22}[13] \oplus \texttt{0x7}$. With known $\Delta X_{22}[5]$ and $\Delta Y_{22}[5]$, we deduce $STK_{22}[5]$. Similarly, we deduce $STK'_{22}[5]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{-24} \cdot 2^{-4} = y^2 \cdot 2^{-28}$ quartets remain.

6. **In round 22:** Guess $STK_{22}[1,7]$, we compute the 2nd, 3rd and 4th of W_{21}. For the 3rd column of X_{21} of (C_1, C_3), we obtain $\Delta X_{21}[2] = \Delta X_{21}[6] = \Delta X_{21}[14]$. Thereafter, we compute $\Delta X_{21}[14]$ from $W_{21}[13]$ and $\Delta Y_{21}[2,6]$ from $W_{21}[2,7]$ and deduce $STK_{21}[2,6]$. Similarly, we deduce $STK'_{21}[2]$ for (C_2, C_4), which acts as an 8-bit filter. Thereafter, we compute $W_{20}[2]$ from $X_{20}[6]$. Since $\Delta X_{20}[2] = \texttt{0xd}$, we deduce $STK_{20}[2]$ for (C_1, C_3) and $STK'_{20}[2]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{-28} \cdot 2^{-8} \cdot 2^{-4} = y^2 \cdot 2^{-40}$ quartets remain.

7. Guess $STK_{22}[0,4]$ to peel off round 22.

8. **In round 21:** Guessing $STK_{21}[1]$, we compute $W_{20}[13]$ and $X_{20}[14]$. $\Delta X_{20}[14] = \texttt{0xd}$ acts as a 4-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{-40} \cdot 2^{-8} = y^2 \cdot 2^{-48}$ quartets remain.

9. Guessing $STK_{21}[7]$, we compute $W_{20}[7]$. Since $\Delta X_{20}[6] = \texttt{0xd}$, we deduce $STK_{20}[6]$ for (C_1, C_3) and $STK'_{20}[6]$ for (C_2, C_4), which acts as a 4-bit filter. $y^2 \cdot 2^{-48} \cdot 2^{-4} = y^2 \cdot 2^{-52}$ quartets remain.

10. Output the top 2^{84-h} counters for the candidates and exhaustively search the other $(k - m_b - m_f) = 32$ unknown key bits to check whether the guessed key is correct.

Table 9: Time complexity and guessed subtweakeys of each step in the `tweakey recovery` process for 24-round `SKINNY-64-128`. `D:` deduced key cells; `G:` guessed key cells.

Step i	Time complexity T_i	Involved subtweakeys
1	$y^2 \cdot \frac{4}{24} = y^2 \cdot 2^{-2.58}$	`D:` $STK_{23}[2,6]$
2	$y^2 \cdot 2^{-8} \cdot \frac{4}{24} = y^2 \cdot 2^{-10.58}$	`D:` $STK_{23}[5]$
3	$y^2 \cdot 2^{-12} \cdot \frac{4}{24} = y^2 \cdot 2^{-14.58}$	`D:` $STK_{23}[3]$
4	$y^2 \cdot 2^{-16} \cdot 2^8 \cdot \frac{4}{24} = y^2 \cdot 2^{-10.58}$	`G:` $STK_{23}[1,7]$, `D:` $STK_{22}[2,6]$
5	$y^2 \cdot 2^{-24} \cdot 2^8 \cdot 2^8 \cdot \frac{4}{24} = y^2 \cdot 2^{-10.58}$	`G:` $STK_{23}[0,4]$, `D:` $STK_{22}[5]$
6	$y^2 \cdot 2^{-28} \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot \frac{4}{24} = y^2 \cdot 2^{-6.58}$	`G:` $STK_{22}[1,7]$, `D:` $STK_{21}[2,6]$
7	$y^2 \cdot 2^{-40} \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot \frac{4}{24} = y^2 \cdot 2^{-10.58}$	`G:` $STK_{22}[0,4]$
8	$y^2 \cdot 2^{-40} \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot 2^4 \cdot \frac{4}{24} = y^2 \cdot 2^{-6.58}$	`G:` $STK_{21}[1]$
9	$y^2 \cdot 2^{-48} \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot 2^8 \cdot 2^4 \cdot 2^4 \cdot \frac{4}{24} = y^2 \cdot 2^{-10.58}$	`G:` $STK_{21}[7]$, `D:` $STK_{20}[6]$
	$\sum_i T_i \approx y^2 \cdot 2^{-2.51}$	

As shown in Table 9, we guess 10 cells of keys and $y^2 \cdot 2^{-52}$ quartets remain in the above steps. Therefore, we obtain $y^2 \cdot 2^{-52} \cdot 2^{40} = y^2 \cdot 2^{-12}$ key counters. Since there are 84-bit key involved in E_f, the counter under each guessed 84-bit key is about $y^2 \cdot 2^{-12}/2^{84} = y^2 \cdot 2^{-96}$ on average.

Complexity. Setting the expected number of right quarters $s = 1$ and the advantage $h = 30$, $y = \sqrt{s} \cdot 2^{n/2 - r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1} = 2^{43.67}$. The data complexity is $2^{61.67}$, the memory complexity is 2^{84} and the time complexity is about $2^{96.83}$. The success probability is about 75.8%.

5.3 The Key-recovery Attack on 30-round SKINNY-128-384

We give a 30-round key-recovery rectangle attack on SKINNY-128-384 by adding 4 rounds before and 4 rounds after the 22-round distinguisher, where

$$\begin{aligned}\alpha &= (\mathtt{81,00,00,00,00,00,00,01,00,00,00,00,00,00,00,00,}),\\ \delta &= (\mathtt{00,00,00,00,00,00,00,00,00,50,00,00,00,00,00,00}).\end{aligned}$$

The probability of the distinguisher is $2^{-n}\cdot\tilde{p}^2\tilde{q}^2t = 2^{-128-101.49} = 2^{-229.49}$. As shown in Fig. 9, we have $r_b = 15c$, $m_b = 19c$, $r_f = 13c$ and $m_f = 16c$. The data collection phase is nearly the same to Sect. 5.1 and we construct $y = \sqrt{s}\cdot 2^{n/2-r_b}\cdot(\tilde{p}\tilde{q}\sqrt{t})^{-1}$ structures.

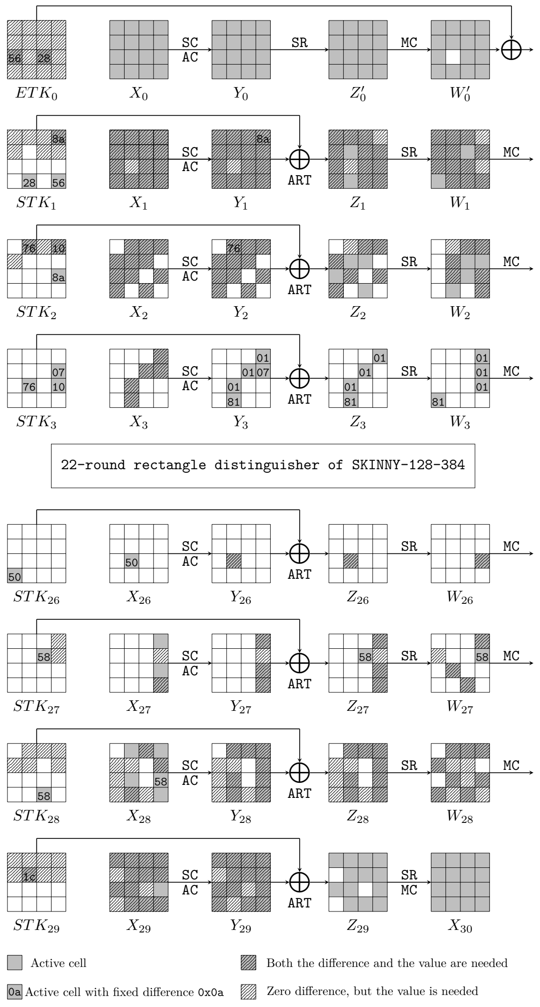


Figure 9: The 30-round attack against SKINNY-128-384

According to Zhao *et al.*'s attack procedures, for each guessing of $2^{m_b} = 2^{152}$ possible values in E_b, there are about $y^2\cdot 2^{2r_b}\cdot 2^{-2(n-r_f)} = y^2\cdot 2^{192}$ quartets remaining. We give the detailed process to recover m_f key bits for E_f. For each quartet remaining, do:

1. **In round 29:** for the 2nd column of X_{29} of (C_1, C_3), we obtain $\Delta X_{29}[1] = \Delta X_{29}[13]$ and $\Delta X_{29}[5] = \Delta X_{29}[9]\oplus\Delta X_{29}[13]$ according to Eq.(9). Since STK_{29} is only xored to the first two rows of the internal state, $X_{29}[9]$ and $X_{29}[13]$ are determined by the ciphertext. With known $\Delta X_{29}[1,5]$ and $\Delta Y_{29}[1,5]$, we deduce $STK_{29}[1,5]$. Similarly,

we deduce $STK'_{29}[1,5]$ for (C_2, C_4). Since the difference $\Delta STK_{29}[1,5]$ is fixed, we get a 16-bit filter. In round 28, since $STK_{29}[1,5]$ is known, we compute $W_{28}[9]$ and $X_{28}[11]$. $\Delta X_{28}[11] = \texttt{0x58}$ acts as an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{192} \cdot 2^{-16} \cdot 2^{-16} = y^2 \cdot 2^{160}$ quartets remain.

2. For the 3rd column of X_{29} of (C_1, C_3), we obtain $\Delta X_{29}[6] = \Delta X_{29}[14]$. With known $\Delta X_{29}[6]$ and $\Delta Y_{29}[6]$,we deduce $STK_{29}[6]$. Similarly, we deduce $STK'_{29}[6]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{160} \cdot 2^{-8} = y^2 \cdot 2^{152}$ quartets remain.

3. For the 4th column of X_{29} of (C_1, C_3), the case is the same to step 1. We deduce $STK_{29}[3,7]$ for (C_1, C_3) and $STK'_{29}[3,7]$ for (C_2, C_4), which acts as a 16-bit filter. $y^2 \cdot 2^{152} \cdot 2^{-16} = y^2 \cdot 2^{136}$ quartets remain.

4. Guess $STK_{29}[0,2,4]$ and compute W_{28} to peel off round 29.

5. **In round 28:** for the 2nd column of X_{28} of (C_1, C_3), we obtain $\Delta X_{28}[1] = \Delta X_{28}[9] = \Delta X_{28}[13]$. With known $\Delta X_{28}[1]$ and $\Delta Y_{28}[1]$, we deduce $STK_{28}[1]$. Similarly, we deduce $STK'_{28}[1]$ for (C_2, C_4), which is an 8-bit filter. $\Delta X_{28}[9] = \Delta X_{28}[13]$ is an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{136} \cdot 2^{-8} \cdot 2^{-16} = y^2 \cdot 2^{112}$ quartets remain.

6. For 4th column of X_{28} of (C_1, C_3), we obtain $\Delta X_{28}[3] = \Delta X_{28}[7] = \Delta X_{28}[15]$. With known $\Delta X_{28}[3,7]$ and $\Delta Y_{28}[3,7]$, we deduce $STK_{28}[3,7]$. Similarly, we deduce $STK'_{28}[3,7]$ for (C_2, C_4), which acts as a 16-bit filter. $y^2 \cdot 2^{112} \cdot 2^{-16} = y^2 \cdot 2^{96}$ quartets remain.

7. Guess $STK_{28}[2,4,5]$ and compute W_{27} to peel off round 28.

8. **In round 27:** for 4th column of X_{27} of (C_1, C_3), we obtain $\Delta X_{27}[3] = \Delta X_{27}[11] = \Delta X_{27}[15]$. With known $\Delta X_{27}[3]$ and $\Delta Y_{27}[3]$, we deduce $STK_{27}[3]$. Similarly, we deduce $STK'_{27}[3]$ for (C_2, C_4), which acts as an 8-bit filter. $\Delta X_{27}[11] = \Delta X_{27}[15]$ acts as an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{96} \cdot 2^{-8} \cdot 2^{-16} = y^2 \cdot 2^{72}$ quartets remain.

9. Guessing $STK_{27}[7]$, we compute $W_{26}[7]$. Thereafter $\Delta X_{26}[9] = \texttt{0x50}$ acts as an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{72} \cdot 2^{-16} = y^2 \cdot 2^{56}$.

Table 10: Time complexity and guessed subtweakeys of each step in the tweakey recovery process for 30-round `SKINNY-128-384`. `D:` deduced key cells; `G:` guessed key cells.

Step i	Time complexity T_i	Involved subtweakeys
1	$y^2 \cdot 2^{192} \cdot \frac{4}{30} = y^2 \cdot 2^{189.1}$	`D:` $STK_{29}[1,5]$
2	$y^2 \cdot 2^{160} \cdot \frac{4}{30} = y^2 \cdot 2^{157.1}$	`D:` $STK_{29}[6]$
3	$y^2 \cdot 2^{152} \cdot \frac{4}{30} = y^2 \cdot 2^{149.1}$	`D:` $STK_{29}[3,7]$
4	$y^2 \cdot 2^{136} \cdot 2^{24} \cdot \frac{4}{30} = y^2 \cdot 2^{157.1}$	`G:` $STK_{29}[0,2,4]$
5	$y^2 \cdot 2^{136} \cdot 2^{24} \cdot \frac{4}{30} = y^2 \cdot 2^{157.1}$	`D:` $STK_{28}[1]$
6	$y^2 \cdot 2^{112} \cdot 2^{24} \cdot \frac{4}{30} = y^2 \cdot 2^{133.1}$	`D:` $STK_{28}[3,7]$
7	$y^2 \cdot 2^{96} \cdot 2^{24} \cdot 2^{24} \cdot \frac{4}{30} = y^2 \cdot 2^{141.1}$	`G:` $STK_{28}[2,4,5]$
8	$y^2 \cdot 2^{96} \cdot 2^{24} \cdot 2^{24} \cdot \frac{4}{30} = y^2 \cdot 2^{141.1}$	`D:` $STK_{27}[3]$
9	$y^2 \cdot 2^{72} \cdot 2^{24} \cdot 2^{24} \cdot 2^{8} \cdot \frac{4}{30} = y^2 \cdot 2^{125.1}$	`G:` $STK_{27}[7]$
	$\sum_i T_i \approx y^2 \cdot 2^{189.1}$	

As shown in Table 10, we totally guess 7 cells of key and $y^2 \cdot 2^{56}$ quartets remain in the above steps. Therefore, we obtain totally $y^2 \cdot 2^{56} \cdot 2^{56} = y^2 \cdot 2^{112}$ key counters. Since there are totally 128-bit key involved in E_f, the counter under each 128-bit key is about $y^2 \cdot 2^{112}/2^{128} = y^2 \cdot 2^{-16}$ on average.

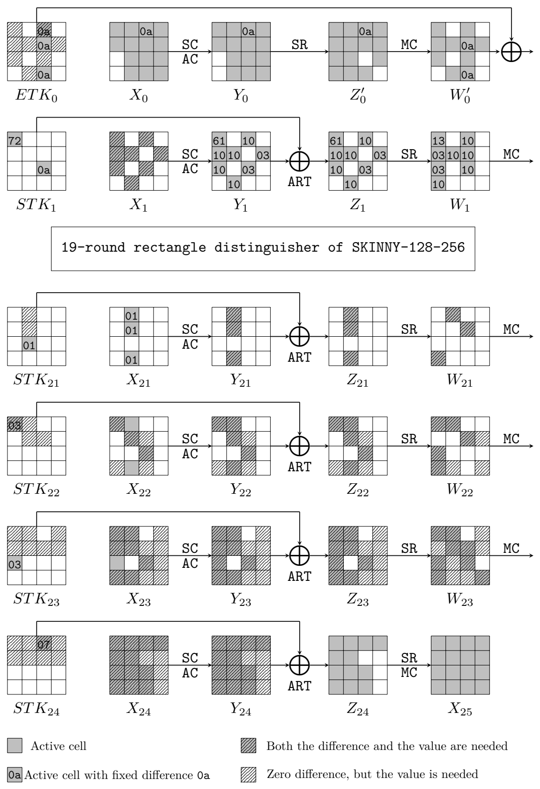
Complexity. Setting the expected number of right quarters $s = 1$ and the advantage $h = 56$, we choose $y = 1$. The data complexity is 2^{122}, the memory complexity is $2^{128.02}$ and the time complexity is about $2^{341.1}$. The success probability is about 84.1%.

5.4 The Key-recovery Attack on 25-round `SKINNY-128-256`

We give a 25-round key-recovery rectangle attack on `SKINNY-128-256` by adding 2 rounds before and 4 rounds after the 19-round distinguisher, where

$$\begin{aligned}\alpha &= (\mathtt{00, 00, 00, 00, 13, 00, 10, 00, 00, 10, 00, 00, 10, 00, 00, 00}),\\ \delta &= (\mathtt{00, 01, 00, 00, 00, 01, 00, 00, 00, 00, 00, 00, 00, 01, 00, 00}).\end{aligned}$$

The probability of the rectangle distinguisher is $2^{-n} \cdot \tilde{p}^2\tilde{q}^2t = 2^{-128-116.97} = 2^{-244.97}$. As shown in Fig. 10, we have $r_b = 8c$ and $m_b = 6c$. In addition, we know $r_f = 12c$ according to ΔZ_{24} and $m_f = 21c$ where $c = 8$. The data collection process is similar to Sect. 5.2 and we construct $y = \sqrt{s} \cdot 2^{n/2-r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1}$ structures.



SC AC
SR
MC
ART
ETK_0 X_0 Y_0 Z'_0 W'_0
STK_1 X_1 Y_1 Z_1 W_1
19-round rectangle distinguisher of SKINNY-128-256
STK_{21} X_{21} Y_{21} Z_{21} W_{21}
STK_{22} X_{22} Y_{22} Z_{22} W_{22}
STK_{23} X_{23} Y_{23} Z_{23} W_{23}
STK_{24} X_{24} Y_{24} Z_{24} X_{25}
Active cell
Active cell with fixed difference 0a
Both the difference and the value are needed
Zero difference, but the value is needed

Figure 10: The 25-round attack against `SKINNY-128-256`

According to Zhao *et al.*'s attack procedures, for each guessing of $2^{m_b} = 2^{48}$ possible values in E_b, there are about $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^{64}$ quartets as (C_1, C_2, C_3, C_4) remaining. We give the detailed process to recover m_f key bits for E_f. For each quartet remaining, do:

1. **In round 24:** for the 2nd column of X_{24} of (C_1, C_3), we obtain $\Delta X_{24}[1] = \Delta X_{24}[5] = \Delta X_{24}[13]$ according to Eq.(9). Since STK_{24} is only xored to the first two rows of the internal state, $X_{24}[13]$ is determined by the ciphertext. With known $\Delta X_{24}[1, 5]$ and $\Delta Y_{24}[1, 5]$, we deduce $STK_{24}[1, 5]$ with Lemma 1. Similarly, we deduce $STK'_{24}[1, 5]$ for

(C_2, C_4) and the pre-fixed $\Delta STK_{24}[1,5]$ acts as a 16-bit filter. $y^2 \cdot 2^{64} \cdot 2^{-16} = y^2 \cdot 2^{48}$ quartets remain.

2. For the 3rd column of X_{24} of (C_1, C_3), we obtain $\Delta X_{24}[2] = \Delta X_{24}[14]$. With known $\Delta X_{24}[2]$ and $\Delta Y_{24}[2]$, we can deduce $STK_{24}[2]$. Similarly, we deduce $STK'_{24}[2]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{48} \cdot 2^{-8} = y^2 \cdot 2^{40}$ quartets remain.

3. For the 1st column, of X_{24} of (C_1, C_3), we obtain $\Delta X_{24}[4] = \Delta X_{24}[8] \oplus \Delta X_{24}[12]$. With known $\Delta X_{24}[4]$ and $\Delta Y_{24}[4]$, we can deduce $STK_{24}[4]$. Similarly, we deduce $STK'_{24}[4]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{40} \cdot 2^{-8} = y^2 \cdot 2^{32}$ quartets remain.

4. Guessing $STK_{24}[0,6]$, we compute the first three columns of W_{23}. For the 2nd column of X_{23} of (C_1, C_3), we obtain $\Delta X_{23}[1] = \Delta X_{23}[5] = \Delta X_{23}[13]$. Thereafter, we compute $\Delta X_{23}[13]$ from $W_{23}[12]$ and $\Delta Y_{23}[1,5]$ from $W_{23}[1,6]$ and deduce $STK_{23}[1,5]$. Similarly, we deduce $STK'_{23}[1,5]$ for (C_2, C_4), which acts as a 16-bit filter. $y^2 \cdot 2^{32} \cdot 2^{-16} = y^2 \cdot 2^{16}$ quartets remain.

5. Guessing $STK_{24}[3,7]$, we compute the 4th column of W_{23} and peel off round 24. For the 1st column of X_{23} of (C_1, C_3), we obtain $\Delta X_{23}[4] = \Delta X_{23}[8] \oplus \Delta X_{23}[12]$. Thereafter, we compute $\Delta X_{23}[8,12]$ from $W_{23}[10,15]$ and $\Delta Y_{23}[4]$ from $W_{23}[5]$ and deduce $STK_{23}[4]$. Similarly, we deduce $STK'_{23}[4]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{16} \cdot 2^{-8} = y^2 \cdot 2^{8}$ quartets remain.

6. **In round 23:** Guess $STK_{23}[0,6]$, we compute the first three columns of W_{22}. For the 2nd column of X_{22} of (C_1, C_3), we obtain $\Delta X_{22}[1] = \Delta X_{22}[5] = \Delta X_{22}[13]$. Thereafter, we compute $\Delta X_{22}[13]$ from $W_{22}[12]$ and $\Delta Y_{22}[1,5]$ from $W_{22}[1,6]$ and deduce $STK_{22}[1,5]$. Similarly, we deduce $STK'_{22}[1,5]$ for (C_2, C_4), which acts as a 16-bit filter. Then in round 22, we can compute $W_{21}[1]$ from $X_{22}[5]$. Due to $\Delta X_{21}[1] = \texttt{0x01}$, we deduce $STK_{21}[1]$ for (C_1, C_3) and $STK'_{21}[1]$ for (C_2, C_4). which acts as an 8-bit filter. $y^2 \cdot 2^{8} \cdot 2^{-16} \cdot 2^{-8} = y^2 \cdot 2^{-16}$ quartets remain.

7. Guess $STK_{23}[3,7]$ and compute W_{22} to peel off round 23.

8. **In round 22:** Guessing $STK_{22}[0]$, we compute $W_{21}[12]$ and $X_{21}[13]$. **In round 21**, $\Delta X_{21}[13] = \texttt{0x01}$ acts as an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{-16} \cdot 2^{-16} = y^2 \cdot 2^{-32}$ quartet remain.

9. Guessing $STK_{22}[6]$, we compute $W_{21}[6]$. Since $\Delta X_{21}[5] = \texttt{0x01}$, we deduce $STK_{21}[5]$ for (C_1, C_3) and $STK'_{21}[5]$ for (C_2, C_4). which acts as an 8-bit filter. $y^2 \cdot 2^{-32} \cdot 2^{-8} = y^2 \cdot 2^{-40}$ quartet remain.

Table 11: Time complexity and guessed subtweakeys of each step in the tweakey recovery process for 25-round `SKINNY-128-256`. `D:` deduced key cells; `G:` guessed key cells.

Step i	Time complexity T_i	Involved subtweakeys
1	$y^2 \cdot 2^{64} \cdot \frac{4}{25} = y^2 \cdot 2^{61.4}$	`D:` $STK_{24}[1,5]$
2	$y^2 \cdot 2^{48} \cdot \frac{4}{25} = y^2 \cdot 2^{45.4}$	`D:` $STK_{24}[2]$
3	$y^2 \cdot 2^{40} \cdot \frac{4}{25} = y^2 \cdot 2^{37.4}$	`D:` $STK_{24}[4]$
4	$y^2 \cdot 2^{32} \cdot 2^{16} \cdot \frac{4}{25} = y^2 \cdot 2^{45.4}$	`G:` $STK_{24}[0,6]$, `D:` $STK_{23}[1,5]$
5	$y^2 \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot \frac{4}{25} = y^2 \cdot 2^{45.4}$	`G:` $STK_{24}[3,7]$, `D:` $STK_{23}[4]$
6	$y^2 \cdot 2^{8} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot \frac{4}{25} = y^2 \cdot 2^{53.4}$	`G:` $STK_{23}[0,6]$, `D:` $STK_{22}[1,5]$, $STK_{21}[1]$
7	$y^2 \cdot 2^{-16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot \frac{4}{25} = y^2 \cdot 2^{45.4}$	`G:` $STK_{23}[3,7]$
8	$y^2 \cdot 2^{-16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{8} \cdot \frac{4}{25} = y^2 \cdot 2^{53.4}$	`G:` $STK_{23}[0]$
9	$y^2 \cdot 2^{-32} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{16} \cdot 2^{8} \cdot 2^{8} \cdot \frac{4}{25} = y^2 \cdot 2^{45.4}$	`G:` $STK_{22}[6]$, `D:` $STK_{21}[5]$
	$\sum_i T_i \approx y^2 \cdot 2^{61.41}$	

As shown in Table 11, we totally guess 10 cells of key combining with remaining $y^2 \cdot 2^{-40}$ quartets in the above steps. We obtain $y^2 \cdot 2^{-40} \cdot 2^{80} = y^2 \cdot 2^{40}$ key counters. Since $m_f = 168$, the key counter for each key is $y^2 \cdot 2^{40}/2^{168} = y^2 \cdot 2^{-128}$ on average.

Complexity. Set the expected number of right quarters $s = 1$ and the advantage $h = 30$, $y = \sqrt{s} \cdot 2^{n/2-r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1} = 2^{58.48}$. The data complexity is $2^{124.48}$, the memory complexity is 2^{168} and the time complexity is about $2^{226.38}$. The success probability is about 80.7%.

6 Application to ForkSkinny

ForkSkinny is a 2nd round candidate in the NIST lightweight authenticated encryption standardization process. It is a Forkcipher designed by Andreeva *et al.* [ALP+19b] under the ForkAE lightweight authenticated encryption framework. A Forkcipher

$$F : \mathcal{K} \times \mathcal{T} \times \{0,1\}^n \to \{0,1\}^{2n}$$

takes n-bit plaintext, and generates a $2n$-bit ciphertext under a given key K and tweak T

$$F_K(p, T) = (c_0, c_1).$$

The primitive is based on the lightweight tweakable block cipher SKINNY, and there are four different instances with variant block sizes and tweakey sizes (see Table 12).

Table 12: Instances of ForkSkinny.

ForkSkinny-[blocksize]-[tweakey]	tweak	$R_{\texttt{init}}$	$R_{\texttt{I}}$	$R_{\texttt{II}}$
ForkSkinny-64-192	64	17	23	23
ForkSkinny-128-192	64	21	27	27
ForkSkinny-128-256	128	21	27	27
ForkSkinny-128-288	128	25	31	31

The construction of ForkSkinny is shown in Fig 11. The encryption of ForkSkinny is split into two steps. The first $R_{\texttt{init}}$ rounds process the input message with the round function of SKINNY under modified constants. Then, the encryption procedure is forked into ForkSkinny$_0$ and ForkSkinny$_1$, where two copies of the output from the first stage are separately processed by the two forks with $R_{\texttt{I}}$ and $R_{\texttt{II}}$ rounds, respectively. The tweakeys are generated by the tweakey schedule for $R_{\texttt{init}} + R_{\texttt{I}} + R_{\texttt{II}}$ rounds in total, and used sequentially in the initial step, ForkSkinny$_0$ and ForkSkinny$_1$. For instance, the last $R_{\texttt{II}}$ round tweakeys are applied in ForkSkinny$_1$.

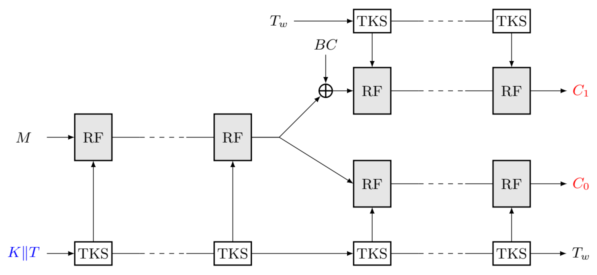


Figure 11: The ForkSkinny framework [ALP+19b].

We use a similar tweaked model to search boomerang distinguishers for the encryption from plaintext M to C_1 in Fig. 11, where only the tweakey schedule is slightly different.

For ForkSkinny-128-256, the subtweakeys used are $STK_0, STK_1, \ldots, STK_{20}$, STK_{48}, $STK_{49}, \ldots, STK_{74}$. As pointed out in [BDL20], there are some master key differences δs that satisfy $\delta = LFSR2^{15}(\delta)$. So we can get differential characteristics with 6 consecutive inactive subtweakeys $STK_{18}, STK_{19}, STK_{20}$, STK_{48}, STK_{49} and STK_{50}. We take advantage of these properties and add constraints to the model in Sect. 4.2.

At ToSC 2020, Bariant *et al.* [BDL20] gave the related-key impossible differential attack on 26(=7+19)-round reduced ForkSkinny-128-256, where $R_{\texttt{init}} = 7$, $R_{\text{I}} = 27$ and $R_{\text{II}} = 19$. We select a 21(=17+4)-round related-tweakey rectangle distinguisher for ForkSkinny-128-256, where $R_{\texttt{init}} = 17$, $R_{\text{I}} = 27$ and $R_{\text{II}} = 4$. The probability of the 4-round middle part is $2^{-29.72}$, which is tested with a data complexity 2^{35} and time of $237.3s$ on one computer equipped with one RTX 2080 Ti. The details of the distinguisher can refer to Table 13 and Table 20. Using the 21-round distinguisher, we give a 28-round key-recovery attack on ForkSkinny-128-256 with 256-bit key ($R_{\texttt{init}} = 20$, $R_{\text{I}} = 27$ and $R_{\text{II}} = 8$), and a 25-round key-recovery attack on ForkSkinny-128-256 with 128-bit key ($R_{\texttt{init}} = 18$, $R_{\text{I}} = 27$ and $R_{\text{II}} = 7$).

Table 13: The 21-round boomerang distinguisher for ForkSkinny-128-256

$r_0 = 8$, $r_m = 4$, $r_1 = 9$, $\tilde{p} = 2^{-30.25}$, $t = 2^{-30.31}$, $\tilde{q} = 2^{-7.48}$, $\tilde{p}^2\tilde{q}^2t = 2^{-105.77}$
$\Delta TK1 =$ 00, 00, e8, 00, 00, 00, 00, 00, 00, 00, 00, 00, 50, 00, 00, 00
$\Delta TK2 =$ 00, 00, fa, 00, 00, 00, 00, 00, 00, 00, 00, 00, aa, 00, 00, 00
$\Delta X_0 =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 88
$\nabla TK1 =$ 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 19
$\nabla TK2 =$ 00, 00, 00, 00, 00, 00, 00, cc, 00, 00, 00, 00, 00, 00, 00, af
$\nabla X_{21} =$ 00, 00, 00, 00, 00, 00, 00, 00, 35, 00, 00, 00, 00, 00, 00, 00

6.1 The Attack on 28-round ForkSkinny-128-256 with 256-bit Key

We give a 28-round key-recovery attack on ForkSkinny-128-256 with 256-bit key by adding 3 rounds before and 4 rounds after the 21-round distinguisher, as shown in Fig. 12. The probability of the distinguisher is $2^{-n} \cdot \tilde{p}^2\tilde{q}^2t = 2^{-128-105.77} = 2^{-233.77}$. We have $r_b = 8c$ and $m_b = 10c$. In addition, there are $r_f = 12c$ according to ΔZ_{27} and $m_f = 17c = 136$ where $c = 8$. The data collection process is similar to Sect. 5.2 and we construct $y = \sqrt{s} \cdot 2^{n/2-r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1}$ structures.

According to Zhao *et al.*'s attack procedures, for each guessing of $2^{m_b} = 2^{80}$ possible values in E_b, there are about $y^2 \cdot 2^{2r_b} \cdot 2^{-2(n-r_f)} = y^2 \cdot 2^{64}$ quartets as (C_1, C_2, C_3, C_4) remaining. We give the detailed process to recover m_f key bits for E_f. For each quartet remaining, do:

1. **In round 27:** for the 1st column of X_{27} of (C_1, C_3), we obtain $\Delta X_{27}[4] = \Delta X_{27}[12]$ according to Eq.(9). $\Delta X_{27}[12]$ is determined since STK_{54} is only xored to the first two rows of the internal state. With known $\Delta X_{27}[4]$ and $\Delta Y_{27}[4]$, we deduce $STK_{54}[4]$ with Lemma 1. Similarly, for (C_2, C_4) we also deduce $STK'_{54}[4]$ and the pre-fixed difference $\Delta STK_{54}[4]$ acts as an 8-bit filter. $y^2 \cdot 2^{64} \cdot 2^{-8} = y^2 \cdot 2^{56}$ quartets remain.

2. For the 2nd column of X_{27}, similar to step 1, we deduce $STK_{54}[5]$ for (C_1, C_3) and $STK'_{54}[5]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{56} \cdot 2^{-8} = y^2 \cdot 2^{48}$ quartets remain.

3. For the 3rd column of X_{27} of (C_1, C_3), we obtain $\Delta X_{27}[2] = \Delta X_{27}[14]$. Therefore $\Delta X_{27}[2]$ is determined, and we deduce $STK_{54}[2]$. Similarly, we can also deduce $STK'_{54}[2]$ for (C_2, C_4) and obtain an 8-bit filter. $y^2 \cdot 2^{48} \cdot 2^{-8} = y^2 \cdot 2^{40}$ quartets remain.

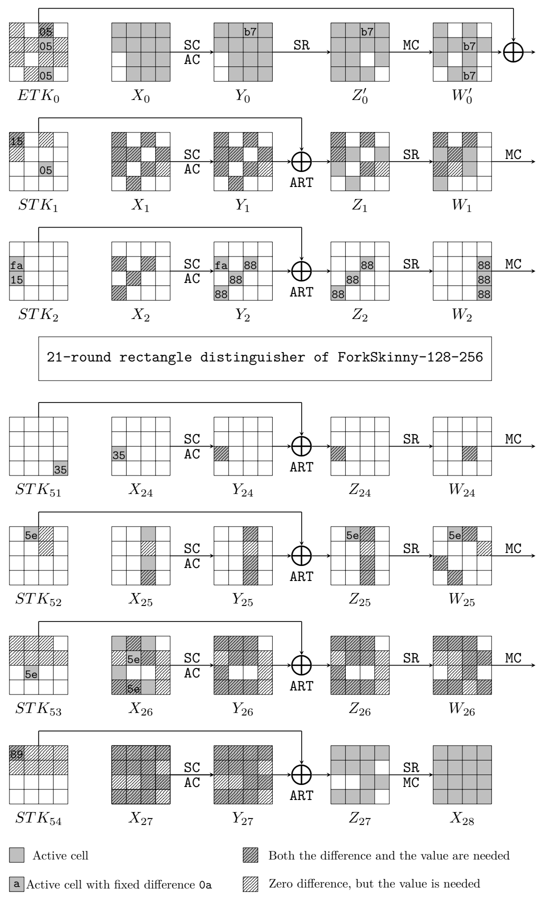


Figure 12: The 28-round attack against `ForkSkinny-128-256` (256-bit key)

4. Guessing $STK_{54}[0]$, the first column of W_{26} is computed. Thereafter, we can compute $\Delta X_{26}[13]$ from $W_{26}[12]$ for both (C_1, C_3) and (C_2, C_4) in round 26. Since $\Delta X_{26}[13] =$ `0x5e`, we get a 16-bit filter. $y^2 \cdot 2^{40} \cdot 2^{-16} = y^2 \cdot 2^{24}$ quartets remain.

5. Guessing $STK_{54}[6]$, the 3rd column of W_{26} is computed. Thereafter, we compute $\Delta Y_{26}[5]$ from $W_{26}[6]$ for both (C_1, C_3) and (C_2, C_4) in round 26. Since $\Delta X_{26}[5] =$ `0x5e`, we deduce $STK_{53}[5]$ for (C_1, C_3) and $STK'_{53}[5]$ for (C_2, C_4), which acts as an 8-bit filter. $y^2 \cdot 2^{24} \cdot 2^{-8} = y^2 \cdot 2^{16}$ quartets remain.

6. Guess $STK_{54}[1, 3, 7]$ and compute Z_{26} to peel off the round 27.

7. **In round 26:** for the 1st column of X_{26} of (C_1, C_3), we obtain $\Delta X_{26}[0] = \Delta X_{26}[8] = \Delta X_{26}[12]$. Therefore $\Delta X_{26}[0]$ is determined and we deduce $STK_{53}[0]$. Similarly, we deduce $STK'_{53}[0]$ for (C_2, C_4), which acts as an 8-bit filter. $\Delta X_{26}[8] = \Delta X_{26}[12]$ is an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{16} \cdot 2^{-8} \cdot 2^{-16} = y^2 \cdot 2^{-8}$ quartets remain.

8. For the 3rd round of X_{26} of (C_1, C_3), we obtain $\Delta X_{26}[2] = \Delta X_{26}[6] = \Delta X_{26}[14]$. Therefore, $\Delta X_{26}[2]$ and $\Delta X_{26}[6]$ are determined and we deduce $STK_{53}[2, 6]$. Similarly, we deduce $STK'_{53}[2, 6]$ for (C_2, C_4), which acts as a 16-bit filter. $y^2 \cdot 2^{-8} \cdot 2^{-16} = y^2 \cdot 2^{-24}$ quartets remain.

9. Guess $STK_{53}[1, 4, 7]$ and compute Z_{25} to peel off round 26.

10. **In round 25:** for 3rd round of X_{25} of (C_1, C_3), we obtain $\Delta X_{25}[2] = \Delta X_{25}[10] = \Delta X_{25}[14]$. Therefore $\Delta X_{25}[2]$ is determined and we deduce $STK_{52}[2]$. Similarly, we deduce $STK'_{52}[2]$ for (C_2, C_4), which acts as an 8-bit filter. $\Delta X_{25}[10] = \Delta X_{25}[14]$ is an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{-24} \cdot 2^{-8} \cdot 2^{-16} = y^2 \cdot 2^{-48}$ quartets remain.

11. **In round 24:** Guessing $STK_{52}[6]$, we compute $W_{24}[10]$. Thereafter, $X_{24}[8]$ is computed and $\Delta X_{24}[8] =$ `0x35` acts as an 8-bit filter for both (C_1, C_3) and (C_2, C_4). $y^2 \cdot 2^{-48} \cdot 2^{-16} = y^2 \cdot 2^{-64}$ quartets remain.

Table 14: Time complexity, guessed and deduced subtweakeys of each step in the tweakey recovery process for 28-round `ForkSkinny-128-256` (256-bit key).

Step i	Time complexity T_i	Involved subtweakeys
1	$y^2 \cdot 2^{64} \cdot \frac{4}{28} = y^2 \cdot 2^{61.19}$	`D:` $STK_{54}[4]$
2	$y^2 \cdot 2^{56} \cdot \frac{4}{28} = y^2 \cdot 2^{53.19}$	`D:` $STK_{54}[5]$
3	$y^2 \cdot 2^{48} \cdot \frac{4}{28} = y^2 \cdot 2^{45.19}$	`D:` $STK_{54}[2]$
4	$y^2 \cdot 2^{40} \cdot 2^{8} \cdot \frac{4}{28} = y^2 \cdot 2^{45.19}$	`G:` $STK_{54}[0]$
5	$y^2 \cdot 2^{24} \cdot 2^{8} \cdot 2^{8} \cdot \frac{4}{28} = y^2 \cdot 2^{37.19}$	`G:` $STK_{54}[6]$, `D:` $STK_{53}[5]$
6	$y^2 \cdot 2^{16} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot \frac{4}{28} = y^2 \cdot 2^{53.19}$	`G:` $STK_{54}[1,3,7]$
7	$y^2 \cdot 2^{16} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot \frac{4}{28} = y^2 \cdot 2^{53.19}$	`D:` $STK_{53}[0]$
8	$y^2 \cdot 2^{-8} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot \frac{4}{28} = y^2 \cdot 2^{29.19}$	`D:` $STK_{53}[2,6]$
9	$y^2 \cdot 2^{-24} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot 2^{24} \cdot \frac{4}{28} = y^2 \cdot 2^{37.19}$	`G:` $STK_{53}[1,4,7]$
10	$y^2 \cdot 2^{-24} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot 2^{24} \cdot \frac{4}{28} = y^2 \cdot 2^{37.19}$	`D:` $STK_{52}[2]$
11	$y^2 \cdot 2^{-48} \cdot 2^{8} \cdot 2^{8} \cdot 2^{24} \cdot 2^{24} \cdot 2^{8} \cdot \frac{4}{28} = y^2 \cdot 2^{21.19}$	`G:` $STK_{52}[6]$
	$\sum_i T_i \approx y^2 \cdot 2^{61.21}$	

As shown in Table 14, we totally guess 9 cells of the key combining with remaining $y^2 \cdot 2^{-64}$ quartets in the above steps, . We obtain $y^2 \cdot 2^{-64} \cdot 2^{72} = y^2 \cdot 2^{8}$ key counters. Since $m_f = 136$, the key counter for each key is $y^2 \cdot 2^{8}/2^{136} = y^2 \cdot 2^{-128}$ on average.

Complexity. Setting the expected number of right quarters $s = 1$ and the advantage $h = 10$, $y = \sqrt{s} \cdot 2^{n/2 - r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1} = 2^{52.88}$. The data complexity is $2^{118.88}$, the memory complexity is 2^{136} and the time complexity is about $2^{246.98}$. The success probability is about 83%.

6.2 The Attack on 25-round `ForkSkinny-128-256` with 128-bit Key

By adding 1 round before and 3 rounds after the 21-round distinguisher, we can attack 25-round `ForkSkinny-128-256` with 128-bit key and 128-bit tweak. There are $r_b = 0c$, $m_b = 0c$, $r_f = 8c$ and $m_f = 8c$. The attack procedure is similar with previous attack in Sect. 6.1 and we omit it here. Setting the expected number of right quarters $s = 1$ and the advantage $h = 20$, $y = \sqrt{s} \cdot 2^{n/2 - r_b} \cdot (\tilde{p}\tilde{q}\sqrt{t})^{-1} = 2^{116.88}$. The data complexity is $4M = 2^{118.88}$, the memory complexity is $5M + 2^{m_f} \approx 2^{119.2}$ and the time complexity is about $4M + 2^{m_b} \cdot y^2 \cdot 2^{2r_b - 2(n - r_f)} \cdot \frac{4}{25} \approx 2^{118.88}$. The success probability is about 83.5%.

7 Discussion and Conclusion

We give a uniform automatic MILP model of related-tweakey rectangle attacks on `SKINNY` and `ForkSkinny`, which includes the key-recovery attack process and the related-tweakey rectangle distinguisher. In the model, we balance the probability of a distinguisher and the dominating factors of the key-recovery process, such as the guessed key bits. Hence, we give the improved related-tweakey rectangle attacks on a few versions of round-reduced

SKINNY and ForkSkinny, which cover 1-2 more rounds than the best previous ones. We would like to state that according to the discussion[4] on the LWC forum, our rectangle attacks on round-reduced SKINNY block cipher do not impact the security of SKINNY-AEAD.

Acknowledgments.

We would like to thank the anonymous reviewers from ToSC for their detailed comments. This paper was supported by the National Key Research and Development Program of China (Grant No. 2018YFA0704701), the Major Program of Guangdong Basic and Applied Research (Grant No. 2019B030302008), the National Natural Science Foundation of China (Nos. 62072270, 61902207 and 61902414), Shandong Province Key Research and Development Project (Nos. 2020ZLYS09 and 2019JZZY010133), and National Cryptography Development Fund (Nos. MMJJ20180101 and MMJJ20170121).

References

[ABC+17] 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 Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi, editors, *Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings*, volume 10355 of *Lecture Notes in Computer Science*, pages 208–228. Springer, 2017.

[ADG+19] Ralph Ankele, Christoph Dobraunig, Jian Guo, Eran Lambooij, Gregor Leander, and Yosuke Todo. Zero-correlation attacks on tweakable block ciphers with linear tweakey expansion. *IACR Transactions on Symmetric Cryptology*, 2019(1):192–235, 2019.

[ALP+19a] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. ForkAE v. *Submission to NIST Lightweight Cryptography Project*, 2019.

[ALP+19b] Elena Andreeva, Virginie Lallemand, Antoon Purnal, Reza Reyhanitabar, Arnab Roy, and Damian Vizár. Forkcipher: A new primitive for authenticated encryption of very short messages. In *Advances in Cryptology - ASIACRYPT 2019 - 25th International Conference on the Theory and Application of Cryptology and Information Security, Kobe, Japan, December 8-12, 2019, Proceedings, Part II*, pages 153–182, 2019.

[BC18] Christina Boura and Anne Canteaut. On the boomerang uniformity of cryptographic sboxes. *IACR Trans. Symmetric Cryptol.*, 2018(3):290–310, 2018.

[BDK01] Eli Biham, Orr Dunkelman, and Nathan Keller. The rectangle attack - rectangling the Serpent. In Birgit Pfitzmann, editor, *Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding*, volume 2045 of *Lecture Notes in Computer Science*, pages 340–357. Springer, 2001.

[BDK05] Eli Biham, Orr Dunkelman, and Nathan Keller. A related-key rectangle attack on the full KASUMI. In Bimal K. Roy, editor, *Advances in Cryptology*

[4]https://groups.google.com/a/list.nist.gov/g/lwc-forum/c/kCNjP0q64Bo?pli=1

- ASIACRYPT 2005, 11th International Conference on the Theory and Application of Cryptology and Information Security, Chennai, India, December 4-8, 2005, Proceedings, volume 3788 of *Lecture Notes in Computer Science*, pages 443–461. Springer, 2005.

[BDL20] Augustin Bariant, Nicolas David, and Gaëtan Leurent. Cryptanalysis of Forkciphers. *IACR Trans. Symmetric Cryptol.*, 2020(1):233–265, 2020.

[BJK$^+$16] 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. Springer, 2016.

[BJK$^+$20] 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. *IACR Transactions on Symmetric Cryptology*, 2020(1):88–121, 2020.

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In Mitsuru Matsui, editor, *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*, volume 5912 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2009.

[BN10] Alex Biryukov and Ivica Nikolic. Automatic search for related-key differential characteristics in byte-oriented block ciphers: Application to AES, Camellia, Khazad and others. In *Advances in Cryptology - EUROCRYPT 2010, 29th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Monaco / French Riviera, May 30 - June 3, 2010. Proceedings*, pages 322–344, 2010.

[BPP$^+$17] 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 Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 321–345. Springer, 2017.

[BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. *J. Cryptology*, 4(1):3–72, 1991.

[BSS$^+$13] 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.

[CHP$^+$17] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. A security analysis of Deoxys and its internal tweakable block ciphers. *IACR Trans. Symmetric Cryptol.*, 2017(3):73–107, 2017.

[CHP$^+$18] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *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, volume 10821 of *Lecture Notes in Computer Science*, pages 683–714. Springer, 2018.

[CSSH19] Qiu Chen, Danping Shi, Siwei Sun, and Lei Hu. Automatic demirci-selçuk meet-in-the-middle attack on SKINNY with key-bridging. In *Information and Communications Security - 21st International Conference, ICICS 2019, Beijing, China, December 15-17, 2019, Revised Selected Papers*, pages 233–247, 2019.

[DDV20] Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the fastest boomerangs application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.

[DF16] Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle and impossible differential attacks. In *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, pages 157–184, 2016.

[DFJ13] Patrick Derbez, Pierre-Alain Fouque, and Jérémy Jean. Improved key recovery attacks on reduced-round AES in the single-key setting. In *Advances in Cryptology - EUROCRYPT 2013, 32nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Athens, Greece, May 26-30, 2013. Proceedings*, pages 371–387, 2013.

[DHLP20] Orr Dunkelman, Senyang Huang, Eran Lambooij, and Stav Perle. Single tweakey cryptanalysis of reduced-round Skinny-64. In Shlomi Dolev, Vladimir Kolesnikov, Sachin Lodha, and Gera Weiss, editors, *Cyber Security Cryptography and Machine Learning - Fourth International Symposium, CSCML 2020, Be'er Sheva, Israel, July 2-3, 2020, Proceedings*, volume 12161 of *Lecture Notes in Computer Science*, pages 1–17. Springer, 2020.

[DKS10] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 393–410. Springer, 2010.

[FN20] Antonio Flórez-Gutiérrez and María Naya-Plasencia. Improving key-recovery in linear attacks: Application to 28-round PRESENT. In *Advances in Cryptology - EUROCRYPT 2020 - 39th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, May 10-14, 2020, Proceedings, Part I*, pages 221–249, 2020.

[HBS20] Hosein Hadipour, Nasour Bagheri, and Ling Song. Improved rectangle attacks on Skinny and CRAFT. *IACR Cryptol. ePrint Arch.*, 2020:1317, 2020.

[IKMP19] Tetsu Iwata, Mustafa Khairallah, Kazuhiko Minematsu, and Thomas Peyrin. Romulus v1. *Submission to NIST Lightweight Cryptography Project*, 2019.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In Palash Sarkar and Tetsu Iwata, editors, *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, volume 8874 of *Lecture Notes in Computer Science*, pages 274–288. Springer, 2014.

[KKS00] 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.

[KLT15] Stefan Kölbl, Gregor Leander, and Tyge Tiessen. Observations on the SIMON block cipher family. In Rosario Gennaro and Matthew Robshaw, editors, *Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I*, volume 9215 of *Lecture Notes in Computer Science*, pages 161–185. Springer, 2015.

[LGS17] Guozhen Liu, Mohona Ghosh, and Ling Song. Security analysis of Skinny under related-tweakey settings. *IACR Transactions on Symmetric Cryptology*, 2017(3):37–72, 2017.

[LS19] Yunwen Liu and Yu Sasaki. Related-key boomerang attacks on GIFT with automated trail search including BCT effect. In *Information Security and Privacy - 24th Australasian Conference, ACISP 2019, Christchurch, New Zealand, July 3-5, 2019, Proceedings*, pages 555–572, 2019.

[Mat93] Mitsuru Matsui. Linear cryptanalysis method for DES cipher. In Tor Helleseth, editor, *Advances in Cryptology - EUROCRYPT '93, Workshop on the Theory and Application of of Cryptographic Techniques, Lofthus, Norway, May 23-27, 1993, Proceedings*, volume 765 of *Lecture Notes in Computer Science*, pages 386–397. Springer, 1993.

[Mur11] Sean Murphy. The return of the cryptographic boomerang. *IEEE Transactions on Information Theory*, 57(4):2517–2521, 2011.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, pages 57–76, 2011.

[oSN20] National Institute of Standards and Technology (NIST). Lightweight cryptography (LWC) standardization process, 2020. https://csrc.nist.gov/Projects/Lightweight-Cryptography/Round-2-Candidates.

[SB02] Ali Aydin Selçuk and Ali Biçak. On probability of success in linear and differential cryptanalysis. In *Security in Communication Networks, Third International Conference, SCN 2002, Amalfi, Italy, September 11-13, 2002. Revised Papers*, volume 2576 of *Lecture Notes in Computer Science*, pages 174–185. Springer, 2002.

[SGL$^+$17] Siwei Sun, David Gérault, 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.

[SHW$^+$14] Siwei Sun, Lei Hu, Peng Wang, Kexin Qiao, Xiaoshuang Ma, and Ling Song. Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers. 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 I*, pages 158–178, 2014.

[SMB18] Sadegh Sadeghi, Tahereh Mohammadi, and Nasour Bagheri. Cryptanalysis of reduced round Skinny block cipher. *IACR Transactions on Symmetric Cryptology*, 2018(3):124–162, 2018.

[SQH19] Ling Song, Xianrui Qin, and Lei Hu. Boomerang connectivity table revisited. application to Skinny and AES. *IACR Transactions on Symmetric Cryptology*, 2019(1):118–141, 2019.

[SSD$^+$18] 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.

[TAY17] Mohamed Tolba, Ahmed Abdelkhalek, and Amr M. Youssef. Impossible differential cryptanalysis of reduced-round Skinny. In Marc Joye and Abderrahmane Nitaj, editors, *Progress in Cryptology - AFRICACRYPT 2017 - 9th International Conference on Cryptology in Africa, Dakar, Senegal, May 24-26, 2017, Proceedings*, volume 10239 of *Lecture Notes in Computer Science*, pages 117–134, 2017.

[Wag99] David A. Wagner. The boomerang attack. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.

[WP19] Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. application to AES variants and Deoxys. *IACR Transactions on Symmetric Cryptology*, 2019(1):142–169, 2019.

[YQC17] Dong Yang, Wen-Feng Qi, and Hua-Jin Chen. Impossible differential attacks on the Skinny family of block ciphers. *IET Inf. Secur.*, 11(6):377–385, 2017.

[ZDC$^+$21] Rui Zong, Xiaoyang Dong, Huaifeng Chen, Yiyuan Luo, Si Wang, and Zheng Li. Towards key-recovery-attack friendly distinguishers: Application to GIFT-128. *IACR Trans. Symmetric Cryptol.*, 2021(1):156–184, 2021.

[ZDJ19] Boxin Zhao, Xiaoyang Dong, and Keting Jia. New related-tweakey boomerang and rectangle attacks on Deoxys-BC including BDT effect. *IACR Trans. Symmetric Cryptol.*, 2019(3):121–151, 2019.

[ZDM$^+$20] Boxin Zhao, Xiaoyang Dong, Willi Meier, Keting Jia, and Gaoli Wang. Generalized related-key rectangle attacks on block ciphers with linear key schedule: applications to Skinny and GIFT. *Designs, Codes and Cryptography*, 88(6):1103–1126, 2020.

Supplementary Material

A Specification of SKINNY

A.1 Tweakey schedule of SKINNY

The round tweakey STK_i is defined as:

$$\begin{cases} t = n : STK_i = TK1_i, \\ t = 2n : STK_i = TK1_i \oplus TK2_i, \\ t = 3n : STK_i = TK1_i \oplus TK2_i \oplus TK3_i. \end{cases} \tag{15}$$

The tweakey arrays $TK1_i$, $TK2_i$ and $TK3_i$ in round i are generated as follows. First, apply the permutation $P = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7]$ on each TKm_{i-1} tweakey arrays:

$$TKm_i[n] \leftarrow TKm_{i-1}[P[n]], 0 \leq n \leq 15, m \in \{1, 2, 3\}.$$

Then, apply an LFSR to update each cell of the first and second rows of $TK2_i$ and $TK3_i$. The details of the LFSRs used in different versions are given in Table 15.

Table 15: The LFSRs used in the tweakey schedule of SKINNY

TKm	c	LFSR
$TK2$	4	$(x_3\|x_2\|x_1\|x_0) \rightarrow (x_2\|x_1\|x_0\|x_3 \oplus x_2)$
	8	$(x_7\|x_6\|x_5\|x_4\|x_3\|x_2\|x_1\|x_0) \rightarrow (x_6\|x_5\|x_4\|x_3\|x_2\|x_1\|x_0\|x_7 \oplus x_5)$
$TK3$	4	$(x_3\|x_2\|x_1\|x_0) \rightarrow (x_0 \oplus x_3\|x_3\|x_2\|x_1)$
	8	$(x_7\|x_6\|x_5\|x_4\|x_3\|x_2\|x_1\|x_0) \rightarrow (x_0 \oplus x_6\|x_7\|x_6\|x_5\|x_4\|x_3\|x_2\|x_1)$

A.2 The linear layer matrix of SKINNY

$$L = \begin{pmatrix}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0
\end{pmatrix} \tag{16}$$

$$L^{-1} = \begin{pmatrix}
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0\\
0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1\\
0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&1\\
0&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&1\\
1&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&1&0
\end{pmatrix} \tag{17}$$

$$L^{E} = \begin{pmatrix}
1&0&0&0&1&0&0&0&0&0&0&0&1&0&0&0\\
0&1&0&0&0&1&0&0&0&0&0&0&0&1&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&0&1&0\\
0&0&0&1&0&0&0&1&0&0&0&0&0&0&0&1\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&1&0&0&0&1&0\\
0&0&0&1&0&0&0&0&0&0&0&1&0&0&0&1\\
1&0&0&0&0&0&0&0&1&0&0&0&1&0&0&0\\
0&1&0&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0
\end{pmatrix} \tag{18}$$

B Boomerang Distinguishers of `SKINNY` and `ForkSkinny`

In this section, we list the differentials of the boomerang distinguishers searched in Sect. 4.2. For each round of the $(r_0+r_m+r_1)$-round distinguisher, we list the input/output differences of the S-box and subtweakey differences for r_0-round upper differential, as well as r_1-round lower differential. For r_m-round middle part of the distinguisher, we only present the input difference of the upper differential and the output difference of the lower differential. In the following table, the differences are given in hexadecimal, "*" denote arbitrary nonzero difference in the computation of the middle part, "-" denote arbitrary difference in the differential.

C Experiments on round-reduced boomerang Distinguishers of `SKINNY` and `ForkSkinny`

Due to the limited computing power, we make experiments on round-reduced boomerang distinguishers for `SKINNY`. The round-reduced distinguishers are listed in Table 22, Table 23,

Table 16: The differentials of the 18-round distinguisher for `Skinny-64-128`, where R8 to R13 is the $r_m = 6$-round middle part, u satisfies DDT[`0x5`][u] > 0 and DDT[$u \oplus$`0xc`][`0x6`] > 0, v satisfies DDT[`0x5`][v] > 0 and DDT[v][`0x6`] > 0.

	Upper differential	Lower differential
R0	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,2 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,5 0,0,5,0,0,0,0,0	
R1	0,0,0,0,0,0,5,0,0,0,0,0,0,0,5,0 0,0,0,0,0,0,u,0,0,0,0,0,0,0,v,0 0,0,0,0,0,0,c,0	
R2	0,v,0,0,0,0,0,0,0,0,0,$u \oplus$ 0xc,0,0,0,0 0,6,0,0,0,0,0,0,0,0,0,6,0,0,0,0 0,0,0,0,6,0,0,0	
R3	0,0,0,0,0,6,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,7,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,7,0,0	
R4/5	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,d,0,0	
R7	0,0,0,0,0,0,0,0,0,0,d,0,0,0,0,0 0,0,0,0,0,0,0,0,0,0,f,0,0,0,0,0 0,0,0,0,0,0,0,f	
R8	f,0,0,0,0,0,0,0,0,0,0,0,f,0,0,0 *,0,0,0,0,0,0,0,0,0,0,0,*,0,0,0 0,0,0,7,0,0,0,0	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,0
R9-R12	middle part	middle part
R13	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,5,0,0,0,0,0	0,*,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,6,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,6,0,0,0,0,0,0
R14/15/16		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
R17		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,d,0,0,0,0,0

Table 17: The differentials of the 22-round distinguisher for `SKINNY-64-192`, where R10 to R15 denote r_m = 6-round middle part, u satisfies DDT[`0xa`][u] > 0 and DDT[$u \oplus$ `0xd`][`0x2`] > 0, v satisfies DDT[`0xa`][v] > 0 and DDT[v][`0x2`] > 0.

	Upper differential	Lower differential
R0	3,0,0,0,0,0,0,2,0,0,0,0,0,0,0,0 d,0,0,0,0,0,0,6,0,0,0,0,0,0,0,0 d,0,0,0,0,0,0,6	
R1-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,a,0,0,0,0,0	
R6	0,0,a,0,0,0,a,0,0,0,0,0,0,0,a,0 0,0,6,0,0,0,u,0,0,0,0,0,0,0,v,0 0,0,6,0,0,0,d,0	
R7	0,v,0,0,0,0,0,0,0,0,0,$u \oplus$ 0xd,0,0,0,0 0,2,0,0,0,0,0,0,0,0,0,2,0,0,0,0 0,0,0,0,2,0,0,0	
R8	0,0,0,0,0,2,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,1,0,0,0,0,0,0,0,0,0,0 0,0,0,0,8,1,0,0	
R9	0,0,0,0,0,0,0,0,0,0,8,0,0,0,0,0 0,0,0,0,0,0,0,0,0,0,5,0,0,0,0,0 0,0,0,0,0,0,5,0	
R10	0,0,0,5,0,0,0,0,0,0,0,0,0,0,0,5 0,0,0,*,0,0,0,0,0,0,0,0,0,0,0,* 0,0,0,a,0,0,d,0	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,0
R11-R14	middle part	middle part
R15	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,f	0,0,*,0,0,0,0,0,0,0,0,0,0,0,0,0 0,0,a,0,0,0,0,0,0,0,0,0,0,0,0,0 0,0,a,0,0,0,0,0
R16-R20		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
R21		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,a,0,0

Table 18: The differentials of the 19-round distinguisher for `Skinny-128-256`, where R9 to R14 denote $r_m = 6$-round middle part, u satisfies DDT[`0x10`][u] > 0 and DDT[u][`0xd4`] > 0, v satisfies DDT[`0xd4`][v] > 0 and DDT[$v \oplus$ `0x08`][`0x98`] > 0, w satisfies DDT[`0xd4`][w] > 0 and DDT[w][`0x98`] > 0.

	Upper differential	Lower differential
R0	0,0,0,0,13,0,10,0,0,10,0,0,10,0,0,0 0,0,0,0,f4,0,u,0,0,u,0,0,u,0,0,0 0,0,0,0,f4,0,0,0	
R1	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,u 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,d4 0,0,d4,0,0,0,0,0	
R2	0,0,0,0,0,0,d4,0,0,0,0,0,0,0,d4,0 0,0,0,0,0,0,v,0,0,0,0,0,0,0,w,0 0,0,0,0,0,0,08,0	
R3	0,w,0,0,0,0,0,0,0,0,0,$v \oplus$0x08,0,0,0,0 0,98,0,0,0,0,0,0,0,0,0,98,0,0,0,0 0,0,0,0,98,0,0,0	
R4	0,0,0,0,0,98,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,f0,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,f0,0,0	
R5/6	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 0,0,0,0,0,31,0,0	
R8	0,0,0,0,0,0,0,0,0,0,31,0,0,0,0,0 0,0,0,0,0,0,0,0,0,0,e0,0,0,0,0,0 0,0,0,0,0,0,0,e0	
R9	e0,0,0,0,0,0,0,0,0,0,0,0,e0,0,0,0 *,0,0,0,0,0,0,0,0,0,0,0,*,0,0,0 0,0,0,52,0,0,0,0	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,0
R10-R13	middle part	middle part
R14	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,a2,0,0,0,0,0	0,0,0,*,0,0,0,0,0,0,0,0,0,0,0,0 0,0,0,80,0,0,0,0,0,0,0,0,0,0,0,0 0,0,0,80,0,0,0,0
R15/16/17		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
R18		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 0,01,0,0,0,0,0,0

Table 19: The differentials of the 22-round distinguisher for SKINNY-128-384, where R10 to R15 denote $r_m = 6$-round middle part, u satisfies DDT[0x07][u] > 0 and DDT[$u \oplus$ 0x22][0x8a] > 0, v satisfies DDT[0x07][v] > 0 and DDT[v][0x8a] > 0.

	Upper differential	Lower differential
R0	81,0,0,0,0,0,0,01,0,0,0,0,0,0,0,0 22,0,0,0,0,0,0,bb,0,0,0,0,0,0,0,0 22,0,0,0,0,0,0,bb	
R1-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 0,0,07,0,0,0,0,0	
R6	0,0,07,0,0,0,07,0,0,0,0,0,0,0,07,0 0,0,bb,0,0,0,u,0,0,0,0,0,0,0,v,0 0,0,bb,0,0,0,22,0	
R7	0,v,0,0,0,0,0,0,0,0,0,$u \oplus$ 0x22,0,0,0,0 0,8a,0,0,0,0,0,0,0,0,0,8a,0,0,0,0 0,0,0,0,8a,0,0,0	
R8	0,0,0,0,0,8a,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,76,0,0,0,0,0,0,0,0,0,0 0,0,0,0,10,76,0,0	
R9	0,0,0,0,0,0,0,0,0,10,0,0,0,0,0,0 0,0,0,0,0,0,0,0,0,50,0,0,0,0,0,0 0,0,0,0,0,0,50,0	
R10	0,0,0,50,0,0,0,0,0,0,0,0,0,0,0,50 0,0,0,*,0,0,0,0,0,0,0,0,0,0,0,* 0,0,0,56,0,0,28,0	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,0
R11-R14	middle part	middle part
R15	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,f0	0,*,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,50,0,0,0,0,0,0,0,0,0,0,0,0,0,0 0,50,0,0,0,0,0,0
R16-R20		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
R21		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 0,0,0,0,50,0,0,0

Table 20: The differentials of the 21-round distinguisher for `ForkSkinny-128-256`, where R8 to R11 is $r_m = 4$-round middle part, u satisfies DDT[`0xbe`][u] > 0 and DDT[$u \oplus$ `0xc6`][`0x2b`] > 0, v satisfies DDT[`0xbe`][v] > 0 and DDT[v][`0x2b`] > 0, w satisfies DDT[`0x44`][w] > 0 and DDT[w][`0x78`] > 0.

	Upper differential	Lower differential
R0	0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,88 0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,12 0,0,12,0,0,0,0,0	
R1	0,0,0,0,0,0,12,0,0,0,0,0,0,0,12,0 0,0,0,0,0,0,u,0,0,0,0,0,0,0,v,0 0,0,0,0,0,0,04,0	
R2	0,v,0,0,0,0,0,0,0,0,0,$u \oplus$0x04,0,0,0,0 0,1c,0,0,0,0,0,0,0,0,0,1c,0,0,0,0 0,0,0,0,1c,0,0,0	
R3	0,0,0,0,0,1c,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,f8,0,0,0,0,0,0,0,0,0,0 0,0,0,0,0,f8,0,0	
R4/5	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 0,0,0,0,0,38,0,0	
R7	0,0,0,0,0,0,0,0,0,0,0,38,0,0,0,0 0,0,0,0,0,0,0,0,0,0,0,f0,0,0,0,0 0,0,0,0,0,0,0,f0	
R8	f0,0,0,0,0,0,0,0,0,0,0,0,f0,0,0,0 *,0,0,0,0,0,0,0,0,0,0,0,*,0,0,0 0,0,0,49,0,0,0,0	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- 0,0,0,0,0,0,0,0
R9-R10	middle part	middle part
R11	-,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- -,-,-,-,-,-,-,-,-,-,-,-,-,-,-,- d0,0,0,0,0,0,0,0	0,0,0,0,0,*,0,*,0,0,*,0,0,*,0,0 0,0,0,0,0,d7,0,01,0,0,01,0,0,01,0,0 0,0,0,0,0,d7,0,0
R12		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,w,0,0,0 0,0,0,0,0,0,0,0
R13		0,0,0,w,0,0,0,0,0,0,0,0,0,0,0,0 0,0,0,9a,0,0,0,0,0,0,0,0,0,0,0,0 0,0,0,9a,0,0,0,0
R14-R19		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
R20		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 0,0,0,0,0,0,0,35

Table 24 and Table 25. The experimental results on one computer equipped with one RTX 2080 Ti are listed in Table 21.

Table 21: Experiments on the round-reduced boomerang distinguishers for `SKINNY`

Version	round	$\tilde{p}^2\tilde{q}^2t$	Probability	Complexity	Time
64-128	15	$2^{-35.96}$	$2^{-36.09}$	2^{40}	5h
64-192	15	$2^{-34.73}$	$2^{-34.69}$	2^{38}	1.9h
128-256	11	$2^{-40.39}$	2^{-41}	2^{42}	16h
128-384	13	$2^{-31.03}$	$2^{-31.17}$	2^{35}	0.3h

Table 22: The 15-round related-tweakey boomerang distinguisher for `SKINNY-64-128`

$r_0 = 5$, $r_m = 6$, $r_1 = 4$, $\tilde{p} = 2^{-6}$, $t = 2^{-23.96}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-35.96}$
$\Delta TK1 =$ 0, 0, 0, 0, 0, a, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0
$\Delta TK2 =$ 0, 0, 0, 0, 0, d, 0, 0, 0, 0, 0, 0, 2, 0, 0, 0
$\Delta X_0 =$ 0, 0, 0, 0, 0, 6, 0, 0, 0, 0, 0, 0, 0, 0, 0, 2
$\nabla TK1 =$ 0, 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla TK2 =$ 0, 0, 0, 0, f, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla X_{15} =$ 0, 0, d, 0, 0, 0, d, 0, 0, 0, 0, 0, 0, 0, d, 0

Table 23: The 15-round related-tweakey boomerang distinguisher for `SKINNY-64-192`

$r_0 = 3$, $r_m = 6$, $r_1 = 6$, $\tilde{p} = 2^{-8}$, $t = 2^{-18.73}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2t = 2^{-34.73}$
$\Delta TK1 =$ 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, c, 0, 0, 0, 8, 0
$\Delta TK2 =$ 0, 0, 0, 0, 8, 0, 0, 0, 0, 0, f, 0, 0, 0, e, 0
$\Delta TK3 =$ 0, 0, 0, 0, b, 0, 0, 0, 0, 0, 5, 0, 0, 0, b, 0
$\Delta X_0 =$ 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0, 0, 0
$\nabla TK1 =$ 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla TK2 =$ 0, 0, 0, d, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla TK3 =$ 0, 0, 0, 4, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0
$\nabla X_{15} =$ 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, a, 0, 0, 0, 0, 0

Table 24: The 11-round related-tweakey boomerang distinguisher for `SKINNY-128-256`

$r_0 = 1$, $r_m = 6$, $r_1 = 4$, $\tilde{p} = 2^{-4}$, $t = 2^{-32.39}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2 t = 2^{-40.39}$
$\Delta TK1 =$ `00, 00, 00, 00, 00, 00, 00, a0, 00, 00, 00, 00, 00, ef, 00, 00`
$\Delta TK2 =$ `00, 00, 00, 00, 00, 00, 00, 40, 00, 00, 00, 00, 00, de, 00, 00`
$\Delta X_0 =$ `00, 00, 00, 00, 00, 00, 00, 00, 00, 00, e0, 00, 00, 00, 00, 00`
$\nabla TK1 =$ `00, 00, 00, 00, ff, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00`
$\nabla TK2 =$ `00, 00, 00, 00, cf, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00`
$\nabla X_{11} =$ `00, 01, 00, 00, 00, 01, 00, 00, 00, 00, 00, 00, 00, 01, 00, 00`

Table 25: The 13-round related-tweakey boomerang distinguisher for `SKINNY-128-384`

$r_0 = 1$, $r_m = 6$, $r_1 = 6$, $\tilde{p} = 2^{-2}$, $t = 2^{-27.03}$, $\tilde{q} = 1$, $\tilde{p}^2\tilde{q}^2 t = 2^{-31.03}$
$\Delta TK1 =$ `00, 00, 00, 00, 00, 00, 56, 00, 00, 00, 00, 00, 1b, 14, 00, 00`
$\Delta TK2 =$ `00, 00, 00, 00, 00, 00, df, 00, 00, 00, 00, 00, 49, 61, 00, 00`
$\Delta TK3 =$ `00, 00, 00, 00, 00, 00, d9, 00, 00, 00, 00, 00, 42, 03, 00, 00`
$\Delta X_0 =$ `00, 00, 00, 00, 00, 00, 00, 00, 00, 10, 00, 00, 00, 00, 00, 00`
$\nabla TK1 =$ `00, 00, 00, 00, 00, 8a, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00`
$\nabla TK2 =$ `00, 00, 00, 00, 00, 87, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00`
$\nabla TK3 =$ `00, 00, 00, 00, 00, 33, 00, 00, 00, 00, 00, 00, 00, 00, 00, 00`
$\nabla X_{13} =$ `00, 00, 00, 00, 00, 00, 00, 00, 00, 50, 00, 00, 00, 00, 00, 00`