IACR Transactions on Symmetric Cryptology
ISSN 2519-173X, Vol. 0, No. 0, pp. 0–0. DOI:10.46586/tosc.v0.i0.0-0

Impossible Boomerang Attacks Revisited

Applications to Deoxys-BC, Joltik-BC and SKINNY

Jianing Zhang, Haoyang Wang(✉) and Deng Tang

Shanghai Jiao Tong University, Shanghai, China
{zhangjn,haoyang.wang,dengtang}@sjtu.edu.cn

Abstract. The impossible boomerang (IB) attack was first introduced by Lu in his doctoral thesis and subsequently published at DCC in 2011. The IB attack is a variant of the impossible differential (ID) attack by incorporating the idea of the boomerang attack. In this paper, we revisit the IB attack, and introduce the incompatibility of two characteristics in boomerang to the construction of an IB distinguisher. With our methodology, all the constructions of IB distinguisher are represented in a unified manner. Moreover, we show that the related-(twea)key IB distinguishers possess more freedom than the ones of ID so that it can cover more rounds.

We also propose a new tool based on Mixed-Integer Quadratically-Constrained Programming (MIQCP) to search for IB attacks. To illustrate the power of the IB attack, we mount attacks against three tweakable block ciphers: `Deoxys-BC`, `Joltik-BC` and `SKINNY`. For `Deoxys-BC`, we propose a related-tweakey IB attack on 14-round `Deoxys-BC-384`, which improves the best previous related-tweakey ID attack by 2 rounds, and we improve the data complexity of the best previous related-tweakey ID attack on 10-round `Deoxys-BC-256`. For `Joltik-BC`, we propose the best attacks against 10-round `Joltik-BC-128` and 14-round `Joltik-BC-192` with related-tweakey IB attack. For `SKINNY`-n-$3n$, we propose a 27-round related-tweakey IB attack, which improves both the time and the memory complexities of the best previous ID attack. We also propose the first related-tweakey IB attack on 28-round `SKINNY`-n-$3n$, which improves the previous best ID attack by one round.

Keywords: Impossible Boomerang Attack · MIQCP · `Deoxys-BC` · `Joltik-BC` · `SKINNY`

1 Introduction

Differential cryptanalysis, one of the most important attacks on block ciphers, was first introduced by Biham and Shamir in 1990 [BS91], and has since been widely studied. The idea is to study the propagation of differences inside an iterated block cipher and construct a high-probability differential. An adversary can use this high-probability differential to recover (part of) the key bits. Built upon the foundation of differential attacks, various derivative cryptanalytic methods have been developed, with *impossible differential attacks* [BBS99a] and *boomerang attacks* [Wag99] being representative examples.

The impossible differential (ID) attack was first proposed independently by Knudsen [Knu98] and Biham [BBS99a]. Unlike traditional differential cryptanalysis, the impossible differential attack uses a differential with a probability of 0. The adversary can use this impossible differential to eliminate wrong key bits. Since its introduction, the impossible differential attack has gained widespread attention and has effectively targeted some block ciphers. The most significant step of the impossible differential attack is to construct a distinguisher that covers as many rounds as possible, and the typical way is to use the *miss-in-the-middle* technique [BBS99b]. In the miss-in-the-middle technique, cryptanalysts try to track the propagation of input and output differences from the encryption and decryption directions, respectively. If there is a contradiction at certain points in between, then an impossible differential has been identified.

The boomerang attack was first introduced by Wagner [Wag99], sharing some similarities with the impossible differential attack in that it involves cascading two characteristics. In the boomerang attack, two short high-probability characteristics are combined to form a longer distinguisher, aiming to achieve a better attack.

At INDOCRYPT 2003, Kim *et al.* [KHS+03] proposed the first computer-aided tool for searching impossible differentials called the $\mathcal{U}$-method. Later, some improved methods such as the WW-method [WW12] and the UID-method [LLWG14] were introduced by Wu *et al.* in 2012 and Luo *et al.* in 2014, respectively. Mixed-integer linear programming (MILP) was introduced to search characteristics by Mouha *et al.* in 2011 [MWGP11], and later was refined by Sun *et al.* in 2014 [SHW+14]. Based on the feasibility of the model, Cui *et al.* [CJF+16] was the first to apply MILP to the search for impossible differential distinguishers. At CRYPTO 2016, Derbez and Fouque [DF16] developed a new Generalized Demirci-Selçuk search algorithm for a large class of block ciphers and applied the algorithm to search for impossible differential attacks. At EUROCRYPT 2017, Sasaki *et al.* [ST17] applied MILP to block ciphers with 8-bit S-boxes and tried to find contradictions from linear layers. In recent works, [ARS+22, HPW22, CLH+23] all use the solvability of the MILP/SMT problems to determine IDs. In [HSE23], Hadipour *et al.* proposed a generic CP-based model to find full impossible differential attacks without using the infeasibility.

For the automatic searching tools for the boomerang attack and its variants, Cid *et al.* [CHP+17] in 2017 firstly introduced MILP to characterize the ladder switch in the boomerang distinguishers for `Deoxys-BC`. Later, Zhao *et al.* [ZDJ19] used the MILP method to search for boomerang distinguishers of `Deoxys-BC` including BDT effect. Delaune *et al.* [DDV20] described a new MILP model to search for truncated boomerang characteristics and a CP model to instantiate the truncated boomerang characteristics for `SKINNY` with the effects of DDT, BCT, UBCT, etc. At EUROCRYPT 2022 [DQSW22], Dong *et al.* managed to search rectangle distinguishers by proposing a new key guessing strategy with MILP and CP models. In [DEFN22], Derbez *et al.* extended the MILP model to search for a complete boomerang attack, which is applied to the attack on `AES-192`.

Building upon the foundations of the impossible differential attack and the boomerang attack, we think it is an interesting idea to combine them together, with the name *impossible boomerang attack*. The impossible boomerang (IB) attack was firstly proposed by Lu in his doctoral thesis [Lu08] and subsequently published in [Lu11]. In [Lu08, Lu11], the author introduced the definition and extended its application to the related-key scenario. With this new technique, Lu proposed several single-key attacks on 6-round `AES-128` and 7-round `AES-192`/`AES-256`, and related-key attacks on 8-round `AES-192` and 9-round `AES-256`. In [CY09], Choy and Yap adapted the $\mathcal{U}$-method [KHS+03] to align with the impossible boomerang attack and computed the maximum length of impossible boomerang distinguishers for ciphers with `MARS`-like [BCD+98]/`RC6`-like [RRSY98] structure.

Our Contributions. In this work, we revisit the impossible boomerang attack, providing a systematic overview of the contradiction conditions, a comparison with the impossible differential attack, and the key recovery process under the related-key setting. Based on Mixed-Integer Quadratically-Constrained Programming (MIQCP), we propose a new automatic searching tool for impossible boomerang attacks and successfully apply it to three block ciphers: `Deoxys-BC`, `Joltik-BC` and `SKINNY`. The main results are summarized in Tables 1 and 2.

Table 1: Summary of our cryptanalytic results. Apart from the ID-type attacks, we also provide the best attacks against the targeted ciphers for a more comprehensive comparison. MITM = Meet-in-the-middle attack, Boom. = Boomerang attack, Rect. = Rectangle attack. All the attacks listed below are under related-(twea)key settings.

Cipher	#R	Key Size	Tweak Size	Attack	#K	Data†	Time	Memory	Ref.
Joltik-BC-128	10	> 109	<19	ID	2	2^{71}	$2^{109.5}$	2^{104}	[ZD18]
	10	> 93	< 35	IB	4	$2^{68.3}$	$2^{93.8}$	$2^{92.6}$	Section 4.3
Joltik-BC-192	11	$= 128$	$= 64$	MITM	2	2^{53}	2^{123}	2^{114}	[LC21]
	13	$= 128$	$= 64$	IB	4	$2^{68.9}$	$2^{122.1}$	2^{96}	Section 4.4
	14	> 183	< 9	IB	4	$2^{66.7}$	$2^{183.65}$	2^{160}	Section 4.5
Deoxys-BC-256	9	$= 128$	$= 128$	ID	2	2^{118}	2^{118}	2^{102}	[MMS18]
	10	> 173	< 83	ID	2	2^{135}	2^{173}	–	[ZDW19]
	10	> 186	< 70	IB	4	$2^{132.8}$	$2^{186.66}$	$2^{181.6}$	Section 4.3
	11	> 222	< 34	Rect.	4	$2^{126.78}$	$2^{222.49}$	2^{128}	[SZY+22]
	11	> 218	< 38	Boom.	4	$2^{122.4}$	$2^{218.65}$	2^{128}	[SZY+22]
Deoxys-BC-384	12	> 329	< 55	ID‡	2	$2^{135.3}$	$2^{329.7}$	2^{312}	Appendix C
	13	$= 256$	$= 128$	IB	4	$2^{133.3}$	$2^{243.5}$	2^{192}	Section 4.4
	14	> 368	< 16	IB	4	$2^{130.9}$	2^{368}	2^{320}	Section 4.5
	14	> 278	< 106	Boom.	4	2^{129}	$2^{278.8}$	2^{129}	[BL23]
	15	> 371	< 13	Rect.	4	$2^{115.7}$	$2^{371.7}$	2^{128}	[SYC+24]
SKINNY-64-192	27	> 189	< 3	ID	2	$2^{63.53}$	2^{189}	2^{184}	[LGS17]
	27	> 183	< 9	ID	2	$2^{63.64}$	$2^{183.26}$	2^{172}	[HSE23]
	27	> 168	< 24	IB	4	$2^{67.1}$	$2^{168.23}$	2^{160}	Section 5.1
	28	> 190	< 2	IB	4	$2^{66.37}$	$2^{190.8}$	2^{184}	Section 5.2
	31	> 182	< 10	Rect.	4	$2^{62.78}$	$2^{182.07}$	$2^{62.79}$	[DQSW22]
SKINNY-128-384	27	> 378	< 6	ID	2	$2^{126.03}$	2^{378}	2^{368}	[LGS17]
	27	> 362	< 22	ID	2	$2^{124.99}$	$2^{362.61}$	2^{344}	[HSE23]
	27	> 337	< 47	IB	4	$2^{131.3}$	2^{337}	2^{320}	Section 5.1
	28	> 382	< 2	IB	4	$2^{130.26}$	$2^{382.8}$	2^{368}	Section 5.2
	32	> 344	< 40	Rect.	4	$2^{123.54}$	$2^{344.78}$	$2^{129.54}$	[SZY+22]

† Some attacks listed are *beyond full-codebook* attacks. For an introduction to beyond full-codebook attacks and the computation of the corresponding data complexity, please refer to the remark in Section 3.4.2. The preceding column "#K" refers to the number of related keys used in the attack.
‡ There is a lack of an impossible differential attack against Deoxys-BC-384 in the public literature. We try to provide the best related-tweakey impossible differential attack against Deoxys-BC-384 for comparison with the impossible boomerang attack.

- We revisit the impossible boomerang attack proposed in [Lu11] and introduce the applicable *Generalized Boomerang Framework* (GBF). For the impossible boomerang, we introduce the generation of contradictions and its advantages over the impossible differential, and then give two key recovery methods. In addition, we propose a MIQCP-based tool to search for complete impossible boomerang attacks.
- For Deoxys-BC, we provide a related-tweakey impossible boomerang attack against 10-round Deoxys-BC-256 and a related-tweakey impossible boomerang attack against 14-round Deoxys-BC-384. As a demonstration of the effectiveness of the impossible boomerang attack, our attack against 14-round Deoxys-BC-384 surpasses the previous best related-tweakey impossible differential attack by two rounds. The distinguishers used in our attacks can cover 1 or 2 more rounds than the previous related-tweakey impossible distinguishers.
- For Joltik-BC-128, we propose an improved 10-round related-tweakey impossible

Table 2: Summary of cryptanalytic distinguishers.

Cipher	#Rounds	Distinguisher	Reference
Joltik-BC-128	6	ID	[CLH+23]
	7	IB	Table 8
Joltik-BC-192	7	ID	[CLH+23]
	9	IB	Table 9
Deoxys-BC-256	6	ID	[ZDW19]
	7	IB	Table 6
Deoxys-BC-384	7	ID	Appendix C
	9	IB	Table 7
SKINNY-64-192	16	ID	[LGS17]
	17	ID	[HSE23]
	18	IB	Figure 15
SKINNY-128-384	16	ID	[LGS17]
	17	ID	[HSE23]
	18	IB	Figures 14 and 16 †

† The distinguisher shown in Figure 16 has been proved to be wrong in [BL24].

boomerang attack compared to the previous best attack. For `Joltik-BC-192`, we present the best related-tweakey attack against 14-round `Joltik-BC-192`. The distinguishers used in our attacks can cover 1 or 2 more rounds than the previous related-tweakey impossible distinguishers.

- For `SKINNY`-n-$3n$, we present a 27-round related-tweakey impossible boomerang attack, with improved time and memory complexity compared to the previous best related-tweakey impossible attack. We also provide the first 28-round related-tweakey impossible boomerang attack, which extends one more round than the previous best related-tweakey impossible attack.

Organization. In Section 2, we give a brief description of boomerang attacks, sandwich attacks, boomerang connectivity table (BCT) and MIQCP, followed by the notations used in this work. In Section 3, we revisit the impossible boomerang attack, then describe a new automatic search tool, and propose two key recovery methods for impossible boomerang attacks. We introduce the new cryptanalytic results on `Joltik-BC` and `Deoxys-BC` in Section 4. The new cryptanalytic results on `SKINNY`-n-$3n$ are provided in Section 5. Finally, Section 6 concludes this paper.

2 Preliminaries

2.1 Boomerang Attacks

The boomerang attack [Wag99] is an extension of the traditional differential cryptanalysis proposed by Wagner, which allows the adversary to use two short characteristics of high probability to construct a long one.

The boomerang attack regards the targeted cipher $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ as a cascade of two sub-ciphers $E = E_1 \circ E_0$, where there are two short differentials $\alpha \to \beta$

and $\gamma \to \delta$ with probability p and q for E_0 and E_1, respectively, as depicted in Figure 1. The probability of the boomerang distinguisher is

$$\Pr[E^{-1}(E(P) \oplus \delta) \oplus E^{-1}(E(P \oplus \alpha) \oplus \delta) = \alpha] = p^2 q^2.$$

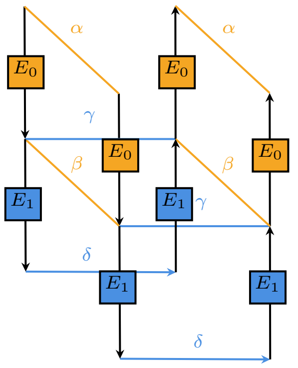


Figure 1: Boomerang attack

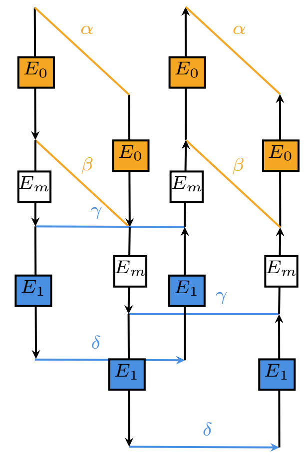


Figure 2: Sandwich attack

The boomerang attack is an adaptive chosen plaintext and ciphertext attack, with the following process:

- Encrypt pair of plaintexts (p_1, p_2) s.t. $p_1 \oplus p_2 = \alpha$ into (c_1, c_2) respectively.
- Get p_3, p_4 by decrypting $c_3 = c_1 \oplus \delta$ and $c_4 = c_2 \oplus \delta$ respectively.
- Check whether $p_3 \oplus p_4 = \alpha$.

2.2 Sandwich Attacks and Boomerang Connectivity Table

In [DKS10], Dunkelman *et al.* proposed the *sandwich attack* (see in Figure 2) to exploit the dependence between two differentials of the boomerang distinguisher, which divides a cipher E into three sub-ciphers: $E = E_1 \circ E_m \circ E_0$. The probability of the sandwich distinguisher is

$$\Pr[E^{-1}(E(P) \oplus \delta) \oplus E^{-1}(E(P \oplus \alpha) \oplus \delta) = \alpha] = \tilde{p}^2 \tilde{q}^2 r,$$

where $\tilde{p}$ (resp. $\tilde{q}$) is the probability of the differential of E_0 (E_1), and r is the probability of generating a right quartet for E_m.

In [CHP$^+$18], Cid *et al.* proposed a tool, named *Boomerang Connectivity Table* (`BCT`), to calculate r when E_m is composed of a single S-box layer. The `BCT` well captures the previous observations including incompatibility [Mur11], the S-box switch and the ladder switch [BK09], and gives more new insights into the boomerang switch. Later, the methods describing multi-round transitions of the boomerang have been proposed by many works [WP19, SQH19, DDV20, HBS21]. These works indicate that a thorough investigation on the construction of the middle layer can lead to a better boomerang attack. Here, we provide the definition of the `BCT`.

Definition 1 ([CHP$^+$18])**.** Let S be an n-bit bijective S-box, and $\Delta_i, \nabla_o \in \mathbb{F}_2^n$. The `BCT` of S is given by a $2^n \times 2^n$ table, in which the entry for (Δ_i, ∇_o) is given by:

$$\mathtt{BCT}(\Delta_i, \nabla_o) = \#\{x \in \mathbb{F}_2^n | S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o) = \Delta_i\}.$$

2.3 Mixed-Integer Quadratically-Constrained Programming

Mixed-Integer Programming (MIP) is a category of mathematical optimization problems that includes Mixed-Integer Linear Programming (MILP), Mixed-Integer Quadratic Programming (MIQP), Mixed-Integer Quadratically-Constrained Programming (MIQCP), etc. These problems were initially applied in the field of Operations Research. Recently, their applications in cryptanalysis have been extensively studied, especially the MILP problem. Our new model employs MIQCP, which involves quadratic constraints and quadratic objective functions. MIQCP has been previously used to model the differential-linear attack in [BGG+23] and [LJC23]. We use the Gurobi solver[1] to solve the MIQCP model in this paper. Here, we provide the mathematical definition of MIQCP.

Definition 2 ([BS12])**.** A Mixed-Integer Quadratically-Constrained Programming (MIQCP) is a problem, where the objective function and constraints can both include linear and quadratic terms, and some or all the decision variables are integer variables. The mathematical definition can be expressed as follows:

$$
\begin{array}{rl}
\texttt{min} \text{ or } \texttt{max} & x^T C x + c^T x \\
\texttt{s.t.} & \left\{ x \in \mathbb{R}^n : \begin{array}{ll} x^T A_k x + a_k^T x \leq b_k & \forall k = 1, \ldots, m \\ l \leq x \leq u & \\ x_i \in \mathbb{Z} & \forall i \in I, I \subseteq N := \{1, \ldots, n\} \end{array} \right\}
\end{array}
$$

where $(C, c) \in \mathcal{S}^n \times \mathbb{R}^n$, $(A_k, a_k, b_k) \in \mathcal{S}^n \times \mathbb{R}^n \times \mathbb{R}$ for all $k = 1, \ldots, m$, $(l, u) \in (\mathbb{R} \cup \{-\infty\})^n \times (\mathbb{R} \cup \{+\infty\})^n$ and $\mathcal{S}^n$ is the set of all $n \times n$ symmetric matrices.

It is worth noting that the distinction between MIQCP and MIQP lies in the types of constraints and objective function. MIQP can only use linear constraints and a quadratic objective function. MIQCP must include quadratic constraints, and the objective function does not matter.

2.4 Notations

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

STK_r	:	Subtweakey of round r
$eSTK_r$	:	Equivalent subtweakey of round r
X_r	:	Internal state before SubBytes (resp. SubCells) in round r for Deoxys-BC (resp. SKINNY)
Y_r	:	Internal state before ShiftRows (resp. AddRoundTweakey) in round r for Deoxys-BC (resp. SKINNY)
Z_r	:	Internal state before MixColumns (resp. ShiftRows) in round r for Deoxys-BC (resp. SKINNY)
W_r	:	Internal state after MixColumns (resp. ShiftRows) in round r for Deoxys-BC (resp. SKINNY)
ΔX	:	Difference of a state X in the upper trail
∇X	:	Difference of a state X in the lower trail
$X_r[i]$	:	i-th cell of a state X in round r
$X_r[i, ..., k]$	:	i-th cell,..., k-th cell of a state X in round r

[1] Gurobi: www.gurobi.com

3 Revisiting the Impossible Boomerang Attack

3.1 Definition of the Impossible Boomerang Distinguisher

The impossible boomerang attack was first introduced by Lu in [Lu08, Lu11], and the definition is given as follows.

Definition 3. Suppose $E : \{0,1\}^n \times \{0,1\}^k \to \{0,1\}^n$ is a block cipher and $K \in \{0,1\}^k$ is a key for E. If there exist a quartet $(\alpha, \alpha', \delta, \delta') \in \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n \times \mathbb{F}_2^n$ satisfying

$$\forall X \in \mathbb{F}_2^n,\ \Pr[E_K^{-1}(E_K(X) \oplus \delta) \oplus E_K^{-1}(E_K(X \oplus \alpha) \oplus \delta') = \alpha'] = 0,$$

then $(\alpha, \alpha', \delta, \delta')$ is called an impossible boomerang distinguisher, written $(\alpha, \alpha') \nrightarrow (\delta, \delta')$.

The paper [Lu11] also describes how to construct an impossible boomerang distinguisher. Specifically, the distinguisher consists of four characteristics of probability 1:
- $\alpha \to \beta$ with probability 1 and $\alpha' \to \beta'$ with probability 1 for E_0;
- $\delta \to \gamma$ with probability 1 and $\delta' \to \gamma'$ with probability 1 for E_1^{-1},

where $\beta, \beta', \gamma, \gamma'$ satisfy the condition $\beta \oplus \beta' \oplus \gamma \oplus \gamma' \neq 0$. The distinguisher is depicted in Figure 3.

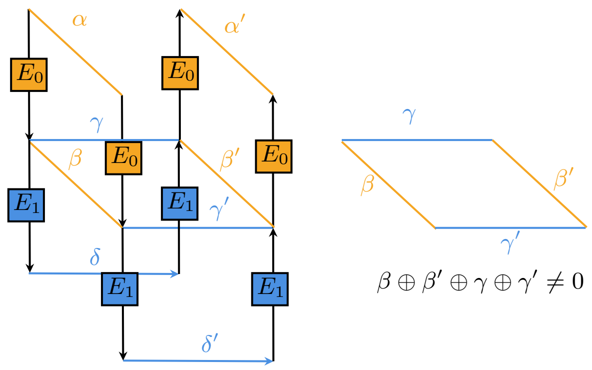


Figure 3: Impossible Boomerang Distinguisher in [Lu11]

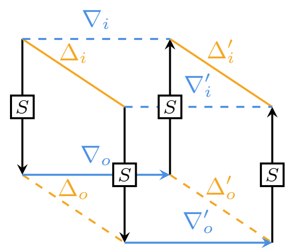


Figure 4: Difference transition in the single S-box layer

3.2 Generalized Boomerang Framework

As we can see from Definition 3, the impossible boomerang distinguisher allows two symmetric characteristics used in E_0 to be different, as well as the ones in E_1. The same idea has been briefly mentioned in [HBS21]. In this paper, to capture this unusual construction of boomerang distinguishers, we propose the *Generalized Boomerang Framework* (GBF). Similar to the traditional boomerang attack, the GBF divides the targeted cipher as two sub-ciphers $E = E_1 \circ E_0$, but there are two differentials $\alpha \xrightarrow{p_1} \beta, \alpha' \xrightarrow{p_2} \beta'$ for E_0 and two differentials $\gamma \xrightarrow{q_1} \delta, \gamma' \xrightarrow{q_2} \delta'$ for E_1. The traditional boomerang is the GBF under the conditions where $\alpha = \alpha'$, $\beta = \beta'$, $\gamma = \gamma'$, and $\delta = \delta'$. In GBF, structures excluding traditional boomerangs are referred to as *asymmetric boomerangs*. The probability of the generalized boomerang distinguisher is $p_1 p_2 q_1 q_2$. Similarly, we can also generalize the sandwich distinguisher.

The `BCT` can also be extended to the GBF, which was first proposed in [LWL22] and named the *Generalized Boomerang Connectivity Table* (`GBCT`).

Definition 4 ([LWL22])**.** Let S be an n-bit bijective S-box, and $\Delta_i, \Delta_i', \nabla_o, \nabla_o' \in \mathbb{F}_2^n$. The GBCT of S is given by a four-dimensional table, in which the entry for $(\Delta_i, \Delta_i', \nabla_o, \nabla_o')$ is given by:

$$\texttt{GBCT}(\Delta_i, \Delta_i', \nabla_o, \nabla_o') = \#\{x \in \mathbb{F}_2^n | S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_o') = \Delta_i'\}.$$

Other generalized tables for multiple rounds in GBF are listed in Appendix A.

3.3 Construct Impossible Boomerang Attacks

The core step for the impossible boomerang attack is to find a boomerang distinguisher that never returns back. One direction is to utilize four distinct characteristics such that their XORed difference is nonzero, as described in Section 3.1. However, a limitation of this direction is that it ignores the dependence of the two sub-ciphers, which has been extensively studied in recent years since the introduction of the BCT. Even earlier, Murphy [Mur11] pointed out that the incompatibility of two characteristics could lead to a boomerang distinguisher of probability 0. Thus, an intuitive idea for constructing an impossible boomerang distinguisher is to explore the incompatibility between the middle layer E_m using advanced tools like BCT, GBCT, etc. The principle of generating an impossible boomerang distinguisher is as follows.

Proposition 1. *A boomerang distinguisher is impossible as long as the probability of generating a right quartet for E_m is zero.*

It is easy to see that this proposition covers the impossible case described in Section 3.1. More importantly, it reveals the construction of an impossible boomerang distinguisher through the incompatibility of multiple rounds, which could possibly lead to a longer distinguisher.

Similar to impossible differentials, we adopt the miss-in-the-middle approach to search impossible boomerangs: For a cipher $E = E_1 \circ E_m \circ E_0$, we can find two forward characteristics (same or different) and two backward characteristics (same or different) both with probability one in order to make the difference transition through the middle layer E_m with probability zero. There are many studies on the switching probability of the middle layer, and therefore a natural idea for constructing impossible boomerang distinguishers is to consider the switching probability of certain cells in E_m using the techniques such as BCT, GBCT, etc., which can be easily modeled by MIQCP (discussed in Section 3.3.3).

In this work, we searched for IB attacks on both traditional boomerangs and asymmetric ones, and found that the attacks based on traditional boomerangs are more effective, so the rest of the paper will focus on traditional boomerangs.

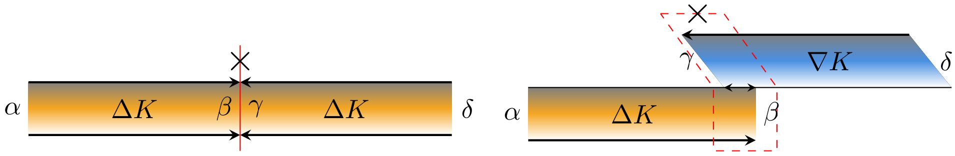
3.3.1 Impossible Boomerangs vs Impossible Differentials

As another cryptanalytic technique using distinguishers with probability zero, impossible differentials have been extensively studied. Then, an interesting discussion would arise from comparing impossible boomerangs with impossible differentials.

For single-key setting, it has been proven in [SLG+16] that the upper bounds for the length of impossible differentials depend on the linear layer. The same approach can be applied to the impossible boomerangs. Related-key attacks [Bih94] allow the attacker uses weaknesses of the encryption function and of the key schedule algorithm to derive information on the unknown keys. In related-key impossible differentials, two related keys are involved (K_a and $K_b = K_a \oplus \Delta K$), while related-key impossible boomerangs could involve four related keys (K_a, $K_b = K_a \oplus \Delta K$, $K_c = K_a \oplus \nabla K$, $K_d = K_a \oplus \nabla K \oplus \Delta K$). Compared to the single-key impossible boomerang in Definition 3, the related-key impossible boomerang with the traditional boomerang structure can be defined as

$$\forall X \in \mathbb{F}_2^n,\ \Pr[E_{K_c}^{-1}(E_{K_a}(X) \oplus \delta) \oplus E_{K_d}^{-1}(E_{K_b}(X \oplus \alpha) \oplus \delta) = \alpha] = 0.$$

This fact allows the adversary to have more freedom to choose key differences for the upper and lower trails independently in the case of the impossible boomerang attack, which is illustrated in Figure 5.



(a) Related-key impossible differential **(b)** Related-key impossible boomerang

Figure 5: Impossible Differential vs Impossible Boomerang (related-key setting)

Therefore, related-key impossible boomerangs are expected to cover more rounds compared to related-key impossible differentials. For example, we refer to our attack on `Deoxys-BC-384`. Specifically, we provide a 9-round related-tweakey impossible boomerang distinguisher and a 7-round related-tweakey impossible differential distinguisher in Table 7 and Figure 13, respectively. The reason that the impossible boomerang is better lies in the fact that the subtweakey difference cancellation (Proposition 2) is only applied once in the impossible differential, whereas in the case of the impossible boomerang, this cancellation is utilized twice (in Round 3-4 and Round 9-10).

3.3.2 Contradictions through Multiple Rounds

In addition to applying the `BCT` to find incompatibility of a single S-box layer, we can also construct an impossible boomerang distinguisher through incompatibility of multiple rounds. *Double Boomerang Connectivity Table* (`DBCT`) [HBS21] is a technique to evaluate the boomerang switch through multiple rounds. Similar to the constraints on cells in a single S-box layer, we can apply quadratic constraints on the propagation of active cells through multiple rounds. The distinguisher in Figure 16 is an example of using the `DBCT` technique.

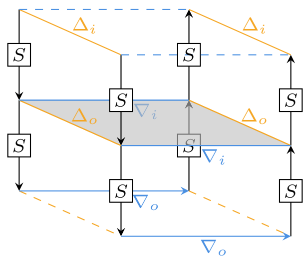


Figure 6: `DBCT` of S-box

Definition 5 ([WP19, DDV20])**.** Let S be an n-bit S-box, and $\Delta_i, \Delta_o, \nabla_i, \nabla_o \in \mathbb{F}_2^n$. The Upper `BCT` (`UBCT`) and the Lower `BCT` (`LBCT`) of S are three-dimensional tables defined as

$$\mathtt{UBCT}(\Delta_i, \Delta_o, \nabla_i) = \# \left\{ x \in \mathbb{F}_2^n \,\middle|\, \begin{array}{l} S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \\ S^{-1}(S(x) \oplus \nabla_i) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla_i) = \Delta_i \end{array} \right\},$$

$$\mathtt{LBCT}(\Delta_o, \nabla_i, \nabla_o) = \# \left\{ x \in \mathbb{F}_2^n \,\middle|\, \begin{array}{l} S(x) \oplus S(x \oplus \nabla_i) = \nabla_o \\ S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_o) \oplus \nabla_o) = \Delta_o \end{array} \right\}.$$

Definition 6 ([HBS21, YSS+22]). Let S be an n-bit S-box, and $\Delta_i, \Delta_o, \nabla_i, \nabla_o \in \mathbb{F}_2^n$. The DBCT of S is a $2^n \times 2^n$ table, in which the entry for (Δ_i, ∇_o) is given by:

$$\texttt{DBCT}(\Delta_i, \nabla_o) = \sum_{\Delta_o, \nabla_i} \texttt{UBCT}(\Delta_i, \Delta_o, \nabla_i) \cdot \texttt{LBCT}(\Delta_o, \nabla_i, \nabla_o).$$

The Attempt to Build Impossibilities Using The DBCT. In the ToSC version of our paper, we attempted to give an example of constructing an impossible boomerang distinguisher of SKINNY using the contradiction through multiple rounds as shown in Figure 7, but it was later proved to be wrong in [BL24]. The complete 18-round distinguisher is shown in Figure 16. The contradiction was assumed to occur in consecutive three rounds: Round 13 to 15. For $\Delta X_{13}[2]$, its difference 0x05 is equal to $\Delta Z_{12}[2]$, which comes from $\Delta STK_{12}[2] =$ 0x05 of ART in Round 12. For $\nabla Z_{14}[6]$, its difference α is equal to $\nabla X_{15}[7]$ due to the linear transformations SR and MC. And $\nabla X_{15}[7]$ is derived from the operation SC on the known difference $\nabla Y_{15}[7] =$ 0x04 in Round 15. Therefore, it seems that we can use

$$\texttt{DBCT}(\texttt{0x05}, \alpha) = 0 \text{ for } \forall \alpha \text{ s.t. } \texttt{DDT}(\alpha, \texttt{0x04}) \neq 0$$

to construct the contradiction. However, we show below that this is not an impossible distinguisher.

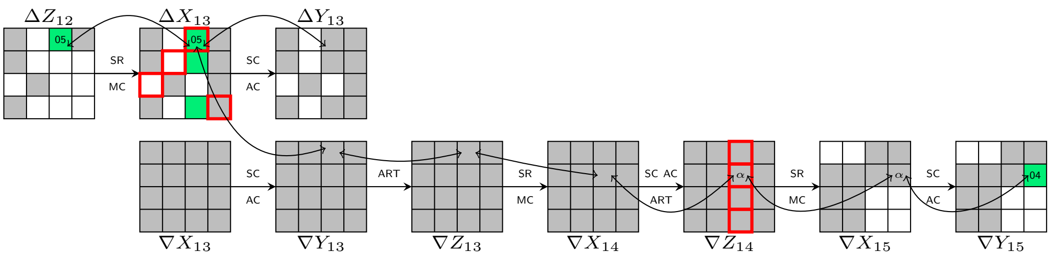


Figure 7: Three-round E_m of the 18-round related-tweakey impossible boomerang distinguisher for SKINNY-128-384 in Figure 16

Limits of The DBCT. In [BCL+24], Bonnetain *et al.* introduced the limitations of using DBCT in identifying multi-round contradictions in IB. The DBCT only considers the case of equal differences in the middle between the two S-boxes, ignoring those $(\Delta_o, \nabla_i, \nabla_i', \Delta_o')$ with $\Delta_o \neq \Delta_o'$ and $\nabla_i \neq \nabla_i'$.

In [BL24], Bonnetain and Lallemand provided a counterexample of Figure 7, thereby proving the flaw in the distinguisher shown in Figure 16. The counterexample is shown in Table 3. For a detailed description of the usage of DBCT, please refer to [BCL+24, BL24].

3.3.3 MIQCP-based Search Tool for Impossible Boomerang Attacks

In this section, we describe a MIQCP-based search tool for impossible boomerang attacks. Differing from the previous search for distinguishers, the quadratic constraints make it easier for us to search for complete attacks. In the following, we will use the round function of Deoxys-BC to describe the constraints of the MIQCP model. We use this model to search for the truncated attack and instantiate it after running the model.

Variables. We assign three attributes to each byte of the internal states and the subtweakeys: z, k, and d, indicating whether it has a zero difference, whether it has a known difference value, and whether it belongs to the distinguisher, respectively. For attributes z and k, if $z = 1$, it means that the byte is zero difference (corresponding to the white

Table 3: An counterexample of an actual quartet for Figure 7 provided by [BL24] (Treat the 2nd, 5th, 8th and 15th cells in X_{13} as a 32-bit word (red boxes in Figure 7).)

	x^1	x^2	x^3	x^4
Input	`0xa7ed0098`	`0xb8ed009d`	`0xa0690090`	`0xbf690095`
Output	`0x8e521bd8`	`0xab523ac8`	`0xeb7d4b8c`	`0xce7d6a9c`
Middle key+constant	`0x00028862`		`0x00028962`	
	(x^1, x^2)		(x^3, x^4)	
ΔX_{13}	`0x1f000005`		`0x1f000005`	
After Sbox layer	`0x7d000026`		`0x5d000006`	
	(x^1, x^3)		(x^2, x^4)	
∇Y_{14}	$\texttt{0x652f}\underbrace{\texttt{50}}_{\alpha}\texttt{54}$		$\texttt{0x652f}\underbrace{\texttt{50}}_{\alpha}\texttt{54}$	

squares in the following figures); if $z = 0, k = 1$, it means that the byte is nonzero known difference (corresponding to the green and pink squares in the following figures); if $z = k = 0$, it means that the byte is unknown difference (corresponding to the gray squares in the following figures). For attribute d, if $d = 1$, it implies that the byte belongs to the distinguisher; if $d = 0$, it implies that the byte belongs to the key recovery phase.

Based on the notations introduced in Section 2.4, we use $\Delta X_r[i]^z$ to represent the attribute z of the difference $\Delta X_r[i]$, and so on. There are some intuitive constraints: for any byte a, we have $a^z \leq a^k$. Let $a \xrightarrow{S} b$, where S is an S-box and a, b are the input and output differences of S, we have $a^z = b^z$. Additionally, the number of rounds R for a complete attack and the number of rounds r_m for the middle layer E_m need to be fixed before running the model. Thus, for the plaintext and ciphertext in the complete attack, we have $\Delta plaintext[i]^d = \nabla ciphertext[i]^d = 0$ and $\nabla X_{r_m}[i]^d = \Delta X_{r_m}[i]^d = 1$.

New Constraints with Quadratic Terms. For attribute d, it indicates whether the byte is in the distinguisher or in the key recovery phase. We use the notation T to represent a transformation in the round function and let $a \xrightarrow{T} b$, where a and b are the input and output differences of the transformation T, respectively. For the upper trail of E_0, there are three possibilities:

$$a^d = 1 \rightarrow \boxed{T} \rightarrow b^d = 1 \qquad a^d = 0 \rightarrow \boxed{T} \rightarrow b^d = 1 \qquad a^d = 0 \rightarrow \boxed{T} \rightarrow b^d = 0$$

and for the lower trail of E_1, there are also three possibilities:

$$a^d = 0 \rightarrow \boxed{T} \rightarrow b^d = 0 \qquad a^d = 1 \rightarrow \boxed{T} \rightarrow b^d = 1 \qquad a^d = 1 \rightarrow \boxed{T} \rightarrow b^d = 0$$

Suppose that L_1 and L_2 respectively represent linear inequalities for the transformation T within the distinguisher and the key recovery phase. Then we can use $a^d \cdot L_1 + (1 - a^d) \cdot L_2$ as the quadratic terms to describe the transformation T. The description of the SubBytes operation in the following text will serve as an example explaining the quadratic term. These quadratic constraints make our model a MIQCP model rather than a MIQP one. Since all variables in this model are binary, it is undeniable that the constraints in this model can be rewritten as linear ones. In [DEFN22], the authors used linear constraints with the same variable d to search for complete boomerang attacks. We fine-tuned their model to adapt to IB attacks and our experiments show that for IB attack MIQCP is more efficient than linear constraints, the rewritten linear inequalities would be more complex and redundant.

SubBytes With the miss-in-the-middle technique, we aim to search for two characteristics with probability 1. When the input difference of S-box takes a nonzero known value, the output difference becomes unknown. For the upper trail of E_0, the constraints would be described as:

$$\begin{aligned} \Delta X_r[i]^d - \Delta Y_r[i]^d &\leq 0 \quad (1)\\ \Delta X_r[i]^d \cdot (\Delta X_r[i]^k - \Delta Y_r[i]^k + 1) + (1 - \Delta X_r[i]^d) \cdot (\Delta Y_r[i]^k - \Delta X_r[i]^k + 1) &\geq 1 \quad (2)\\ \Delta X_r[i]^d \cdot (\Delta Y_r[i]^k - \Delta Y_r[i]^z + 1) + (1 - \Delta X_r[i]^d) \cdot (\Delta X_r[i]^k - \Delta X_r[i]^z + 1) &= 1 \quad (3) \end{aligned}$$

The linear inequality (1) can be intuitively derived from the three possibilities mentioned above for E_0. For inequalities (2) and (3), only one additive term will work, corresponding to the propagation in the distinguisher or in the key recovery phase. For instance, when $\Delta X_r[i]^d = 1$, both (2) and (3) will only activate the first term, constraining the difference propagation in the distinguisher: zero input difference ($z = 1, k = 1$) to zero output difference ($z = 1, k = 1$), nonzero known difference ($z = 0, k = 1$) to unknown output difference ($z = 0, k = 0$), and unknown input difference ($z = 0, k = 0$) to unknown output difference ($z = 0, k = 0$).

The constraints for the lower trail of E_1 are symmetric:

$$\begin{aligned} \nabla X_r[i]^d - \nabla Y_r[i]^d &\geq 0\\ \nabla Y_r[i]^d \cdot (\nabla Y_r[i]^k - \nabla X_r[i]^k + 1) + (1 - \nabla Y_r[i]^d) \cdot (\nabla X_r[i]^k - \nabla Y_r[i]^k + 1) &\geq 1\\ \nabla Y_r[i]^d \cdot (\nabla X_r[i]^k - \nabla X_r[i]^z + 1) + (1 - \nabla Y_r[i]^d) \cdot (\nabla Y_r[i]^k - \nabla Y_r[i]^z + 1) &= 1 \end{aligned}$$

ShiftRows The ShiftRows operation preserves the values of all attributes of the variable.

MixColumns For the constraints for MixColumns, we use some linear constraints on the MDS matrix proposed in [DEFN22]. Let $(b_1, b_2, b_3, b_4) = \mathsf{MC}(a_1, a_2, a_3, a_4)$, then we have

$$a_1^u + a_2^u + a_3^u + a_4^u + b_1^u + b_2^u + b_3^u + b_4^u \in \{0, 1, 2, 3, 8\}.$$

This could be translated into linear constraints by adding extra dummy variables $e_{u,i}$ and $e'_{u,i}$ corresponding to each u and i:

$$\begin{aligned} \sum_{j=0}^{3} \Delta Z_r[i+j]^u + \sum_{j=0}^{3} \Delta W_r[i+j]^u &\leq 8 - 5e_{u,i}, \text{ for } u \in \{z,k,d\}, i \in \{0,4,8,12\} \quad (4)\\ \sum_{j=0}^{3} \Delta Z_r[i+j]^u + \sum_{j=0}^{3} \Delta W_r[i+j]^u &\geq 8 - 8e_{u,i}, \text{ for } u \in \{z,k,d\}, i \in \{0,4,8,12\} \quad (5)\\ \sum_{j=0}^{3} \nabla Z_r[i+j]^u + \sum_{j=0}^{3} \nabla W_r[i+j]^u &\leq 8 - 5e'_{u,i}, \text{ for } u \in \{z,k,d\}, i \in \{0,4,8,12\} \quad (6)\\ \sum_{j=0}^{3} \nabla Z_r[i+j]^u + \sum_{j=0}^{3} \nabla W_r[i+j]^u &\geq 8 - 8e'_{u,i}, \text{ for } u \in \{z,k,d\}, i \in \{0,4,8,12\} \quad (7) \end{aligned}$$

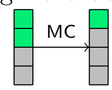
$$\begin{aligned} 4\Delta Z_r[i]^d &\leq \sum_{j=0}^{3} \Delta W_r[4\lfloor i/4 \rfloor + j]^d \quad (8)\\ 4\nabla W_r[i]^d &\leq \sum_{j=0}^{3} \nabla Z_r[4\lfloor i/4 \rfloor + j]^d \quad (9) \end{aligned}$$

Besides, we introduce the following quadratic constraints to more accurately describe the difference propagation of MixColumns operation by adding two dummy variables e_u

and e_l:

$$\Delta Z_r[i]^d \cdot \sum_{j=0}^{3} \Delta W_r[4\lfloor i/4 \rfloor + j]^k + (1 - \Delta Z_r[i]^d) \cdot \sum_{j=0}^{3} \Delta Z_r[4\lfloor i/4 \rfloor + j]^k \quad = \quad 4 - 4e_u \quad (10)$$

$$\nabla W_r[i]^d \cdot \sum_{j=0}^{3} \nabla Z_r[4\lfloor i/4 \rfloor + j]^k + (1 - \nabla W_r[i]^d) \cdot \sum_{j=0}^{3} \nabla W_r[4\lfloor i/4 \rfloor + j]^k \quad = \quad 4 - 4e_l \quad (11)$$

Specifically, inequalities (4)-(9) provide a rough characterization of the MDS-type MixColumns and can not eliminate cases such as 

(the green color stands for nonzero known difference and the gray color stands for unknown difference). The problem can be solved with inequalities (10) and (11).

AddRoundTweakey For the primary XOR operation within the AddRoundTweakey, expressed as $a \oplus b = c$, we treat it as an ordered operation to exclude the possible case $a[z = 0, k = 0] \oplus b[z = 0, k = 0] = c[z = 0, k = 1]$ that is feasible under the previous constraint: $a^u + b^u + c^u \neq 2, u \in \{z, k\}$.

Table 4: Possible values for $a \oplus b = c$

(a^z, a^k)	(b^z, b^k)	(c^z, c^k)
(1, 1)	(1, 1)	(1, 1)
(1, 1)	(0, 1)	(0, 1)
(0, 1)	(1, 1)	(0, 1)
(0, 1)	(0, 1)	(1, 1)
(0, 1)	(0, 1)	(0, 1)
(1, 1)	(0, 0)	(0, 0)
(0, 0)	(1, 1)	(0, 0)
(0, 1)	(0, 0)	(0, 0)
(0, 0)	(0, 1)	(0, 0)
(0, 0)	(0, 0)	(0, 0)

Considering the attributes z and k of bytes a, b and c, there are a total of 10 possibilities (shown in Table 4). We have the following inequality constraints for these cases:

$$\begin{cases} a^k - a^z + b^k - b^z + c^z - c^k >= 0 \\ a^z - b^z + b^k - c^z >= 0 \\ a^k - a^z + b^z - c^z >= 0 \\ c^k - a^k - b^k >= -1 \\ a^k - c^k >= 0 \\ b^k - c^k >= 0 \end{cases}$$

Construct Contradiction The distinguisher retrieved by the model is based on the contradiction we constructed rather than automatically captured by the model in previous works. For the r_m-th round E_m which is composed of a single S-box layer, there exists at least one byte whose input and output differences are both nonzero known values:

$$\sum_{i=0}^{15} (\Delta X_{r_m}[i]^k - \Delta X_{r_m}[i]^z) \cdot (\nabla Y_{r_m}[i]^k - \nabla Y_{r_m}[i]^z) \geq 1. \quad (12)$$

After running the MIQCP model, we use `BCT` and tweakey schedule to obtain specific instantiation that satisfy the truncated characteristics. Inequality (12) can provide a very simple example comparing linear constraints with quadratic constraints. If we rewrite it as a linear constraint $\Delta X_{r_m}[i]^k - \Delta X_{r_m}[i]^z + \nabla Y_{r_m}[i]^k - \nabla Y_{r_m}[i]^z = 2$, and then need to run the model 16 times for i ranging from 0 to 15.

As for the contradictions through multiple rounds, it is necessary to analyze the round function of the cipher to construct contradictions. It is possible that the model needs to be run multiple times to obtain the final result. Thanks to the efficiency of our model, the overall time takes a few minutes even when multiple iterations are required (single execution requiring only several seconds on Intel(R) Xeon(R) Gold 5220R CPU @ 2.20GHz). Taking Figure 7 as an example, we use the following constraints to construct the contradiction:

$$\begin{aligned}(\Delta X_{r_m}[2]^k - \Delta X_{r_m}[2]^k) \cdot (3 - \nabla Y_{r_m+2}[7]^z - \nabla Y_{r_m+2}[11]^z - \nabla Y_{r_m+2}[15]^z) &= 1 \\ \nabla Y_{r_m+2}[7]^k + \nabla Y_{r_m+2}[11]^k + \nabla Y_{r_m+2}[15]^k &= 3\end{aligned}$$

Objective Function. Generally, both the beginning and the end of an impossible boomerang have very few active bytes. We aim to activate as few bytes as possible during the key recovery phase, thus reducing the number of bytes with unknown difference ($k = 0$) of plaintext and ciphertext, and consequently lowering the complexity. Thus, we have the following objective function and ask the solver for the maximum value:

$$\sum_{i=0}^{15} (\Delta plaintext[i]^k + \nabla ciphertext[i]^k),$$

in which the plaintext and ciphertext refer to the actual ones in a complete attack.

Advantages and Limitations. Previously, most automatic tools focused on searching for impossible differential distinguishers. Those tools were designed by specifying a set of input differences and a set of output dfferences in the model and asking the solver to iterate through all input and output differences in the given sets. If the solver outputs an error code with "*infeasible*", it implies the detection of an impossible differential. The advantage of this method is that it can detect impossible differentials with arbitrary types of contradictions. However, the search space is large, and it is time-consuming.

In our new model, we no longer need to specify the sets of input and output differences. Instead, we use quadratic constraints to construct contradictions in E_m and describe the propagation of differences during the distinguisher and key recovery phase. This significantly reduces the search space compared to previous models, allowing results to be obtained in seconds. The limitation of the new model is that it cannot capture all types of contradictions like the previous model. We need to sequentially describe contradictions to run the model for several times. The model cannot promise the optimal results for IB attacks because there has not been a thorough study on the types of contradictions yet.

3.4 Key Recovery Attacks under Related-Key Setting

The impossible boomerang combines the concepts of impossible differentials and boomerangs. Therefore, we propose two approaches for its key recovery attack: the impossible differential style and the boomerang style. In the following, we will introduce the two methods, both under the related-key setting with the targeted cipher having linear key schedule. The introduction follows the procedures and notations from the previous works [BNS14, BLNS18, ZDJ19].

As illustrated in Figure 8, assume that there is an impossible boomerang distinguisher $(\alpha, \alpha) \nrightarrow (\delta, \delta)$. We denote the targeted cipher as $E = E_f \circ E_{dist} \circ E_b$, where E_{dist} denotes the rounds covered by the impossible boomerang distinguisher, and E_b and E_f denote the rounds added at the beginning and at the end of the distinguisher, respectively. We denote by K the size of master key, by s the size of S-box, by r_b (resp. r_f) the dimension

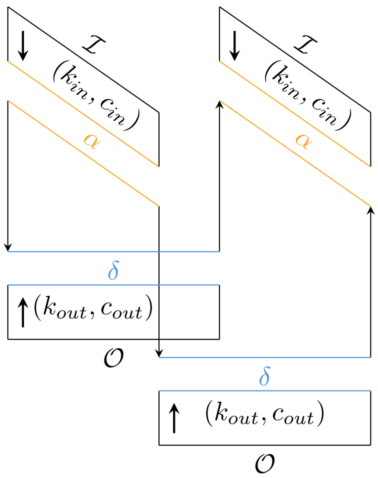


Figure 8: Outline of impossible boomerang key recovery attack

of vector space $\mathcal{I}$ (resp. $\mathcal{O}$). Let k_{in} (resp. k_{out}) denotes the number of subkey bits involved in E_b (resp. E_f), c_{in} (resp. c_{out}) denotes the number of bit-conditions that have to be verified during E_b (resp. E_f).

The goal of the key recovery attack is to discard the keys that allow the differentials $\mathcal{I} \to \alpha$ and $\mathcal{O} \to \delta$ at the same time at the both sides of the boomerang.

3.4.1 Impossible Differential Style

Construct a structure of 2^{r_b} plaintexts, and each can combine 2^{2r_b} plaintext pairs. In total, 2^{2N+4r_b} plaintext quartets can be constructed if 2^N structures are prepared. After filtration on the ciphertext side, $Q = 2^{2N+4r_b-2(n-r_f)}$ quartets will remain. For a given key, the probability that a quartet satisfying the differences $\mathcal{I}$ and $\mathcal{O}$ verifies all the bit-conditions in E_b and E_f is $2^{-2(c_{in}+c_{out})}$. Thus we have

$$2^{\alpha} \geq 2^{|k_{in} \cup k_{out}|}(1 - 2^{-2(c_{in}+c_{out})})^{Q},$$

where α denotes the number of subkey bits need to be exhaustively searched of the $\mathcal{K} = |k_{in} \cup k_{out}|$ subkey bits after incorrect keys are rejected.

Data Complexity. The formula above could be rewritten as:

$$Q \geq 2^{2(c_{in}+c_{out})} \cdot \frac{\mathcal{K} - \alpha}{\log_2 e}.$$

Thus, the data complexity of the attack is $D = 2^{N+r_b+2}$.

Time Complexity. As for the key recovery phase, we could adopt the early abort technique [LKKD08], which is popular in impossible differential cryptanalysis. Similar to impossible differential attack, the time complexity of impossible boomerang attacks consists of three terms. The first term is the time of preparing $Q = 2^{2N+4r_b-2(n-r_f)}$ quartets, denoted by C_Q. The second is the time of guessing all candidate keys $k_{in} \cup k_{out}$ and the cost can be approximated by $C_G = \left(Q + 2^{\mathcal{K}} \frac{Q}{2^{2(c_{in}+c_{out})}}\right) C'_E$, where C'_E is the ratio of the cost for one partial encryption to the full encryption. Finally, the last term is the cost for brute force, including the remaining key candidates after the sieving and the subkey bits not involved in key recovery procedure, given by $C_B = 2^{K-\mathcal{K}+\alpha}$. Considering the cost of one encryption as C_E, we have a total time complexity

$$\begin{aligned} T &= (C_Q + C_G + C_B)C_E \\ &= \left(C_Q + \left(Q + 2^{\mathcal{K}}\frac{Q}{2^{2(c_{in}+c_{out})}}\right)C'_E + 2^{K-\mathcal{K}+\alpha}\right)C_E. \end{aligned}$$

Memory Complexity. The memory complexity M of this attack is bounded by $Q + 2^{\mathcal{K}}$.

3.4.2 Boomerang Style

1. Construct 2^N structures of 2^{r_b} plaintexts each, each of them taking all the possible values of r_b active bits.
2. For each structure, query the ciphertexts corresponding to 2^{r_b} plaintexts under four related keys: K_1,$K_2 = K_1 \oplus \Delta K$,$K_3 = K_1 \oplus \nabla K$,$K_4 = K_1 \oplus \nabla K \oplus \Delta K$, respectively. We denote by S_i the plaintext-ciphertext sets encrypted by K_i, where $i \in \{1, 2, 3, 4\}$, and insert S_2 and S_4 into hash tables H_1 and H_2 indexed by the r_b bits of plaintexts.
3. Guess $|k_{in}|$ subkey bits involved in E_b:
   (a) For each structure, partially encrypt $P_1 \in S_1$ to the beginning of the distinguisher, XOR the obtained state with α, then decrypt it to produce the plaintext and search for a collision in H_1 to find P_2. It is expected for one collision for each P_1. Conduct the same operation to the set S_3 to find expected pairs (P_3, P_4). Two new sets can be obtained:
   $$L_1 = \{(P_1, C_1, P_2, C_2) : (P_1, C_1) \in S_1, (P_2, C_2) \in S_2, E_{b_{K_1}}(P_1) \oplus E_{b_{K_2}}(P_2) = \alpha\},$$
   $$L_2 = \{(P_3, C_3, P_4, C_4) : (P_3, C_3) \in S_3, (P_4, C_4) \in S_4, E_{b_{K_3}}(P_3) \oplus E_{b_{K_4}}(P_4) = \alpha\}.$$
   (b) The sizes of L_1 and L_2 are both $2^N \cdot 2^{r_b}$. Insert L_1 into a hash table H_3 indexed by $n - r_f$ bits of C_1 and $n - r_f$ bits of C_2. For each element (P_3, C_3, P_4, C_4) of L_2, we lookup the hash table H_3 to find the corresponding (P_1, C_1, P_2, C_2) satisfying $C_1 \oplus C_3 \in \mathcal{O}$ and $C_2 \oplus C_4 \in \mathcal{O}$. Finally, there are $Q = 2^{2N+2r_b-2(n-r_f)}$ quartets can be constructed.
   (c) Guess the $|k_{out}|$ subkey bits involved in E_f and eliminate the candidate keys which satisfy the differential $\mathcal{O} \to \delta$. As this is a type of impossible attacks, we employ the widely-used technique of early abort to eliminate incorrect keys, as commonly done in impossible differential attacks.
4. Exhaustively search the remaining key candidates and the unknown $K - |k_{in} \cup k_{out}|$ subkey bits.

Complexity. The average number of quartets Q required to be left with at most 2^α key candidates is given by the formula:

$$2^\alpha \geq 2^{|k_{in} \cup k_{out}|}(1 - 2^{-2c_{out}})^Q.$$

The data complexity is $D = 2^{N+r_b+2}$ chosen plaintexts and do $2^{|k_{in}|}(2 \cdot 2^{N+r_b} + 2^{N+r_b}) = 3 \cdot 2^{|k_{in}|+N+r_b}$ table lookups to prepare quartets. The total time complexity, including data collection, key guessing and brute force, is

$$T = \left(2^{N+r_b+2} + \left(2^{|k_{in}|}(2 \cdot 2^{N+r_b}) + Q + 2^{\mathcal{K}}\frac{Q}{2^{2c_{out}}}\right)C'_E + 2^{K-\mathcal{K}+\alpha}\right)C_E,$$

where C'_E is the ratio of the cost of partial encryption to the full encryption, $\mathcal{K} = |k_{in} \cup k_{out}|$ denotes the targeted key space, and C_E denotes the cost of one encryption. The memory complexity M is bounded by $4 \cdot 2^N \cdot 2^{r_b} + 2^N \cdot 2^{r_b} + Q + 2^{\mathcal{K}} = 5 \cdot 2^{N+r_b} + Q + 2^{\mathcal{K}}$.

Beyond Full-Codebook. For the block cipher with block size n, an attack against it that requires $D > 2^n$ plaintexts/ciphertexts is known as a *beyond full-codebook* attack. Recall that a tweakable block cipher takes as input an n-bit plaintext and a t-bit tweak, it is reasonable to assume that an attacker may have available an amount of data $D \gg 2^n$ to carry out an attack, as long as $D \leq 2^{n+t}$. Ciphers adopting the TWEAKEY framework [JNP14], such as Deoxys-BC and SKINNY, offer further flexibility in setting the limit of data resources available for an attack. The construction allows one to add a tweak of (almost) any length to a key-alternating block cipher and/or to extend the key space of the block cipher to (almost) any size. This provides cryptanalysts with a potentially optimal strategy to attack the ciphers: select the key size k as large as possible, which results on a higher security claim, as long as the size of the tweak t is large enough to supply the required data to run the attack. In fact, the beyond full-codebook attacks are considered to be realistic and effective against real-world tweakable block ciphers, and have been applied in previous works [BHT16, ABC+17, CHP+17, ZDW19].

Suppose the tweak size is t, the tweakey size is h, the number of related keys used in the attack is rk, we have two natural constraints for the related-tweakey impossible boomerang attack: (1) the data complexity under each related key $D' = \frac{D}{rk} = 2^{N+r_b}$ should be less than 2^{n+t}, and the total data complexity D should be less than $2^{n+t+\log_2 rk}$; (2) the time complexity T should be less than 2^{h-t}.

4 Applications to Deoxys-BC and Joltik-BC

4.1 Description of Deoxys-BC

Deoxys [JNPS16] is an authenticated encryption scheme selected as one of the finalists for the CAESAR competition. As its internal primitive, Deoxys-BC is a 128-bit block cipher conforming to the TWEAKEY framework [JNP14]. Deoxys-BC has two versions according to different tweakey sizes: for Deoxys-BC-256 the tweakey size is 256 bits, while for Deoxys-BC-384 it is 384 bits.

Deoxys-BC is an AES-like design, it adopts an iterative substitution-permutation network (SPN) that transforms the internal states through a round function similar to that of AES. Deoxys-BC-256 has 14 rounds, while Deoxys-BC-384 has 16 rounds. The ordering of the internal state and the tweakey state is represented by a 4×4 matrix:

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

Each round function consists of the four transformations in the order specified below:

- AddRoundTweakey (ART): XOR the 128-bit round subtweakey to the internal state.
- SubBytes (SB): Apply the 8-bit AES S-box $\mathcal{S}$ to the 16 bytes of the internal state.
- ShiftRows (SR): Rotate the 4-byte i-th row left by i positions, $i = 0, 1, 2, 3$.
- MixColumns (MC): Multiply the internal state by the 4×4 MDS matrix of AES.

At the end of the last round, a final AddRoundTweakey operation is applied to the internal state to produce the ciphertext.

Tweakey Schedule. Different from the key schedule of AES, Deoxys-BC used a linear tweakey schedule under the TWEAKEY framework. We denote the concatenation of the key K and the tweak K as KT, i.e. $KT = K||T$. For Deoxys-BC-256, the size of KT is 256 bits with the first (most significant) 128 bits denoted as W_1, the second W_2, while the 384 bits tweakey of Deoxys-BC-384 is divided into W_1, W_2 and W_3 per 128 bits sequentially. For Deoxys-BC-256, a subtweakey of i-th round is defined as $STK_i = TK_i^1 \oplus TK_i^2 \oplus RC_i$ while for the case of Deoxys-BC-384 it is defined as $STK_i = TK_i^1 \oplus TK_i^2 \oplus TK_i^3 \oplus RC_i$.

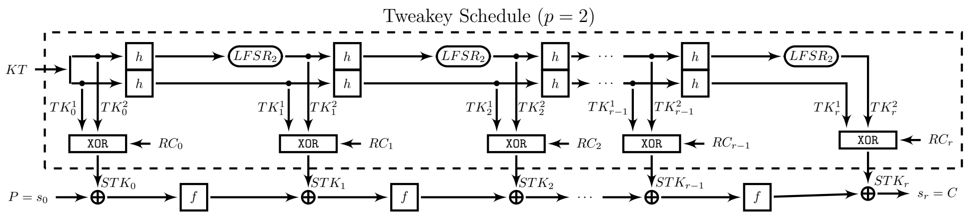


Figure 9: Tweakey schedule of `Deoxys-BC-256`

The 128-bit words TK_i^1, TK_i^2, TK_i^3 are outputs produced by tweakey schedule algorithm, initialized with $TK_0^1 = W_1$ and $TK_0^2 = W_2$ for `Deoxys-BC-256` and with $TK_0^1 = W_1, TK_0^2 = W_2$ and $TK_0^3 = W_3$ for `Deoxys-BC-384`. The tweakey schedule algorithm is defined as

$$TK_{i+1}^1 = h(TK_i^1), TK_{i+1}^2 = h(LFSR_2(TK_i^2)), TK_{i+1}^3 = h(LFSR_3(TK_i^3)),$$

where the byte permutation h is defined as:

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

The $LFSR_2$ and $LFSR_3$ functions are the application of an LFSR to each of the 16 bytes of a tweakey 128-bit word. The two LFSRs used are given in Table 5.

Table 5: Two LFSRs used in `Deoxys-BC` tweakey schedule

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

Additionally, for the specifics of round constants RC_i, please refer to [JNPS16]. Figure 9 illustrates an instantiation of the `TWEAKEY` framework for `Deoxys-BC-256`, the one for `Deoxys-BC-384` is similar.

Proposition 2 (Subtweakey Difference Cancellation [JNPS16])**.** *For* `Deoxys-BC-256`*, suppose that a single cell of TK^1 and TK^2 are active. Let a_1 and a_2 be differences of the active cell, respectively. Thus, the subtweakey difference of the first round is $a_2 \oplus a_1$, and in the i-th round, the subtweakey difference is $a_2 \oplus LFSR_2^i(a_1)$. Since a_1 and a_2 are both nonzero differences, $a_2 \oplus LFSR_2^i(a_1) = 0$ can happen only once over 15 consecutive subtweakeys. For* `Deoxys-BC-384`*, suppose that a single cell of TK^1, TK^2 and TK^3 are active. Let a_1, a_2 and a_3 be differences of the active cell, respectively. Thus, the subtweakey difference of the first round is $a_1 \oplus a_2 \oplus a_3$, and in the i-th round, the subtweakey difference is $a_1 \oplus LFSR_2^i(a_2) \oplus LFSR_3^i(a_3)$. Since a_1, a_2 and a_3 are both nonzero differences, the cancellation $a_1 \oplus LFSR_2^i(a_2) \oplus LFSR_3^i(a_3) = 0$ can happen twice for 15 consecutive subtweakeys.*

4.2 Description of `Joltik-BC`

`Joltik-BC` is a lightweight ad-hoc tweakable block cipher of the authenticated encryption scheme `Joltik` [JNP15]. Similar to `Deoxys-BC`, `Joltik-BC` uses an `AES`-like round function and conforms to the `TWEAKEY` framework. Adopting a lightweight and hardware-oriented design, `Joltik-BC` has a 64-bit state, and it has two versions `Joltik-BC-128` and `Joltik-BC-192` according to different tweakey sizes. The number of rounds is 24 for

Joltik-BC-128 and 32 for Joltik-BC-192. Joltik-BC uses a 4-bit S-box and an involutory MDS matrix, and the subtweakey update function in the tweakey schedule is a bit different from Deoxys-BC. For a more detailed specification of Joltik-BC, please refer to [JNP15]. The tweakey schedules of Joltik-BC family also have the property explained in Proposition 2.

4.3 Related-Tweakey Impossible Boomerang Attack on 10-Round Deoxys-BC-256 and 10-Round Joltik-BC-128

For all the attacks in this paper, we omit the MixColumns in the last round as it is a linear operation. Due to the similarities in the round function and tweakey schedule of Deoxys-BC and Joltik-BC, we can mount a 10-round related-tweakey impossible boomerang attack on these two ciphers. We prefix 1 round at the beginning and append 2 rounds at the end of a 7-round distinguisher to mount the attack, as shown in Figure 10. The distinguishers for Deoxys-BC-256 and Joltik-BC-128 are listed in Table 6 and Table 8, respectively. We denote the cell size as c, with $c = 4$ for Joltik-BC-128 and $c = 8$ for Deoxys-BC-256. The key recovery part follows the impossible differential style, as proposed in Section 3.4.1, and the attack begins by constructing quartets on the plaintext side.

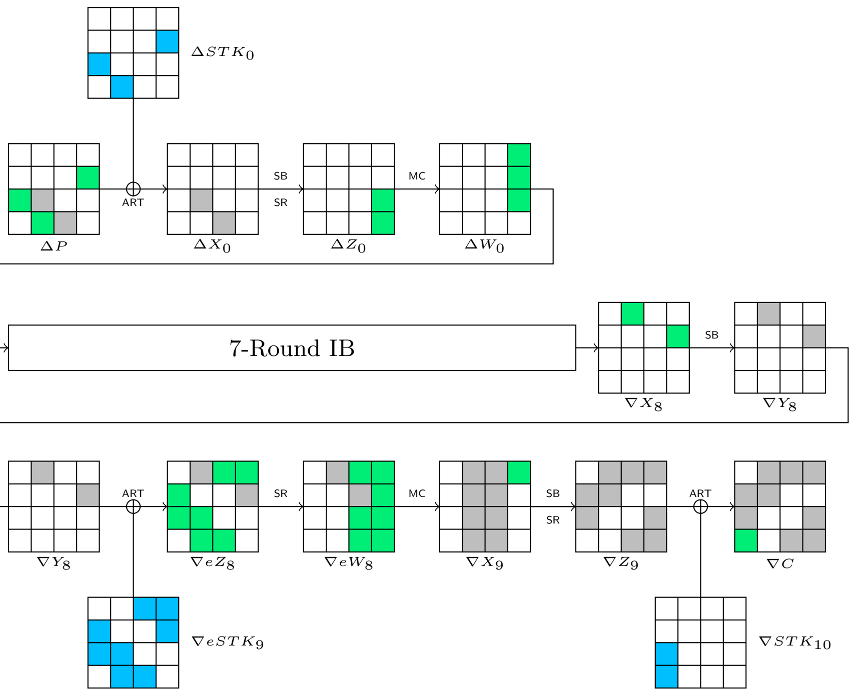
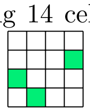


Figure 10: The related-tweakey impossible boomerang attack against 10-round Deoxys-BC-256 and Joltik-BC-128. The white color stands for zero difference in both tweakey and internal states, the green (blue) color stands for known-nonzero difference in internal (tweakey) states, and the grey color stands for unknown difference.

Data Collection. Prepare a structure S of size 2^{2c} by traversing the 2 gray cells $\Delta P[6, 11]$ of the plaintext and fixing the remaining 14 cells to constants. Then, prepare another structure S' by XORing the difference  to each element of S. We can obtain 2^{4c} ordered pairs (P_1, P_2) from the two structures, each XORed difference conforms to

ΔP. For 2^n structures, we can construct $(2^{n+4c})^2 = 2^{2n+8c}$ ordered plaintext quartets (P_1, P_2, P_3, P_4). Encrypt P_1, P_2, P_3, P_4 with K_1,$K_2 = K_1 \oplus \Delta K$,$K_3 = K_1 \oplus \nabla K$,$K_4 = K_1 \oplus \Delta K \oplus \nabla K$, and get the corresponding ciphertext quartets (C_1, C_2, C_3, C_4). Then filter the ciphertext quartets according to the 7 known differences of ∇C. Finally, there are $2^{2n+8c-14c} = 2^{2n-6c}$ ciphertext quartets remaining.

Guess-and-Filter. We make use of the 2^{2n-6c} quartets to eliminate wrong key bits, and then exhaust the remaining key bits to recover the full key. The procedure of the tweakey recovery phase is briefly described in Algorithm 1.

Algorithm 1: Guess-and-Filter Phase of Related-Tweakey Impossible Boomerang Attacks on 10-round `Deoxys-BC-256` and `Joltik-BC-128`

for 2^c *guesses of* $STK_0[6]$ **do**
  for 2^{2n-6c} *remaining quartets* **do**
    Filter with known $\Delta Z_0[14]$;
  Obtain 2^{2n-8c} remaining quartets;
  for 2^c *guesses of* $STK_0[11]$ **do**
    for 2^{2n-8c} *remaining quartets* **do**
      Filter with known $\Delta Z_0[15]$.
    Obtain 2^{2n-10c} remaining quartets;
    for 2^c *guesses of* $STK_{10}[12]$ **do**
      for 2^{2n-10c} *remaining quartets* **do**
        Filter with known $\nabla X_9[12]$.
      Obtain 2^{2n-12c} remaining quartets;
      for 2^{4c} *guesses of* $STK_{10}[1, 4, 11, 14]$ **do**
        for 2^{2n-12c} *remaining quartets* **do**
          Filter with known $\nabla eW_8[5-7]$.
        Obtain 2^{2n-18c} remaining quartets;
        for 2^{4c} *guesses of* $STK_{10}[2, 5, 8, 15]$ **do**
          for 2^{2n-18c} *remaining quartets* **do**
            Filter with known $\nabla eW_8[8, 10, 11]$.
          Obtain 2^{2n-24c} remaining quartets;
          for 2^c *guesses of* $eSTK_9[4]$ **do**
            for 2^{2n-24c} *remaining quartets* **do**
              Filter with known $\nabla X_8[4]$.
            Obtain 2^{2n-26c} remaining quartets;
            for 2^c *guesses of* $eSTK_9[13]$ **do**
              for 2^{2n-26c} *remaining quartets* **do**
                Use the known $\nabla X_8[13]$ to filter out the wrong subtweakeys.

1. Guess 2^c possible values of $STK_0[6]$ and partially encrypt (P_1, P_2, P_3, P_4) for one round, then use the known difference cell $\Delta Z_0[14]$ to filter the quartets. There are about $2^{2n-6c} \cdot 2^{-2c} = 2^{2n-8c}$ remaining quartets. The time complexity of this step is $2^c \cdot 2^{2n-6c+2} \cdot \frac{1}{16 \cdot 10} \approx 2^{2n-5c-5.32}$.
2. Guess 2^c possible values of $STK_0[11]$ and partially encrypt (P_1, P_2, P_3, P_4) for one round, then use the known difference cell $\Delta Z_0[15]$ to filter the quartets. There are

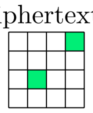
about 2^{2n-10c} remaining quartets. The time complexity of this step is $2^{2c} \cdot 2^{2n-8c+2} \cdot \frac{1}{16 \cdot 10} \approx 2^{2n-6c-5.32}$.

3. Guess 2^c possible values of $STK_{10}[12]$ and partially decrypt (C_1, C_2, C_3, C_4) for one round. Use the known difference cell $\nabla X_9[12]$ to filter quartets and about 2^{2n-12c} quartets left. The time complexity of this step is $2^{3c} \cdot 2^{2n-10c+2} \cdot \frac{1}{16 \cdot 10} \approx 2^{2n-7c-5.32}$.
4. Guess 2^{4c} possible values of $STK_{10}[1, 4, 11, 14]$. Use the three cells $\nabla eW_8[5-7]$ with zero difference from the operations of SB and MC in Round 9 to filter quartets and about 2^{2n-18c} quartets left. The time complexity of this step is $2^{7c} \cdot 2^{2n-12c+2} \cdot \frac{4}{16 \cdot 10} \approx 2^{2n-5c-3.32}$.
5. Guess 2^{4c} possible values of $STK_{10}[2, 5, 8, 15]$. Use the three cells $\nabla eW_8[8, 10, 11]$ with known difference value from the operations of SB and MC in Round 9 to filter quartets and about 2^{2n-24c} quartets left. The time complexity of this step is $2^{11c} \cdot 2^{2n-18c+2} \cdot \frac{4}{16 \cdot 10} \approx 2^{2n-7c-3.32}$.
6. Guess 2^c possible values of $eSTK_9[4]$[2]. Use the known $\nabla X_8[4]$ to filter quartets and keep only 2^{2n-26c} quartets remaining. The time complexity of this step is $2^{12c} \cdot 2^{2n-24c+2} \cdot \frac{1}{16 \cdot 10} \approx 2^{2n-12c-5.32}$.
7. Guess 2^c possible values of $eSTK_9[13]$. Use the known $\nabla X_8[13]$ to filter out wrong candidate subtweakeys. The time complexity of this step is $2^{13c} \cdot 2^{2n-26c+2} \cdot \frac{1}{16 \cdot 10} \approx 2^{2n-13c-5.32}$.

Complexity. In this attack, we have $2^\alpha = 2^{13c} \cdot (1 - 2^{-22c})^{2^{2n-6c}}$. For `Joltik-BC-128`, $c = 4$, we choose $\alpha = 15, n \approx 58.3$, thus the time complexity of the attack is approximately $2^{2n-5c-5.32} + 2^{2n-6c-5.32} + 2^{2n-5c-3.32} + 2^{128-13c+15} \approx 2^{93.8}$, the data complexity is $2^{68.3}$ and the memory complexity is $2^{92.6}$. For `Deoxys-BC-256`, $c = 8$, we choose $\alpha = 30, n \approx 114.8$, thus the time complexity of the attack is approximately $2^{2n-5c-5.32} + 2^{2n-5c-3.32} + 2^{256-13c+30} \approx 2^{186.66}$, the data complexity is $2^{132.8}$ and the memory complexity is $2^{181.6}$.

4.4 Related-Tweakey Impossible Boomerang Attack on 13-Round `Deoxys-BC-384` and 13-Round `Joltik-BC-192`

The 13-round related-tweakey impossible boomerang attack against `Deoxys-BC-384` and `Joltik-BC-192` is based on a 9-round distinguisher each, which are listed in Table 7 and Table 9, respectively. We prefix two rounds at the beginning and append two rounds at the end of the distinguisher to mount the attacks, as shown in Figure 11. The key recovery part follows the boomerang style, as proposed in Section 3.4.2, and the attacks begin by constructing quartets on the ciphertext side.

Data Collection. Prepare a structure S of size 2^{6c} by traversing the 6 gray cells $C[2, 3, 5, 8, 9, 15]$ of the ciphertext and fixing the other 10 cells to constants. For the ciphertexts in S, we query the corresponding plaintexts under two related tweakeys K_1 and $K_2 = K_1 \oplus \Delta K$ and denote the plaintext-ciphertext sets by S_1 and S_2. Then, prepare another structure S' by XORing the difference  to each element of S. We can also get two plaintext-ciphertext sets S_3 and S_4 under the related tweakeys $K_3 = K_1 \oplus \nabla K$ and $K_4 = K_1 \oplus \Delta K \oplus \nabla K$. Then, we insert S_3 and S_4 into hash tables H_1 and H_2 indexed by the 6 gray cells $C[2, 3, 5, 8, 9, 15]$.

For K_1, we guess 2^{8c} possible values of 6 cells $STK_{13}[2, 3, 5, 8, 9, 15]$ and 2 cells $eSTK_{12}[2, 8]$. For each guess, partially decrypt the ciphertexts $C_1 \in S_1$ under the key K_1 to the position at X_{11}, then XOR the decrypted states with the fixed difference ∇X_{11}, after that partially encrypt the XORed states to get the ciphertexts using the known

[2]For the r-th round subtweakey STK_r, its equivalent subtweakey is defined as $eSTK_r = \mathsf{SR}^{-1} \circ \mathsf{MC}^{-1}(STK_r)$.

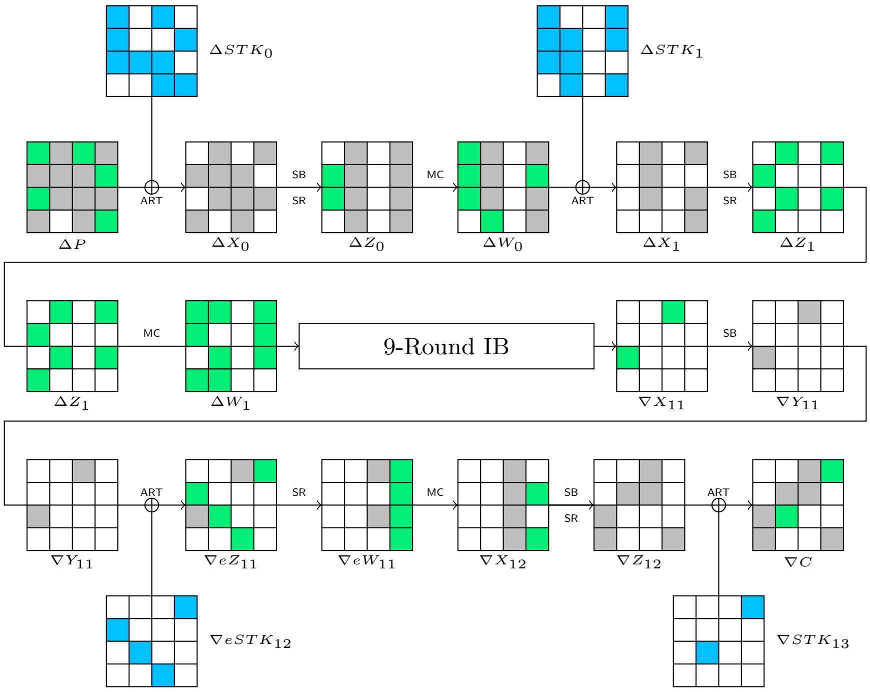


Figure 11: The related-tweakey impossible boomerang attack against 13-round `Deoxys-BC-384` and `Joltik-BC-192`

cells of K_3, finally lookup the hash table H_1 to find collisions indexed by the 6 cells $C[2,3,5,8,9,15]$. The expected number of collisions is 2^{6c}, and we store all the corresponding plaintext-ciphertext pairs into a set

$$L_1 = \{(P_1, C_1, P_3, C_3) : (P_1, C_1) \in S_1, (P_3, C_3) \in S_3, E^{-1}_{f_{K_1}}(C_1) \oplus E^{-1}_{f_{K_3}}(C_3) = \nabla X_{11}\}.$$

Similarly, we can get another set L_2:

$$L_2 = \{(P_2, C_2, P_4, C_4) : (P_2, C_2) \in S_2, (P_4, C_4) \in S_4, E^{-1}_{f_{K_2}}(C_2) \oplus E^{-1}_{f_{K_4}}(C_4) = \nabla X_{11}\}.$$

Then, we insert L_1 in a hash table H_3 indexed by 12 cells $P_1[0,2,7,8,13,15]$ and $P_3[0,2,7,8,13,15]$. For each element (P_2, C_2, P_4, C_4) of L_2, we find the corresponding (P_1, C_1, P_3, C_3) satisfying $P_1 \oplus P_2 \in \Delta P$ and $P_3 \oplus P_4 \in \Delta P$. There are about $2^{6c+n} \cdot 2^{6c+n} \cdot 2^{-12c} = 2^{2n}$ quartets constructed using 2^n structures. In total, we do $2^{8c} \cdot (2 \cdot 2^{6c+n} + 2^{6c+n}) = 3 \cdot 2^{14c+n}$ table lookups and $2^{n+6c+2} + 2^{8c} \cdot 2 \cdot 2^{n+6c} \cdot \frac{8}{16 \cdot 13} \approx 2^{n+14c-3.7}$ encryptions to prepare quartets.

Guess-and-Filter. For each of the 2^{8c} guesses in the data collection phase, we use the 2^{2n} quartets to eliminate wrong key bits, and then exhaust the remaining key bits to recover the full key:

1. For each of the 2^{2n} quartets, we guess 2^c possible values of $STK_0[5]$ and partially encrypt (P_1, P_2, P_3, P_4) for one round. Use the known $\Delta Z_0[1]$ to filter quartets and about 2^{2n-2c} quartets left. The time complexity of this step is $2^{9c} \cdot 2^{2n+2} \cdot \frac{1}{16 \cdot 13} \approx 2^{2n+9c-5.7}$.
2. Guess 2^c possible values of $STK_0[10]$ and partially encrypt (P_1, P_2, P_3, P_4) for one round. Use the known $\Delta Z_0[2]$ to filter quartets and about 2^{2n-4c} quartets left. The time complexity of this step is $2^{10c} \cdot 2^{2n-2c+2} \cdot \frac{1}{16 \cdot 13} \approx 2^{2n+8c-5.7}$.

3. Guess 2^{4c} possible values of $STK_0[3,4,9,14]$ and partially encrypt (P_1,P_2,P_3,P_4) for one round. Use the known $\Delta W_0[7]$ after the MC operation in Round 0 to filter quartets and about only 2^{2n-6c} quartets left. The time complexity of this step is $2^{14c}\cdot 2^{2n-4c+2}\cdot \frac{4}{16\cdot 13}\approx 2^{2n+10c-3.7}$.
4. Guess 2^c possible values of $STK_1[4]$ and use known difference $\Delta Z_1[4]$ to filter quartets. There are about 2^{2n-8c} remaining quartets. Then, guess 2^c possible values of $STK_1[5]$, use $\Delta Z_1[1]$ to filter quartets and 2^{2n-10c} quartets will remain. For the remaining 2^{2n-10c} quartets, guess 2^c possible values of $STK_1[6]$ and use $\Delta Z_1[14]$ to filter the quartets. There are about 2^{2n-12c} remaining quartets that meet the above conditions. The time complexity of this step is $2^{15c}\cdot 2^{2n-6c+2}\cdot \frac{1}{16\cdot 13}+2^{16c}\cdot 2^{2n-8c+2}\cdot \frac{1}{16\cdot 13}+2^{17c}\cdot 2^{2n-10c+2}\cdot \frac{1}{16\cdot 13}\approx 2^{2n+9c-5.7}$.
5. Guess 2^{4c} possible values of $STK_0[1,6,11,12]$ and partially encrypt (P_1,P_2,P_3,P_4) for one round. Use the known $\Delta W_0[13]$ after the MC operation in Round 0 to filter quartets and about 2^{2n-14c} quartets left. The time complexity of this step is $2^{21c}\cdot 2^{2n-12c+2}\cdot \frac{4}{16\cdot 13}\approx 2^{2n+9c-3.7}$.
6. Guess 2^c possible values of $STK_1[12]$ and partially encrypt the remaining quartets. Use the condition of known $\Delta Z_1[12]$ to filter quartets and about 2^{2n-16c} quartets will remain. The time complexity of this step is $2^{22c}\cdot 2^{2n-14c+2}\cdot \frac{1}{16\cdot 13}\approx 2^{2n+8c-5.7}$.
7. Guess 2^c possible values of $STK_1[14]$. Use the condition of known $\Delta Z_1[6]$ to filter quartets and about 2^{2n-18c} quartets will remain. The time complexity of this step is $2^{23c}\cdot 2^{2n-16c+2}\cdot \frac{1}{16\cdot 13}\approx 2^{2n+7c-5.7}$.
8. Guess 2^c possible values of $STK_1[15]$. Use the condition of known $\Delta Z_1[3]$ to filter out wrong subtweakeys. The time complexity of this step is $2^{24c}\cdot 2^{2n-18c+2}\cdot \frac{1}{16\cdot 13}\approx 2^{2n+6c-5.7}$.

Complexity. We reduce all the guessed subtweakey bits to the master tweakey and find that all the 16 cells in the master tweakey have been derived. In the guess-and-filter phase, we have $2^{\alpha}=2^{24c}\cdot(1-2^{-20c})^{2^{2n}}$. For `Joltik-BC-192` with $c=4$, we choose $\alpha=20, n\approx 42.9$, thus the time complexity of the whole attack is $2^{n+14c-3.7}+2^{2n+10c-3.7}+2^{192-24c+\alpha}\approx 2^{122.1}$, the data complexity is $2^{6c+n+2}=2^{68.9}$ and the memory complexity is 2^{96}. For `Deoxys-BC-384` with $c=8$, we choose $\alpha=50, n\approx 83.3$, thus the time complexity of the whole attack is approximately $2^{n+14c-3.7}+2^{2n+10c-3.7}+2^{384-24c+\alpha}\approx 2^{243.5}$, the data complexity is $2^{6c+n+2}=2^{133.3}$ and the memory complexity is 2^{192}.

4.5 Related-Tweakey Impossible Boomerang Attack on 14-Round `Deoxys-BC-384` and 14-Round `Joltik-BC-192`

The 13-round attack can be directly extended to a 14-round attack by appending one round to the last round, which makes the states in the final round fully active.

At the data collection phase, we need to guess the full STK_{14}, compared to the 13-round attack. We prepare a structure of ciphertexts of size 2^x, and then by guessing 2^{8c+16c} possible values of the subtweakeys STK_{12}, STK_{13} and STK_{14}, we can obtain the set L_1 and L_2 of size 2^{2x-16c} for each guess. Then $2^{2(2x-16c)}\cdot 2^{-12c}=2^{4x-44c}$ quartets can be constructed. The time complexity of data collection is $2^{24c}\cdot 2\cdot 2^x\cdot \frac{24}{16\cdot 14}=2^{x+24c-2.22}$. The guess-and-filter phase follows exactly the same as the 13-round attack in Section 4.4. Finally, we have $2^{\alpha}=2^{40c}\cdot(1-2^{-20c})^{2^{4x-44c}}$.

For `Joltik-BC-192`, $c=4$, we choose $\alpha=150, x\approx 64.7$, thus the time complexity of the whole attack is $2^{x+24c-2.22}+2^{4x-18c-3.7}+2^{192-40c+\alpha}\approx 2^{183.65}$, the data complexity is $2^{x+2}=2^{66.7}$ and the memory complexity is 2^{160}. For `Deoxys-BC-384`, $c=8$, we choose $\alpha=300, x\approx 128.9$, thus the time complexity of the whole attack is approximately 2^{368}, the data complexity is $2^{130.9}$ and the memory complexity is 2^{320}.

5 Applications to SKINNY

5.1 Related-Tweakey Impossible Boomerang Attacks on 27-Round SKINNY-n-$3n$

In this section, we provide a 27-round related-tweakey impossible boomerang attack against SKINNY-n-$3n$, the specification of SKINNY is given in Appendix E. Though the last round of SKINNY completes the full round function, we omit the SR and MC operations in the last round as they are linear operations. The impossible boomerang distinguishers used in this attack are depicted in Figure 14 and Figure 15. We prefix 4 rounds at the beginning and append 5 rounds at the end of the 18-round distinguisher ($\Delta Y_4 \to \nabla X_{22}$) to mount a 27-round related-tweakey impossible boomerang attack, as shown in Figure 12. The subtweakey bits involved in E_b and E_f are listed in Table 10. The guess-and-filter part follows the boomerang style (introduced in Section 3.4.2) and the attack begins by constructing quartets on the ciphertext side.

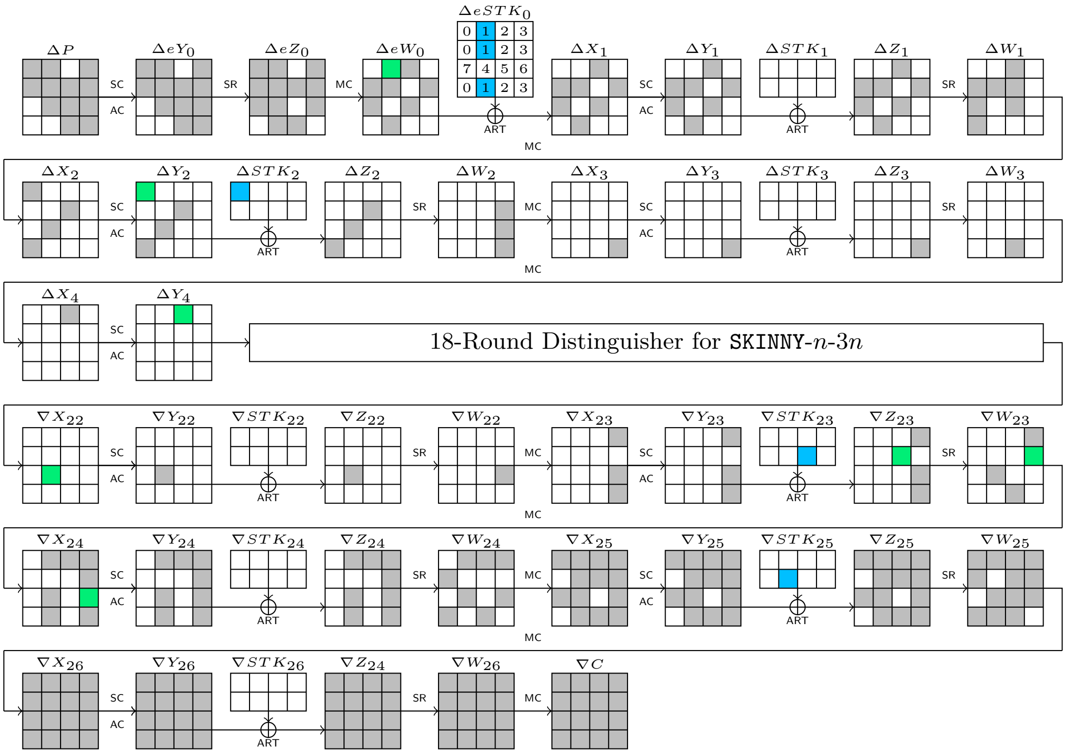


Figure 12: The related-tweakey impossible boomerang attack against 27-round SKINNY-n-$3n$

Data Collection. We prepare a set of ciphertexts of size 2^x, and then by guessing 2^{24c} possible values of $STK_{26}[0-7], STK_{25}[0-7], STK_{24}[1,2,3,4,5,7], STK_{23}[3,7]$, we can obtain two sets L_1 and L_2 of size 2^{2x-16c} for each guess. Then, $2^{2(2x-16c)} = 2^{4x-32c}$ quartets satisfying $\nabla C \to \nabla X_{22}$ can be constructed. Because of the equivalent representation of the first key $eSTK_0 = \mathsf{MC} \circ \mathsf{SR}(STK_0)$ in the first round, we can filter the quartets obtained by ΔeW_0 and about $2^{4x-32c-2\cdot 9c} = 2^{4x-50c}$ quartets will remain. In total, $2^{24c}(2\cdot 2^x + 2^{2x-16c}) = 2^{x+24c+1} + 2^{2x+8c}$ table lookups and $2^{24c}\cdot 2\cdot 2^x \cdot \frac{48}{16\cdot 27} = 2^{x+24c-2.17}$ encryptions are needed in this phase.

Guess-and-Filter. For each of the 2^{24c} guesses in the data collection phase, we use the 2^{4x-50c} quartets to discard wrong key bits, and then exhaust the remaining key bits to recover the full key.

1. Satisfying Round 1:
   (a) Guess 2^{2c} possible values of $eSTK_0[2,8]$. Use the condition $\Delta W_1[2] = \Delta W_1[10]$ from the MC operation in round 1 to filter the quartets, and about $2^{4x-50c-2c} = 2^{4x-52c}$ quartets will remain. The time complexity of this step is $2^{26c} \cdot 2^{4x-50c+2} \cdot \frac{2}{27 \cdot 16} = 2^{4x-24c-5.75}$.
   (b) Guess 2^{c} possible values of $eSTK_0[5]$. Use the condition $\Delta W_1[6] = \Delta W_1[10]$ from the MC operation in round 1 to filter the quartets, and about 2^{4x-54c} quartets will remain. The time complexity of this step is $2^{27c} \cdot 2^{4x-52c+2} \cdot \frac{1}{27 \cdot 16} = 2^{4x-25c-6.75}$.
   (c) Guess 2^{2c} possible values of $eSTK_0[7,10]$. Use the condition $\Delta W_1[4] = \Delta W_1[8]$ from the MC operation in round 1 to filter the quartets, and about 2^{4x-56c} quartets will remain. The time complexity of this step is $2^{29c} \cdot 2^{4x-54c+2} \cdot \frac{2}{27 \cdot 16} = 2^{4x-25c-5.75}$.
2. Satisfying Round 2:
   (a) Guess 2^{2c} possible values of $eSTK_0[0], STK_1[0]$. Use the condition $\Delta Y_2[0] = \Delta STK_2[0]$ to filter the quartets and about 2^{4x-58c} quartets will remain. The time complexity of this step is $2^{31c} \cdot 2^{4x-56c+2} \cdot \frac{2}{27 \cdot 16} = 2^{4x-25c-5.75}$.
   (b) Guess 2^{2c} possible values of $eSTK_0[11], STK_1[4]$. Use the condition $\Delta W_2[11] = \Delta W_2[15]$ from the MC operation in round 1 to filter the quartets, and about 2^{4x-60c} quartets will remain. The time complexity of this step is $2^{33c} \cdot 2^{4x-58c+2} \cdot \frac{2}{27 \cdot 16} = 2^{4x-25c-5.75}$.
   (c) Guess 2^{c} possible values of $STK_1[2]$. Use the condition $\Delta W_2[7] = \Delta W_2[11]$ from the MC operation in round 1 to filter the quartets, and about 2^{4x-62c} quartets will remain. The time complexity of this step is $2^{34c} \cdot 2^{4x-60c+2} \cdot \frac{1}{27 \cdot 16} = 2^{4x-26c-6.75}$.
3. Satisfying Round 4: Guess 2^{6c} possible values of $eSTK_0[9]$, $STK_1[3,5,7]$, $STK_2[3]$ and $STK_3[2]$. (For the involved subtweakey cell $STK_2[2]$, we can uniquely determine its value in the tweakey schedule by the previously guessed values of $STK_{26}[3]$, $STK_{24}[5]$ and $eSTK_0[0]$. The same principle applies to another cell, $STK_2[7]$.) Use the known $\Delta Y_4[2]$ of the beginning of the distinguisher to filter out wrong subtweakeys. The time complexity of this step is $2^{40c} \cdot 2^{4x-62c+2} \cdot \frac{6}{27 \cdot 16} = 2^{4x-22c-4.17}$.

Complexity. We have $2^{\alpha} = 2^{40c} \cdot (1 - 2^{-14c})^{2^{4x-50c}}$. When $c = 4, \alpha = 130$, we choose $x \approx 65.1$. In total, the time complexity of this attack is $2^{x+24c-2.17} + 2^{4x-22c-4.17} + 2^{48c-40c+\alpha} \approx 2^{168.23}$. The data complexity is $2^{67.1}$ ciphertexts. The memory complexity is 2^{160}. When $c = 8, \alpha = 265$, we choose $x \approx 129.3$, thus the time complexity of this attack is 2^{337}. The data complexity is $2^{131.3}$ ciphertexts and the memory complexity is 2^{320}.

5.2 Related-Tweakey Impossible Boomerang Attacks on 28-Round SKINNY-n-$3n$

The 27-round attack can be directly extended to 28-round attack by appending one round to the last round. The overall procedure of the attack is similar to Section 5.1.

Data Collection. We prepare a set of ciphertexts of size 2^{x}, and then by guessing 2^{32c} possible values of $STK_{27}[0-7], STK_{26}[0-7], STK_{25}[0-7], STK_{24}[1,2,3,4,5,7]$ and $STK_{23}[3,7]$, we can obtain the sets L_1 and L_2 of size 2^{2x-16c} for each guess. Then, we can construct $2^{2(2x-16c)} = 2^{4x-32c}$ quartets satisfying $\nabla C \to \nabla X_{22}$. After filtering with

ΔeW_0, about $2^{4x-32c-2\cdot 9c} = 2^{4x-50c}$ quartets will remain. In total, $2^{32c}(2\cdot 2^x + 2^{2x-16c}) = 2^{x+32c+1} + 2^{2x+16c}$ table lookups are needed in this phase. The time complexity of this step is $2^{32c} \cdot 2 \cdot 2^x \cdot \frac{32}{16\cdot 28} = 2^{x+32c-2.8}$ approximately.

Guess-and-Filter. For each of the 2^{32c} guesses in the data collection phase, we use the 2^{4x-50c} quartets obtained to recover subtweakeys.

1. Round 1: Guess 2^{2c} possible values of $eSTK_0[2,8]$. Use the condition $\Delta W_1[2] = \Delta W_1[10]$ to filter the quartets and about $2^{4x-50c-2c} = 2^{4x-52c}$ quartets will remain. The time complexity of this step is $2^{34c} \cdot 2^{4x-50c+2} \cdot \frac{2}{28\cdot 16} = 2^{4x-16c-5.8}$.
2. Round 1: Guess 2^c possible values of $eSTK_0[5]$. Use the condition $\Delta W_1[6] = \Delta W_1[10]$ to filter the quartets and about 2^{4x-54c} quartets will remain. The time complexity of this step is $2^{35c} \cdot 2^{4x-52c+2} \cdot \frac{1}{28\cdot 16} = 2^{4x-17c-6.8}$.
3. Round 1: Guess 2^{2c} possible values of $eSTK_0[7,10]$. Use the condition $\Delta W_1[4] = \Delta W_1[8]$ to filter the quartets and about 2^{4x-56c} quartets will remain. The time complexity of this step is $2^{37c} \cdot 2^{4x-54c+2} \cdot \frac{2}{28\cdot 16} = 2^{4x-17c-5.8}$.
4. Round 2: Guess 2^{2c} possible values of $eSTK_0[0], STK_1[0]$. Use the known value of $\Delta Y_2[0]$ to filter the quartets and about 2^{4x-58c} quartets will remain. The time complexity of this step is $2^{39c} \cdot 2^{4x-56c+2} \cdot \frac{2}{28\cdot 16} = 2^{4x-17c-5.8}$.
5. Round 2: Guess 2^c possible values of $eSTK_0[11]$. Use the condition $\Delta W_2[11] = \Delta W_2[15]$ to filter the quartets and about 2^{4x-60c} quartets will remain. The time complexity of this step is $2^{40c} \cdot 2^{4x-58c+2} \cdot \frac{1}{28\cdot 16} = 2^{4x-18c-6.8}$.
6. Round 2: Guess 2^c possible values of $STK_1[2]$. Use the condition $\Delta W_2[7] = \Delta W_2[11]$ to filter the quartets and about 2^{4x-62c} quartets will remain. The time complexity of this step is $2^{41c} \cdot 2^{4x-60c+2} \cdot \frac{1}{28\cdot 16} = 2^{4x-19c-6.8}$.
7. Round 4: Guess 2^{5c} possible values of $eSTK_0[9], STK_1[3,5,7], STK_2[3]$. Use the known $\Delta Y_4[2]$ to filter out wrong subtweakeys. The time complexity of this step is $2^{46c} \cdot 2^{4x-62c+2} \cdot \frac{5}{28\cdot 16} = 2^{4x-16c-4.22}$.

Complexity. We have $2^\alpha = 2^{46c} \cdot (1-2^{-14c})^{2^{4x-50c}}$. When $c = 4, \alpha = 180$, we choose $x \approx 64.37$, thus the time complexity of this attack is $2^{x+32c-2.8} + 2^{4x-16c-5.8} + 2^{4x-16c-4.22} + 2^{48c-46c+\alpha} \approx 2^{190.8}$ approximately. The data complexity is $2^{66.37}$ ciphertexts and the memory complexity is 2^{184}. When $c = 8, \alpha = 365$, we choose $x \approx 128.26$, thus the time complexity of this attack is $2^{382.8}$ approximately. The data complexity is $2^{130.26}$ ciphertexts and the memory complexity is 2^{368}.

6 Conclusions

In this paper, we revisit the impossible boomerang attack. We introduce a systematic overview of the generation of impossible boomerang distinguishers, analyze the advantages of impossible boomerang attacks over impossible differential attacks, and propose two key recovery methods for impossible boomerang attacks. Based on MIQCP, we propose an automatic tool for searching complete impossible boomerang attacks and successfully apply it to three tweakable block ciphers: `Deoxys-BC`, `Joltik-BC` and `SKINNY`. In particular, the results for `Deoxys-BC-384`, `Joltik-BC-128`, `Joltik-BC-192`, `SKINNY-64-192` and `SKINNY-128-384` have all improved the best previous related-tweakey impossible differential attacks, demonstrating the power of the impossible boomerang attack. Our cryptanalytic results show that the impossible boomerang attack needs more attention in the design and analysis of block ciphers. In addition, the MIQCP tool has the potential to be extended for modeling other cryptanalytic methods, due to the convenience provided by the quadratic constraints in describing block ciphers.

Acknowledgments

The authors would like to thank the anonymous reviewers for their valuable comments and suggestions to improve the quality of the paper. The authors would also like to thank the authors of [BL24] for their valuable feedback. This work was supported by the National Natural Science Foundation of China (Grants No. 62302293 and 62272303).

References

[ABC+17] Ralph Ankele, Subhadeep Banik, Avik Chakraborti, Eik List, Florian Mendel, Siang Meng Sim, and Gaoli Wang. Related-key impossible-differential attack on reduced-round SKINNY. In *ACNS 17*, volume 10355 of *LNCS*, pages 208–228. Springer, Heidelberg, July 2017.

[ARS+22] Seyyed Arash Azimi, Adrián Ranea, Mahmoud Salmasizadeh, Javad Mohajeri, Mohammad Reza Aref, and Vincent Rijmen. A bit-vector differential model for the modular addition by a constant and its applications to differential and impossible-differential cryptanalysis. *Des. Codes Cryptogr.*, 90(8):1797–1855, 2022.

[BBS99a] Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of Skipjack reduced to 31 rounds using impossible differentials. In *EUROCRYPT'99*, volume 1592 of *LNCS*, pages 12–23. Springer, Heidelberg, May 1999.

[BBS99b] Eli Biham, Alex Biryukov, and Adi Shamir. Miss in the middle attacks on IDEA and Khufu. In *FSE'99*, volume 1636 of *LNCS*, pages 124–138. Springer, Heidelberg, March 1999.

[BCD+98] Carolynn Burwick, Don Coppersmith, Edward DAvignon, Rosario Gennaro, Shai Halevi, Charanjit Jutla, Stephen M Matyas Jr, Luke OConnor, Mohammad Peyravian, David Safford, et al. Mars-a candidate cipher for aes. *NIST AES Proposal*, 268:80, 1998.

[BCL+24] Xavier Bonnetain, Margarita Cordero, Virginie Lallemand, Marine Minier, and María Naya-Plasencia. On impossible boomerang attacks application to simon and skinnyee. *IACR Trans. Symmetric Cryptol.*, 2024(2):222–253, 2024.

[BGG+23] Emanuele Bellini, David Gérault, Juan Grados, Rusydi H. Makarim, and Thomas Peyrin. Fully automated differential-linear attacks against ARX ciphers. In *Topics in Cryptology - CT-RSA 2023 - Cryptographers' Track at the RSA Conference 2023, San Francisco, CA, USA, April 24-27, 2023, Proceedings*, volume 13871, pages 252–276. Springer, 2023.

[BHT16] Mihir Bellare, Viet Tung Hoang, and Stefano Tessaro. Message-recovery attacks on feistel-based format preserving encryption. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 444–455. ACM, 2016.

[Bih94] Eli Biham. New types of cryptanalytic attacks using related keys. *Journal of Cryptology*, 7(4):229–246, December 1994.

[BJK+16] Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In

CRYPTO 2016, Part II, volume 9815 of *LNCS*, pages 123–153. Springer, Heidelberg, August 2016.

[BK09] Alex Biryukov and Dmitry Khovratovich. Related-key cryptanalysis of the full AES-192 and AES-256. In *ASIACRYPT 2009*, volume 5912 of *LNCS*, pages 1–18. Springer, Heidelberg, December 2009.

[BL23] Augustin Bariant and Gaëtan Leurent. Truncated boomerang attacks and application to AES-based ciphers. In *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 3–35. Springer, Heidelberg, April 2023.

[BL24] Xavier Bonnetain and Virginie Lallemand. A note on the use of the double boomerang connectivity table (DBCT) for spotting impossibilities. Cryptology ePrint Archive, Paper 2024/1218, 2024. https://eprint.iacr.org/2024/1218.

[BLNS18] Christina Boura, Virginie Lallemand, María Naya-Plasencia, and Valentin Suder. Making the impossible possible. *Journal of Cryptology*, 31(1):101–133, January 2018.

[BNS14] Christina Boura, María Naya-Plasencia, and Valentin Suder. Scrutinizing and improving impossible differential attacks: Applications to CLEFIA, Camellia, LBlock and Simon. In *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 179–199. Springer, Heidelberg, December 2014.

[BS91] Eli Biham and Adi Shamir. Differential cryptanalysis of DES-like cryptosystems. In *CRYPTO'90*, volume 537 of *LNCS*, pages 2–21. Springer, Heidelberg, August 1991.

[BS12] Samuel Burer and Anureet Saxena. The milp road to miqcp. In *Mixed Integer Nonlinear Programming*, pages 373–405, New York, NY, 2012. Springer New York.

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

[CHP+18] Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In *EUROCRYPT 2018, Part II*, volume 10821 of *LNCS*, pages 683–714. Springer, Heidelberg, April / May 2018.

[CJF+16] Tingting Cui, Keting Jia, Kai Fu, Shiyao Chen, and Meiqin Wang. New automatic search tool for impossible differentials and zero-correlation linear approximations. Cryptology ePrint Archive, Report 2016/689, 2016. https://eprint.iacr.org/2016/689.

[CLH+23] Huiqin Chen, Yongqiang Li, Xichao Hu, Zhengbin Liu, Lin Jiao, and Mingsheng Wang. Automatic search model for related-tweakey impossible differential cryptanalysis. In *ACNS 2023 Satellite Workshops, Kyoto, Japan, June 19-22, 2023, Proceedings*, volume 13907 of *Lecture Notes in Computer Science*, pages 3–22. Springer, 2023.

[CY09] Jiali Choy and Huihui Yap. Impossible boomerang attack for block cipher structures. In *IWSEC 09*, volume 5824 of *LNCS*, pages 22–37. Springer, Heidelberg, October 2009.

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

[DEFN22] Patrick Derbez, Marie Euler, Pierre-Alain Fouque, and Phuong Hoa Nguyen. Revisiting related-key boomerang attacks on AES using computer-aided tool. In *ASIACRYPT 2022, Part III*, volume 13793 of *LNCS*, pages 68–88. Springer, Heidelberg, December 2022.

[DF16] Patrick Derbez and Pierre-Alain Fouque. Automatic search of meet-in-the-middle and impossible differential attacks. In *CRYPTO 2016, Part II*, volume 9815 of *LNCS*, pages 157–184. Springer, Heidelberg, August 2016.

[DKS10] Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3G telephony. In *CRYPTO 2010*, volume 6223 of *LNCS*, pages 393–410. Springer, Heidelberg, August 2010.

[DQSW22] Xiaoyang Dong, Lingyue Qin, Siwei Sun, and Xiaoyun Wang. Key guessing strategies for linear key-schedule algorithms in rectangle attacks. In *EUROCRYPT 2022, Part III*, volume 13277 of *LNCS*, pages 3–33. Springer, Heidelberg, May / June 2022.

[HBS21] Hosein Hadipour, Nasour Bagheri, and Ling Song. Improved rectangle attacks on SKINNY and CRAFT. *IACR Trans. Symm. Cryptol.*, 2021(2):140–198, 2021.

[HPW22] Kai Hu, Thomas Peyrin, and Meiqin Wang. Finding all impossible differentials when considering the DDT. Cryptology ePrint Archive, Paper 2022/1034, 2022. https://eprint.iacr.org/2022/1034.

[HSE23] Hosein Hadipour, Sadegh Sadeghi, and Maria Eichlseder. Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks. In *EUROCRYPT 2023, Part IV*, volume 14007 of *LNCS*, pages 128–157. Springer, Heidelberg, April 2023.

[JNP14] Jérémy Jean, Ivica Nikolic, and Thomas Peyrin. Tweaks and keys for block ciphers: The TWEAKEY framework. In *ASIACRYPT 2014, Part II*, volume 8874 of *LNCS*, pages 274–288. Springer, Heidelberg, December 2014.

[JNP15] Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Joltik v1. 3. *CAESAR Round*, 2, 2015.

[JNPS16] Jérémy Jean, Ivica Nikolic, Thomas Peyrin, and Yannick Seurin. Deoxys v1. 41. *Submitted to CAESAR*, 124, 2016.

[KHS+03] Jongsung Kim, Seokhie Hong, Jaechul Sung, Changhoon Lee, and Sangjin Lee. Impossible differential cryptanalysis for block cipher structures. In *INDOCRYPT 2003*, volume 2904 of *LNCS*, pages 82–96. Springer, Heidelberg, December 2003.

[Knu98] Lars Knudsen. Deal-a 128-bit block cipher. *complexity*, 258(2):216, 1998.

[LC21] Manman Li and Shaozhen Chen. Improved meet-in-the-middle attacks on reduced-round Joltik-BC. *IET Inf. Secur.*, 15(3):247–255, 2021.

[LGS17] Guozhen Liu, Mohona Ghosh, and Ling Song. Security analysis of SKINNY under related-tweakey settings (long paper). *IACR Trans. Symm. Cryptol.*, 2017(3):37–72, 2017.

[LJC23] Guangqiu Lv, Chenhui Jin, and Ting Cui. A miqcp-based automatic search algorithm for differential-linear trails of ARX ciphers(long paper). Cryptology ePrint Archive, Paper 2023/259, 2023. https://eprint.iacr.org/2023/259.

[LKKD08] Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Improving the efficiency of impossible differential cryptanalysis of reduced Camellia and MISTY1. In *CT-RSA 2008*, volume 4964 of *LNCS*, pages 370–386. Springer, Heidelberg, April 2008.

[LLWG14] Yiyuan Luo, Xuejia Lai, Zhongming Wu, and Guang Gong. A unified method for finding impossible differentials of block cipher structures. *Inf. Sci.*, 263:211–220, 2014.

[Lu08] Jiqiang Lu. *Cryptanalysis of block ciphers*. PhD thesis, University of London UK, 2008.

[Lu11] Jiqiang Lu. The (related-key) impossible boomerang attack and its application to the AES block cipher. *Des. Codes Cryptogr.*, 60(2):123–143, 2011.

[LWL22] Chenmeng Li, Baofeng Wu, and Dongdai Lin. Generalized boomerang connectivity table and improved cryptanalysis of GIFT. In *Inscrypt 2022, Beijing, China, December 11-13, 2022*, volume 13837 of *Lecture Notes in Computer Science*, pages 213–233. Springer, 2022.

[MMS18] Alireza Mehrdad, Farokhlagha Moazami, and Hadi Soleimany. Impossible differential cryptanalysis on Deoxys-BC-256. Cryptology ePrint Archive, Paper 2018/048, 2018. https://eprint.iacr.org/2018/048.

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

[MWGP11] Nicky Mouha, Qingju Wang, Dawu Gu, and Bart Preneel. Differential and linear cryptanalysis using mixed-integer linear programming. In *Inscrypt 2011, Beijing, China, November 30 - December 3, 2011.*, volume 7537 of *Lecture Notes in Computer Science*, pages 57–76. Springer, 2011.

[RRSY98] Ronald L Rivest, Matthew JB Robshaw, Ray Sidney, and Yiqun Lisa Yin. The rc6tm block cipher. In *First advanced encryption standard (AES) conference*, page 16, 1998.

[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 *ASIACRYPT 2014, Part I*, volume 8873 of *LNCS*, pages 158–178. Springer, Heidelberg, December 2014.

[SLG+16] Bing Sun, Meicheng Liu, Jian Guo, Vincent Rijmen, and Ruilin Li. Provable security evaluation of structures against impossible differential and zero correlation linear cryptanalysis. In *EUROCRYPT 2016, Part I*, volume 9665 of *LNCS*, pages 196–213. Springer, Heidelberg, May 2016.

[SQH19] Ling Song, Xianrui Qin, and Lei Hu. Boomerang connectivity table revisited. *IACR Trans. Symm. Cryptol.*, 2019(1):118–141, 2019.

[ST17] Yu Sasaki and Yosuke Todo. New impossible differential search tool from design and cryptanalysis aspects - revealing structural properties of several ciphers. In *EUROCRYPT 2017, Part III*, volume 10212 of *LNCS*, pages 185–215. Springer, Heidelberg, April / May 2017.

[SYC+24] Ling Song, Qianqian Yang, Yincen Chen, Lei Hu, and Jian Weng. Probabilistic extensions: A one-step framework for finding rectangle attacks and beyond. In *Advances in Cryptology – EUROCRYPT 2024*, pages 339–367. Springer Nature Switzerland, 2024.

[SZY+22] Ling Song, Nana Zhang, Qianqian Yang, Danping Shi, Jiahao Zhao, Lei Hu, and Jian Weng. Optimizing rectangle attacks: A unified and generic framework for key recovery. In *ASIACRYPT 2022, Part I*, volume 13791 of *LNCS*, pages 410–440. Springer, Heidelberg, December 2022.

[Wag99] David Wagner. The boomerang attack. In *FSE'99*, volume 1636 of *LNCS*, pages 156–170. Springer, Heidelberg, March 1999.

[WP19] Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. *IACR Trans. Symm. Cryptol.*, 2019(1):142–169, 2019.

[WW12] Shengbao Wu and Mingsheng Wang. Automatic search of truncated impossible differentials for word-oriented block ciphers. In *INDOCRYPT 2012*, volume 7668 of *LNCS*, pages 283–302. Springer, Heidelberg, December 2012.

[YSS+22] Qianqian Yang, Ling Song, Siwei Sun, Danping Shi, and Lei Hu. New properties of the double boomerang connectivity table. *IACR Trans. Symm. Cryptol.*, 2022(4):208–242, 2022.

[ZD18] Rui Zong and Xiaoyang Dong. Milp-aided related-tweak/key impossible differential attack and its applications to QARMA, Joltik-BC. Cryptology ePrint Archive, Paper 2018/142, 2018. https://eprint.iacr.org/2018/142.

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

[ZDW19] Rui Zong, Xiaoyang Dong, and Xiaoyun Wang. Related-tweakey impossible differential attack on reduced-round Deoxys-BC-256. *Sci. China Inf. Sci.*, 62(3):32102:1–32102:12, 2019.

A Cryptanalytic Tables in Generalized Boomerang Framework

For the previously extended techniques for E_m with multiple rounds, all of them can be generalized to be applicable in GBF:

Definition 7 (Generalized Upper BCT (GUBCT))**.** Let S be an n-bit bijective S-box, and $\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_o, \nabla'_o \in \mathbb{F}_2^n$. The GUBCT of S is a six-dimensional table, in which the entry for $(\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_o, \nabla'_o)$ is given by:

$$\mathtt{GUBCT}(\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_o, \nabla'_o) = \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \\ S(x) \oplus \nabla_o \oplus S(x \oplus \Delta_i) \oplus \nabla'_o = \Delta'_o \\ S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla'_o) = \Delta'_i \end{array} \right\}.$$

Definition 8 (Generalized Lower BCT (GLBCT))**.** Let S be an n-bit bijective S-box, and $\Delta_i, \Delta'_i, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o \in \mathbb{F}_2^n$. The GLBCT of S is a six-dimensional table, in which the entry for $(\Delta_i, \Delta'_i, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o)$ is given by:

$$\mathtt{GLBCT}(\Delta_i, \Delta'_i, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o) = \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} x \oplus S^{-1}(S(x) \oplus \nabla_o) = \nabla_i \\ x \oplus \Delta_i \oplus x \oplus \nabla_i \oplus \Delta'_i = \nabla'_i \\ S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla'_o) = \Delta'_i \end{array} \right\}.$$

Definition 9 (Generalized Extended BCT (GEBCT))**.** Let S be an n-bit bijective S-box, and $\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o \in \mathbb{F}_2^n$. The GEBCT of S is a eight-dimensional table, in which the entry for $(\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o)$ is given by:

$$\begin{aligned} &\mathtt{GEBCT}(\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o) \\ &= \#\left\{ x \in \mathbb{F}_2^n \;\middle|\; \begin{array}{l} S(x) \oplus S(x \oplus \Delta_i) = \Delta_o \\ S(x) \oplus \nabla_o \oplus S(x \oplus \Delta_i) \oplus \nabla'_o = \Delta'_o \\ x \oplus S^{-1}(S(x) \oplus \nabla_o) = \nabla_i \\ x \oplus \Delta_i \oplus x \oplus \nabla_i \oplus \Delta'_i = \nabla'_i \\ S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_i) \oplus \nabla'_o) = \Delta'_i \end{array} \right\}. \end{aligned}$$

Definition 10 (Generalized Double BCT (GDBCT))**.** Let S be an n-bit bijective S-box, and $\Delta_i, \Delta'_i, \nabla_o, \nabla'_o \in \mathbb{F}_2^n$. The GDBCT of S is a four-dimensional table, in which the entry for $(\Delta_i, \Delta'_i, \nabla_o, \nabla'_o)$ is given by:

$$\mathtt{GDBCT}(\Delta_i, \Delta'_i, \nabla_o, \nabla'_o) = \sum_{\Delta_o, \Delta'_o, \nabla_i, \nabla'_i} \mathtt{GUBCT}(\Delta_i, \Delta'_i, \Delta_o, \Delta'_o, \nabla_i, \nabla'_i) \cdot \mathtt{GLBCT}(\Delta_o, \Delta'_o, \nabla_i, \nabla'_i, \nabla_o, \nabla'_o).$$

The GBCT exhibits the following two apparent properties:

Property 1 (Commutativity)**.**

$$\begin{aligned} \mathtt{GBCT}(\Delta_i, \Delta'_i, \nabla_o, \nabla'_o) &= \mathtt{GBCT}(\Delta'_i, \Delta_i, \nabla_o, \nabla'_o) \\ &= \mathtt{GBCT}(\Delta_i, \Delta'_i, \nabla'_o, \nabla_o) \\ &= \mathtt{GBCT}(\Delta'_i, \Delta_i, \nabla'_o, \nabla_o) \end{aligned}$$

Property 2 (Symmetry)**.**

$$\mathtt{GBCT}(\Delta_i, \Delta_i, \nabla_o, \nabla_o) = \mathtt{BCT}(\Delta_i, \nabla_o)$$

Similar to GBCT, the properties of commutativity and symmetry are equally applicable to the generalized tables above. Under the condition of identical opposite differentials, they can be transformed into UBCT, LBCT, and so forth.

B Related-Tweakey Impossible Boomerang Distinguishers for `Deoxys-BC`

Table 6: The 7-round related-tweakey impossible boomerang distinguisher for `Deoxys-BC-256` (Contradiction: 15-th byte in Round 4, $\texttt{BCT}(2c, 2c) = 0$)

ΔTK_0^1: 00 00 5c 00 00 00 00 5c 00 00 00 00 00 b3 00 00
ΔTK_0^2: 00 00 57 00 00 00 00 57 00 00 00 00 00 6c 00 00
∇TK_0^1: 00 00 00 00 d9 00 00 00 00 00 00 00 00 e1 00 00
∇TK_0^2: 00 00 00 00 25 00 00 00 00 00 00 00 00 95 00 00

R	ΔSTK	ΔX	ΔY	ΔW
0				00 00 00 6a 00 00 00 f2 00 00 00 f2 00 00 00 00
1	00 00 00 6a 00 00 00 f2 00 00 00 f2 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
2	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3	00 00 e4 00 00 00 00 00 00 00 d5 00 e4 00 00 00	00 00 e4 00 00 00 00 00 00 00 d5 00 e4 00 00 00	00 00 ?? 00 00 00 00 00 00 00 ?? 00 ?? 00 00 00	?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 00
4	00 00 00 00 00 7e 00 00 00 00 2c 00 00 00 00 2c	?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 2c		
	∇STK	∇X	∇Y	∇W
4			3a ?? ?? ?? ?? 9d ?? ?? ?? ?? e4 ?? ?? ?? ?? 2c	00 59 ?? 00 00 00 00 ?? 00 00 00 00 6f ?? 00 00
5	00 59 00 00 00 00 00 00 00 00 00 00 6f 00 00 00	00 00 ?? 00 00 00 00 ?? 00 00 00 00 00 ?? 00 00	00 00 fd 00 00 00 00 1c 00 00 00 00 00 c5 00 00	00 00 00 00 00 00 00 00 00 00 b5 00 00 00 91 00
6	00 00 00 00 00 00 00 00 00 00 b5 00 00 00 91 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
7	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
8	00 6a 00 00 00 00 00 23 00 00 00 00 00 00 00 00	00 6a 00 00 00 00 00 23 00 00 00 00 00 00 00 00		

Table 7: The 9-round related-tweakey impossible boomerang distinguisher for `Deoxys-BC-384` (Contradiction: 15-th byte in Round 6, $\mathtt{BCT}(4f, 4f) = 0$)

ΔTK_0^1: cb c5 bc 00 00 00 60 00 35 00 95 a5 00 89 00 5e
ΔTK_0^2: f7 48 4d 00 00 00 9b 00 e2 00 d1 d3 00 af 00 26
ΔTK_0^3: 95 34 bc 00 00 00 fa 00 35 00 cb ce 00 89 00 5e
∇TK_0^1: 00 00 00 00 00 3d 00 4a 00 00 00 00 00 00 00 00
∇TK_0^2: 00 00 00 00 00 e4 00 86 00 00 00 00 00 00 00 00
∇TK_0^3: 00 00 00 00 00 19 00 7a 00 00 00 00 00 00 00 00

R	ΔSTK	ΔX	ΔY	ΔW
1				71 e1 00 26 13 00 00 98 00 f1 00 f2 d7 e9 00 00
2	71 e1 00 26 13 00 00 98 00 f1 00 f2 d7 e9 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
3	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
4	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
5	00 d7 e9 00 00 26 71 e1 00 00 98 13 00 f1 00 f2	00 d7 e9 00 00 26 71 e1 00 00 98 13 00 f1 00 f2	00 ?? ?? 00 00 ?? ?? ?? 00 00 ?? ?? 00 ?? 00 ??	?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 00 ?? ?? ?? 00
6	00 78 aa 53 00 e5 bc 00 00 ef 00 6b 00 00 13 4f	?? ?? ?? 53 ?? ?? ?? 00 ?? ?? ?? 6b ?? ?? ?? 4f		
	∇STK	∇X	∇Y	∇W
6			80 ?? ?? ?? ?? c0 ?? ?? ?? ?? c5 ?? ?? ?? ?? 4f	ca ?? 00 00 00 00 ?? 00 00 00 5d ?? 00 00 00 00
7	ca 00 00 00 00 00 00 00 00 00 5d 00 00 00 00 00	00 ?? 00 00 00 00 ?? 00 00 00 00 ?? 00 00 00 00	00 ff 00 00 00 00 67 00 00 00 00 4c 00 00 00 00	00 00 00 00 00 e5 00 00 00 00 00 00 00 31 00 00
8	00 00 00 00 00 e5 00 00 00 00 00 00 00 31 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
9	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
10	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
11	00 00 31 00 00 00 00 00 e5 00 00 00 00 00 00 00	00 00 31 00 00 00 00 00 e5 00 00 00 00 00 00 00		

C Related-Tweakey Impossible Differential Attack against 12-Round Deoxys-BC-384

We mount a 12-round related-tweakey impossible differential attack on Deoxys-BC-384 by prefixing 2 rounds at the beginning and appending 3 rounds at the end of a 7-round distinguisher ($\Delta W_1 \rightarrow \Delta X_9$), shown in Figure 13.

Figure 13: 12-round related-tweakey impossible differential attack against Deoxys-BC-384

Data Collection. Construct 2^n structures that each of them traverses all 2^{96} possible values of $\Delta P[1,2,3,4,6,7,8,9,11,12,13,14]$, then we get 2^{192} plaintext pairs for each structure. Encrypt the plaintexts under two related tweakeys (K_A, K_B), choose the pairs satisfying ciphertext differences and 2^{192+n} pairs would remain.

Guess-and-Filter For each of the remaining pairs:

1. Guess 2^{32} possible values of $STK_0[3,4,9,14]$ and partially encrypt (P_A, P_B) for one round, then check whether $W_{A,0}[5] \oplus W_{B,0}[5] = W_{A,0}[6] \oplus W_{B,0}[6] = W_{A,0}[7] \oplus W_{B,0}[7] = 0$. Keep only $2^{192+n} \cdot 2^{-24} = 2^{168+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+192+1} \cdot 2^{32} \cdot \frac{4}{16 \cdot 12} = 2^{n+219.4}$.
2. Guess 2^{32} possible values of $STK_0[2,7,8,13]$ and partially encrypt (P_A, P_B) for one round, then check whether $W_{A,0}[8] \oplus W_{B,0}[8] = W_{A,0}[11] \oplus W_{B,0}[11] = 0, W_{A,0}[10] \oplus W_{B,0}[10] = \Delta STK_1[10]$. Keep only $2^{168+n} \cdot 2^{-24} = 2^{144+n}$ pairs that meet above conditions remain. Time complexity of this step is $2^{n+168+1} \cdot 2^{64} \cdot \frac{4}{16 \cdot 12} = 2^{n+227.4}$.
3. Guess 2^{32} possible values of $STK_0[1,6,11,12]$ and partially encrypt (P_A, P_B) for one round, then check whether $W_{A,0}[12] \oplus W_{B,0}[12] = W_{A,0}[13] \oplus W_{B,0}[13] = 0, W_{A,0}[15] \oplus W_{B,0}[15] = \Delta STK_1[15]$. Keep only $2^{144+n} \cdot 2^{-24} = 2^{120+n}$ pairs that

meet above conditions remain. Time complexity of this step is $2^{n+144+1} \cdot 2^{96} \cdot \frac{4}{16 \cdot 12} = 2^{n+235.4}$.

4. Guess 2^8 possible values of $STK_1[4]$ and partially encrypt for one round, then check whether $Z_{A,1}[4] \oplus Z_{B,1}[4] = \Delta Z_1[4]$. Keep only $2^{120+n} \cdot 2^{-8} = 2^{112+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+120+1} \cdot 2^{104} \cdot \frac{1}{16 \cdot 12} = 2^{217.4+n}$.
5. Guess 2^8 possible values of $STK_1[9]$ and partially encrypt for one round, then check whether $Z_{A,1}[5] \oplus Z_{B,1}[5] = \Delta Z_1[5]$. Keep only $2^{112+n} \cdot 2^{-8} = 2^{104+n}$ pairs that meet above conditions remain. Time complexity of this step is $2^{n+112+1} \cdot 2^{112} \cdot \frac{1}{16 \cdot 12} = 2^{217.4+n}$.
6. Guess 2^8 possible values of $STK_1[14]$ and partially encrypt for one round, then check whether $Z_{A,1}[6] \oplus Z_{B,1}[6] = \Delta Z_1[6]$. Keep only $2^{104+n} \cdot 2^{-8} = 2^{96+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+104+1} \cdot 2^{120} \cdot \frac{1}{16 \cdot 12} = 2^{217.4+n}$.
7. Guess 2^{32} possible values of $STK_{12}[1, 4, 11, 14]$ and partially decrypt for one round, then check whether $eW_{A,10}[4-6] \oplus eW_{B,10}[4-6] = \Delta eW_{10}[4-6]$. Keep only $2^{96+n} \cdot 2^{-24} = 2^{72+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+96+1} \cdot 2^{152} \cdot \frac{4}{16 \cdot 12} = 2^{243.4+n}$.
8. Guess 2^{32} possible values of $STK_{12}[2, 5, 8, 15]$ and partially decrypt for one round, then check whether $eW_{A,10}[8, 9, 11] \oplus eW_{B,10}[8, 9, 11] = \Delta eW_{10}[8, 9, 11]$. Keep only $2^{72+n} \cdot 2^{-24} = 2^{48+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+72+1} \cdot 2^{184} \cdot \frac{4}{16 \cdot 12} = 2^{251.4+n}$.
9. Guess 2^{32} possible values of $STK_{12}[0, 7, 10, 13]$ and partially decrypt for one round, then check whether $eW_{A,10}[2, 3] \oplus eW_{B,10}[2, 3] = \Delta eW_{10}[2, 3]$. Keep only $2^{48+n} \cdot 2^{-16} = 2^{32+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+48+1} \cdot 2^{216} \cdot \frac{4}{16 \cdot 12} = 2^{259.4+n}$.
10. Guess 2^{32} possible values of $STK_{12}[3, 6, 9, 12]$ and partially decrypt for one round, then check whether $eW_{A,10}[12, 15] \oplus eW_{B,10}[12, 15] = \Delta eW_{10}[12, 15]$. Keep only $2^{32+n} \cdot 2^{-16} = 2^{16+n}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+32+1} \cdot 2^{248} \cdot \frac{4}{16 \cdot 12} = 2^{275.4+n}$.
11. Guess 2^8 possible values of $eSTK_{11}[5]$ and partially decrypt for one round, then check whether $X_{A,10}[5] \oplus X_{B,10}[5] = \Delta X_{10}[5]$. Keep only $2^{16+n} \cdot 2^{-8} = 2^{8+n}$ pairs that meet above conditions remain. Time complexity of this step is $2^{n+16+1} \cdot 2^{256} \cdot \frac{1}{16 \cdot 12} = 2^{265.4+n}$.
12. Guess 2^8 possible values of $eSTK_{11}[6]$ and partially decrypt for one round, then check whether $X_{A,10}[6] \oplus X_{B,10}[6] = \Delta X_{10}[6]$. Keep only $2^{8+n} \cdot 2^{-8} = 2^n$ pairs that meet above condition remain. Time complexity of this step is $2^{n+8+1} \cdot 2^{264} \cdot \frac{1}{16 \cdot 12} = 2^{265.4+n}$.
13. Guess 2^{32} possible values of $eSTK_{11}[0-3]$ and partially decrypt for one round, then check whether $eW_{A,9}[0-1] \oplus eW_{B,9}[0-1] = \Delta eW_9[0-1]$. Keep only $2^n \cdot 2^{-16} = 2^{n-16}$ pairs that meet above condition remain. Time complexity of this step is $2^{n+1} \cdot 2^{296} \cdot \frac{4}{16 \cdot 12} = 2^{291.4+n}$.
14. Guess 2^8 possible values of $eSTK_{10}[10]$ and partially decrypt for one round, then check whether $X_{A,9}[10] \oplus X_{B,9}[10] = \Delta X_9[10]$. Keep only $2^{n-16} \cdot 2^{-8} = 2^{n-24}$ pairs that meet above condition remain. Time complexity of this step is $2^{n-16+1} \cdot 2^{304} \cdot \frac{1}{16 \cdot 12} = 2^{281.4+n}$.
15. Guess 2^8 possible values of $eSTK_{10}[15]$ and partially decrypt for one round, then use the condition $X_{A,9}[15] \oplus X_{B,9}[15] = \Delta X_9[15]$ to filter out wrong tweakey bits. Time complexity of this step is $2^{n-24+1} \cdot 2^{312} \cdot \frac{1}{16 \cdot 12} = 2^{281.4+n}$.

Complexity. In this attack, we have $2^\alpha = 2^{312}(1-2^{-28.8})^{2^{192+n}}$. We choose $\alpha = 200, n \approx 38.3$, thus the time complexity of the whole attack is $2^{291.4+n} \approx 2^{329.7}$ approximately. The data complexity is $2^{135.3}$ and the memory complexity is 2^{312}.

D Related-Tweakey Impossible Boomerang Distinguishers for Joltik-BC

Table 8: The 7-round related-tweakey impossible boomerang distinguisher for `Joltik-BC-128` (Contradiction: 15-th byte in Round 4, $\texttt{BCT}(8,8) = 0$)

ΔTK_0^1: 0 0 c 0 0 0 0 7 0 0 0 0 0 f 0 0
ΔTK_0^2: 0 0 3 0 0 0 0 5 0 0 0 0 0 7 0 0
∇TK_0^1: 0 0 0 0 f 0 0 0 0 0 0 0 0 7 0 0
∇TK_0^2: 0 0 0 0 6 0 0 0 0 0 0 0 0 8 0 0

R	ΔSTK	ΔX	ΔY	ΔW
0				0 0 0 1 0 0 0 a 0 0 0 d 0 0 0 0
1	0 0 0 1 0 0 0 a 0 0 0 d 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 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 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
3	0 0 9 0 0 0 0 0 0 0 2 0 7 0 0 0	0 0 9 0 0 0 0 0 0 0 2 0 7 0 0 0	0 0 ? 0 0 0 0 0 0 0 ? 0 ? 0 0 0	? ? ? 0 ? ? ? 0 ? ? ? 0 ? ? ? 0
4	0 0 0 0 0 6 0 0 0 0 9 0 0 0 0 8	? ? ? 0 ? ? ? 0 ? ? ? 0 ? ? ? 8		
	∇STK	∇X	∇Y	∇W
4			2 ? ? ? ? 4 ? ? ? ? 6 ? ? ? ? 8	0 2 ? 0 0 0 0 ? 0 0 0 0 8 ? 0 0
5	0 2 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 0	0 0 7 0 0 0 0 2 0 0 0 0 0 9 0 0	0 0 0 0 0 0 0 0 0 0 1 0 0 0 d 0
6	0 0 0 0 0 0 0 0 0 0 1 0 0 0 d 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
7	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
8	0 2 0 0 0 0 0 9 0 0 0 0 0 0 0 0	0 2 0 0 0 0 0 9 0 0 0 0 0 0 0 0		

Table 9: The 9-round related-tweakey impossible boomerang distinguisher for `Joltik-BC-192` (Contradiction: 15-th byte in Round 6, $\texttt{BCT}(6, 9) = 0$)

ΔTK_0^1: 3 d 9 0 0 0 a 0 b 0 f 6 0 5 0 8
ΔTK_0^2: 3 d 9 0 0 0 a 0 b 0 f 6 0 5 0 8
ΔTK_0^3: f c b 0 0 0 4 0 1 0 6 d 0 2 0 e
∇TK_0^1: 0 0 0 0 0 1 0 9 0 0 0 0 0 0 0 0
∇TK_0^2: 0 0 0 0 0 a 0 5 0 0 0 0 0 0 0 0
∇TK_0^3: 0 0 0 0 0 e 0 7 0 0 0 0 0 0 0 0

R	ΔSTK	ΔX	ΔY	ΔW
1				8 c 0 5 f 0 0 9 0 2 0 d 4 b 0 0
2	8 c 0 5 f 0 0 9 0 2 0 d 4 b 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 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 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 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 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
5	0 6 7 0 0 e c a 0 0 4 1 0 3 0 2	0 6 7 0 0 e c a 0 0 4 1 0 3 0 2	0 ? ? 0 0 ? ? ? 0 0 ? ? 0 ? 0 ?	? ? ? 0 ? ? ? 0 ? ? ? 0 ? ? ? 0
6	0 c 2 3 0 f 7 0 0 e 0 9 0 0 1 6	? ? ? 3 ? ? ? 0 ? ? ? 9 ? ? ? 6		
	∇STK	∇X	∇Y	∇W
6			2 ? ? ? ? 8 ? ? ? ? 1 ? ? ? ? 9	2 ? 0 0 0 0 ? 0 0 0 6 ? 0 0 0 0
7	2 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 0 0 0	0 2 0 0 0 0 1 0 0 0 0 c 0 0 0 0	0 0 0 0 0 a 0 0 0 0 0 0 0 5 0 0
8	0 0 0 0 0 a 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 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 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 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 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 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
11	0 0 e 0 0 0 0 0 f 0 0 0 0 0 0 0	0 0 e 0 0 0 0 0 f 0 0 0 0 0 0 0		

E Description of SKINNY

SKINNY is a tweakable block cipher family following the TWEAKEY framework, first proposed in [BJK+16]. SKINNY family has 6 versions, denoted by SKINNY-n-t: $n \in \{64, 128\}$ is the block size and $t \in \{n, 2n, 3n\}$ is the tweakey size. The cell size c is 4 for SKINNY-64 and 8 for SKINNY-128. The number r of rounds is 32 for SKINNY-64-64, 36 for SKINNY-64-128, 40 for SKINNY-64-192 and SKINNY-128-128, 48 for SKINNY-128-256 and 56 for SKINNY-128-384. The ordering of the internal state and the tweakey state is represented by a 4×4 matrix:

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

The SKINNY round function applies five transformations: SubCells (SC), AddConstants (AC), AddRoundTweakey (ART), ShiftRows (SR), MixColumns (MC):

- SubCells (SC): Apply a 4-bit (resp. 8-bit) S-box on each cell for SKINNY-64 (resp. SKINNY-128).
- AddConstants (AC): XOR the round constant to the internal state,
- AddRoundTweakey (ART): XOR the first and second rows of subtweakey with the corresponding cells in the internal state,
- ShiftRows (SR): Rotate the 4-cell i-th row right by i positions, $i = 0, 1, 2, 3$,
- MixColumns (MC): Multiply the internal state by a binary matrix $\mathrm{M} = \begin{pmatrix} 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$.

We denote the internal states in the r-th round as follows:

$$X_r \xrightarrow[\mathsf{AC}]{\mathsf{SC}} Y_r \xrightarrow[STK_r]{\mathsf{ART}} Z_i \xrightarrow{\mathsf{SR}} W_i \xrightarrow{\mathsf{MC}} X_{r+1}.$$

Similar to Deoxys-BC, the tweakey schedule of SKINNY is a linear algorithm and satisfies the property explained in Proposition 2. It divides the master tweakey into z tweakey arrays $(TK1, ..., TKz)$ with n-bit length each, where $z = \frac{t}{n} \in \{1, 2, 3\}$. $TK1$, $TK2$ and $TK3$ follow three independent update functions. The subtweakey used in r-th round STK_r is generated from:

- $STK_r = TK1_r$ when $z = 1$,
- $STK_r = TK1_r \oplus TK2_r$ when $z = 2$,
- $STK_r = TK1_r \oplus TK2_r \oplus TK3_r$ when $z = 3$,

where $TK1_r, TK2_r, TK3_r$ denote the tweakey arrays in round r and are generated as follows. First, a permutation h is applied to each tweakey array as $TKz_{r+1}[i] \leftarrow TKz_r[h[i]]$. Next, each cell of the first and second rows of $TK2_r$ and $TK3_r$ are individually updated with an LFSR. For more details about SKINNY, please refer to [BJK+16].

F Related-Tweakey Impossible Boomerang Distinguishers for SKINNY-n-$3n$

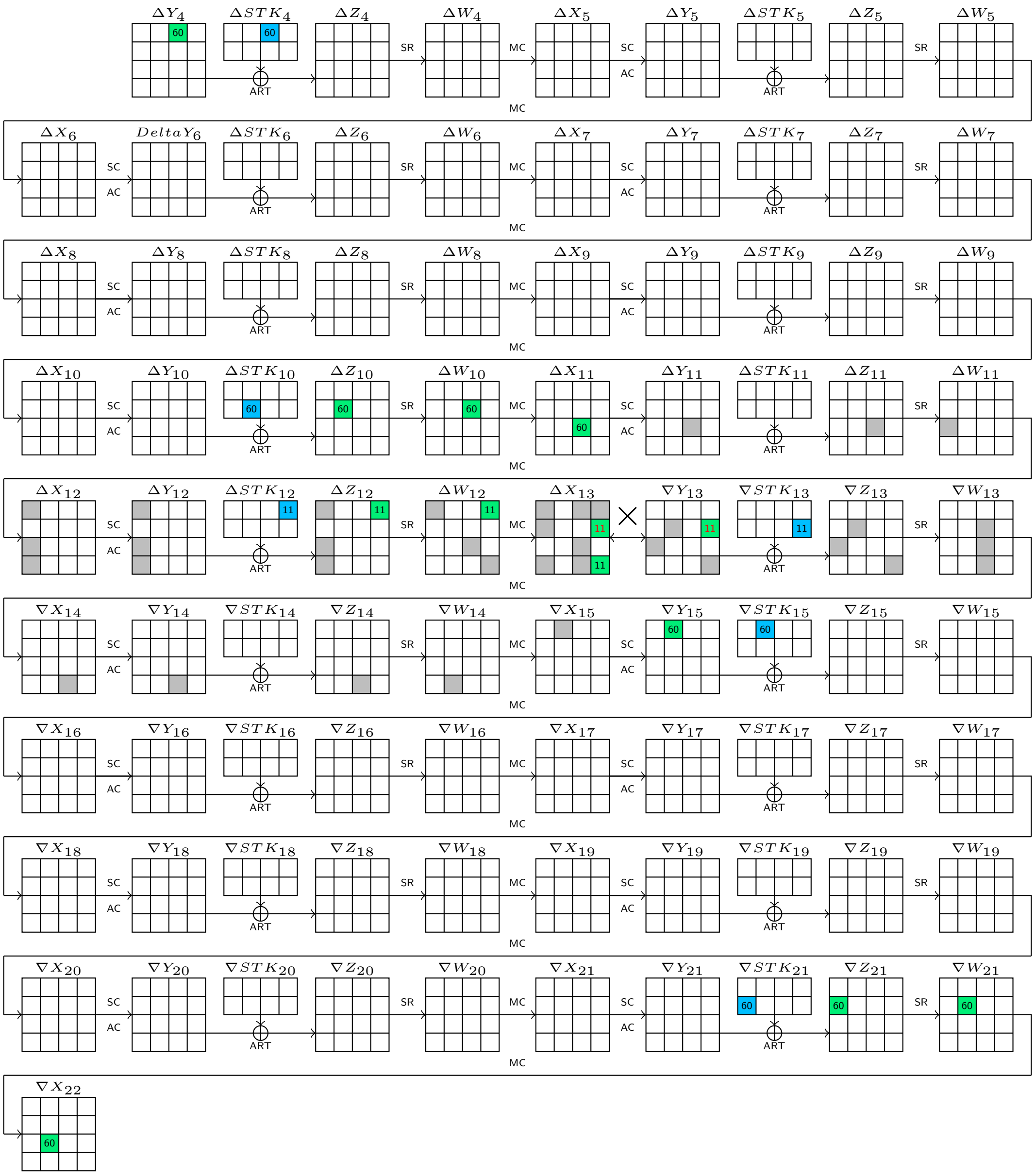


Figure 14: The 18-round related-tweakey impossible boomerang distinguisher for SKINNY-128-384 with BCT effect (Contradiction: 7-th cell in Round 13, $\texttt{BCT}(11, 11) = 0$)

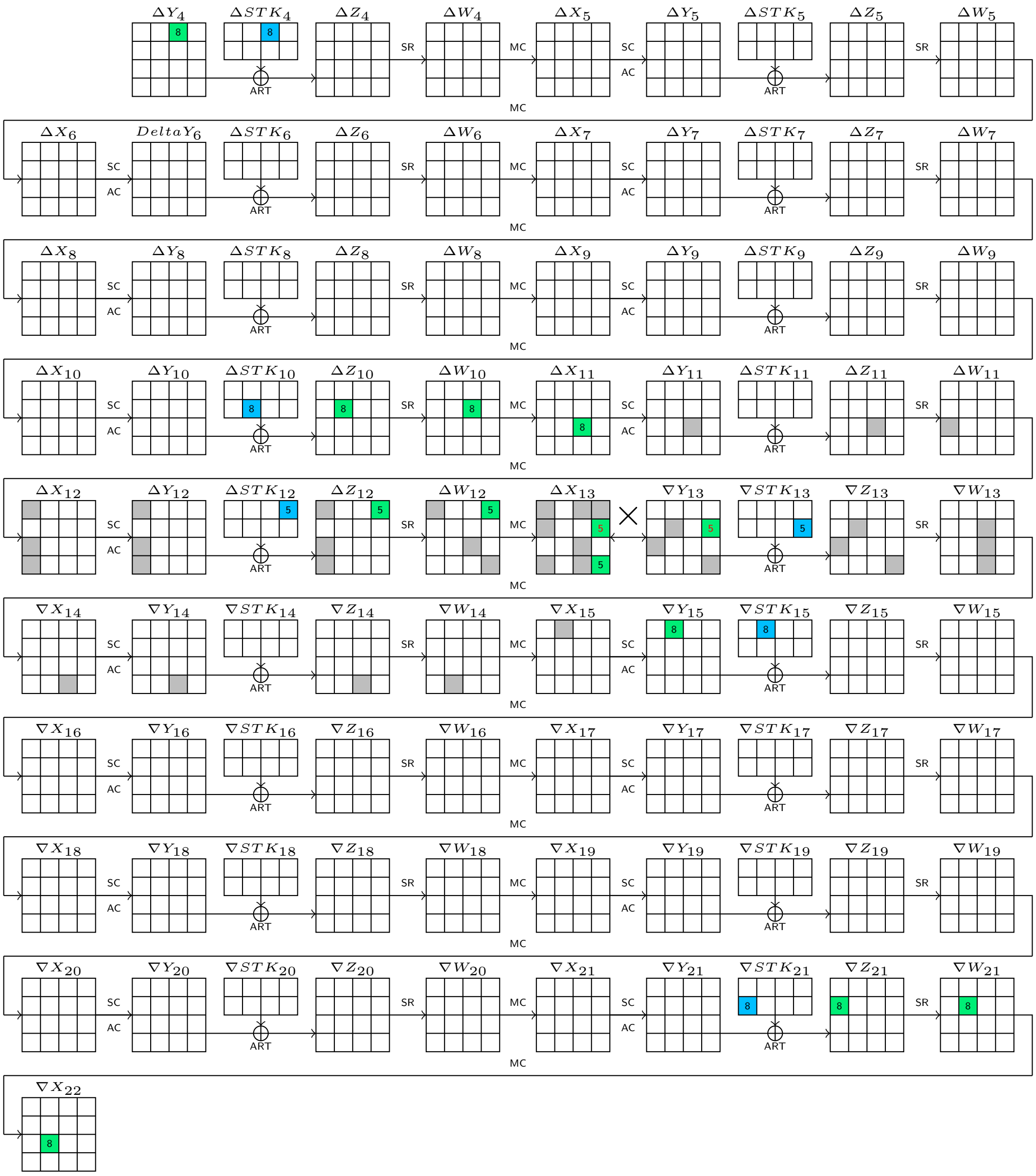


Figure 15: The 18-round related-tweakey impossible boomerang distinguisher for `SKINNY-64-192` with `BCT` effect (Contradiction: 7-th cell in Round 13, $\texttt{BCT}(5, 5) = 0$)

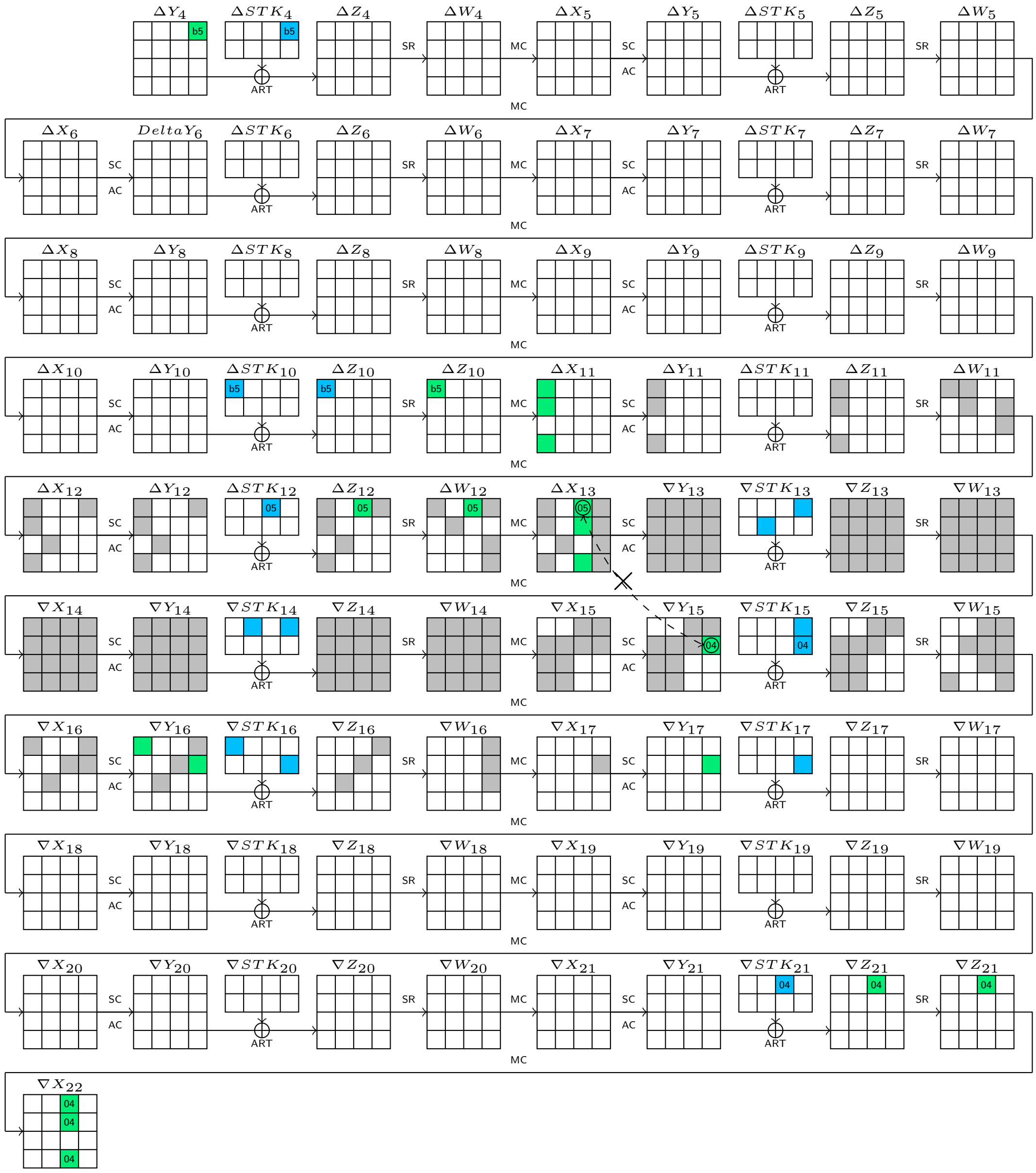


Figure 16: The 18-round related-tweakey impossible boomerang distinguisher for SKINNY-128-384 with DBCT effect

G Subtweakey cells involved in E_b and E_f of the attack against 27-round SKINNY-n-$3n$

Table 10: Subtweakey cells involved in E_b and E_f of the attack against 27-round SKINNY-n-$3n$

	Filter	Involved Subtweakey Cells
E_b	$\Delta W_1[2] = \Delta W_1[10]$	$eSTK_0[2, 8]$
	$\Delta W_1[6] = \Delta W_1[10]$	$eSTK_0[5, 8]$
	$\Delta W_1[4] = \Delta W_1[8]$	$eSTK_0[7, 10]$
	$\Delta Y_2[0]$	$eSTK_0[0, 10, 13], STK_1[0]$
	$\Delta W_2[11] = \Delta W_2[15]$	$eSTK_0[0, 4, 10, 11], STK_1[0, 4]$
	$\Delta W_2[7] = \Delta W_2[11]$	$eSTK_0[2, 4, 11], STK_1[2, 4]$
	$\Delta Y_4[2]$	$eSTK_0[2, 3, 4, 5, 7, 8, 9, 10, 11, 12, 15]$, $STK_1[2, 3, 4, 5, 7], STK_2[2, 3, 7], STK_3[2]$
E_f	$\nabla X_{25}[1] = \nabla X_{25}[13]$	$STK_{25}[1]$
	$\nabla X_{25}[5] \oplus \nabla X_{25}[9] = \nabla X_{25}[13]$	$STK_{25}[5]$
	$\nabla X_{25}[6] = \nabla X_{25}[14]$	$STK_{25}[6]$
	$\nabla X_{25}[7] \oplus \nabla X_{25}[11] = \nabla X_{25}[15]$	$STK_{25}[7]$
	$\nabla X_{25}[3] = \nabla X_{25}[15]$	$STK_{25}[3]$
	$\nabla X_{24}[11]$	$STK_{25}[5]$
	$\nabla X_{24}[9] = \nabla X_{24}[13]$	$STK_{25}[0, 7]$
	$\nabla X_{24}[1] = \nabla X_{24}[13]$	$STK_{25}[0, 5], STK_{24}[1]$
	$\nabla X_{24}[7] = \nabla X_{24}[15]$	$STK_{25}[2, 4], STK_{24}[7]$
	$\nabla X_{24}[3] = \nabla X_{24}[15]$	$STK_{25}[2, 7], STK_{24}[3]$
	$\nabla X_{23}[11] = \nabla X_{23}[15]$	$STK_{25}[0, 1, 6], STK_{24}[2, 5]$
	$\nabla X_{23}[3] = \nabla X_{23}[15]$	$STK_{25}[1, 4, 6], STK_{24}[2, 7], STK_{23}[3]$
	$\nabla X_{22}[9]$	$STK_{25}[1, 3, 5, 6], STK_{24}[2, 4], STK_{23}[7]$
		$STK_{26}[0 - 7]$