Security Analysis of SKINNY under Related-Tweakey Settings

Guozhen Liu[1,2], Mohona Ghosh[1] and Ling Song[1,3](✉)

[1] Nanyang Technological University, Singapore

[2] Shanghai JiaoTong University, China

[3] State Key Laboratory of Information Security,
Institute of Information Engineering, Chinese Academy of Sciences, China

liuguozhen@sjtu.edu.cn, {mohonaghosh,songling}@ntu.edu.sg

Abstract.
In CRYPTO'16, a new family of tweakable lightweight block ciphers - SKINNY was introduced. Denoting the variants of SKINNY as SKINNY-n-t, where n represents the block size and t represents the tweakey length, the design specifies $t \in \{n, 2n, 3n\}$. In this work, we evaluate the security of SKINNY against differential cryptanalysis in the related-tweakey model. First, we investigate truncated related-tweakey differential trails of SKINNY and search for longest impossible and rectangle distinguishers where there is only one active cell in the input and the output. Based on the distinguishers obtained, 19, 23 and 27 rounds of SKINNY-n-n, SKINNY-n-$2n$ and SKINNY-n-$3n$ can be attacked respectively. Moreover, actual differential trails for SKINNY under related-tweakey model are also explored and optimal differential trails of SKINNY-64 within certain number of rounds are searched with an indirect searching method based on Mixed-Integer Linear Programming. The results show a trend that as the number of rounds increases, the probability of optimal differential trails is much lower than the probability derived from lower bounds of active Sboxes in SKINNY.

Keywords: Block cipher · SKINNY · Impossible Differential Attack · Rectangle Attack · Related-Tweakey

1 Introduction

Ubiquitous computing is rapidly emerging as the new computing paradigm in information technology sector. The mass development and deployment of pervasive devices such as RFID tags, sensors, smartcards etc. promises many benefits such as lower implementation costs, optimized performance and increased efficiency. At the same time, these devices demand harsh costs constraints like lower memory availability, lower area requirements and power constraints. Ensuring strong security from cryptographic point of view under such circumstances becomes a striving issue. Lightweight cryptography is a field of cryptography which encompasses the current state-of-the-art cryptographic algorithms that are tailored for implementation in constrained environments and directly cater to the security concerns of low cost devices. With the growing interest of symmetric cryptographic community in this field, several lightweight variants of traditional cryptographic primitives such as *lightweight block ciphers* (PRESENT [BKL+07], LED [GPPR11], SIMON [BSS+13] etc.), *lightweight hash functions* (Spongent [BKL+11], Photon [GPP11], Quark [AHMN13] etc.) and *lightweight stream ciphers* (Mickey [BD08], Grain [HJMM08],

Trivium [CP08] etc.) have been proposed and studied in literature to address the design and security goals of lightweight ciphers.

In this work, we focus on the security analysis of SKINNY which is the latest addition to the list of lightweight block ciphers. Proposed by Beierle et al. in CRYPTO'16 [BJK+16b], the design of SKINNY has many attractive features. Firstly, its design can be seen as a first step towards bridging the gap between high operational efficiency vs. strong security. By careful analysis and thorough investigation, the designers of SKINNY show how non-optimal but very light internal crypto components can be combined together to provide a cipher which has competitive performance as well as strong security guarantees in both single key as well as related key settings. Secondly, inspired from the TWEAKEY framework [JNP14], SKINNY replaces its key input with a tweakey input. This provides the users of SKINNY an added advantage of enjoying the benefits of a tweakable block cipher.

The official SKINNY specification [BJK+16c] defines two block sizes, i.e., 64-bit and 128-bit. Depending on the block length n, the tweakey length t can be n, $2n$ or $3n$. Consequently, if we denote a variant of SKINNY as SKINNY-n-t, then the six variants of SKINNY are - SKINNY-64-64, SKINNY-64-128, SKINNY-64-192, SKINNY-128-128, SKINNY-128-256 and SKINNY-128-384. In the design document of SKINNY [BJK+16c], the designers provide a detailed security evaluation of SKINNY against the traditional block cipher cryptanalysis. However, apart from the differential and linear attacks, for which the lower bounds on the number of active S-boxes under the single key as well as related-tweakey settings have been provided, for other attack types such as MITM, impossible, integral attacks etc., the analysis has been restricted to single key model only. Moreover, in these attacks, only SKINNY-n-n variants have been investigated. This motivates us to analyze the security of all SKINNY variants under the related-tweakey model. We utilize related-tweakey impossible and rectangle attacks for our analysis.

Our Contribution. Our main results are summarized in Table 1. We focus on attacks with a data complexity below codebook. For most of our attacks, we employ truncated differential trails to serve our purpose. In all these results, we construct distinguishers where there is only one active cell in the input and the output difference. Moreover, the position of the active cell has been chosen such that maximum number of rounds can be extended in the forward and backward direction from the distinguisher. Under these conditions, the distinguishers so constructed are the longest. Based on truncated distinguishers, 19, 23 and 27 rounds of SKINNY-n-n, SKINNY-n-$2n$ and SKINNY-n-$3n$ can be attacked respectively. Additionally, extensions to the case where a tweak is used are discussed for SKINNY-n-$2n$ and SKINNY-n-$3n$ under impossible attacks.

Besides, actual differential trails for SKINNY under related-tweakey model are also explored. From the rectangle attacks based on actual trails it is shown that employing actual differential trails helps in getting longer distinguishers and sometimes getting better cryptanalytic results. What's more, optimal trails of SKINNY-64 within certain number of rounds show a positive trend for the designers that as the number of rounds increases, the probability of optimal differential trails is much lower than the probability derived from lower bounds of active Sboxes in SKINNY.

Recently, the designers of SKINNY announced a competition where they invited the cryptographic community to break SKINNY-64-128 and SKINNY-128-128 under the following categories [BJK+16a].

- SKINNY-64-128: 18- or 20- or 22- or 24- or 26-rounds
- SKINNY-128-128: 22- or 24- or 26- or 28- or 30-rounds

Through our attacks, we are able to cryptanalyze upto 23-rounds of SKINNY-64-128.

Table 1: Summary of cryptanalytic results on SKINNY

Vers.	Block Size	Rounds	Data†	Time	Memory	Dist. Type	Attack	Ref.
n-n	64	19	$2^{61.47}$	$2^{63.03}$	2^{56}	Trunc. Diff.	Imposs.	§ 3
	128	19	$2^{122.47}$	$2^{124.60}$	2^{112}	Trunc. Diff.	Imposs.	
n-2n	64	23	$2^{62.47}$	$2^{124.2}$	2^{124}	Trunc. Diff.	Imposs.	§ 3
	128	23	$2^{124.47}$	$2^{243.6}$	2^{248}	Trunc. Diff.	Imposs.	
n-3n	64	27	2^{63}	$2^{154.9}$	2^{80}	Trunc. Diff.	Rect.	§ 4
	128	27	$2^{126.5}$	2^{322}	2^{160}	Trunc. Diff.	Rect.	
	128	27	2^{112}	2^{300}	2^{144}	Diff.	Rect.	§ 5

† Attacks with a data complexity below codebook are focused.

Organization. This work is organized as follows. In Section 2, we provide a brief description of SKINNY and its properties, followed by the important notations adopted throughout the work. Section 3 gives details of the 23-round attack on SKINNY-n-2n and results on other SKINNY variants under related-tweakey impossible differential attack. Section 4 discusses the 22-round attack on SKINNY-n-2n using the related-tweakey rectangle attack and presents results on other variants. In the above sections, truncated differential trails are used for analysis. In Section 5, we investigate actual differential characteristics for SKINNY and present related-tweakey rectangle attacks based on them. Section 6 summarizes and concludes this paper.

2 Preliminaries

In this section, we first describe SKINNY and then mention the key notations and definitions used in our cryptanalysis to facilitate better understanding.

2.1 Description of SKINNY

The SKINNY block cipher adopts substitution-permutation network and elements of TWEAKEY framework [JNP14] in its design. Represented as SKINNY-n-t, where, n/ t denotes the block size/ tweakey size respectively, this block cipher has six variants namely - SKINNY-64-64, SKINNY-64-128, SKINNY-64/192, SKINNY-128/128, SKINNY-128/256 and SKINNY- 128/384. The number of rounds in each variant are 32, 36, 40, 40, 48 and 56 respectively. Both the 64-bit and 128-bit internal states are represented as 4×4 array of cells with each cell being a nibble in case of $n = 64$ bits and a byte in case of $n = 128$ bits. The tweakey state is seen as a group of z 4×4 arrays, where, $z = \frac{t}{n}$ and $z \in \{1, 2, 3\}$. The arrays are marked as TK-1 / TK-1, TK-2 / TK-1, TK-2, TK-3 for $z = 1$ / 2 / 3 respectively. [1] In all the cases, the cells are numbered row-wise as shown in Fig. 1. Each round consists of 5 basic operations as shown in Fig. 2.

1. *SubCells (SC)* - The non-linear substitution layer uses a 4-bit S-box in case of $n = 64$ bits and a 8-bit S-box in case of $n = 128$ bits. The maximal differential probability of both 4-bit and 8-bit S-boxes is 2^{-2}.

2. *AddConstants (AC)* - This step involves xoring three round constants to the first three cells of the first column of an internal state.

[1] The notations followed here are adopted from the original SKINNY document [BJK+16b] to facilitate better understanding in the subsequent sections.

0	1	2	3
4	5	6	7
8	9	10	11
12	13	14	15

Figure 1: Cell numbering in a state of Skinny

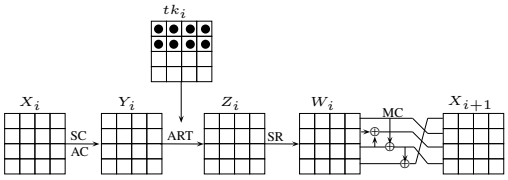


Figure 2: i^{th} round of SKINNY. Only the first two rows (cells marked with •) in the round tweakey tk_i are xor'ed in each round

3. *AddRoundTweakey (ART)* - In this step, the first two rows of the round tweakey (tk_i) are xor'ed with the first two rows of the corresponding internal state. The cells of the round tweakey are of the same size as the respective cell size of the internal state. The round tweakey (tk_i) is defined as:

   - $z = 1$: $tk_i = (TK\text{-}1)_i$
   - $z = 2$: $tk_i = (TK\text{-}1)_i \oplus (TK\text{-}2)_i$
   - $z = 3$: $tk_i = (TK\text{-}1)_i \oplus (TK\text{-}2)_i \oplus (TK\text{-}3)_i$

   The tweakeys $(TK\text{-}1)_i$, $(TK\text{-}2)_i$ and $(TK\text{-}3)_i$ for each round i are generated by a *tweakey scheduling algorithm.*

4. *ShiftRows (SR)* - The linear shift rows operation performs circular right shift on each row of the internal state. The number of shifts in each row j is j for $0 \leq j \leq 3$.

5. *MixColumns (MC)* - This linear transformation pre-multiplies each column of the internal state by a 4×4 binary matrix M shown below. The inverse MixColumns operation (M^{-1}) can be computed as shown below.

$$M = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix} \qquad M^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix}$$

Tweakey Scheduling Algorithm (TSA). - The tweakey schedule of SKINNY is a linear scheduling algorithm. The tweakey input is first loaded with a n, $2n$ or $3n$-bit tweakey input. Accordingly, we have TK-1 with $z = 1$ or (TK-1, TK-2) with $z = 2$ or (TK-1, TK-2, TK-3) with $z = 3$. The cells in each of these 4×4 TK-m arrays (for $m \in \{1, 2, 3\}$) are numbered row-wise. The round tweakeys are then generated using the tweakey scheduling algorithm as follows:

- *Permutation Phase*: In this phase, a permutation P defined as:

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

  is first applied to each of the TK-m arrays as: $TK\text{-}m[i] \leftarrow TK\text{-}m[P[i]]$ for all $0 \leq i \leq 15$ depending on $z = 1$, 2 or 3.

- *LFSR Updation Phase*: In this phase, all cells of the first two rows of TK-2 / (TK-2, TK-3) for $z = 2$ / 3 are individually updated using a 4-bit (if the cell is a nibble) or a 8-bit (if the cell is a byte) LFSR. Note that TK-1 is not updated in this phase.

The first two rows of each of the $(TK\text{-}m)_i$ arrays are used to generate the corresponding round tweakey tk_i as discussed earlier. This process is repeated until all round tweakeys have been generated. For complete details of the state updation process and the tweakey scheduling algorithm, one can refer [BJK+16b].

2.2 Properties of SKINNY

Some interesting properties of SKINNY that were utilized during our attacks are as follows:

1. The SKINNY MixColumns matrix is not an MDS matrix. Therefore, during the tweakey recovery phase of an attack, sometimes it is not enough to know the values of only the active cells in the output column of MC operation to determine the value of the active cell in the input column and more cells need to be guessed. To highlight this issue, consider Fig. 3.

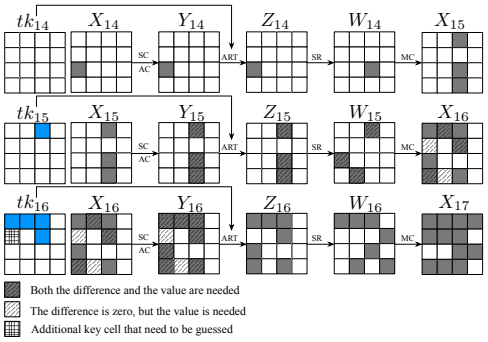


Figure 3: Property of *MixColumns* of SKINNY

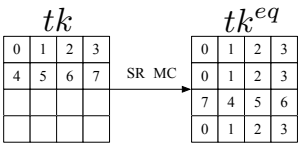


Figure 4: Reordering of ART, SR and MC operations in a round

   In this figure, we have a truncated differential trail with probability 1 from ΔX_{14} to ΔX_{17}. Now, if we backtrack this trail from ΔX_{17} and wish to know the difference at $\Delta X_{14}[8]$, it is necessary to determine the value as well as difference at $Y_{14}[8]$. This requires us to know the values as well as differences at $X_{15}[2, 10, 14]$ which in turn can be computed if we know values as well as differences in $Y_{15}[2, 10, 14] = W_{15}[2, 8, 13]$. Now, if suppose, values as well as differences in the active cells of the first column of X_{16} are known, only $\Delta W_{15}[8]$ can be computed. To compute $W_{15}[8]$, value of $X_{16}[4]$ needs to be known as well as $W_{15}[8] = X_{16}[4] \oplus X_{16}[12]$. This in turn leads to an additional tweakey cell guess in tk_{16} as $tk_{16}[4]$ needs to be guessed to determine $X_{16}[4]$.

   This property serves as one of the factors in determining the number of rounds covered in the tweakey recovery phase of our attacks.

2. The order of ART, SR and MC operations in any round can be changed by first applying SR and MC operations and then xoring the intermediate state with an equivalent round tweakey input. We denote this equivalent tweakey by $tk^{eq} = MC(SR(tk))$ as shown in Fig. 4.

3. The TSA of SKINNY is linear. Hence, if we know the value of differences injected in the master tweakey, the exact differences in all the other round keys can be determined.

4. Only the first two rows of each round tweakey are xor'ed with the intermediate state in each round. Thus, because of SKINNY TSA, the tweakey cells operating in the

i^{th} round will next appear in $i^{th}+2$ round and so on. Moreover, both the phases of SKINNY TSA individually update each cell and that too synchronously in all the z-arrays (i.e., same cell is modified across all the z-arrays). Combining these two properties together helps in improving the cryptanalytic prospects since difference cancellations can be done in multiple rounds and more rounds can be covered.

5. *Tweakey differential cancellation.* As described in the specification [BJK+16c], for a given cell, only a single cancellation of the interconnected LFSR can happen every 30 rounds for TK2, and two cancellations for TK3. As a consequence, we can introduce tweakey differences to cancel state differences and obtain more rounds in a differential trail. Additionally, as only half of the internal states are XORed with the subtweakeys at each round, if we bring in a tweakey difference in one cell position of the tweakey state, (which cancels the internal state difference) we will have *three* and *five* rounds of *fully inactive internal states for* TK2 and TK3 respectively.

We also utilize the following *Lemma* in our attacks.

Lemma 1. *For any given SKINNY S-box (c = 4 or 8), say S and any non-zero input - output difference pair, say (δ_{in}, δ_{out}), there exists one solution in average, say y, for which the equation, $S_i(y) \oplus S_i(y \oplus \delta_{in}) = \delta_{out}$, holds true.*

2.3 Notations and Definitions

The following notations are followed throughout the rest of the paper.

$\mathbf{P}$	:	Plaintext
$\mathbf{C}$	:	Ciphertext
$\mathbf{c}$	:	Cell size, where $c \in \{4, 8\}$
$\mathbf{n}$	:	Block size, where $n \in \{64, 128\}$
$\mathbf{r}$	:	Number of rounds broken
$\mathbf{R_d}$	:	Rounds covered by the distingusiher
$\mathbf{R_b}$	:	Rounds extended backward in the tweakey recovery phase
$\mathbf{R_f}$	:	Rounds extended forward in the tweakey recovery phase
$\mathbf{i}$	:	Round number i, where, $1 \leq i \leq r$
$\mathbf{tk_i}$	:	Round tweakey of round i
$\mathbf{X_i}$	:	State before SC, AC in round i
$\mathbf{Y_i}$	:	State before ART in round i
$\mathbf{Z_i}$	:	State before SR in round i
$\mathbf{W_i}$	:	State before MC in round i
$\mathbf{Row(j)}$	:	j^{th} row, where, $1 \leq j \leq 4$
$\mathbf{Col(k)}$	:	k^{th} column, where, $1 \leq k \leq 4$
$\mathbf{S}$	:	Substitution of Sbox
$\mathbf{\Delta s}$	:	Difference in a state s
$\mathbf{s_i[m]}$	:	m^{th} cell of a state s in round i, where $0 \leq m \leq 15$
$\mathbf{s_i[p, \ldots, r]}$	:	p^{th} cell, ... , r^{th} cell of state s in round i, where $0 \leq p, r \leq 15$

The time complexity of the attack is measured in terms of number of r-round SKINNY encryptions required and the memory accesses. The memory complexity is measured in units of 64-/128-bit SKINNY blocks required.

In [BJK+16b], the designers of SKINNY define the adversarial model to be $TK1$, $TK2$ and $TK3$ for the scenarios where an attacker can inject differences in the tweakey state based on the respective SKINNY variant used. We follow the same notation in our attacks.

3 Related-Tweakey Impossible Differential Attack

In this section, we present our related-tweakey impossible differential attack on SKINNY. We investigate truncated impossible differential trails of SKINNY under certain fixed tweakey differences to mount our attacks. Impossible differential cryptanalysis was first proposed independently by Biham et al. [BBS05] and Knudsen [Knu98]. The main idea of this attack is to find an input difference that can never lead to a particular output difference, i.e., probability of such a differential trail is zero. Then, one can derive the right key by discarding the keys which suggest this impossible differential. Under related-tweakey settings, the development of this differential is studied for two encryptions under related-tweakeys, where the relation between the two secret tweakeys is assumed to be known to the attacker.

Through our related-tweakey impossible differential attacks, we can break 19, 23 and 27 rounds of SKINNY under $TK1$, $TK2$ and $TK3$ respectively. In this section, details of the 23-round attack on SKINNY-n-$2n$ are discussed, and extended to the cases where part of the tweakey is tweak. The attacks of other versions work in a similar manner. The attack consists of two phases: *Distinguisher construction phase* and *Tweakey recovery phase*.

3.1 Related-Tweakey Impossible Distinguisher

In this phase, we first construct a 14-round related-tweakey impossible distinguisher for SKINNY-n-$2n$ by leveraging the tweakey differential cancellation property. Our distinguisher is placed between Round 5 to Round 19. A 7.5-round related-tweakey differential in the forward direction (having prob. 1) starting at Y_5 (after the SC and AC operations in Round 5) is concatenated to a 6.5-round related-tweakey differential (having prob. 1) starting in the reverse direction from Y_{19} (before the ART operation in Round 19). The contradiction happens in Round 13 at $X_{13}[11]$. The 14-round related-tweakey impossible differential is:

$$(00b0 \mid 0000 \mid 0000 \mid 0000) \xrightarrow{14r} \not\rightarrow (0000 \mid 0000 \mid 00N0 \mid 0000)$$

where b denotes a fixed non-zero difference and N denotes any non-zero difference. The entire impossible differential path is illustrated in Fig. 5. The tweakey difference is only injected in one cell.

We construct a 12-round distinguisher under TK1 and owing to the tweakey cancellations 16-round impossible distinguishers can be constructed under TK3. For more information regarding the impossible distinguishers under TK1 and TK3, we refer the readers to Appendix A.

3.2 Tweakey Recovery Attacks

In this section, we use the 14-round related-tweakey impossible differential distinguisher to attack 23-rounds of SKINNY-n-$2n$ using a pair of related-tweakeys. Before the attack is explained, we introduce some more notations which are borrowed from [BNS14]. Suppose an impossible differential $(\Delta X \not\rightarrow \Delta Y)$ has been constructed for E' under a pair of related-tweakeys and is used to attack $E = E_f \circ E' \circ E_b$. Through E_b^{-1} (E_f), ΔX (ΔY) is propagated to Δ_{in} (Δ_{out}) with probability 1. Let c_{in} (c_{out}) be the number of bit conditions that need to be verified for $\Delta_{in} \rightarrow \Delta X$ $(\Delta Y \leftarrow \Delta_{out})$, and k_{in} (k_{out}) is the key information involved in E_b (E_f).

Consider the two secret related-tweakey inputs to be - $(TK\text{-}1)^1 \mid\mid (TK\text{-}2)^1$ and $(TK\text{-}1)^2 \mid\mid (TK\text{-}2)^2$. Consider further the difference Δ injected in round 1 subtweakey to be: $\Delta = (0\ d\ 0\ 0 \mid 0\ 0\ 0\ 0 \mid 0\ 0\ 0\ 0 \mid 0\ 0\ 0\ 0)$, i.e., the related round 1 subtweakeys differ at the 2^{nd} cell-position and the difference d is known to the attacker. The 14-round distinguisher

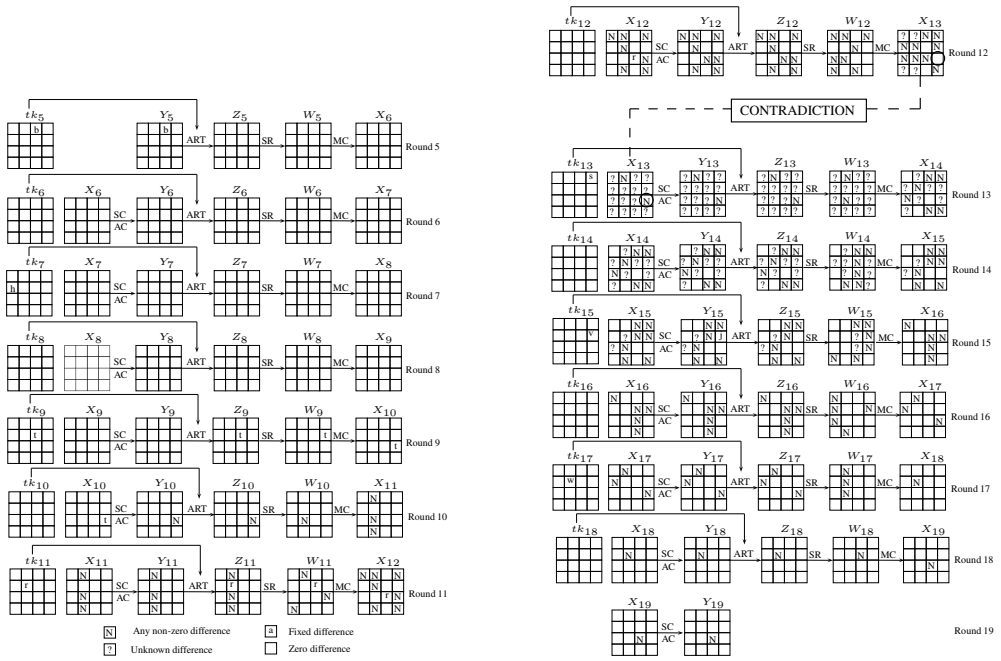


Figure 5: 14-round related-tweakey impossible distinguisher on SKINNY-n-$2n$.

is extended 4.5 rounds at the top and 4.5 rounds in the bottom to cover 9 rounds in the tweakey recovery phase as shown in Fig. 6. Here E' covers 14 rounds, E_f (E_b) covers 4.5 (4.5) rounds, $c_{in} = |\Delta_{in}| = 7c$, $c_{out} = 15c$, $|\Delta_{out}| = 16c$ and $|k_{in} \cup k_{out}| = 31c$ where c is the cell size.

As discussed in Section 2.2, the order of ART, SR and MC operations in round 1 can be changed. Since, the design of SKINNY does not include a pre-whitening key, the input difference at W_1 (denoted as P^{eq} in Fig. 6) can be considered as the plaintext difference and the equivalent plaintexts so obtained can be rolled back to compute the actual plaintexts. In the following discussion, we start our tweakey recovery attack at W_1 and call the inputs at this position as the plaintext inputs. The steps of our attack are as follows:

Data collection Consider a pair of structures S_1 and S_2, where, each structure consists of $2^{|\Delta_{in}|} = 2^{7c}$ plaintexts and for each plaintext pair $P_1 \in S_1$ & $P_2 \in S_2$, $P_1 \oplus P_2 = (0\ 0\ N\ 0 \mid N\ N\ 0\ N \mid N\ 0\ N\ 0 \mid 0\ N\ 0\ 0)$, where N denotes any cell value. The total number of possible plaintext pairs is $2^{2|\Delta_{in}|} = 2^{14c}$. Invert back the plaintexts in S_1 and S_2 by one keyless round to get the original plaintexts. Encrypt the pool S_1 under $(TK\text{-}1)^1 \,||\, (TK\text{-}2)^1$ and the pool S_2 under $(TK\text{-}1)^2 \,||\, (TK\text{-}2)^2$ to obtain the corresponding ciphertexts. Generate 2^{x}[2] such pair of structures and repeat this for each pair of structures. In total there are $M = 2^{x+2|\Delta_{in}|} = 2^{x+14c}$ plaintext pairs. This step requires a total of $2^{x+|\Delta_{in}|+1} = 2^{x+7c+1}$ encryption calls.

Tweakey recovery For each of the M pairs:

(a) Compute $\Delta X_{23}[10, 12, 14]$ from the knowledge of the ciphertext pair values as computing these cells do not require any tweakey information. Due to MC operation on the active nibbles of $Col(1)$ and $Col(3)$ of W_{22}, it can be seen

[2]More than one pair of structures are used. The case which requires less than one pair of structures is discussed in Appendix.

Round i	Row(1)	Row(2)
1	(0 d 0 0)	(0 0 0 0)
3	(c 0 0 0)	(0 0 0 0)
5	(0 0 b 0)	(0 0 0 0)
19	(a 0 0 0)	(0 0 0 0)
21	(0 0 e 0)	(0 0 0 0)
23	(0 0 0 0)	(f 0 0 0)

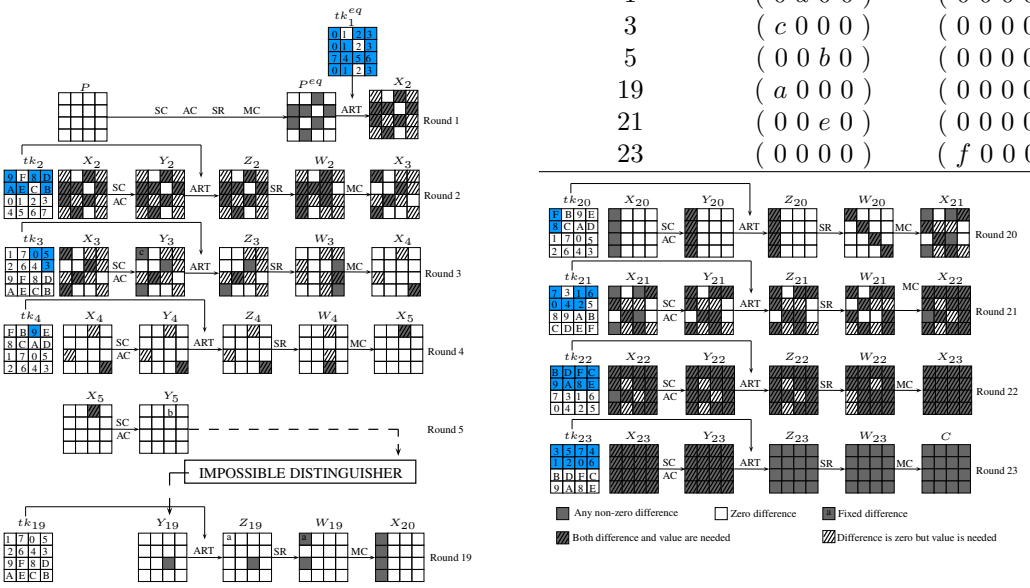


Figure 6: 23-round attack against SKINNY-n-$2n$.

that $\Delta X_{23}[0] = \Delta X_{23}[4] = \Delta X_{23}[12]$ and $\Delta X_{23}[6] = \Delta X_{23}[10] \oplus \Delta X_{23}[14]$ and thus can be determined. From this information, $X_{23}[0]$, $Y_{23}[0]$ and thus $tk_{23}[0]$ can be found using Lemma 1 as $\Delta Y_{23}[0]$ is known from the ciphertexts. Similarly, the tweakey cells $tk_{23}[4, 6]$ can also be derived. Guess $tk_{23}[1, 2, 3, 5, 7]$. Compute Z_{22} and ΔZ_{22}[0-4, 6-9, 11-12, 14-15]. The time complexity of this step is $M \cdot 2^{5c}$ one-round encryptions and a total of $M \cdot 2^{5c}$ tests need to be done in the next step.

(b) Compute $\Delta X_{22}[8, 11, 12, 14, 15]$ from the knowledge of Z_{22} and ΔZ_{22} known cells. Due to MC operation on the active nibbles of $Col(1, 3, 4)$ of W_{21}, $\Delta X_{23}[0, 2, 4, 6, 7]$ can be determined as - $\Delta X_{22}[0] = \Delta X_{22}[4]$, $\Delta X_{22}[4] = \Delta X_{22}[8] \oplus \Delta X_{22}[12]$, $\Delta X_{22}[2] = \Delta X_{22}[6] = \Delta X_{23}[14]$ and $\Delta X_{22}[7] = \Delta X_{22}[11] \oplus \Delta X_{22}[15]$. Use this information and the known differences of ΔZ_{22} to derive tk_{22}[0, 2, 4, 6, 7] by using Lemma 1. Guess $tk_{22}[1, 3, 5]$. Compute Z_{21} and ΔZ_{21} as shown in Fig. 6. The time complexity of this step is $M \cdot 2^{8c}$ one-round encryptions and a total of $M \cdot 2^{8c}$ tests need to be done in the next step.

(c) Compute $\Delta X_{21}[10, 12, 14]$ from the knowledge of Z_{21} and ΔZ_{21} known cells. Check if $\Delta X_{21}[10] = \Delta X_{21}[14]$. This acts as one-cell filter. As $\Delta X_{21}[0] = \Delta X_{21}[4] = \Delta X_{21}[12]$ and $\Delta X_{21}[2] = \Delta X_{21}[10]$, use this information and the known differences of ΔZ_{21} to derive tk_{21} [0, 2, 4] by using Lemma 1. At this stage, the attacker can uniquely determine $(TK\text{-}1)^1[0, 1, 7]$ and $(TK\text{-}2)^1[0, 1, 7]$. This helps the her to determine $tk_1[0, 1, 7]$ and $tk_3[2]$. This step has a time complexity of $M \cdot 2^{8c}$ one-round encryptions and the number of tests left for the next step is $M \cdot 2^{7c}$.

(d) Since $tk_1^{eq}[0, 1, 4, 5, 8, 12, 13]$ are known from the previous steps, $\Delta Y_2[5, 8]$ can be computed. Check if $\Delta Y_2[5] = \Delta Y_2[8]$ or not. This acts as a one-cell filter. Also, since $\Delta Y_2[2] = \Delta Y_2[8]$, use this information and the plaintext difference to derive $tk_1[2]$ by using Lemma 1. At this stage, the attacker can uniquely determine $(TK\text{-}1)^1[2]$ and $(TK\text{-}2)^1[2]$ and compute $tk_1[2]$ and $tk_{21}[6]$. The

time complexity of this step is $M \cdot 2^{7c}$ and the number of tests left for the next step is $M \cdot 2^{6c}$.

(e) Guess $tk_{21}[3,5]$. Compute ΔZ_{20} and $Z_{20}[0,4,8,12]$. Compute $\Delta X_{20}[8,12]$. Now, since $\Delta X_{20}[0] = \Delta X_{20}[12]$ and $\Delta X_{20}[4] = \Delta X_{20}[8] \oplus \Delta X_{20}[12]$, $tk_{20}[0,4]$ can be derived using Lemma 1. The knowledge of $\Delta X_{20}[4]$ also allows her to compute $\Delta W_{19}[0]$. Check if $\Delta W_{19}[0] = a$ or not where, the difference a is known to the attacker. We obtain a filter of one cell. At this stage, the attacker can uniquely determine $(TK\text{-}1)^1[4,6,8,15]$ and $(TK\text{-}2)^1[4,6,8,15]$. This allows her to compute $tk_1[4,6]$ and $tk_2[1,2]$. This step has a time complexity of $M \cdot 2^{8c}$ one-round encryptions and the number of tests left after this step is $M \cdot 2^{7c}$.

(f) Guess $tk_1[5]$ and compute $\Delta Y_2[10]$. Since, $\Delta Y_2[7] = \Delta Y_2[10]$ and from the plaintexts $\Delta X_2[7]$ is known, determine $X_2[7]$ and thus $tk_1^{eq}[7] = tk_1[3]$. At this stage, the attacker can uniquely determine $(TK\text{-}1)^1[3,5]$ and $(TK\text{-}2)^1[3,5]$ and thus compute $tk_3[3,7]$. As per the MC^{-1} definition, $\Delta W_2[12] = \Delta X_3[0] \oplus \Delta X_3[12]$. Since, $\Delta W_2[12]$ $(= \Delta Y_2[13])$ and $\Delta X_3[12]$ $(= \Delta W_2[4] = \Delta Y_2[7])$ are known to the attacker, she can compute $\Delta X_3[0]$. Since, $\Delta Y_3[0] = c$ is also known to her, $X_3[0]$ can be determined. Using $X_3[0]$, $W_2[0] = X_3[0] \oplus W_2[8] \oplus W_2[12]$ can be determined. This helps the attacker to derive $tk_2[0]$ using Lemma 1. Further, $X_3[12] = W_2[0] \oplus W_2[12]$ can also be computed. This allows her to compute $Y_3[12]$ and $\Delta Y_3[12]$. It can be easily verified that $\Delta Y_3[12] = \Delta Y_3[6] = \Delta Y_3[9]$. It can also be observed that $\Delta X_3[6] = \Delta Y_2[5]$ and $\Delta X_3[12] = \Delta Y_2[4]$ due to MC^{-1} operation in Round 3. Thus, $X_3[6,9]$ and $Y_3[6,9]$ can be determined. Now, $X_3[6] = W_2[2]$ and $W_2[2]$ is already known to the attacker as $W_2[2] = Y_2[2] \oplus tk_2[2]$. Hence, a one-cell filter is obtained. Using $X_3[9]$, obtain $W_2[5]$ and derive $TK_2[4]$. Attacker can now uniquely determine $(TK\text{-}1)^1[10]$ and $(TK\text{-}2)^1[10]$. This step has a time complexity of $M \cdot 2^{8c}$ one-round encryptions and the number of tests left after this step is $M \cdot 2^{7c}$.

(g) Guess $tk_2[3]$. This allows the attacker to compute $Y_3[3]$ and further $X_4[15]$ and $\Delta X_4[15]$. Through, this knowledge she can determine $\Delta X_5[2]$. Since, $\Delta Y_5[2] = b$ is already known to her, using S-box equations she can compute $X_5[2]$. Note that $S(X_4[2]) \oplus S(X_4[8]) = X_5[2] \oplus tk_4[2]$ where, the right hand side of the equation is known to the attacker and the left hand side involves information of the only two unknown subtweakey cells $tk_2[5], tk_2[7]$. The attacker constructs in advance a table that stores input values $X_4[2], X_4[8]$ for the two Sbox for each possible right hand value and uses it for the retrieval of inputs. There are 2^c combinations of $X_4[2], X_4[8]$ for each right hand value. Then, $tk_2[5] = S^{-1}(X_4[8] \oplus tk_3[7] \oplus Y_3[7]) \oplus W_2[8]$, $tk_2[7] = S^{-1}(X_4[2] \oplus tk_3[2] \oplus Y_3[2] \oplus Y_3[15]) \oplus W_2[10]$ can be calculated. The time complexity of this step is $M \cdot 2^{8c}$ and the number of tests verifying the impossible distinguisher is $M \cdot 2^{9c}$, i.e., for each pair there are 2^{9c} 31-cell keys on average that verify the impossible distinguisher and are wrong.

The time complexity of analyzing M pairs is $M \cdot 2^{8c}$. The total number of tweakeys left is:

$$TK_{rem} = 2^{|k_{in} \cup k_{out}|}(1 - 2^{-(c_{in}+c_{out})})^M = 2^{31c}(1 - 2^{-22c})^{2^{x+14c}} \quad (1)$$

Suppose $(1 - 2^{-(c_{in}+c_{out})})^M = 2^{-g}$, $1 < g \leq |k_{in} \cup k_{out}|$, which means g-bit key material is recovered, then we have $M = 2^{c_{in}+c_{out}} g\ln 2$ since $(1 - 2^{-(c_{in}+c_{out})})^M \approx e^{-M2^{-(c_{in}+c_{out})}}$. Meanwhile $M = 2^{x+2|\Delta_{in}|-n+|\Delta_{out}|}$, so $2^x = 2^{c_{in}+c_{out}+n-2|\Delta_{in}|-|\Delta_{out}|} \times g\ln 2$, and data complexity is $D = 2^{x+|\Delta_{in}|+1} = 2^{c_{in}+c_{out}+n+1-|\Delta_{in}|-|\Delta_{out}|} g\ln 2$.

Brute force For the tweakeys that remain, we guess the remaining tweakey cells (1 cell) and exhaustively search the $TK_{rem} \times 2^c = 2^{n-g}$ tweakeys to find the correct tweakey.

Attack Complexities. The attack described above requires a data complexity of $D = 2^{c_{in}+c_{out}+n+1-|\Delta_{in}|-|\Delta_{out}|}g\ln 2 = 2^{n+1-c}g\ln 2$ chosen plaintexts. The total time complexity is the summation of the time consumption of all the steps:

$$T = D + M \cdot 2^{8c} + TK_{rem} \cdot 2^{c}.$$

The memory complexity is the storage for one structure and wrong keys. For, $c = 4$, we set $g = 4$, then $D = 2^{62.47}$, $M \cdot 2^{8c} = 2^{121.47}$, $TK_{rem} \cdot 2^{c} = 2^{124}$, $T = 2^{124.2}$ and the memory complexity is 2^{124}. For, $c = 8$, we set $g = 16$, then $D = 2^{124.47}$, $M \cdot 2^{8c} = 2^{243.47}$, $TK_{rem} \cdot 2^{c} = 2^{240}$, $T = 2^{243.6}$ and the memory complexity is 2^{248}.

Attack where tweak is used. Suppose a w-bit tweak is used. The tweak is loaded into the first w bits of the first tweakey state $TK - 1$, followed by the key material. For $0 \leq w \leq 9c$, we have $|k_{in} \cup k_{out}| = 2n - w - c < 2n - w$ and the above attack still has a time complexity below 2^{2n-w}, so 23-round SKINNY-n-$2n$ is valid in such cases. For $9c < w \leq 12c$, we still have $|k_{in} \cup k_{out}| = 2n - w - c < 2n - w$, but the time complexity is higher than 2^{2n-w}. For $w = 9c$, results are as follows. For $c = 4$, we set $t = 3$, then $D = 2^{62.06}$, $T = 2^{90.03.03}$ and the memory complexity is 2^{88}. For $c = 8$, we set $t = 6$, then $D = 2^{123.06}$, $T = 2^{179.03}$ and the memory complexity is 2^{176}.

For SKINNY-n-n and SKINNY-n-$3n$, our results are as shown in Table 2. For more information regarding the impossible distinguishers under TK1 and TK3, we refer the readers to Appendix A. The 19-round attack of SKINNY-n-n and the 27-round attack of SKINNY-n-$3n$ is presented in Appendix B.

Table 2: Results of related-tweakey impossible differential attack on SKINNY

Model	Version	#Rounds	R_d	R_b	R_f	Data	Time	Memory
TK1	64-64	19	12	4.5	2.5	$2^{61.47}$	$2^{63.03}$	2^{56}
	128-128	19	12	4.5	2.5	$2^{122.47}$	$2^{124.60}$	2^{112}
TK2	64-128	23	14	4.5	4.5	$2^{62.47}$	$2^{124.2}$	2^{124}
	128-256	23	14	4.5	4.5	$2^{124.47}$	$2^{243.6}$	2^{248}
TK3	64-192	27	16	6.5	4.5	$2^{63.53}$	2^{189}	2^{184}
	128-384	27	16	6.5	4.5	$2^{126.03}$	2^{378}	2^{368}

4 Truncated Related-Tweakey Differential Trails and Rectangle Attacks

In the last section, we constructed impossible distinguishers by connecting two truncated differential trails which propagate forward and backward with probability 1 under one related-tweakey. In this section, boomerang (rectangle) distinguishers are constructed by combining two truncated differential trails which propagate forward and backward with probability 1 under two distinct related-tweakeys.

This section first gives a brief introduction of boomerang attacks and rectangle attacks, and then presents rectangle distinguishers of SKINNY based on which 17, 22 and 27 rounds of SKINNY can be attacked under TK1, TK2 and TK3 respectively.

4.1 Boomerang Attacks and Rectangle Attacks

Boomerang attack, proposed by David Wagner in FSE 1999 [Wag99], allows an attacker to concatenate two short differential trails. This proves beneficial in cases where long

differential trails have a very low probability or it is difficult to search a long differential trail. In the basic setting of the attack , the block cipher is treated as a cascade of two sub-ciphers E_0 and E_1, each having a high probability short differential of its own. These differentials are then combined in a chosen plaintext and ciphertext attack setting to first construct a boomerang distinguisher and then use the distinguisher to recover the secret key.

Suppose E_0 covers the first l rounds of encryption and E_1 covers the rest $(r-l)$ rounds of encryption. Let us further suppose, there exists a differential $\alpha \rightarrow \beta$ through E_0 with a high probability p. Similarly, there exists a differential $\delta \rightarrow \gamma$ through $(E_1)^{-1}$ which has a high probability q. The boomerang attack then proceeds as follows:

1. Consider two plaintexts X, Y such that $Y = X \oplus \alpha$. Obtain their corresponding ciphertexts X'', Y'' respectively.

2. The probability that $E_0(X) \oplus E_0(Y) = X' \oplus Y' = \beta$ is $\mathbf{p}$.

3. Obtain, $Z'' = X'' \oplus \delta$ and $W'' = Y'' \oplus \delta$. If we apply $(E_1)^{-1}$ to each of the pairs (X'', Z'') and (Y'', W''), then with probability $\mathbf{q^2}$, $(E_1)^{-1}(X'') \oplus (E_1)^{-1}(Z'') = X' \oplus Z' = \gamma$ and $(E_1)^{-1}(Y'') \oplus (E_1)^{-1}(W'') = Y' \oplus W' = \gamma$.

4. Then, the following statement holds true: With probability $\mathbf{pq^2}$, $(E_1)^{-1}(Z'') \oplus (E_1)^{-1}(W'') = \beta$. This is because,

$$\begin{aligned}(E_1)^{-1}(Z'') \oplus (E_1)^{-1}(W'') &= Z' \oplus W' \\ &= X' \oplus Z' \oplus X' \oplus Y' \oplus Y' \oplus W' \\ &= \gamma \oplus \beta \oplus \gamma \\ &= \beta\end{aligned}$$

5. Thus, with probability $\mathbf{p^2q^2}$, $E^{-1}(Z'') \oplus E^{-1}(W'') = Z \oplus W = \alpha$.

6. Now if, $(\mathbf{pq}) > 2^{-n/2}$, then a valid distinguisher is constructed. This is because, for a random permutation, the expected probability that $Z' \oplus W' = \alpha$ is 2^{-n}.

Therefore, if $\mathbf{p^2q^2}$ is sufficiently large, then the boomerang distinguisher can effectively distinguish between $E(\cdot)$ and a randomly chosen permutation, given a sufficient number of adaptive chosen plaintexts and ciphertexts. The plaintexts (X, Y, Z, W) together are termed as a *quartet* and satisfy the following property:

$$X \oplus Y \oplus Z \oplus W = 0$$

The basic boomerang attack explained above requires adaptive chosen plain-/ciphertexts. Later, Kelsey et al. [KKS00] developed amplified boomerangs which are pure chosen-plaintext attacks. In the case of amplified boomerang attacks, the attacker chooses certain amount of plaintext pairs and let the oracle encrypt them. Any two pairs form a quartet (X, Y, Z, W), and the difference γ before E_1 holds with probability 2^{-n} for a quartet. Thus one can expect a right quartet where $X'' \oplus Z'' = Y'' \oplus W'' = \delta$ with probability $2^{-n}p^2q^2$. For a random permutation the expected probability is 2^{-2n}, so if $2^{-n}p^2q^2 > 2^{-2n}$, a distinguisher can be constructed. In [BDK01], Biham et al. made further improvements which allow any value of β and γ to occur as long as $\beta \neq \gamma$ and renamed the attack as *rectangle attack*. As a result, the probability of right quartet is increased to $\hat{p}\hat{q}$, where $\sqrt{\Sigma_i \Pr^2(\alpha \rightarrow \beta_i)}$ and $\sqrt{\Sigma_j \Pr^2(\gamma_j \rightarrow \delta)}$.

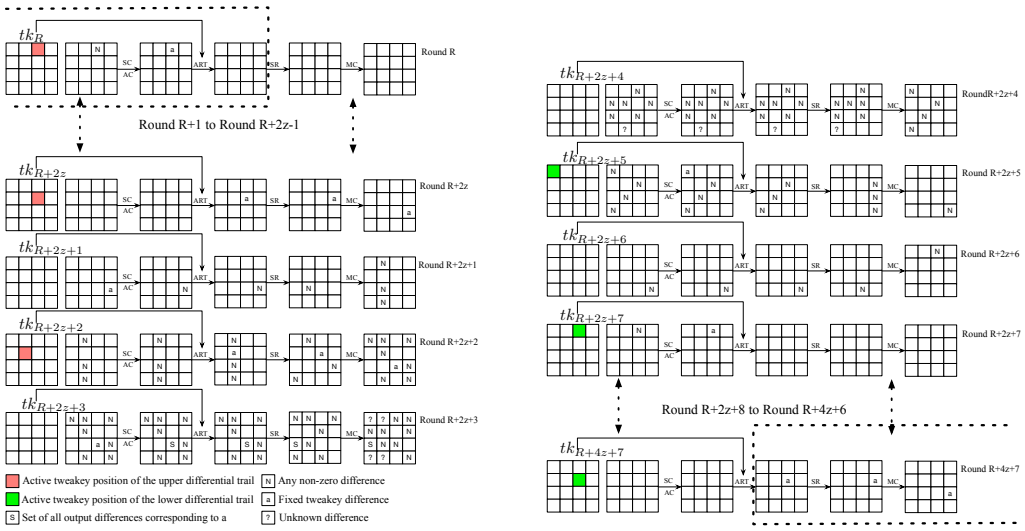


Figure 7: Description of the generalized related-tweakey rectangle distinguisher.

4.2 Related-Tweakey Rectangle Distinguishers

Our distinguisher takes the advantage of the differential cancellation behavior of the tweakey schedule to cover as many rounds as possible.

Fig. 7 is a description of the generalized related-tweakey rectangle distinguisher.

For the sake of generalization, the distinguisher begins at round R and the subtweakey difference a is constant. As shown in Fig. 7, an input difference will transform to any possible output difference which is denoted by N. The notation ? means that we are not sure whether the cell is active or not. There is only one active tweakey cell in both the upper and the lower truncated differential trails. In the following discussion we take SKINNY-64 as an example, where each cell is a nibble.

In a typical rectangle distinguisher, usually, two differential trails are constructed, i.e., $\alpha \to \beta$ and $\gamma \leftarrow \delta$. Note that in the related-tweakey case, we specify a tweakey difference Δ_1 for upper trail $\alpha \to \beta$ and a tweakey difference Δ_2 for lower trail $\gamma \leftarrow \delta$. The position of the active tweakey cell is chosen by the rule that the truncated differential trail from round $R + 2z$ to round $R + 2z + 3$ is the optimal differential trail as well as the truncated differential trail from round $R + 2z + 7$ to round $R + 2z + 4$.

The $\alpha \xrightarrow{\Delta_1} \beta$ differential trail begins with a state difference α that contains one active cell in a carefully chosen position relating to the key state permutation. At Round R, a tweakey difference of one cell position is introduced to cancel the state difference. Therefore, we will have $2z - 1$ rounds of fully non-active internal state in the subsequent encryption process,e.g., we have 5 rounds of fully non-active internal states for TK3.

At round $R + 2z$, the key state introduces a cell difference a to the internal states with the corresponding related-tweakey difference. The introduced internal difference continues to diffuse for four rounds according to the round function of SKINNY. At round $2z + 2$, another tweakey difference a is introduced, which complicates the diffusion of the difference pattern through rounds.

Similarly, at the bottom of the distinguisher the input difference δ (with the same value as the tweakey difference a) of the differential trail $\gamma \xleftarrow{\Delta_2} \delta$ is cancelled out by the tweakey difference Δ_2 which results in $2z - 1$ rounds of fully non-active internal states. At round $2z + 7$, difference a is inserted by the subtweakey difference and continues to distribute for 4 rounds just as the upper trail $\alpha \xrightarrow{\Delta_1} \beta$.

A rectangle distinguisher will work as long as the two upper trails $\alpha \xrightarrow{\Delta_1} \beta$ as well as the two lower trails $\gamma \xleftarrow{\Delta_2} \delta$ agree with each other(note that $\beta \neq \gamma$). Here we denote the two upper trails as $\alpha \xrightarrow{\Delta_1} \beta_1$ and $\alpha \xrightarrow{\Delta_1} \beta_2$ respectively. Equally, we denote the two lower trails as $\gamma_1 \xleftarrow{\Delta_2} \delta$ and $\gamma_2 \xleftarrow{\Delta_2} \delta$. Thus, a rectangle distinguisher will succeed as long as $\beta_1 = \beta_2$ and $\gamma_1 = \gamma_2$ while $\beta \neq \gamma$ are satisfied.

Instead of considering specific differential trails with high probability, we focus on all the possible differential trails, which means the output difference β of the upper trail $\alpha \xrightarrow{\Delta_1} \beta$ can be any possible values. We employ the same strategy when analyzing the differential probability of the lower trail $\gamma \xleftarrow{\Delta_2} \delta$, i.e., γ_1 and γ_2 can be any possible values.

Thus, the probability[3] of the upper trial $\alpha \xrightarrow{\Delta_1} \beta$ of the related-tweakey rectangle distinguisher is $\sum_{\beta_1,\beta_2} P_r(\alpha \xrightarrow{\Delta_1} \beta_1) \cdot P_r(\alpha \xrightarrow{\Delta_1} \beta_2) \cdot P_r(\beta_1 = \beta_2) = (2^{-3.9})^7 \cdot 2^{-3} = 2^{-30.3}$. Similarly, the probability of the lower trail $\gamma \xleftarrow{\Delta_2} \delta$ of the related-tweakey rectangle distinguisher is $\sum_{\gamma_1,\gamma_2} P_r(\gamma_1 \xleftarrow{\Delta_2} \delta) \cdot P_r(\gamma_2 \xleftarrow{\Delta_2} \delta) \cdot P_r(\gamma_1 = \gamma_2) = (2^{-3.9})^6 \cdot 2^{-4} = 2^{-27.4}$. The total probability of the related-tweakey rectangle distinguisher would be $2^{-30.3} \cdot 2^{-27.4} = 2^{-57.7}$. To be consist with conventional notations, we define $\hat{p}^2\hat{q}^2 = \sum_{\beta_1,\beta_2} P_r(\alpha \xrightarrow{\Delta_1} \beta_1) \cdot P_r(\alpha \xrightarrow{\Delta_1} \beta_2) \cdot P_r(\beta_1 = \beta_2) \cdot \sum_{\gamma_1,\gamma_2} P_r(\gamma_1 \xleftarrow{\Delta_2} \delta) \cdot P_r(\gamma_2 \xleftarrow{\Delta_2} \delta) \cdot P_r(\gamma_1 = \gamma_2)$.

The generalized related-tweakey distinguisher is suitable for both the 64-bit block size versions and 128-bit block size versions although we need to adjust the probability of the distinguisher for distinctive versions in use, which is easy work. Particularly, the distinguisher is more effective on more tweaks versions, e.g., the distinguisher is 19 rounds on TK3 while only 15 rounds on TK2.

Discussion. In order to take full advantage of the tweakey differential cancellation property, the position of the active tweakey cell is determined cautiously taking into account two main factors. Firstly, the tweakey difference ought to offset the state difference at the beginning of the distinguisher, resulting in $2z - 1$ rounds of fully non-active rounds. Secondly, the truncated differential trail formed due to the tweakey difference should have at most 8 active cell positions in the end, otherwise, the distinguisher won't work. With these requirements, we searched all the possible active cell positions of the tweakey and found 5 positions for the upper truncated differential trail and 4 positions for the lower truncated differential trail that satisfy our requirements.

The above distinguisher is constructed using one of the optimal truncated trails covering maximum number of rounds and having minimal probability. Moreover, the chosen active tweakey position also ensures that the truncated differential trails in the extended rounds of the tweakey recovery process are optimal (enabling maximum rounds to be covered), indicating that it is the best distinguisher under all the considertations.

Experimental verification of the distinguisher. The validity of our distinguishers is verified with an experiment on SKINNY-64-128 which aims at finding a right quartet for the 14-round boomerang distinguisher with probability 2^{-40}. The result shows that averagely there is one right quartet among $2^{35.7}$ trials. The experiment not only verifies the correctness of our distinguishers but also demonstrates that the probability of our distinguishers are not overestimated. One of the right quartets obtained and more details of the experiment is shown in Appendix D.

[3]We use $1/15 = 2^{-3.9}$ and 2^{-3} to compute the probability of two output difference (i.e., β_1 and β_2) of the truncated differential to collide rather than a general 2^{-4}.

4.3 Key Recovery Algorithm Based on Related-Tweakey Rectangle Distinguishers

Our algorithm for using related-tweakey rectangle distinguisher in a key recovery attack is adapted from Biham et al's algorithm [BDK02] where in their rectangle key recovery attack there is no key difference. Note that this adapted key recovery algorithm is applicable to other block ciphers under related-tweakey (-key) model as long as the key schedule is linear. Here we just give the main results of the adapted tweakey recovery algorithm, and details of the algorithm are provided in Appendix C.

We continue to use the notations and conventions of boomerang and rectangle attacks. The cipher E is expressed in a concatenation form $E = E_f \circ E_1 \circ E_0 \circ E_b$ and $E' = E_1 \circ E_0$ is the rectangle distinguisher(see Fig. 8). E_b and E_f are the rounds extended to the backward and forward direction of the distinguisher.

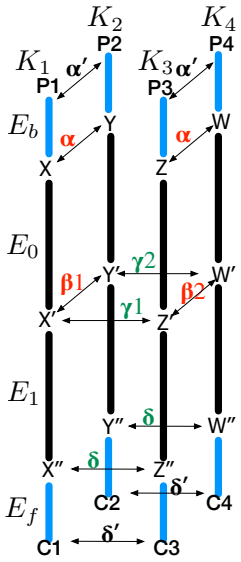
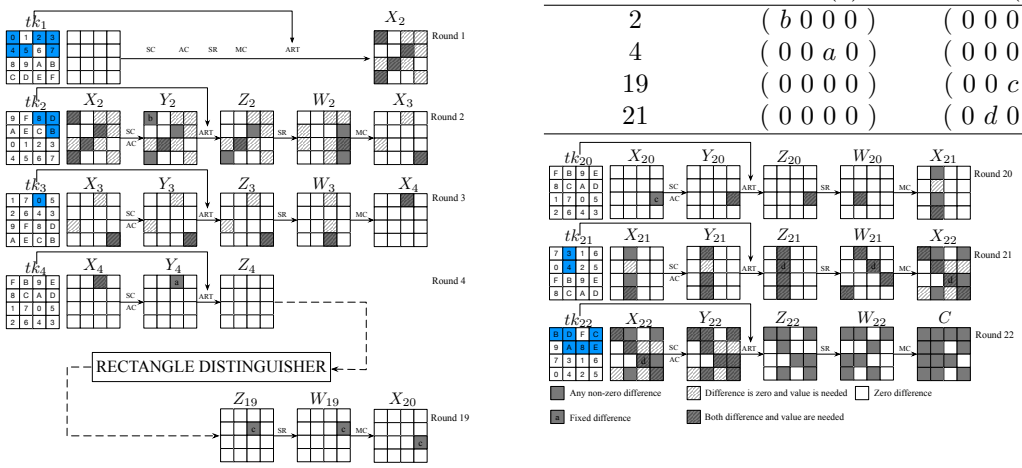


Figure 8: Key rovery model of related-tweakey rectangle attack

In a related-tweakey rectangle attack, the quartet of (P_1, P_2, P_3, P_4) as plaintexts and (C_1, C_2, C_3, C_4) as ciphertexts is encrypted with four related-tweakeys (K_1, K_2, K_3, K_4) which are related with each other with specific tweakey differences. Let the input difference α of the distinguisher diffuse to the backward direction of the cipher E for several rounds (which is E_b section of E) with the related-tweakey difference Δ_1 and store all the possible output differences α' (which are the actual plaintext differences) corresponding to α in a set U_b. Likewise, let the output difference δ of the distinguisher diffuse several rounds to the forward direction of the cipher with a related-tweakey difference Δ_2 and store all the possible output differences δ'(which are the actual ciphertext differences) in a set U_f.

Before we continue we introduce some additional notations. Let V_b be the space spanned by the values in U_b. Let $r_b = \log_2|V_b|$ and $t_b = \log_2|U_b|$. Let m_b be the the number of subtweakey bits which involve in E_b and affect the difference of plaintexts when encrypting pairs whose difference after E_b is α. Similarly, we define U_f, V_f, r_f, t_f, m_f for E_f, i.e., U_f is the set of ciphertext differences that may cause a difference δ before E_f under the tweakey difference Δ_2, V_f is the space spanned by values of U_f and $r_f = \log_2|V_f|, t_f = \log_2|U_f|$. m_f is the number of subtweakey bits which involve in E_f and affect the difference of ciphertexts when decrypting pairs whose difference before E_f is δ. Using the adapted tweakey recovery algorithm, the complexities of a rectangle attack are



Round i	Row(1)	Row(2)
2	(b 0 0 0)	(0 0 0 0)
4	(0 0 a 0)	(0 0 0 0)
19	(0 0 0 0)	(0 0 c 0)
21	(0 0 0 0)	(0 d 0 0)

Figure 9: Tweakey rovery attack on 22-round SKINNY-64-128.

as follows.

- Data complexity: $D = 4M$ chosen plaintexts, where $M = \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q}$ and s is the expected number of right quartets;
- Time complexity: $4M + 2 \cdot M^2 \cdot 2^{r_f - n} + 2 \cdot M^2 \cdot 2^{t_f - n} + M^2 \cdot 2^{2t_f + 2t_b - 2n}(1 + 2^{t_b - r_b}) + M^2 \cdot 2^{t_b + t_f - 2n+1}(2^{m_b + t_f} + 2^{m_f + t_b})$ memory accesses;
- Memory complexity: $4M + 2^{t_b} + 2^{t_f} + 2^{m_b + m_f}$.

4.4 Related-Tweakey Rectangle Attacks

This subsection takes SKINNY-64-128 as an example to illustrate our attacks and the attack works similarly for other versions so we just provide the results.

The following Fig. 9 is an dedicated attak of 22 rounds SKINNY-64-128 using the 15-round related-tweakey rectangle distinguisher and extending 3.5 rounds before and after the distinguisher.

Firstly search a number of right quartets with the algorithm introduced in the previous section. Then check whether the subtweakey guess can pass the test that $E_b^{K_1}(P_1) \oplus E_b^{K_2}(P_2) = E_b^{K_3}(P_3) \oplus E_b^{K_4}(P_4) = \alpha$ and $(E_f^{K1})^{-1}(C_1) \oplus (E_f^{K3})^{-1}(C_3) = (E_f^{K2})^{-1}(C_2) \oplus (E_f^{K4})^{-1}(C_4) = \delta$. The subtweakey guess that counts with the highest value would be the right subtweakey guess.

In a concrete attack, we choose the cell position and value of tweakey difference so that a distinguisher with the optimal probability can be obtained. As there is no whitening key added before the first round, when recovering the subtweakey we can attack one round for free. The subtweakey bits that afftect the input difference α of the distinguisher are marked specially in each round(refer to Fig. 9).

At this attack senario, we choose $r_b = log_2(15^4) = 15.6$, $t_b = log_2(15^3) + log_2 6 = 14.3$, $m_b = 10c = 40$, $r_f = log_2(15^8) = 31.3$, $t_f = log_2(15^7) + log_2 8 = 30.3$ and $m_f = 8c = 32$. As we have analyzed previously, the probability of the related-tweakey rectangle distinguisher is $\hat{p}^2\hat{q}^2 = (2^{-3.9})^7 \cdot 2^{-2.4} \cdot (2^{-3.9})^6 \cdot 2^{-4} = 2^{-57.1}$.

The total data complexity for 22-round SKINNY64-128 is $D = 4 \cdot \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q} = 4 \cdot \sqrt{s} \cdot 2^{32} \cdot 2^{28.5} = \sqrt{s} \cdot 2^{62.5}$, i.e. if we choose $s = 2$, the data complexity would be 2^{63}. The time complexity required is $2^{109.9}$ memory access and the memory complexity is 2^{63}.

If we reduce the related-tweakey rectangle distinguisher by one round, i.e., a 14-round distinguisher, the data complexity and time complexity could be highly reduced. In a 14-round distinguisher, the probability would be $\hat{p}^2\hat{q}^2 = (2^{-3.9})^3 \cdot (2^{-3.9})^6 \cdot 2^{-4} = 2^{-39.1}$. Using the same tweakey recovery procedure to extend the distinguisher 3.5 rounds at both sides of the distinguisher, the data complexity for 21-round attack is $D = 4 \cdot \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q} = 4 \cdot \sqrt{s} \cdot 2^3 2 \cdot 2^{19.5} = \sqrt{s} \cdot 2^{53.5} = 2^{54}$(with $s = 2$). The corresponding time complexity is $2^{92.9}$ memory access and the memory complexity is 2^{54}.

Table 3 summarizes the results of related-tweakey rectangle attacks on all versions of SKINNY[4]. It shows that the attack is more effective on TK2 and TK3 versions which is consistent with the fact that the attack takes advantage of the key schedule of SKINNY, e.g., 27 rounds of SKINNY-64-192 can be covered. Besides, we can see that the attack covers the same rounds whatever the block size is (64-bit or 128-bit).

Table 3: Results of related-tweakey rectangle attack on SKINNY

		#Rounds	R_d	R_b	R_f	Data	Time	Memory
TK1	64-64	17	11	3.5	2.5	2^{54}	$2^{54.7}$	2^{54}
	128-128	17	11	3.5	2.5	$2^{106.5}$	2^{112}	$2^{106.5}$
TK2	64-128	21	14	3.5	3.5	2^{54}	$2^{87.9}$	2^{54}
	64-128	22	15	3.5	3.5	2^{63}	$2^{109.9}$	2^{63}
	128-256	22	15	3.5	3.5	$2^{126.5}$	2^{234}	$2^{126.5}$
TK3	64-192	27	19	4.5	3.5	2^{63}	$2^{154.9}$	2^{80}
	128-384	27	19	4.5	3.5	$2^{126.5}$	2^{322}	2^{160}

5 Related-Tweakey Differential Trails and Rectangle Attacks

In the previous sections truncated differential trails are investigated and used to attack reduced versions of SKINNY. This section focuses on differential trails of SKINNY. In the specification of SKINNY [BJK+16b], the authors only gave lower bounds on the number of differential active Sboxes in SKINNY. It is not clear whether exact differential trails satisfying the lower bounds exist or not, especially for SKINNY-128 which employs a differentially non-optimal 8-bit Sbox. This section sheds some light on it and gives some good related-tweakey differential trails for both SKINYY-64 and SKINNY-128 on which boomerang distinguishers can also be constructed.

5.1 Strategies for Finding Differential Trails

In this section our aim is to find optimal differential trails for SKINNY-64 and good differential trails for SKINNY-128 in a reasonable time. It is challenging to directly find optimal differential trails, even for block ciphers using 4-bit Sboxes, so we propose an indirect method for finding optimal differential trails for SKINNY-64, which is described as follows.

1. For an r-round SKINNY-64, find all truncated differential trails with the minimal number of active Sboxes $AS = AS_{\min}$, where $AS_{\min}$ denotes the minimal number of active Sboxes of truncated differential trails;

[4]The values in the table are roughly estimated and the exact complexity would be slightly deviated corresponding to the actual attack implementation.

Table 4: Bounds on the number of active Sboxes and probability of trails in SKINNY-64 under related-tweakey models

#rounds		6	7	8	9	10	11	12	13
TK1	$AS_{\min}$	6	10	13	16	23	32	38	41
	$-\log_2 p$	12	20	26	32	46	64	76	82
TK2	$AS_{\min}$	2	3	6	9	12	16	21	25
	$-\log_2 p_1$	4	6	12	20	None*	35	49	55
	$-\log_2 p_2$	4	6	12	20	28	35	48	55
TK3	$AS_{\min}$	0	1	2	3	6	10	13	16
	$-\log_2 p_1$	0	2	4	6	12	20	28	43
	$-\log_2 p_2$	0	2	4	6	12	20	28	38

* No solution is found for all truncated differentials with the minimal number of active Sboxes.

2. Based on the truncated differential trails, search for a best differential trail, namely the trail with the highest probability. If the best differential trail obtained has probability $p = 2^{-2AS_{\min}+i}$, $i \in \{0, 1, 2\}$, then this trail must be the optimal trail for an r-round SKINNY-64; otherwise, go to Step 3;

3. For $AS = AS_{\min} + 1$ to $\lfloor -\log_2 p/2 \rfloor$, find all truncated differential trails with AS active Sboxes. Based on the truncated differentials, search for a best differential trail with probability p'. If $p > p'$, let $p = p'$. Until $p = 2^{-2AS+i}, i \in \{0, 1, 2\}$ is satisfied, then the trail related to p is the optimal trail for an r-round SKINNY.

Following the designers of SKINNY [BJK+16b], we generate a Mixed-Integer Linear Programming (MILP) model to get truncated differential trails. Basic ideas of converting a differential searching problem into inequalities over integers are introduced in [MWGP11, SHS+13, SHW+14]. Once the active pattern, i.e., a truncated differential trail is given, the search for finding the optimal trail with MILP solvers is greatly sped up. In this way optimal differential trails can be found for SKINNY-64 as long as the number of truncated differential trails that needs to be traversed is reasonable, say less than 5000.

However, for SKINNY-128, 8-bit Sboxes are too heavy for Mixed-Integer Linear Programming solvers, so a dedicated searching algorithm used after obtaining a truncated differential trail with the minimal number of active Sboxes. Our idea is that given a truncated differential trail, there exists a few free active bytes for which all nonzero values are possible, and by traversing all the free active bytes a best differential trail which follows the given truncated differential can be found. In the case of SKINNY-128, only one truncated differential of minimal number of active Sboxes is considered.

5.2 Results of Differential Trails

Table 4 lists the results of SKINNY-64 from 6 rounds to 13 rounds under related-tweakey model. For each of TK1, TK2 and TK3, the first line shows the number of minimal active Sboxes; the second line presents the probability p_1 of best trails following the truncated differentials with the minimal number active Sboxes; and the last line shows the probability p_2 of optimal trails[5]. Under TK1, differential trails with probability $2^{-2AS_{\min}}$ are found in all cases. However, under TK2, and TK3, as the number of rounds increases, the probability of the optimal trail is much lower than $2^{-2AS_{\min}}$.

For SKINNY-128, only the cases that are promising in rectangle attacks are considered and the results are shown in Talbe 5. Note that the highest probability for the 8-bit Sbox used in SKINNY-128 is also 2^{-2}. As can be seen from the results that to make the

[5]For TK1, the second line and the third line are identical, so only one line is kept.

total number of active Sboxes lower, the average probability of each Sbox is also much lower. For example, the minimal number of active Sboxes of 9-round SKINNY-128-256 is 9, and following one of the truncated differential that satisfies this bound, the best trail has probability $2^{-32.42}$. While extending an 8-round trail with 6 active Sboxes one round back, we get a 9-round trail with 10 active Sboxes and probability 2^{-20}.

Table 5: Bounds on the number of active Sboxes and probability of trails in SKINNY-128 under related-tweakey models

#rounds		6	7	8	9	10	11	12
TK1	$AS_{\min}$	6	10	13				
	$-\log_2 p$	12	20	54(36)**				
TK2	$AS_{\min}$			6	9			
	$-\log_2 p$			12	32.42(20)			
TK3	$AS_{\min}$					6	10	13
	$-\log_2 p$					13	21	63.42(38)

** The values in parentheses are the probability of r-round differential trails obtained by extending a $(r-1)$-round differential trail.

These results show a trend that it is not likely to reach the bounds of $2^{-2AS_{\min}}$ as the number of rounds increase.

5.3 Related-Tweakey Rectangle Attacks

Based on differential trails in the previous subsection, rectangle distinguishers can be constructed. We follow the notations in Section 4. Suppose $E' = E_1 \circ E_0$ is the rectangle distinguisher. The probability of the upper trail for E_0 under tweakey difference Δ_1 (resp. lower trail for E_1 under Δ_2) $\Pr(\alpha \xrightarrow{\Delta_1} \beta)$ (resp. $\Pr(\gamma \xrightarrow{\Delta_2} \delta)$) is denoted by p (resp. q). If multiple trails are considered for E_0 or E_1, we denote $\sqrt{\Sigma_i \Pr^2(\alpha \xrightarrow{\Delta_1} \beta_i)}$ (resp. $\sqrt{\Sigma_j \Pr^2(\gamma_j \xrightarrow{\Delta_2} \delta)}$) by $\hat{p}$ (resp. $\hat{q}$). In the tweakey recovery attack, the same notations of $E = E_f \circ E_1 \circ E_0 \circ E_b$, m_b, r_b, t_b, m_f, r_f and t_f are used as in Section 4 and the number of right quartets s is set to be 4. Under TK1 where the key size is as large as the block size, the rectangle attack doesn't work as well as under TK2 and TK3. Thus, in this section only distinguishers under TK2 and TK3 are presented, together with the tweakey recovery attacks using the adapted key recovery algorithm in Appendix C.

5.3.1 21-Round attack on SKINNY-64-128

we construct a 17-round rectangle distinguisher by combining an 8-round upper trail with a 9-round lower trail. Details of these two trails are shown in Table 9. For E_0, if we fix the input difference and the tweakey difference according to the 8-round trail, there are 5477 trails which belong to 1563 differentials, and $\hat{p} = 2^{-7.15}$. Similarly for E_1, if we fix the output difference and the tweakey difference according to the 9-round trail, there are 24 trails which belong to 6 differentials, and $\hat{q} = 2^{-17.21}$. By extending 3 rounds backward and one round forward, we get following parameters: $r_b = 13c, t_b = 8c, m_b = c = 10c$, $r_f = 12c, t_f = 7c, m_f = 4c$ where $c = 4$ is the cell size. Using the adapted key recovery algorithm, a 21-round version of SKINNY-64-128 can be attacked with data complexity of $D = 2^{59.36}$ chosen plaintexts, time complexity of $2^{115.72}$ memory accesses and memory complexity of $2^{59.36}$ blocks.

5.3.2 22-Round attack on SKINNY-128-256

Following the truncated differentials with the minimal number of active Sboxes, we found an 8-round trail of probability 2^{-12} and a 9-round trail of probability $2^{-32.42}$. These two trails are displayed in Table 10 By extending the 8-round trail one round backward, we get a 9-round trail with probability 2^{-20} which is higher than the probability of the 9-round trail obtained directly from a 9-round truncated differential with the minimal number of active Sboxes. We choose the 9-round trail with probability $2^{-32.42}$ for E_0 and the other one with probability 2^{-20} for E_1. If the output difference and the tweakey difference are fixed, there are two trails with the same probability for E_1, so $\hat{q} = 2^{-19.50}$ and $\hat{p}\hat{q} = 2^{-51.92}$ for the 18-round rectangle distinguisher. Using the distinguisher from Round 4 to Round 21, we can attack 22 rounds with following parameters: $r_b = 14c, t_b = 8c, m_b = 10c$, $r_f = 12c, t_f = 8c, m_f = 5c$ where $c = 8$ is the cell size. Figure 10 gives a visualized view of the key recovery attack. Using the adapted key recovery algorithm, a 22-round version of SKINNY-128-256 can be attacked with data complexity of $D = 2^{118.92}$ chosen plaintexts, time complexity of $2^{250.84}$ memory accesses and memory complexity of 2^{120} blocks.

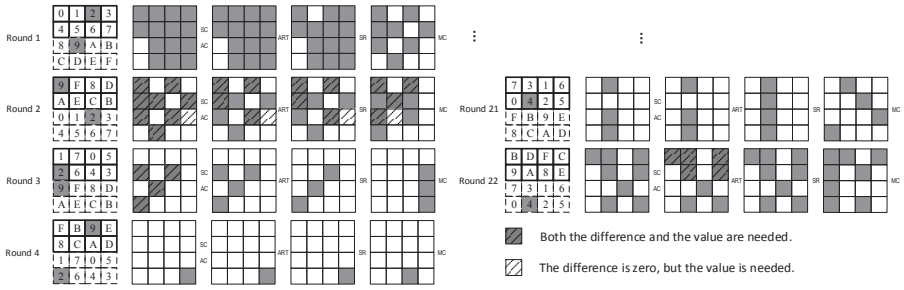


Figure 10: Tweakey recovery attack of 22-round SKINNY-128-256 using a 18-round related-tweakey rectangle distinguisher from Round 4 to Round 21

5.3.3 26-Round attack on SKINNY-64-192

The best 11-round related-tweakey differential trail for SKINNY-64-192 has probability 2^{-20} as shown in Table 11. The same trail is used for both E_0 and E_1 to get a 22-round distinguisher. If we fix input difference (output difference) and the tweakey difference, there are many trails. Taking 5000 trails into consideration, $\hat{p} = 2^{-14.51}$, $\hat{q} = 2^{-12.96}$. By extending 2 rounds backward and 2 rounds forward, we get following parameters: $r_b = 13c, t_b = 8c, m_b = 6c$, $r_f = 16c, t_f = 13c, m_f = 12c$ where $c = 4$. Consequently, 26 rounds of SKINNY-64-192 can be attacked with data complexity of $D = 2^{62.47}$ chosen plaintexts, time complexity of $2^{160.94}$ memory accesses and memory complexity of 2^{72} blocks.

5.3.4 27-Round attack on SKINNY-128-384

The best 11-round related-tweakey differential trail for SKINNY-128-384 has probability 2^{-21}, and with the same input difference and output difference there are two tails of the same probability. By extending this 11-round differential trail backward for one round we get a 12-round trail with probability 2^{-38}. We connect the 11-round trail and the 12-round trail to get a 23-round rectangle distinguisher. Using Boomerang switching technique [BK09] at the meeting point of two trails, four Sboxes of the lower trail can be saved. If we fix the tweakey difference and the output difference, there are 2^{10} trails with the same probability, thus $\hat{q} = 2^{-25}$. By extending 2 rounds backward and 2 rounds forward, we get following parameters: $r_b = 13c, t_b = 8c, m_b = 6c$, $r_f = 16c, t_f = 13c, m_f = 12c$

where $c = 8$. Thus, a 27-round SKINNY-128-384 can be attacked with data complexity of $D = 2^{112}$ chosen plaintexts, time complexity of 2^{300} memory accesses and memory complexity of 2^{144} blocks.

Results of related-tweakey rectangle attacks on SKINNY-n-$2n$ and SKINNY-n-$3n$ are summarized in Table 6. As can be seen that even though differential trails based rectangle distinguishers cover more rounds compared with rectangle distinguishers based on truncated differential trails as in Table 3, the total number of rounds attacked is not necessarily more.

Table 6: Results of related-tweakey rectangle attacks on SKINNY-n-$2n$ and SKINNY-n-$3n$

Model	Version	#Rounds	R_d	R_b	R_f	Data	Time	Memory
TK2	64-128	21	17	3	1	$2^{59.36}$	$2^{115.72}$	$2^{59.36}$
	128-256	22	18	3	1	$2^{118.92}$	$2^{250.84}$	2^{120}
TK3	64-192	26	22	2	2	$2^{62.47}$	$2^{160.94}$	2^{72}
	128-384	27	23	2	2	2^{112}	2^{300}	2^{144}

6 Conclusion

This paper analyzes the security of SKINNY under related-tweakey model using impossible differential and rectangle attacks. The results show that 19, 23 and 27 rounds of SKINNY-n-n/2n/3n can be attacked respectively with truncated differential trails. This paper also analyzes the security of SKINNY using actual differential trails and presents results for the rectangle attacks under TK2 and TK3. For SKINNY-128, better attack complexities are obtained (in terms of data and memory) compared to truncated differential trails. For SKINNY-64, the results show a trend that as the number of rounds increases, the probability of optimal differential trails is much lower than the probability derived from lower bounds of active Sboxes in SKINNY. Another interesting outcome of this work shows that the increased number of rounds in the distinguisher may not necessarily lead to more rounds attacked.

References

[AHMN13] Jean-Philippe Aumasson, Luca Henzen, Willi Meier, and María Naya-Plasencia. Quark: A lightweight hash. *J. Cryptology*, 26(2):313–339, 2013.

[BBS05] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. *J. Cryptology*, 18(4):291–311, 2005.

[BD08] Steve Babbage and Matthew Dodd. The MICKEY stream ciphers. In Matthew J. B. Robshaw and Olivier Billet, editors, *New Stream Cipher Designs - The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 191–209. Springer, 2008.

[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, Pro-*

ceeding, volume 2045 of *Lecture Notes in Computer Science*, pages 340–357. Springer, 2001.

[BDK02] Eli Biham, Orr Dunkelman, and Nathan Keller. New results on boomerang and rectangle attacks. In Joan Daemen and Vincent Rijmen, editors, *Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers*, volume 2365 of *Lecture Notes in Computer Science*, pages 1–16. Springer, 2002.

[BJK+16a] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. https://sites.google.com/site/skinnycipher/cryptanalysis-competition, 2016.

[BJK+16b] 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 Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016.

[BJK+16c] 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. *IACR Cryptology ePrint Archive*, 2016:660, 2016.

[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.

[BKL+07] Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.

[BKL+11] Andrey Bogdanov, Miroslav Knezevic, Gregor Leander, Deniz Toz, Kerem Varici, and Ingrid Verbauwhede. spongent: A lightweight hash function. In Preneel and Takagi [PT11], pages 312–325.

[BNS14] Christina Boura, María Naya-Plasencia, and Valentin Suder. Scrutinizing and improving impossible differential attacks: Applications to clefia, camellia, lblock and simon. In Sarkar and Iwata [SI14], pages 179–199.

[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.

[CP08] Christophe De Cannière and Bart Preneel. Trivium. In Matthew J. B. Robshaw and Olivier Billet, editors, *New Stream Cipher Designs - The eSTREAM*

Finalists, volume 4986 of *Lecture Notes in Computer Science*, pages 244–266. Springer, 2008.

[GPP11] Jian Guo, Thomas Peyrin, and Axel Poschmann. The PHOTON family of lightweight hash functions. In Phillip Rogaway, editor, *Advances in Cryptology - CRYPTO 2011 - 31st Annual Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2011. Proceedings*, volume 6841 of *Lecture Notes in Computer Science*, pages 222–239. Springer, 2011.

[GPPR11] Jian Guo, Thomas Peyrin, Axel Poschmann, and Matthew J. B. Robshaw. The LED block cipher. In Preneel and Takagi [PT11], pages 326–341.

[HJMM08] Martin Hell, Thomas Johansson, Alexander Maximov, and Willi Meier. The grain family of stream ciphers. In Matthew J. B. Robshaw and Olivier Billet, editors, *New Stream Cipher Designs - The eSTREAM Finalists*, volume 4986 of *Lecture Notes in Computer Science*, pages 179–190. Springer, 2008.

[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 Bruce Schneier, editor, *Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings*, volume 1978 of *Lecture Notes in Computer Science*, pages 75–93. Springer, 2000.

[Knu98] Lars Knudsen. Deal - a 128-bit block cipher. In *NIST AES Proposal*, 1998.

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In Chuankun Wu, Moti Yung, and Dongdai Lin, editors, *Information Security and Cryptology - 7th International Conference, Inscrypt 2011, Beijing, China, November 30 - December 3, 2011. Revised Selected Papers*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[PT11] Bart Preneel and Tsuyoshi Takagi, editors. *Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings*, volume 6917 of *Lecture Notes in Computer Science*. Springer, 2011.

[SHS+13] Siwei Sun, Lei Hu, Ling Song, Yonghong Xie, and Peng Wang. Automatic security evaluation of block ciphers with s-bp structures against related-key differential attacks. In Dongdai Lin, Shouhuai Xu, and Moti Yung, editors, *Information Security and Cryptology - 9th International Conference, Inscrypt 2013, Guangzhou, China, November 27-30, 2013, Revised Selected Papers*, volume 8567 of *Lecture Notes in Computer Science*, pages 39–51. Springer, 2013.

[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 Sarkar and Iwata [SI14], pages 158–178.

[SI14] 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 I*, volume 8873 of *Lecture Notes in Computer Science*. Springer, 2014.

[Wag99] David 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.

A Truncated Impossible Differentials

Table 7 lists all truncated impossible differentials we found under both single-tweakey model and related-tweakey models. These truncated impossible differentials are found by reusing the Mixed Integer Linear Programming model as in Section 5.

Table 7: Truncated impossible differentials

Model	(active cell position in input, active cell position in output)
SK 11 rounds	(12, 8), (13, 8), (12, 11),(14, 9), (14, 11), (15, 10) (12, 9), (13, 9), (13, 10),(15, 8), (14, 10), (15, 11)
Model	**(active cell position in key, active cell position in output)**
TK1 12 rounds	(8, 8), (9, 10), (13, 8), (10, 9), (12, 10), (13, 11) **(9, 9)**, (10, 8), (13, 9), (14, 9), (14, 10), (15, 10)
TK2 14 rounds	(8, 8), (12, 9), (14, 8), (**10, 10**), (12, 10), (9, 8) (8, 9), (14, 9), (15, 9), (11, 10), (15, 10), (14, 11)
TK3 16 rounds	(9, 8), (12, 9), **(10, 10)**, (8, 10), (15, 8), (12, 8) (9, 9), (10, 9), (11, 10), (13, 10), (11, 9), (12, 11)
Model	active cell position in key
TK1 12 rounds	8, 9, 10, 11, 12, 13, 14, 15
TK2 14 rounds	
TK3 16 rounds	

Note that, in the specification of SKINNY [BJK+16c] the authors state that there are 16 such truncated impossible differentials under single-tweakey model where only one cell is active in both the input and output, while we only find 12.

For related-tweakey models, no result is reported in [BJK+16c]. Our impossible differentials under related-tweakey model start with a fully passive state and a master key with only a special active cell which makes first $2z - 1$ subtweakeys with zero difference and end with one active cell after the SubCells operation of Round r where r is the number of rounds in the distinguisher, so each truncated impossible differential is represented with a tuple. As can be seen in Table 7, there are 12 such impossible differentials under TK1/TK2/TK3 with 12/14/16 rounds. Table 7 also lists impossible differentials that start as well as end with a fully passive state.

Note. Impossible attacks based on impossible differentials which start and end with a fully passive state always have a data complexity beyond codebook. This is explained by the formula of data complexity. First, let's check how many number structures should be

used. Following the notations explained in Section 3.2, suppose $(1-2^{-(c_{in}+c_{out})})^M = 2^{-g}$, and then we have $M = 2^{c_{in}+c_{out}} g\ln 2$. Therefore $M \cdot 2^{n-|\Delta_{out}|}$ pairs are obtained from plaintexts. If $M \cdot 2^{n-|\Delta_{out}|} > 2^{2|\Delta_{in}|}$, more than one structure should be used; otherwise, one structure is enough. The first case has been discussed in Section 3.2, so we come to the second case. In the second case, $D = 2\sqrt{M \cdot 2^{n-|\Delta_{out}|}}$. In a nutshell,

$$D = \begin{cases} 2^{c_{in}+c_{out}+n+1-|\Delta_{in}|-|\Delta_{out}|} g\ln 2 & \text{if } M \cdot 2^{n-|\Delta_{out}|} > 2^{2|\Delta_{in}|}, \\ 2\sqrt{2^{c_{in}+c_{out}+n-|\Delta_{out}|} g\ln 2} & \text{if } M \cdot 2^{n-|\Delta_{out}|} \leq 2^{2|\Delta_{in}|}. \end{cases}$$

Note that, $c_{in} = |\Delta_{in}|$, $c_{out} = |\Delta_{out}|$ for these distinguishers. Then the above formula of data complexity can be simplified as

$$D = \begin{cases} 2^{n+1} g\ln 2 & \text{if } |\Delta_{in}| < n + \log_2 g\ln 2, \\ 2\sqrt{2^{c_{in}+n} g\ln 2} & \text{if } |\Delta_{in}| \geq n + \log_2 g\ln 2. \end{cases}$$

It is learnt from the formula that the data complexity is beyond codebook, i.e., $D > 2^n$.

B Related Tweakey Impossible Differential Attack for SKINNY-n-n and SKINNY-n-3n

In this section, we present the details of our 19-round attack on SKINNY-n-n using related tweakey impossible differential attack. After that we just list parameters and results of SKINNY-n-3n.

B.1 SKINNY-n-n

Our 12-round distinguisher is placed between Round 5 to Round 17. A 6.5-round related-tweakey differential in the forward direction (having prob. 1) starting at Y_5 (after the SC and AC operations in Round 5) is concatenated to a 5.5-round related-tweakey differential (having prob. 1) starting in the reverse direction from Y_{17} (before the ART operation in Round 17). The contradiction happens in Round 12 at $X_{12}[12]$. The 12-round related-tweakey impossible differential is:

$$(0a00 \mid 0000 \mid 0000 \mid 0000) \xrightarrow{12r} (0000 \mid 0000 \mid 0N00 \mid 0000)$$

where a denotes a fixed non-zero difference and N denotes any non-zero difference. This 12-round distinguisher is extended 4.5 rounds at the top and 2.5 rounds in the bottom to cover 7 rounds in the tweakey recovery phase as shown in Fig. 11. Following the notations explained in Section 3.2, here E' covers 12 rounds, E_f (E_b) covers 2.5 (3.5) rounds, and $c_{in} = |\Delta_{in}| = 8c, c_{out} = 7c$, $|\Delta_{out}| = 7c$ and $|k_{in} \cup k_{out}| = 14c$ where c is the cell size. The steps of our tweakey recovery phase are as follows:

Data collection Consider a pair of structures S_1 and S_2, where, each structure consists of $2^{|\Delta_{in}|} = 2^{8c}$ plaintexts and for each plaintext pair $P_1 \in S_1$ & $P_2 \in S_2$, $P_1 \oplus P_2 = (0\ N\ 0\ N \mid N\ 0\ N\ N \mid 0\ N\ 0\ N \mid N\ 0\ 0\ 0)$, where N denotes any cell value. The total number of possible plaintext pairs is 2^{16c}. Invert back the plaintexts in S_1 and S_2 by one keyless round to get the original plaintexts. Encrypt the pool S_1 under $(TK\text{-}1)^1$ and the pool S_2 under $(TK\text{-}1)^2$ to obtain the corresponding ciphertexts. For each ciphertext pair, check whether $n - |\Delta_{out}| = 8c$ bits are zero or not, i.e., $\Delta W_{19}[1, 2, 3, 4, 10, 11, 12, 14]$ should be zero. Generate 2^x such pair of structures and repeat this for each pair of structures. In total there are $2^{x+2|\Delta_{in}|} = 2^{x+16c}$ plaintext pairs. This step requires $2^{x+|\Delta_{in}|+1} = 2^{x+8c+1}$ encryptions. Out of the total 2^{x+16c} ciphertext pairs, $M = 2^{x+2|\Delta_{in}|-n+|\Delta_{out}|} = 2^{x+8c}$ pairs are expected

Round i	Row(1)	Row(2)
2	(0 0 0 0)	(0 0 0 a)
16	(0 0 0 a)	(0 0 0 0)
18	(0 0 0 0)	(0 0 0 a)

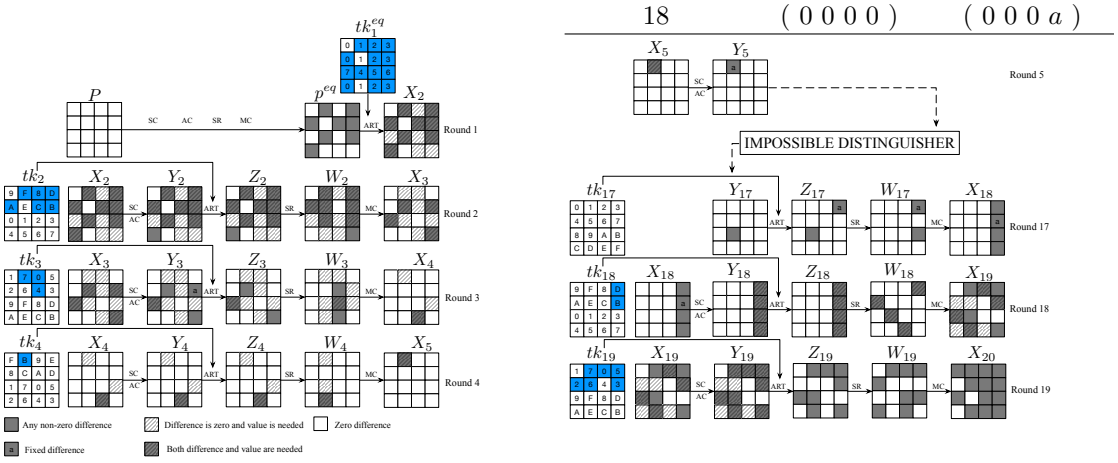


Figure 11: 19-round attack against SKINNY-n-n.

to remain. With the help of a hash table, this step takes a memory complexity of $2^{|\Delta_{in}|} = 2^{8c}$.[6]

Tweakey recovery For M pairs:

(a) Check whether $\Delta X_{19}[9] = \Delta X_{19}[13]$ holds. Since $\Delta X_{19}[1] = \Delta X_{19}[13]$ and $\Delta Y_{19}[1]$ is known from the ciphertexts, use Lemma 1 to compute $X_{19}[1], Y_{19}[1]$ and $tk_{19}[1]$. Compute $\Delta X_{19}[15]$. Since $\Delta X_{19}[3] = \Delta X_{19}[7] = \Delta X_{19}[15]$, and $\Delta Y_{19}[3]$, $\Delta Y_{19}[7]$ are known from the ciphertexts, compute $tk_{19}[3], tk_{19}[7]$. This step takes a time complexity of M and the number of tests left for the next step is $M \cdot 2^{-c}$.

(b) Guess $tk_{19}[2]$. Using $tk_{19}[2]$, compute $W_{18}[14], \Delta W_{18}[14]$ and then compute $X_{18}[15], \Delta X_{18}[15]$. Since $\Delta Y_{18}[11]$ can be computed from $\Delta X_{19}[1, 5, 13]$ and $\Delta Y_{18}[11] = \Delta Y_{18}[11] \oplus a$, $Y_{18}[11]$ can be recovered by Lemma 1, and then $tk_{19}[5]$ also. Similarly, $\Delta X_{18}[3] = \Delta X_{18}[15]$, $\Delta Y_{18}[3], Z_{18}[3]$ can be computed from $X_{19}, \Delta X_{19}$, so $tk_{18}[3]$ can be calculated by solving the equation of the Sbox. This step takes a time complexity of M and the number of tests left for the next step is M.

(c) With $tk_1[0] = tk_{19}[2], tk_1[6] = tk_{19}[5]$, compute $\Delta W_2[5], \Delta W_2[9]$, and check whether $\Delta W_2[5] = \Delta W_2[9]$ or not. This is a one-cell filter. Since $\Delta Y_2[1] = \Delta W_2[5]$ and $\Delta X_2[1]$ is known from the plaintexts, $tk_1[2]$ can be computed by solving the equation of the Sbox. With $tk_1[0], tk_1[3] = tk_{19}[7], \Delta W_2[3], \Delta W_2[15]$ can be computed. Since $\Delta W_2[7] = \Delta W_2[11] = \Delta W_2[3] + \Delta W_2[15]$, and $\Delta X_2[6], \Delta X_2[9]$ are known from the plaintexts, $tk_1[2], tk_1[5]$ can be computed. Now $\Delta W_2[3, 7, 11, 15]$ is known as well as $W_2[3, 7, 11, 15]$, if the equivalent subtweakey is applied after MC. Since $tk_2[3] = tk_{18}[3]$ is known, $\Delta Y_3[7]$ can be

[6]For each pair of structures under $(TK\text{-}1)^1$ and $(TK\text{-}1)^2$, insert the messages of one structure into a hash table according to the the value of $W_{19}[1, 2, 3, 4, 10, 11, 12, 14]$, and then use messages in the other structure to find message pairs under the related tweakeys such that $W_{19}[1, 2, 3, 4, 10, 11, 12, 14]$ are equal by searching the built hash table. An attacker can do this for each structure pair. In this way, the time complexity is equal to the data complexity, and the memory complexity remains to be the size of one structure.

computed and it can be checked whether $\Delta Y_3[7] = a$. This is a one-cell filter. Similarly, $\Delta Y_3[15]$ can be computed. Again, since $\Delta Y_3[5] = \Delta Y_3[8] = \Delta Y_3[15]$, and $\Delta X_3[5], \Delta X_3[8]$ are known from the plaintexts, $tk_2[1], tk_2[7]$ can be calculated by solving the equations of the Sbox. This step takes a time complexity of M and the number of tests left for the next step is $M \cdot 2^{-2c}$.

(d) With $tk_{19}[4] = tk_1[2], tk_{18}[7] = tk_2[7]$, compute $\Delta X_{18}[7], X_{18}[7]$ and check whether $\Delta X_{18}[7] = a$. This is a one-cell filter. This step takes a time complexity of $M \cdot 2^{-2c}$ and the number of tests left for the next step is $M \cdot 2^{-3c}$.

(e) Guess $tk_2[2], tk_2[4], tk_2[6]$ and compute $\Delta Y_5[1]$ and check whether $\Delta Y_5[1] = a$. This step takes a time complexity of M and the number of tests verifying the impossible distinguisher is $M \cdot 2^{-c}$, i.e., for each pair there is 2^{-c} 14-cell key on average that verifies the impossible distinguisher and is wrong.

The total number of tweakeys left is:

$$TK_{rem} = 2^{14c}(1 - 2^{-15c})^M \tag{2}$$

Brute force For the tweakeys that remain, we guess the other two tweakey cells and exhaustively search the $TK_{rem} \times 2^{2c}$ tweakeys to find the correct tweakey.

Attack Complexities. The time complexity of the tweakey recovery phase is M. Following the formulas derived in Section 3.2, the data complexity of the attack is $D = 2^{n+1-c} g\ln 2$ chosen plaintexts and the total time complexity is:

$$T = D + M + TK_{rem} \times 2^{2c}.$$

The memory complexity is the storage for one structure and wrong keys. For, $c = 4$, we set $g = 2$, then $D = 2^{61.47}, M = 2^{60.47}$, $TK_{rem} \times 2^{2c} = 2^{62}$, $T = 2^{63.03}$ and the memory complexity is 2^{56}. For, $c = 8$, we set $g = 4$, then $D = 2^{122.47}, M = 2^{121.47}$, $TK_{rem} \times 2^{2c} = 2^{124}$, $T = 2^{124.60}$ and the memory complexity is 2^{112}.

B.2 SKINNY-n-$3n$

A 16-round distinguisher E' is placed between Round 7 to Round 21 to attack 27 rounds of SKINNY-n-$3n$. In the attack, $|k_{in} \cup k_{out}| = 46c$, $c_{in} = 16c$, $|\Delta_{in}| = 16c$, $c_{out} = 15c$, $|\Delta_{out}| = 16c$. Less than one structure is used. Suppose 2^m messages are generated under two related-tweakey respectively. Then $M = 2^{2m}$,$D = 2^{m+1} = 2 \cdot \sqrt{M}$. Suppose $(1 - 2^{-(c_{in}+c_{out})})^M = 2^{-g}$. Then $M = 2^{c_{in}+c_{out}} g\ln 2$. The time complexity is

$$T = D + M \cdot 2^{|k_{in} \cup k_{out}| - (c_{in}+c_{out})} + 2^{3n-g}.$$

For $c = 4$, set $g = 3$, then $D = 2^{15.5c+1}\sqrt{g\ln 2} = 2^{63.53}$, $T = 2^{189}$, memory complexity is 2^{184}. For $c = 8$, set $g = 6$, then $D = 2^{15.5c+1}\sqrt{g\ln 2} = 2^{126.03}$, $T = 2^{378}$, memory complexity is 2^{368}.

Attack where tweak is used. Suppose a w-bit tweak is used. The tweak is loaded into the first w bits of the first tweakey state $TK - 1$, followed by the key material. For $0 \leq w \leq 10c$, we have $|k_{in} \cup k_{out}| = 3n - w - 2c < 2n - w$ and the above attack still has a time complexity below 2^{3n-w}, so 27-round SKINNY-n-$2n$ is valid in such cases. For $w = 10c$, results are as follows. For $c = 4$, we set $g = 3$, then $D = 2^{63.53}$, $T = 2^{149}$ and the memory complexity is 2^{144}. For $c = 8$, we set $g = 6$, then $D = 2^{126.03}$, $T = 2^{298}$ and the memory complexity is 2^{288}.

C Key Recovery Algorithm Based on Related-Tweakey Rectangle Distinguishers

The algorithm in this section is adapted from Biham et al's algorithm where there is no key difference and applicable to other block ciphers under related key model as long as the key schedule is linear. We follow the notations in , our algorithm proceeds as follows:

1. Let K_1 be the secret key and $K_2 = K_1 \oplus \Delta_1$, $K_3 = K_1 \oplus \Delta_2$ and $K_4 = K_1 \oplus \Delta_1 \oplus \Delta_2$. Create $y = \sqrt{s} \cdot 2^{n/2-r_b}/\hat{p}\hat{q}$ structures of 2^{r_b} plaintexts each, where s is the expected number of right quartets. Encrypt these y structures with K_1, K_2 respectively. Also, create y structures of 2^{r_b} plaintexts each and encrypt these y structures with K_3, K_4 respectively.

2. Initialize a list of $2^{m_b+m_f}$ counters, each of which corresponds a $(m_b + m_f)$-bit subkey guess.

3. Under each key there are $M = y2^{r_b}$ ciphertexts. Denote the ciphertext sets under $K_i, i \in \{1, 2, 3, 4\}$ by $L_i, i \in \{1, 2, 3, 4\}$. Process (L_1, L_3) and (L_2, L_4) independently. Insert the L_1 ciphertexts into a hash table H_1 according to the $n - r_f$ ciphertext bits that are set to 0 in V_f. Then for each ciphertext in L_3, try to find collisions of this ciphertext and ciphertexts in H_1. If a ciphertext pair from L_1 and L_3 agrees on the $n - r_f$ bits, check whether the ciphertext difference is in U_f. Do the same thing for (L_2, L_4).

4. For each collision $(C_1, C_3) \in L_1 \times L_3$ which remains after Step 2, denote C_i's structure under K_j by $S_{C_i}^{K_j}$ and attach to C_1 the index of $S_{C_3}^{K_3}$. For each collision $(C_2, C_4) \in L_2 \times L_4$, attach to C_2 the index of $S_{C_4}^{K^4}$.

5. In each structure S under K_1, K_2, we search for two ciphertexts $C_1 \in S^{K_1}, C_2 \in S^{K_2}$ which are attached to some other structures under K_3 and K_4 respectively. When we find such a pair, first check that structures that C_1 and C_2 are attached to are the same, and that the corresponding plaintext difference $P_1 \oplus P_2$ is in U_b. Also, check the difference of the plaintexts which P_1 and P_2 are related to.

6. For all the quartets which passed Step 5 denote by (P_1, P_2, P_4, P_4) the plaintexts of a quartet and by (C_1, C_2, C_3, C_4) the corresponding ciphertexts under (K_1, K_2, K_3, K_4). Increment the counters of $(m_b + m_f)$-bit subkeys which satisfy that $E_b^{K_1}(P_1) \oplus E_b^{K_2}(P_2) = E_b^{K_3}(P_3) \oplus E_b^{K_4}(P_4) = \alpha$ and $(E_f^{K1})^{-1}(C_1) \oplus (E_f^{K3})^{-1}(C_3) = (E_f^{K2})^{-1}(C_2) \oplus (E_f^{K4})^{-1}(C_4) = \delta$.

7. Output the subkey with maximal number of hits.

The data complexity of the attack is $D = 4M$ chosen plaintexts. The time complexity of Step 1 is D encryptions. The time complexity of Step 2 is only $2^{m_b+m_f}$ memory access using suitable data structures.

Step 3 requires $2M$ memory access for the insertion for both (L_1, L_3) and (L_2, L_4) and thus $4M$ memory accesses in total. The colliding pairs for both (L_1, L_3) and (L_2, L_4) is $M^2 \cdot 2^{r_f-n}$. After checking that the ciphertext difference is in U_f, about $M^2 \cdot 2^{t_f-n}$ colliding pairs remain for both (L_1, L_3) and (L_2, L_4) is $M^2 \cdot 2^{r_f-n}$. The time complexity of this step is $4M + 2 \cdot M^2 \cdot 2^{r_f-n}$ memory accesses.

Step 4 requires one memory access for each pair remained after Step 3. The time complexity is $2 \cdot M^2 \cdot 2^{t_f-n}$ memory accesses.

Step 5 searches for possible quartets. In structures under K_1 (or K_2), there are $M^2 \cdot 2^{t_f-n}$ attachments which distribute over y structures. Therefore in each structure

K_1 (or K_2), we have about $M^2 \cdot 2^{t_f-n}/y = M \cdot 2^{t_f-r_b-n}$ attachments on average. In the same structure under K_1 and K_2, there are $(M \cdot 2^{t_f+r_b-n})^2/y$ pairs of (C_1, C_2) where the ciphertexts say C_3 and C_4 that C_1 and C_2 are related to are also in the same structure, i.e. $M^2 \cdot 2^{2t_f+2r_b-2n}$ possible quartets (C_1, C_2, C_3, C_4) under (K_1, K_2, K_3, K_4). Suppose that the corresponding plaintext is (P_1, P_2, P_3, P_4). We check that both $P_1 \oplus P_2$ and $P_3 \oplus P_4$ are in U_b. The probability that both $P_1 \oplus P_2$ and $P_3 \oplus P_4$ are in U_b is $2^{(t_b-r_b)\times 2}$, so $M^2 \cdot 2^{2t_f+2t_b-2n}$ quartets will be left. In total this step takes $M^2 \cdot 2^{2t_f+2r_b-2n}(1+2^{t_b-rb})$ memory accesses.

Step 6 deduces the right subkey from the remaining quartets. Note that a right quartets satisfies $E_b^{K_1}(P_1) \oplus E_b^{K_2}(P_2) = E_b^{K_3}(P_3) \oplus E_b^{K_4}(P_4) = \alpha$ and key differences Δ_1, Δ_2 are chosen in advance and known. With these two key differences, K_2, K_3, K_4 can be computed from K_1. A right quartet must agree on the m_f bits of K_1. There are 2^{t_b} possible input differences that may lead to α difference after E_b. Therefore, $2^{m_b-t_b}$ subkeys on average take one of the difference in U_b to α. For two pairs in a quartet, they agree on $(2^{m_b-t_b})^2/(2 \cdot 2^{m_b}) = 2^{m_b-2t_b-1}$ subkeys for E_b. Do the same analysis for E_f part with the corresponding ciphertexts, and we get $2^{m_f-2t_f-1}$ subkey suggestions for E_f. Each remaining quartet suggests $2^{m_b+m_f-2t_b-2t_f-2}$ subkeys. There are $M^2 \cdot 2^{2t_f+2t_b-2n} \cdot 2^{m_b+m_f-2t_b-2t_f-2} = M^2 \cdot 2^{m_b+m_f-2n-2}$ hits. For a wrong key, there are $M^2 \cdot 2^{-2n-2}$ $(< 1/4)$ hits, while for a right key there are $c(c > 1)$ hits. Using hash tables, this step can be implemented with $M^2 \cdot 2^{2t_f+2t_b-2n} \cdot 2^{m_b-t_b} \cdot 2 + M^2 \cdot 2^{2t_f+2t_b-2n} \cdot 2^{m_f-t_f} \cdot 2 = M^2 \cdot 2^{t_b+t_f-2n+1}(2^{m_b+t_f} + 2^{m_f+t_b})$ memory accesses.

Step 7 requires $1 \sim 4$ memory accesses using efficient data structures.

More information about the original algorithm of Biham et al. can be found in [BDK02].

Overall, this algorithm requires $D = 4y2^{r_b} = 4 \cdot \sqrt{s} \cdot 2^{r_b} \cdot 2^{n/2-r_b}/\hat{p}\hat{q} = 4 \cdot \sqrt{s} \cdot 2^{n/2}/\hat{p}\hat{q}$ chosen plaintexts, and time complexity of $4M + 2 \cdot M^2 \cdot 2^{r_f-n} + 2 \cdot M^2 \cdot 2^{t_f-n} + M^2 \cdot 2^{2t_f+2t_b-2n}(1+2^{t_b-r_b}) + M^2 \cdot 2^{t_b+t_f-2n+1}(2^{m_b+t_f}+2^{m_f+t_b})$ memory accesses. The memory complexity is $4M + 2^{t_b} + 2^{t_f} + 2^{m_b+m_f}$.

D Boomerang Quartets

Even though rectangle distinguishers are used to attack SKINNY, our experiment works on a boomerang distinguisher of SKINNY-64-128, since a rectangle distinguisher is valid as long as the corresponding boomerang distinguisher is valid. What's more, the probability of a rectangle distinguisher is $2^{-n}p^2q^2$, while the probability of the corresponding boomerang is p^2q^2 which is more practical for verification.

The experiment is implemented in standard C programming language. We aim at finding at least one right quartet that follow our related-tweakey boomerang distinguisher within $1/p^2q^2$ tested quartets. For the 13-round distinguisher, one right quartet is found among a total of 2^{12} tested quartets while the estimated probability of the distinguisher is 2^{-27}. In total we found $2^{10.2}$ right quartets in a searching space of 2^{20} quartets. For the 14-round distinguisher, 34 right quartets are found in a searching space of $2^{40.8}$, while the estimated probability is 2^{-40}. One of the right quartets is displayed in Table 8.

Note that our distinguisher starts from AddConstant instead of SubCells. According to our 14-round distinguisher, the plaintext difference is simply one cell difference of position 2 (we choose 1 as the actual difference value) so as to the ciphertext difference whose active cell position is 12 (we choose 2 as the actual difference value).

Table 8: One of the quartets of the 15-round boomerang distinguisher of SKINNY-64-128

K_1	ed19 f85b 920d 6862	8953 f24b fd90 8f60
Δ_1	00e0 0000 0000 0000	00f0 0000 0000 0000
Δ_2	0000 0e00 0000 0000	0000 0d00 0000 0000
P_1, C_1	8ae9 28a6 9000 0000	0b08 912a e543 25e0
P_2, C_2	8af9 28a6 9000 0000	a4c8 c51b bc2c 646b
P_3, C_3	993a cad5 00b8 af00	0b08 912a e541 25e0
P_4, C_4	994a cad5 00b8 af00	a4c8 c51b bc2e 646b

E Differential Trails

In this section, we list the differential trails used in rectangle attacks in Section 5. Note that we only present the master key difference (since the subtweakey differences are determined by master key difference) and the state differences before and after SubCells of each round, where each cell (byte or nibble) of zero difference is denote by '0' and each non-zero cell is given in hexadecimal, ordered from left to right. For the sake of brevity, a round with whole zero state difference may be omitted.

Table 9: Trails for SKINNY-64-128

	8-round upper trail $p = 2^{-12}$	9-round lower trail $q = 2^{-20}$
ΔK	0,0,0,0, 0,0,0,0, 6,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 9,0,0,0, 0,0,0,0	0,0,c,0, 0,0,0,0, 0,0,0,0, e,0,0,0 0,0,f,0, 0,0,0,0, 0,0,0,0, b,0,0,0
R1	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, 0,0,0,8	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, 0,0,0,3
R2	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, 0,0,0,0	0,0,0,0, 0,0,3,0, 0,0,0,0, 0,0,3,0 0,0,0,0, 0,0,d,0, 0,0,0,0, 0,0,c,0
R3	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,c,0,0, 0,0,0,0, 0,0,0,4, 0,0,0,0 0,2,0,0, 0,0,0,0, 0,0,0,2, 0,0,0,0
R4	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,0,0,0, 0,2,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,1,0,0, 0,0,0,0, 0,0,0,0
R7	0,0,0,0, 0,0,0,0, 0,0,0,b, 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, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
R8	0,1,0,0, 0,0,0,0, 0,1,0,0, 0,1,0,0 0,8,0,0, 0,0,0,0, 0,8,0,0, 0,8,0,0	0,0,0,0, 0,0,0,0, 0,0,4,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,2,0, 0,0,0,0
R9		2,0,0,0, 0,0,0,0, 0,0,0,0, 2,0,0,0 6,0,0,0, 0,0,0,0, 0,0,0,0, 5,0,0,0

Table 10: Trails for SKINNY-128-256

	8-round trail $q = 2^{-12}$	9-round trail $p = 2^{-32.42}$
ΔK	0,0,0,0, 0,0,0,0, 55,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 8a,0,0,0, 0,0,0,0	0,0,bb,0, 0,0,0,0, 0,0,0,0, a8,0,0,0 0,0,db,0, 0,0,0,0, 0,0,0,0, 15,0,0,0
R1	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,08 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,10	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,11 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,60
R2	0,0,10,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,40,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,0,0,0, 0,0,60,0, 0,0,0,0, 0,0,60,0 0,0,0,0, 0,0,96,0, 0,0,0,0, 0,0,44,0
R3	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,44,0,0, 0,0,0,0, 0,0,0,14, 0,0,0,0 0,d5,0,0, 0,0,0,0, 0,0,0,d5, 0,0,0,0
R4	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,0,0,0, 0,d5,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,02,0,0, 0,0,0,0, 0,0,0,0
R7	0,0,0,0, 0,0,0,0, 0,0,0,01, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,20, 0,0,0,0	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
R8	0,20,0,0, 0,0,0,0, 0,20,0,0, 0,20,0,0 0,80,0,0, 0,0,0,0, 0,80,0,0, 0,80,0,0	0,0,0,0, 0,0,0,0, 0,0,56,0, 0,0,0,0 0,0,0,0, 0,0,0,0, 0,0,08,0, 0,0,0,0
R9		08,0,0,0, 0,0,0,0, 0,0,0,0, 08,0,0,0 10,0,0,0, 0,0,0,0, 0,0,0,0, 10,0,0,0

Table 11: Trails for SKINNY under TK3

	11-round trail for SKINNY-64 $p = 2^{-20}$	11-round trail for SKINNY-128 $q = 2^{-21}$
ΔK	0,a,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,44,0,0, 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	0,04,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
	0,d,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	0,51,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
R1	0,2,0,0, 1,0,0,0, 0,0,0,1, 0,0,1,0	0,0c,0,0, 02,0,0,0, 0,0,0,02, 0,0,02,0
	0,5,0,0, b,0,0,0, 0,0,0,b, 0,0,b,0	0,11,0,0, 08,0,0,0, 0,0,0,08, 0,0,08,0
R2	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,b,0,0	0,0,0,0, 0,0,0,0, 0,0,0,0, 0,08,0,0
	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, 0,10,0,0
R3	1,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	10,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
	8,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0	40,0,0,0, 0,0,0,0, 0,0,0,0, 0,0,0,0
R10	0,0,0,0, 0,0,0,0, 0,0,0,8, 0,0,0,0	0,0,0,0, 0,0,0,0, 0,0,0,40, 0,0,0,0
	0,0,0,0, 0,0,0,0, 0,0,0,4, 0,0,0,0	0,0,0,0, 0,0,0,0, 0,0,0,04, 0,0,0,0
R11	0,4,0,0, 0,0,0,0, 0,4,0,0, 0,4,0,0	0,04,0,0, 0,0,0,0, 0,04,0,0, 0,04,0,0
	0,2,0,0, 0,0,0,0, 0,2,0,0, 0,2,0,0	0,01,0,0, 0,0,0,0, 0,01,0,0, 0,01,0,0