Impossible Boomerang Distinguishers Revisited

Xichao Hu[1], Lin Jiao[1], Dengguo Feng[1], Yonglin Hao[1], Xinxin Gong[1], Yongqiang Li[2,3], Siwei Sun[1,4]

[1] State Key Laboratory of Cryptology, Beijing, China
xchao_h@163.com,jiaolin_jl@126.com
[2] Key Laboratory of Cyberspace Security Defense, Institute of Information Engineering, Chinese Academy of Sciences, Beijing, China
[3] School of Cyber Security, University of Chinese Academy of Sciences, Beijing, China
[4] University of Chinese Academy of Sciences, Beijing, China

Abstract. The Impossible Boomerang Attack (IBA) has shown significant power in evaluating the security of block ciphers, such as `AES`. However, current studies still lack foundational theory, user guild and universal method for constructing IBDs. This paper addresses these gaps through comprehensive research. Theoretically, we establish a new framework for constructing a series of IBDs by differential propagation, state propagation, and generalized boomerang tables. We rigorously prove their inclusion relations, resulting in a complete theory and hierarchical apply strategy for both single-key and related-key settings. We further analyze IBD constructions in two types of related-key settings: two-related-keys with arbitrary schedules and four-related-keys with linear schedules, structurally in a unified way. Technically, we develop a scheduling algorithm and a general SAT-based method to search for IBDs across various block cipher designs, including SPN, Feistel, and ARX. Additionally, we propose several strategies to enhance the search process. As applications, we derive (RK-)IBDs for 10 block ciphers, almost for the first time. Compared to impossible differentials, our IBDs are at least as effective, such as `DES` and `PRESENT`. Notably, we achieve 1 more round on `PRINTcipher48` in single-key setting; 2 more rounds on `AES-128`, and 1 or 2 more rounds on `SPECK` variants in two-related-keys settings; 1, 4, 2 more rounds on `GIFT-64`, `CHAM-64/128` and `CHAM-128/256` in four-related-keys settings. We also obtain full-round RK-IBDs on `GOST`. Compared to current IBDs, we achieve 1, 1 more rounds on `SKINNY-64/192` and `SKINNYee`. Furthermore, as an applied case of derived IBDs, we present a 31-round IBA on `SKINNYee`, which is the first 31-round attack on `SKINNYee` and the best result to date.

Keywords: Impossible Boomerang · Single-key · Related-key · Block cipher · SPN · Feistel · ARX.

1 Introduction

The impossible boomerang attack (IBA) is a combination of impossible differential attack [1,2] and boomerang attack [3]. The core of IBA lies in the construction of an Impossible Boomerang Distinguisher (IBD). Once an optimized

IBD is established, it is feasible to extend the analysis forward and backward by a specified number of rounds for key recovery, thereby enabling an effective IBA. Since its proposal, the IBA has played a significant role in analyzing block ciphers, particularly on `AES`. However, compared to its two predecessors, subsequent advancements in the IBA have been limited. To date, there are mainly four key studies on IBD constructions, which all used a miss-in-the-middle approach.

The first two studies constructed IBDs by treating a block cipher E as two sub-ciphers $E^0 \circ E^1$. They used two forward differentials for E^0 and two backward differentials for E^1, all with probability 1, to create a contradiction by ensuring a non-zero XOR of the four output differences. In J. Lu's initial study [4,5], this method has successfully broken 6-round `AES-128`, 7-round `AES-192` and 7-round `AES-256` in single-key setting, as well as 8-round `AES-192` and 9-round `AES-256` in related-key setting, using a 4-round IBD. However, these results were derived manually. To automate the contradiction search, Choy and Yap introduced $\mathcal{UB}$-method [6], which transforms differential propagation into matrix manipulation based on defined criteria. This method was applied to some generalized Feistel ciphers, but only considered the general structure, ignoring component details like S-boxes, linear layers, and key schedules. Additionally, such division of block ciphers is limited because it overlooks the dependence between the two sub-ciphers, as noted by Murphy [7].

The two subsequent studies constructed IBDs based on current research of BDs. Referring to the sandwich attack [8,9], they divides the block cipher E into three parts $E^1 \circ E^m \circ E^0$, where differentials with probability 1 for E^0 and E^1 are still required as well as the difference transition through E^m with probability 0. The contradiction within E^m is derived from boomerang switch constraints based on the Boomerang Connectivity/Difference Tables (BCT or BDT) [10,11]. The $\mathcal{ZWT}$-method [12] constructed Related-Key IBD (RK-IBD) using BCT or Double Boomerang Connectivity Table (DBCT) [13,14] for SPN ciphers, which is modeled by a tool based on MIQCP. It has been applied to three tweakable block ciphers: `Deoxys-BC`, `Joltik-BC` and `SKINNY`. The $\mathcal{BCL}$-method [15] constructed RK-IBDs for Feistel ciphers with quadratic round functions using Feistel Boomerang Connectivity Table (FBCT) [16] searched by an SMT solver, as well as (RK-)IBDs for SPN ciphers based on BCT or DBCT. As an application, it presented a 23-round IBA on `Simon-32/64` based on a 7-round RK-IBD and a 29-round IBA on `SKINNYee` based on an actually 22-round RK-IBD[5]. However, constructing and modeling DBCT becomes challenging if the linear layer of the block cipher is not byte- or nibble-based and sparse. Besides, $\mathcal{ZWT}$-method and $\mathcal{BCL}$-method for S-box based block ciphers detect the positions of S-boxes with known and input-output differences with some flags and their propagation rule, then check for contradictions at these positions according to BCT, rather than adding the information of BCT to the search process. Therefore, even though they use BCT or DBCT, they may still overlook some details, which will affect the construction of distinguishers covering more rounds.

[5] The original result in [15] presented a 21-round IBD, while it can be extended by one additional round without accounting for the first SC operation.

Since the security of a block cipher against IBA can be evaluated by its IBDs, constructing IBDs is crucial. In light of current research, the solutions to the following three fundamental problems are still missing.

Question 1. *Are existing construction methods for IBDs equivalent to the essential definition? Within the same difference search space, is there a better (RK-)IBD?*

We observe that, existing methods impose certain limitations on the form and scope of contradictions. Firstly, they divide the stages of block ciphers, thereby restricting the positions of contradictions. If the contradictions span a significant number of continuous rounds, these methods appear to be insufficient for achieving such constructions. For example, current methods cannot reveal the contradiction that exists across six consecutive rounds of S-boxes for SKINNY-ee, which further improve the RK-IBD base on one-round contradiction in [16]. Secondly, they all require differentials with probability 1 for E^0 and E^1. For most block ciphers, such differentials are impractical for covering many rounds, limiting the development of longer IBDs. Overall, there is a lack of a unified construction method that can both accurately capture the essential definition of IBDs and be effectively transformed into a search model. Additionally, foundational theory is missing to evaluate relationships between different construction methods, hindering definitive assessments of block ciphers' resistance to IBA.

Question 2. *Can existing construction methods of IBDs fully exploit the details of block cipher and generalize to all current structures?*

Firstly, we examine the key schedule. In single-key setting, the key schedule cannot be exploited because existing methods construct IBDs based on differential propagation, which inherently counteracts the impact of key schedule, making it ineffective. In related-key setting, these methods can only construct RK-IBDs for block ciphers with linear key schedules under specific difference pattern of (μ, μ, ν, ν). Additionally, there is no discussion on automatic search methods of IBDs in block ciphers with nonlinear key schedules, despite the potential for further exploration. Secondly, we examine the details of cipher components. The $\mathcal{UB}$-method fails to account for the specifics of S-boxes and linear layers. The $\mathcal{ZWT}$-method and $\mathcal{BCL}$-method are limited to SPN ciphers with byte/nibble-based sparse matrices and Feistel ciphers with quadratic round functions. They do not apply to SPN ciphers with MDS matrices (such as `AES`) and bit permutations (such as `PRESENT`), or Feistel ciphers with non-quadratic round functions (such as `DES`). Furthermore, ARX ciphers also fall outside their scope of applicability. In summary, there is a lack of a universal method for constructing IBDs that can effectively generate contradictions by fully utilizing cipher details.

Question 3. *For specific block ciphers with distinct characteristics, how can we select the IBD construction method to effectively obtain IBDs? How can we optimize the application of each IBD search method to achieve the best evaluation results given the current computational resources?*

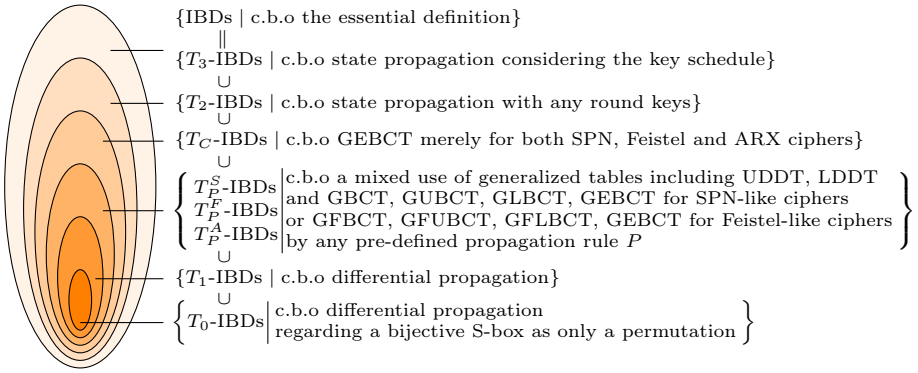

Fig. 1: The inclusion relations between the newly-defined IBDs
†Here c.b.o is the abbreviation of "constructed based on". T_2-IBDs apply only to single-key setting. T_0-IBDs apply only to bijective S-boxes.

Relying solely on differential-based search methods for IBDs is insufficient, while directly forming search models based on various Boomerang tables (BT) introduces considerable complication. Currently, there is no hierarchical apply strategy for applying different IBD construction methods to specific block ciphers under varying evaluation criteria.

Our contribution. Motivated by the strong threat posed by IBA, as well as the lack of solutions to these fundamental questions, we initiate comprehensive research on constructing IBDs.

Firstly, we establish a complete theory for constructing IBDs (introduced in Section 3). Drawing upon current analysis methods for impossible differential attacks and boomerang attacks, we propose a series of construction methods of (RK-)IBDs, encompassing differential propagation, state propagation, and generalized BT perspectives. Specifically, we first introduce an IBD construction method based on state propagation that fully exploits design details of various ciphers like SPN, Feistel, and ARX without assumptions and cipher divisions. We generalize all BTs into IBD construction framework. Furthermore, we prove the inclusion relationships among these IBD constructions, leading to a tight conclusion that aligns with the essential definition of IBD, as illustrated in Fig 1. Additionally, for Feistel-like and SPN-like ciphers with full-(branch)-state round-key-additions in single-key setting, GEBCT-based IBDs are equivalent to state-propagation-based IBDs under any-key-assumption.

Based on the theoretical conclusion, we propose a hierarchical apply strategy for IBD constructions when choosing the appropriate method.

- Use T_0-IBD for a rough estimation (lower bound) and T_3-IBD for a precise evaluation (upper bound) of the number of rounds of IBDs.
- If solving time permits, prioritize constructing T_3-IBDs. When encountering efficiency bottlenecks of solvers for searching for T_3-IBDs, use T_C-IBDs,

T_P-IBDs, T_2-IBDs, T_1-IBDs, T_0-IBDs and hybrid techniques as effective supplements for round estimation of IBDs.
– To select an appropriate search type of IBD from the diverse set of IBD construction methods with varying precisions, we develop a scheduling algorithm that leverages multi-process technology in computer engineering to achieve an optimal balance between search capability and accuracy.

Secondly, we extend the analysis of IBD in related-key settings (introduced in Section 4). In addition to current RK-IBDs, we focus on constructions with two types of key schedules in a unified way, defined as RT_j^i-IBDs for $j = 0, 1, 3, P, C$ in i-related-keys setting for $i = 2, 4$ as follows:

- Two-related-keys setting: RK-IBDs under the key difference of $(\kappa, \kappa, 0, 0)$ (applicable to block cipher with any key schedule, especially the nonlinear one).
- Four-related-keys setting: RK-IBDs under the key difference of $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ (applicable to block cipher with linear key schedules, generalizing current specific key difference pattern to a broader context).

By carefully offsetting round key differences and state differences in initial/final rounds of RK-IBD, we derive the required key and input/output difference patterns for central rounds. This enables efficient searching for potential central RK-IBD and structurally extend it by probability-1 related-key differentials. For block ciphers with non-linear key schedules, the default of lower trail key differences in two-related-keys setting allow extension at the beginning of the RK-IBD; whereas for block ciphers with linear key schedules, the definitive round key differences enable extension at both ends.

Thirdly, we develop a general SAT-based automatic method based on our newly established theoretical framework (introduced in Section 5), to search for T_j-IBDs for $j = 0, 1, 2, 3, P, C$ and RT_j^i-IBDs for $i = 2, 4$ and $j = 0, 1, 3, P, C$. Aligned with our scheduling algorithm according to the hierarchical apply strategy, it forms a universal model for various block cipher designs. This model incorporates IBD construction methods including differential propagation, state propagation, and generalized BT. Based on this model, we can obtain the optimal IBD result for the currently executable search in the shortest time. Meanwhile, the bottleneck issue in directly constructing models for generalized BTs with a large number of variables has been successfully addressed. Additionally, we propose several search and verify strategies. Especially, we remove the current constraint of probability-1 differentials for E^0 and E^1.

Under our complete construct theory, hierarchical apply strategy and general search method, we have addressed the fundamental questions of IBD construction. We apply this theoretical framework to various block ciphers (introduced in Section 6), including SPN, Feistel network and ARX designs. The selections cover the common classifications of block ciphers and serve to verify the effectiveness of our approach. In single-key setting, we analyze `AES` [17] (SPN with large S-boxes and an MDS matrix), `DES` [23] (Feistel with non-bijective S-boxes, no limit on round function), `PRESENT-80` [24] (lightweight with bit permutation),

Table 1: A Summary of the results for IBDs.

Cipher	Type	Round	Number	Time(hours)	Method	Note
Single-key setting						
AES	ID	4	many	-	[17]	
	ID	5	none	-	[18]	
	IBD	4	less	-	Manual[4]	
	1 ABT T_0-IBD	**4**	**61440**	**149.04**	**Section 6.1**	
	1 ABT T_0-IBD	5	none	203.44	Section 6.1	
DES	1 Ab ID	7	394	0.57	ST[a]	
	1 Ab ID	8	none	0.91	ST[a]	
	1 Ab T_1-IBD	**7**	**1904**	**327.64**	**Section 6.1**	**1-st IBD**
	1 Ab T_1-IBD	8	none	372.38	Section 6.1	
PRESENT-80	ID	6	many	-	[19]	
	1 AN ID	7	none	-	[20]	
	1 AN T_2-IBD	**6**	**58**	**7.13**	**Section 6.1**	**1-st IBD**
	1 AN T_3-IBD	7	none	24.52	Section 6.1	
PRINTcipher48	1 Ab ID	4	many	-	[20]	
	1 Ab ID	5	none	-	[20]	
	1 Ab T_3-IBD	**5**	**2**	**14.75**	**Section 6.1**	**1-st IBD**
	1 Ab T_3-IBD	6	none	40.07	Section 6.1	**1-round>ID**
Two-related-keys setting						
AES-128	1 ABT RK-ID	3	64	0.39	ST[a]	
	1 ABT RK-ID	4	none	0.52	ST[a]	
	1 ABT RT_0^2-IBD	**5**	**768**	**14.44**	**Section 6.2**	**1-st IBD**
	1 ABT RT_0^2-IBD	6	none	18.68	Section 6.2	**2-round>ID**
SPECK-2w/4w	RK-ID	7	many	-	[21]	w=16,24,32,64
	RK-ID	8	none	-	[21]	
SPECK-2w/3w	RK-ID	6	many	-	[21]	w=24,32,48,64
	RK-ID	7	none	-	[21]	
SPECK-32/64	**RT_3^2-IBD**	**8**	**377**	**0.18**	**Section 6.2**	
	RT_3^2-IBD	9	none	0.97	Section 6.2	
SPECK-48/72	**RT_3^2-IBD**	**7**	**6**	**0.06**	**Section 6.2**	
	RT_3^2-IBD	8	none	0.26	Section 6.2	
SPECK-48/96	**RT_3^2-IBD**	**8**	**6**	**0.09**	**Section 6.2**	
	RT_3^2-IBD	9	none	0.60	Section 6.2	
SPECK-64/96	**RT_3^2-IBD**	**8**	**4**	**0.29**	**Section 6.2**	**1-st IBD**
	RT_3^2-IBD	9	none	0.60	Section 6.2	**1-/2-round>ID**
SPECK-64/128	**RT_3^2-IBD**	**9**	**4**	**0.28**	**Section 6.2**	
	RT_3^2-IBD	10	none	0.99	Section 6.2	
SPECK-96/144	**RT_3^2-IBD**	**8**	**4**	**0.22**	**Section 6.2**	
	RT_3^2-IBD	9	none	0.65	Section 6.2	
SPECK-128/192	**RT_3^2-IBD**	**8**	**4**	**0.33**	**Section 6.2**	
	RT_3^2-IBD	9	none	1.18	Section 6.2	
SPECK-128/256	**RT_3^2-IBD**	**9**	**4**	**0.41**	**Section 6.2**	
	RT_3^2-IBD	10	none	1.78	Section 6.2	
Four-related-keys setting						
SKINNY-64/192	RK-ID	17	1	-	[22]	
	RK-IBD	18	1	-	$\mathcal{ZWT}$[12]	
	RT_3^4-IBD	**19**	**3**	**136.12**	**Section 6.2**	**2-round>ID**
	RT_3^4-IBD	20	none	267.21	Section 6.2	**1-round>IBD**
SKINNYee	RK-IBD	22	103	-	$\mathcal{BCL}$[15]	
	RT_3^4-IBD	**23**	**5**	**209.78**	**Section 6.2**	**1-round>RK-IBD**
	RT_3^4-IBD	24	none	488.89	Section 6.2	
GIFT-64	1 AN RK-ID	12	48	-	[21]	
	1 AN RK-ID	13-16	none	-	[21]	
	1 AN RT_3^4-IBD	**13**	**48**	**0.51**	**Section 6.2**	**1-st IBD**
	1 AN RT_3^4-IBD	14	none	1.91	Section 6.2	**1-round>ID**

continued on next page

continued from previous page

Four-related-keys setting

Cipher	Type	Round	Number	Time(hours)	Method	Note
CHAM-64/128	RK-ID	26	many	-	[21]	
	RK-ID	27	none	-	[21]	
	$\boldsymbol{RT_3^4}$**-IBD**	**30**	**3**	**0.15**	**Section 6.2**	**1-st IBD**
	RT_3^4-IBD	31-32	none	0.22	Section 6.2	**4-round>ID**
CHAM-128/256	RK-ID	26	many	-	[21]	
	RK-ID	27	none	-	[21]	
	$\boldsymbol{RT_3^4}$**-IBD**	**28**	**4**	**0.48**	**Section 6.2**	**1-st IBD**
	RT_3^4-IBD	29-30	none	0.63	Section 6.2	**2-round>ID**
GOST-FB/PS	$\boldsymbol{RT_3^4}$**-IBD**	**full-round**	**2**	**0.08**	**Section 6.2**	**1-st IBD**

†The method marked with ^{a} represents our implementation rather than reference. ABT: active byte truncated, Ab: active bit, AN: active nibble. All (RK-)IBDs are confined to a search space by meticulous selection. 1-st IBD represents it is the first IBD result of this cipher. r-round>ID (resp. IBD) represents our (RK-)IBDs cover r round more than current (RK-)IDs (resp. (RK-)IBDs).

and PRINTcipher48 [25] (SPN with key-dependent permutation). In two-related-keys setting, we analyze AES-128, and SPECK [26] (ARX). In four-related-keys setting, we analyze DES, SKINNY-64/192 [27] and SKINNYee [28] (tweakable block ciphers), GIFT [29] (lightweight with bit permutation), CHAM [30] (ARX), and GOST [31] (Feistel). The results are presented in Table 1. To illustrate the efficacy of the obtained IBDs, we utilize SKINNYee as a case study and provide its corresponding 31-round IBA, which is the first 31-round attack on SKINNYee and the best result to date. The findings indicate that our method outperforms the current methods for IBDs. Moreover, specific results suggest that IBDs offer an advantage over IDs. Given the importance of ID attacks as a fundamental analysis technique, it is crucial to seriously consider and employ IBA.

2 Preliminaries

The primary notations used hereafter are detailed as follows.

- Let k denote the master key and rk_i the i-th round key, where $rk_i = \mathrm{KS}_i(k)$ is generated by the key schedule KS.
- Let $E_k^{(r)}(x)$ denote an r-round block cipher that encrypts input $x \in \mathbb{F}_2^n$ with master key $k \in \mathbb{F}_2^m$ to produce output $y = E_k^{(r)}(x) \in \mathbb{F}_2^n$.
- Let E_{i,rk_i} denote the i-th round of $E_k^{(r)}$ with round key rk_i. Then $E_k^{(r)}(x) = E_{r-1,rk_{r-1}} \circ \cdots \circ E_{0,rk_0}(x)$. In clear contexts, $E_k^{(r)}$ is abbreviated as E or E_k, and E_{i,rk_i} is abbreviated as E_i.
- Let S, SL, LL, and AK denote an S-box, a parallel S-boxes operation, a linear operation and a round-key-XOR operation, respectively.

We review the basic definitions of differential analysis. For a function $f\colon \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, the probability that input difference α propagates to output difference β under key difference κ is given by $P_{f,\kappa}(\alpha,\beta) = \#\{(k,x) \in \mathbb{F}_2^m \times \mathbb{F}_2^n \mid f(k,x) \oplus f(k \oplus \kappa, x \oplus \alpha) = \beta\}/2^{n+m}$. If $P_{f,\kappa}(\alpha,\beta) \neq 0$, it is denoted

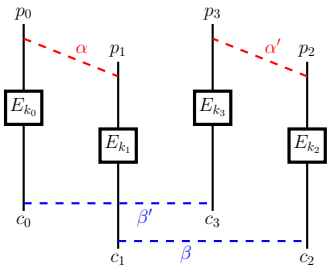

Fig. 2: The illustration of RK-IBD

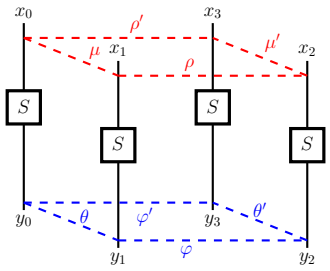

Fig. 3: Generalized BTs

as $\alpha \overset{f,\kappa}{\to} \beta$. Define $\mathrm{DP}_{f,\kappa}(\alpha) = \{\beta | \alpha \overset{f,\kappa}{\to} \beta\}$. Particularly, in single-key setting, $P_f(\alpha,\beta) = \#\{x \in \mathbb{F}_2^n \mid f(x) \oplus f(x \oplus \alpha) = \beta\}/2^n$, $\mathrm{DP}_f(\alpha) = \{\beta | \alpha \overset{f}{\to} \beta\}$. Additionally, for a composite function $f : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, where $f = f_{r-1} \circ \cdots \circ f_1 \circ f_0$, an r-round related-key differential characteristic is defined as a series of differences $\Omega = (\alpha_0, \ldots, \alpha_r)$ under the key difference $\Gamma = (\kappa_0, \ldots, \kappa_r)$, where $\alpha_i \overset{f_i,\kappa_i}{\to} \alpha_{i+1}$ for $0 \le i \le r-1$, and the probability of Ω is given by $P_{f,\Gamma}(\Omega) = \prod_{i=0}^{r-1} P_{f_i,\kappa_i}(\alpha_i, \alpha_{i+1})$. The probability of differential defined by (α_0, α_r) is given by $P_{f,\Gamma}(\alpha_0, \alpha_r) = \sum_{\alpha_1,\ldots,\alpha_{r-1}} P_{f,\Gamma}(\Omega)$. In single-key setting, Γ is omit. Given two differences $\gamma \in \mathbb{F}_2^n, \theta \in \mathbb{F}_2^m$, the DDT for an $n \times m$-bit function is defined as $\mathrm{DDT}(\gamma,\theta) = \#\{x \in \mathbb{F}_2^n \mid f(x) \oplus f(x \oplus \gamma) = \theta\}$.

Next, we review the definitions of BTs.

Definition 1 ([10]). *Given differences $\gamma, \theta, \delta \in \mathbb{F}_2^n$, the BCT for an n-bit S-box is defined as* $\mathrm{BCT}(\gamma,\delta) = \#\left\{x \in \mathbb{F}_2^n \mid S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma\right\}$.

Let $E = E^1 \circ E^m \circ E^0$ be an r-round block cipher with $r = r_0 + r_1 + 1$, where E^0, E^m and E^1 denote the initial r_0 rounds, middle 1 round and final r_1 rounds of E respectively. Suppose $\alpha \overset{E^0}{\to} \gamma$ and $\beta \overset{(E^1)^{-1}}{\to} \delta$, then

$$\Pr(E^{-1}(E(x) \oplus \beta) \oplus E^{-1}(E(x \oplus \alpha) \oplus \beta) = \alpha) = (P_{E^0}(\alpha,\gamma))^2 (P_{E^1}(\delta,\beta))^2 P_m,$$

where $P_m = \prod_{i=0}^{t} (\mathrm{BCT}(\gamma_i,\delta_i)/2^n)$ for SPN E, assuming that there are t n-bit S-boxes in E^m with the input difference γ_i and output difference δ_i. To apply boomerang switch in multiple rounds, more BTs have been proposed. Especially, a method for establishing BDs using mixed tables was proposed in [32].

Definition 2 ([11,13,14,32]). *Given four differences $\gamma, \theta, \lambda, \delta \in \mathbb{F}_2^n$, the UBCT, LBCT and EBCT for an n-bit S-box are defined as*

$$\mathrm{UBCT}(\gamma,\theta,\delta) = \#\left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x \oplus \gamma) = \theta, \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\},$$

$$\mathrm{LBCT}(\gamma,\lambda,\delta) = \#\left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x \oplus \lambda) = \delta, \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\},$$

$$\mathrm{EBCT}(\gamma,\theta,\lambda,\delta) = \#\left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x \oplus \gamma) = \theta, S(x) \oplus S(x \oplus \lambda) = \delta, \\ S^{-1}(S(x) \oplus \delta) \oplus S^{-1}(S(x \oplus \gamma) \oplus \delta) = \gamma \end{array} \right\},$$

and the DBCT is defined as $\mathrm{DBCT}(\gamma,\delta)=\sum_{\theta,\lambda} UBCT(\gamma,\theta,\lambda)\cdot LBCT(\theta,\lambda,\delta)$.

In addition to the BCTs for SPN ciphers, BTs are also defined for Feistel and ARX ciphers with corresponding properties. We present these tables in Appendix A. The essential definition of (RK-)IBD is defined as follows, as shown in Fig 2.

Definition 3 ([4]). *Given a block cipher* $E:\mathbb{F}_2^n\times\mathbb{F}_2^m\to\mathbb{F}_2^n$ *with four related master keys* $k_i\in\mathbb{F}_2^m$*, if for four differences* $\alpha,\alpha',\beta,\beta'$*, any pair of plaintexts* (x_1,x_2) *cannot satisfy* $E_{k_1}(x_1)\oplus E_{k_2}(x_2)=\beta, E_{k_0}(x_1\oplus\alpha)\oplus E_{k_3}(x_2\oplus\alpha')=\beta'$ *simultaneously, then* $(\alpha,\alpha',\beta,\beta')$ *is called an RK-IBD for* E*. Particularly, if* $k=k_0=k_1=k_2=k_3$*,* $(\alpha,\alpha',\beta,\beta')$ *is called an IBD of* E *in single-key setting, denoted by* $(\alpha,\alpha')\nrightarrow(\beta,\beta')$.

Given an IBD or RK-IBD, an attacker can extend the number of rounds before and after the distinguisher to launch a key recovery attack, known as IBA.

3 New Construct Theory for IBDs in Single-Key Setting

In this section, we develop a new theory for constructing IBDs by from the aspects of differential and state propagation, as well as generalized BTs for SP-N, Feistel and ARX ciphers. We also prove the interrelationships among these construction methods. Proofs of theorems and propositions are provided in Appendix B.

3.1 Constructing IBDs from the aspect of differential propagation

We present two IBD-definitions based on differential propagation. Firstly, we present two boomerang trails based on $\mathrm{DP}_f(\alpha)$ and its relaxed variant.

Definition 4. *Given an* r*-round block cipher* $E=E^1\circ E^0$*, for two input differences* α,α' *and two output differences* β,β'*,*

- *if there exist* $\gamma\in\overline{\mathrm{DP}}_{E^0}(\alpha)$*,* $\gamma'\in\overline{\mathrm{DP}}_{E^0}(\alpha')$*,* $\delta\in\overline{\mathrm{DP}}_{(E^1)^{-1}}(\beta)$*, and* $\delta'\in\overline{\mathrm{DP}}_{(E^1)^{-1}}(\ \beta')$*, such that* $\gamma\oplus\gamma'\oplus\delta\oplus\delta'=0$*, then*

$$(\alpha,\alpha')\to\cdots\to\underbrace{(\gamma,\gamma')(\delta,\delta')}_{\gamma\oplus\gamma'\oplus\delta\oplus\delta'=0}\to\cdots\to(\beta,\beta') \tag{1}$$

  is called an r*-round* T_0 *boomerang trail. Here,* $\overline{\mathrm{DP}}_f(\alpha)$ *is a relaxed variant of* $\mathrm{DP}_f(\alpha)$ *by considering all details of operations of* f *except S-boxes.*
- *if there exist* $\gamma\in\mathrm{DP}_{E^0}(\alpha)$*,* $\gamma'\in\mathrm{DP}_{E^0}(\alpha')$*,* $\delta\in\mathrm{DP}_{(E^1)^{-1}}(\beta)$*, and* $\delta'\in\mathrm{DP}_{(E^1)^{-1}}(\ \beta')$*, such that* $\gamma\oplus\gamma'\oplus\delta\oplus\delta'=0$*, then Trail 1 is called an* r*-round* T_1 *boomerang trail.*

Accordingly, we present the following two IBD-construction methods: T_0-IBD and T_1-IBD. T_0-IBD is newly proposed, while T_1-IBD generalizes the method from [4]. We also prove their inclusion relationship.

Construction 1 (T_0-IBD). *Given an r-round block cipher E, for two input differences α, α' and two output differences β, β', if no r-round T_0 boomerang trail exists, then $(\alpha, \alpha', \beta, \beta')$ is an IBD, called an r-round T_0-IBD.*

Construction 2 (T_1-IBD). *Given an r-round block cipher E, for two input differences α, α' and two output differences β, β', if no r-round T_1 boomerang trail exists, then $(\alpha, \alpha', \beta, \beta')$ is an IBD, called an r-round T_1-IBD.*

Theorem 1. *An r-round T_0-IBD $(\alpha, \alpha', \beta, \beta')$ is an r-round T_1-IBD.*

T_0-IBD is suitable for block ciphers with bijective S-boxes, treating them as permutations. T_0-IBD takes the advantage of efficiently searching and assessing a lower bound on the number of rounds of IBDs. T_1-IBD offers broader applicability, within which the differential propagation rule of each component can be characterized. The examples of T_0-IBD and T_1-IBD are provided in Section 6.1, e.g. the applications to `AES` and `DES` respectively.

3.2 Constructing IBDs from the aspect of state propagation

Inspired by the concept proposed in [20], which uses the propagation of two states to construct IDs, we first extend this idea to construct IBDs by the propagation of four states. This method adapts to any block cipher and considers all details of components, as well as independent keys and key relations in single-key setting.

Definition 5. *Let $E = E_{r-1,rk_{r-1}} \circ \cdots \circ E_{0,rk_0}$ be an r-round block cipher with l-bit round keys. Given four differences $\alpha, \alpha', \beta, \beta'$, let $I = \{(x_0, x_1, x_2, x_3) \mid x_0 \oplus x_1 = \alpha, x_2 \oplus x_3 = \alpha'\}$ and $O = \{(y_0, y_1, y_2, y_3) \mid y_1 \oplus y_2 = \beta, y_0 \oplus y_3 = \beta'\}$. If there exist $(x_0^0, x_1^0, x_2^0, x_3^0) \in I$, $(x_0^r, x_1^r, x_2^r, x_3^r) \in O$ and $(rk_0, \cdots, rk_{r-1}) \in (\mathbb{F}_2^l)^r$, such that $x_j^{i+1} = E_{i,rk_i}(x_j^i)$ for $0 \le i \le r-1, 0 \le j \le 4$, then $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$ is called an r-round T_2 boomerang trail.*

Definition 6. *Let $E = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ be an r-round block cipher with the key schedule* KS. *Given four differences $\alpha, \alpha', \beta, \beta'$, let $I = \{(x_0, x_1, x_2, x_3) \mid x_0 \oplus x_1 = \alpha, x_2 \oplus x_3 = \alpha'\}$ and $O = \{(y_0, y_1, y_2, y_3) \mid y_1 \oplus y_2 = \beta, y_0 \oplus y_3 = \beta'\}$. If there exist $(x_0^0, x_1^0, x_2^0, x_3^0) \in I$, $(x_0^r, x_1^r, x_2^r, x_3^r) \in O$ and an master key k such that $x_j^{i+1} = E_{i,\mathrm{KS}_i(k)}(x_j^i)$ for $0 \le i \le r-1, 0 \le j \le 4$, then $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$ is called an r-round T_3 boomerang trail.*

These definitions enable us to assess the impact of round keys on IBD construction for the first time.

Construction 3 (T_2-IBD). *Given an r-round block cipher E, for two input differences α, α' and two output differences β, β', if no r-round T_2 boomerang trail exists, then $(\alpha, \alpha', \beta, \beta')$ is an IBD, called an r-round T_2-IBD.*

Construction 4 (T_3-IBD). *Given an r-round block cipher E, for two input differences α, α' and two output differences β, β', if no r-round T_3 boomerang trail exists, then $(\alpha, \alpha', \beta, \beta')$ is an IBD, called an r-round T_3-IBD.*

The inclusion relationships between T_2-IBD and T_3-IBD, as well as between T_1-IBD and T_2-IBD, are directly derived from their definitions. The examples of T_2-IBD and T_3-IBD are provided in Section 6.1, e.g. the applications to `PRESENT` and `PRINTcipher` respectively. However, the definitions of T_1-IBD and T_2-IBD are not equivalent; for example, the T_2-IBD of `PRESENT` presented in Section 6.1 is not a T_1-IBD.

Theorem 2. *An r-round T_2-IBD $(\alpha, \alpha', \beta, \beta')$ is an r-round T_3-IBD. An r-round T_1-IBD $(\alpha, \alpha', \beta, \beta')$ is an r-round T_2-IBD.*

Next, we prove that Construction 4 is the tightest method for constructing IBDs.

Theorem 3. *T_3-IBD is equivalent to the essential definition of IBDs given in Definition 3.*

3.3 Constructing IBDs from the aspect of generalized BTs

BTs are extensively utilized in BD constructions for SPN and Feistel ciphers, with limited application in ARX ciphers. Theoretically, it is logical to extend BD to IBD and construct IBDs based on generalized BTs. In this section, we discuss constructing IBDs based on generalized BTs for SPN, Feistel and ARX ciphers.

For SPN ciphers, the original boomerang attack assumes independence between two sub-ciphers E^0 and E^1. However, this assumption may not hold for selected differential characteristics, as shown in [7]. An r-round T_1 boomerang trail derived in this manner could actually be an IBD, potentially overlooked under this assumption. This issue is addressed by GBCT [33]. In this paper, we present a slightly different definition.

Definition 7. *Given four differences $\mu, \mu', \varphi, \varphi' \in \mathbb{F}_2^n$, the GBCT for an n-bit S-box is defined as* $\mathrm{GBCT}(\mu, \mu', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi')$, *where* $\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.

Furthermore, as Song et al. [34] observed, dependence can significantly influence multiple rounds, such as up to 6 rounds for `SKINNY`. While BCT cannot eliminate the incompatibility over multiple rounds, various tables like UBCT, LBCT, and EBCT have been defined. We generalize these concepts to IBD[6].

Definition 8. *Given eight differences $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi' \in \mathbb{F}_2^n$ where $\rho' = \mu \oplus \mu' \oplus \rho$, $\varphi' = \theta \oplus \theta' \oplus \varphi$, the GUBCT, GLBCT and GEBCT for an n-bit S-box are defined as*
$\mathrm{GUBCT}(\mu, \mu', \theta, \theta', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 =$

[6] Concurrent Work: [12] proposed similar definitions but did not develop search methods for IBDs using these tables. In contrast, our work provide a complete theoretical analysis and modeling method.

$\mu', S(x_0)\oplus S(x_1) = \theta, S(x_2)\oplus S(x_3) = \theta', S(x_1)\oplus S(x_2) = \varphi, S(x_0)\oplus S(x_3) = \varphi'\}$,
$\mathrm{GLBCT}(\mu, \mu', \rho, \rho', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$,
$\mathrm{GEBCT}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_0) \oplus S(x_1) = \theta, S(x_2) \oplus S(x_3) = \theta', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.

The above tables are supplemented with two additional notations for clarity:
$\mathrm{UDDT}(\mu, \mu', \theta, \theta') = \#\,\mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \theta, \theta')$, where
$\mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \theta, \theta') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', S(x_0) \oplus S(x_1) = \theta, S(x_2) \oplus S(x_3) = \theta'\}$,
$\mathrm{LDDT}(\rho, \rho', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{LDDT}}(\rho, \rho', \varphi, \varphi')$, where
$\mathrm{IN}_{\mathrm{LDDT}}(\rho, \rho', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.
A schematic diagram for these generalized BTs is shown in Fig 3.

Proposition 1. $\mathrm{UDDT}, \mathrm{LDDT}, \mathrm{GBCT}, \mathrm{GUBCT}, \mathrm{GLBCT}, \mathrm{GEBCT}$ *have the following relations:*

1. $\exists \eta, \eta'$, *s.t.* $\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \eta, \eta')$,
2. $\exists \omega, \omega'$, *s.t.* $\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{LDDT}}(\omega, \omega', \varphi, \varphi')$,
3. $\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \theta, \theta')$,
4. $\exists \omega, \omega'$, $\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{LDDT}}(\omega, \omega', \varphi, \varphi')$,
5. $\exists \eta, \eta'$, *s.t.* $\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \eta, \eta')$,
6. $\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{LDDT}}(\rho, \rho', \varphi, \varphi')$,
7. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{UDDT}}(\mu, \mu', \theta, \theta')$,
8. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{LDDT}}(\rho, \rho', \varphi, \varphi')$,
9. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi')$,
10. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi')$,
11. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi')$.

We first propose an approach that makes a mixed use of DDT, GBCT, GUBCT, GLBCT, and GEBCT to construct IBDs in a manner similar to optimizing BDs. At first, we define a set of propagation rules for an SPN cipher E with t S-boxes $(S_0, \ldots, S_{t-1})$ in total as

$$\mathcal{AP}_E^S = \{(p_0, \ldots, p_{t-1}) | p_i \in \{\mathrm{UDDT}, \mathrm{LDDT}, \mathrm{GBCT}, \mathrm{GUBCT}, \mathrm{GLBCT}, \mathrm{GEBCT}\}\}.$$

Then $P = (p_0, \ldots, p_{t-1}) \in \mathcal{AP}_E^S$ denotes that the propagation rule through the i-th S-box follows p_i.

Definition 9. *Let $E = E_{r-1} \circ \cdots \circ E_0$ be an r-round SPN cipher. Let $P = (P_0, \ldots, P_{r-1})$ be a predefined propagation rule of E, where $P_i \in \mathcal{AP}_{E_i}^S$ denotes a propagation rule of E_i for $i \in \{0, \ldots, r-1\}$. Let $\epsilon_0^i, \epsilon_1^i, \epsilon_2^i, \epsilon_3^i$ be the four input differences and $\epsilon_0^{i+1}, \epsilon_1^{i+1}, \epsilon_2^{i+1}, \epsilon_3^{i+1}$ be the four output differences of E_i for $i \in$*

$\{0,\ldots,r-1\}$. *For two input differences* α,α' *and two output differences* β,β' *of* E*, if there exists a trail*

$$(\epsilon_0^0=\alpha,\epsilon_1^0,\epsilon_2^0=\alpha',\epsilon_3^0)\xrightarrow{P_0}\cdots\xrightarrow{P_{r-1}}(\epsilon_0^r,\epsilon_1^r=\beta,\epsilon_2^r,\epsilon_3^r=\beta'),\tag{2}$$

then it is called an r*-round* T_P^S *boomerang trail. Here,* $\xrightarrow{P_i}$ *represents the propagation rule through S-boxes in* E_i *follows* P_i*.*

Propagation $P\in\mathcal{AP}_E^S$ must be connectable. Accordingly, we have the following construction.

Construction 5 (T_P^S-IBD). *Given an* r*-round SPN cipher* E *and a predefined rule* $P\in\mathcal{AP}_E^S$*, for two input differences* α,α' *and two output differences* β,β'*, if no* r*-round* T_P^S *boomerang trail exists, then* $(\alpha,\alpha',\beta,\beta')$ *is an IBD, called an* r*-round* T_P^S*-IBD.*

T_P^S-IBD corresponds to a serious of IBDs for different predefined propagation rules P. Here, it is necessary to provide a clarification for DBCT used to detect contradictions within two consecutive rounds in $\mathcal{ZWT}$-method and $\mathcal{BCL}$-method [12,15]. The generalized DBCT (GDBCT) for an n-bit S-box is defined as

$$\mathrm{GDBCT}(\gamma,\gamma',\delta,\delta')=\sum_{\theta,\theta',\lambda,\lambda'}\mathrm{GUBCT}(\gamma,\gamma',\theta,\theta',\lambda,\lambda')\cdot\mathrm{GLBCT}(\theta,\theta',\lambda,\lambda',\delta,\delta'),$$

where $\gamma,\gamma',\theta,\theta',\lambda,\lambda',\delta,\delta'\in\mathbb{F}_2^n$ in [12]. Unlike DBCT Definition 2, GDBCT does not require that $\gamma=\gamma'$, $\delta=\delta'$, $\theta=\theta'$, and $\lambda=\lambda'$ for the input, output and middle differences, which are extremely stringent when applying DBCT in the middle two rounds of IBD. However, current methods have not made use of GDBCT to detect contradictions. Additionally, a GDBCT-based boomerang trail corresponds to a set of boomerang trails based on a composition of GUBCT and GLBCT. Thus, the IBDs constructed by GDBCT also is one type of T_P^S-IBD. Furthermore, T_1-IBD is a special example of T_P^S-IBD.

Theorem 4. *For any predefined rule* $P\in\mathcal{AP}_E^S$*, an* r*-round* T_1*-IBD* $(\alpha,\alpha',\beta,\beta')$ *is an* r*-round* T_P^S*-IBD.*

Proposition 1 shows that GEBCT belongs to all other tables. Therefore, we consider to construct IBDs using only GEBCT.

Definition 10. *Let* $E=E_{r-1}\circ\cdots\circ E_0$ *be an* r*-round block cipher. Let* $\epsilon_0^i,\epsilon_1^i,\epsilon_2^i,\epsilon_3^i$ *be the four input differences and* $\epsilon_0^{i+1},\epsilon_1^{i+1},\epsilon_2^{i+1},\epsilon_3^{i+1}$ *be the four output differences of* E_i *for* $i\in\{0,\ldots,r-1\}$*. For two input differences* α,α' *and two output differences* β,β' *of* E*, if there exists a trail*

$$(\epsilon_0^0=\alpha,\epsilon_1^0,\epsilon_2^0=\alpha',\epsilon_3^0)\xrightarrow{GEBCT}\cdots\xrightarrow{GEBCT}(\epsilon_0^r,\epsilon_1^r=\beta,\epsilon_2^r,\epsilon_3^r=\beta'),\tag{3}$$

then it is called an r*-round* T_C *boomerang trail.*

Construction 6 (T_C-IBD). *Given an* r*-round block cipher* E*, for two input differences* α,α' *and two output differences* β,β'*, if no* r*-round* T_C *boomerang trail exists, then* $(\alpha,\alpha',\beta,\beta')$ *is an IBD, called an* r*-round* T_C*-IBD.*

From the definitions, an r-round T_C boomerang trail is a special r-round T_P^S boomerang trail. Thus, we have the following inclusion relationship.

Theorem 5. *For any predefined rule $P \in \mathcal{AP}_E^S$, an r-round T_P^S-IBD $(\alpha, \alpha', \beta, \beta')$ is an r-round T_C-IBD.*

Subsequently, we prove inclusion relationship between T_C-IBD and T_2-IBD.

Theorem 6. *An r-round T_C-IBD $(\alpha, \alpha', \beta, \beta')$ is an r-round T_2-IBD.*

Additionally, we prove the equivalence of T_C-IBD and T_2-IBD for SPN with full-state round-key-addition operations.

Proposition 2. *Given an SPN cipher with full-state round-key-addition operations, $(\alpha, \alpha', \beta, \beta')$ is an r-round T_C-IBD if and only if it is an r-round T_2-IBD.*

Similarly, for Feistel and ARX ciphers, we explore the IBD construction methods based on generalized BTs in Appendix A. The process is similar: For Feistel ciphers, we generalize the definition of FBCT and FBDT to define GFBCT and GFUBCT/GFLBCT; we define T_P^F-IBD and study the relationships between T_P^F-IBD and T_i-IBD $(0 \leq i \leq 3)$ as well as T_C-IBD. For ARX ciphers, we redefine a modular addition as an S-box, inspired by [35], and the IBD constructions based on our generalized BTs for SPN and Feistel ciphers and their inclusion relationships all apply to ARX ciphers.

In summary, we have the following relations:

Summary 1 *For a block cipher E, let S_{T_i} denotes the set of all T_i-IBDs and S_{IBD} be the set corresponding to the essential definition of IBDs, then*

$$(S_{T_0} \subseteq) S_{T_1} \subseteq S_{T_P} \subseteq S_{T_C} \subseteq S_{T_2} \subseteq S_{T_3} = S_{IBD}.$$

The inclusion relationship for T_0-IBD specifically applies to block ciphers with bijective S-boxes. Here, T_P^S, T_P^F and T_P^A are collectively denoted as T_P. Especially, for an (equivalently transformed) SPN or Feistel cipher E with full-(branch)-state round-key-addition operations, we further have $S_{T_C^S} = S_{T_2}$ or $S_{T_C^F} = S_{T_2}$.

In brief, the differential-propagation-based IBDs do not surpass those generalized-BTs-based IBDs, and further those state-propagation-based IBDs, which are equivalent to the essential IBD definition when considering the key schedules. While constructing IBDs using BTs from BDs seems reasonable, there still exist a gap between the optimal IBD and the construction using even a mixed use of generalized BTs. Moreover, searching for IBDs using continuous generalized BTs is lack and challenging, which requires more advanced modeling techniques.

So far, we have developed a complete theory of IBD constructions. Furthermore, we propose a hierarchical apply strategy for IBD constructions when choosing the appropriate method as follows.

- The inclusion relationship in Summary 1 and the equivalence in Theorem 3 allow us to use T_0-IBD for a rough estimation (lower bound) and T_3-IBD for a precise evaluation (upper bound) of the number of rounds of IBDs.

- As no approach for constructing IBDs surpasses Construction 4, searching for T_3-IBDs remains prioritized, when solving time permits. When it encounters the efficiency bottleneck of solvers for searching T_3-IBDs, T_C-IBDs, T_P-IBDs, T_2-IBDs, T_1-IBDs, T_0-IBDs and hybrid techniques provide valuable supplements and effective estimation of the number of rounds for IBDs.
- In the field of automatic search, a critical consideration is the trade-off between precision and computational consumption. Given a diverse set of IBD construction methods, each with distinct precision level, we develop a scheduling algorithm to achieve an optimal balance between search time and accuracy. The algorithm begins with random sampling input and output differences of IBDs within the designated search space, and then initiates multiple processes, each associated with an IBD construction method. Each process uses its corresponding IBD model to validate whether the sampled data qualifies as an IBD. The algorithm records the total executing time for all sampled data and terminates the process whose time exceeds the preset threshold. Among the IBD construction methods that complete within the time limit, select the highest-priority method for subsequent full-space search. The overall algorithm is detailed in Algorithm 1 in Appendix C.

Once the target is determined, the search process is initiated. To enhance implementation efficiency, we also employ multi-process technology for parallel computations of divided search space parts, significantly improving search performance. All in all, the complete theory and strategy paves the way for subsequent research on IBD constructions.

4 New Construct Theory for IBDs in Related-Key Setting

The power of RK-IBDs becomes more pronounced according to Definition 2, as attackers have greater control over the relationships between related keys. In this section, we construct the RK-IBDs in two-related-keys setting and four-related-keys setting. The definitions of T_j boomerang trails for $j = 0, 1, 3, P, C$ given in Section 3 can be naturally extended to the related-key setting.

Definition 11. *Given an r-round block cipher $E = E^1 \circ E^0$ with the key schedule* KS*, for two input differences α, α', two output differences β, β', and four key differences κ_0, κ_1, κ_2 and $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$,*

- *if there exist $\gamma \in \overline{\mathrm{DP}}_{E^0,\kappa_0}(\alpha)$, $\gamma' \in \overline{\mathrm{DP}}_{E^0,\kappa_1}(\alpha')$, $\delta \in \overline{\mathrm{DP}}_{(E^1)^{-1},\kappa_2}(\beta)$, and $\delta' \in \overline{\mathrm{DP}}_{(E^1)^{-1},\kappa_3}(\beta')$, such that $\gamma \oplus \gamma' \oplus \delta \oplus \delta' = 0$, then Trail 1 is called an r-round T_0 related-key boomerang trail, where $\overline{\mathrm{DP}}_{f,\kappa}(\alpha)$ is a relaxed variant of $\mathrm{DP}_{f,\kappa}(\alpha)$ by considering all details of operations of f except S-boxes.*
- *if there exist $\gamma \in \mathrm{DP}_{E^0,\kappa_0}(\alpha)$, $\gamma' \in \mathrm{DP}_{E^0,\kappa_1}(\alpha')$, $\delta \in \mathrm{DP}_{(E^1)^{-1},\kappa_2}(\beta)$, and $\delta' \in \mathrm{DP}_{(E^1)^{-1},\kappa_3}(\beta')$, such that $\gamma \oplus \gamma' \oplus \delta \oplus \delta' = 0$, then Trail 1 is called an r-round T_1 related-key boomerang trail.*

Definition 12. *Given $E = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ be an r-round block cipher with the key schedule* KS*. For two input differences α, α', two output*

differences β, β', and four key differences κ_0, κ_1, κ_2 and $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, let $I = \{(x_0, x_1, x_2, x_3) \mid x_0 \oplus x_1 = \alpha, x_2 \oplus x_3 = \alpha'\}$ and $O = \{(y_0, y_1, y_2, y_3) \mid y_1 \oplus y_2 = \beta, y_0 \oplus y_3 = \beta'\}$. If there exist $(x_0^0, x_1^0, x_2^0, x_3^0) \in I$, $(x_0^r, x_1^r, x_2^r, x_3^r) \in O$ and master keys k_j such that $x_j^{i+1} = E_{i,\mathrm{KS}_i(k_j)}(x_j^i)$ for $0 \le i \le r-1, 0 \le j \le 3$, under the key differences $k_0 \oplus k_1 = \kappa_0, k_2 \oplus k_3 = \kappa_1, k_1 \oplus k_2 = \kappa_2, k_0 \oplus k_3 = \kappa_3$, then $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$ is called an r-round T_3 related-key boomerang trail.

Let $\mathcal{AP}_E^S$, $\mathcal{AP}_E^F$ and $\mathcal{AP}_E^A$ be collectively denoted as $\mathcal{AP}_E$, i.e., a set of propagation rules for an block cipher E, then the following definition apply to SPN, Feistel and ARX ciphers.

Definition 13. *Given $E = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ be an r-round block cipher with the key schedule* KS*. Let $\epsilon_0^i, \epsilon_1^i, \epsilon_2^i, \epsilon_3^i$ be the four input differences and $\epsilon_0^{i+1}, \epsilon_1^{i+1}, \epsilon_2^{i+1}, \epsilon_3^{i+1}$ be the four output differences of $E_{i,\mathrm{KS}_i(k_0)}$, $E_{i,\mathrm{KS}_i(k_1)}$, $E_{i,\mathrm{KS}_i(k_2)}$, $E_{i,\mathrm{KS}_i(k_3)}$ for $i \in \{0, \ldots, r-1\}$. For two input differences α, α', two output differences β, β', and four key differences κ_0, κ_1, κ_2 and $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$,*

- *if there exists a Trail 2 under the key differences $k_0 \oplus k_1 = \kappa_0, k_2 \oplus k_3 = \kappa_1, k_1 \oplus k_2 = \kappa_2, k_0 \oplus k_3 = \kappa_3$, then it is called an r-round T_P related-key boomerang trail. Here, $P = (P_0, \ldots, P_{r-1})$ is a predefined propagation rule of E, where $P_i \in \mathcal{AP}_{E_i}$ for $i \in \{0, \ldots, r-1\}$, and $\xrightarrow{P_i}$ represents the propagation rule through S-boxes in E_i follows P_i.*
- *if there exists a Trail 3 under the key differences $k_0 \oplus k_1 = \kappa_0, k_2 \oplus k_3 = \kappa_1, k_1 \oplus k_2 = \kappa_2, k_0 \oplus k_3 = \kappa_3$, then it is called an r-round T_C related-key boomerang trail.*

The RK-IBDs in two-related-keys setting was first proposed by J. Lu [5], focusing solely on differential propagation rather than state propagation and generalized BTs. Therefore, we introduce the following definitions.

Construction 7 (RT_i^2-IBD). *Given an r-round block cipher E with the key schedule* KS*, for two input differences α, α' and two output differences β, β', if no r-round T_i related-key boomerang trail under the key differences $(\kappa, \kappa, 0, 0)$ exists, where $k_0 \oplus k_1 = \kappa, k_2 \oplus k_3 = \kappa, k_1 \oplus k_2 = 0, k_0 \oplus k_3 = 0$, then $(\alpha, \alpha', \beta, \beta')$ is called an r-round RK-IBD in two-related-keys setting, denoted as RT_i^2-IBD for $i = 0, 1, 3, P, C$.*

Accordingly, the RK-IBDs in four-related-keys setting are defined as follows.

Construction 8 (RT_i^4-IBD). *Given an r-round block cipher E with the key schedule* KS*, for two input differences α, α' and two output differences β, β', if no r-round T_i related-key boomerang trail under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ exists, where $k_0 \oplus k_1 = \kappa_0$, $k_2 \oplus k_3 = \kappa_1$, $k_1 \oplus k_2 = \kappa_2$, $k_0 \oplus k_3 = \kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, then $(\alpha, \alpha', \beta, \beta')$ is called an r-round RK-IBD in four-related-keys setting, denoted as RT_i^4-IBD for $i = 0, 1, 3, P, C$.*

Using the same proving method, these RK-IBDs in the related-key setting are superior to RT_1^i-IBD but inferior to RT_3^i-IBD for $i = 2, 4$.

Summary 2 *For a block cipher E, let $S_{RT_j^i}$ denotes the set of RT_j^i-IBDs, then*

$$(S_{RT_0^i} \subseteq) S_{RT_1^i} \subseteq S_{RT_P^i} \subseteq S_{RT_C^i} \subseteq S_{RT_3^i},$$

for $i = 2, 4$ and $S_{RT_3^4} = S_{IBD}$. The inclusion relationship for RT_0-IBD specifically applies to block ciphers with bijective S-boxes.

Additionally, as explained in Section 3, hierarchical apply strategy for IBD constructions also applies to related-key settings.

To search for RK-IBDs in practice, the selection of key differences involves greater technicality. This view has been long known to our community, but formal and general selection rules have not yet been established. In particular, by examining different scenarios, we make the description of RK-IBD constructions more expressive and accurate.

In two-related-keys setting. We set the two input differences with $\alpha = \alpha'$. The two-related-keys setting is applicable to block ciphers with linear or nonlinear key schedules.

A direct strategy is to set key differences κ that offset the input differences α (usually define $\kappa = \mathcal{L}(\alpha)$, where $\mathcal{L}$ represents a linear transformation according to specific ciphers), thereby allowing several initial rounds of E to proceed without any differences. Subsequently, for each (α, κ) pair, we search for output differences (β, β') such that $(\alpha, \alpha, \beta, \beta')$ forms an r-round RT_i^2-IBD for $i = 0, 1, 3$ under key differences $(\kappa, \kappa, 0, 0)$. This approach ensures that the difference propagates through the non-linear operations in the initial rounds with probability 1, thereby achieving the objective of an RK-IBD covering more rounds.

An advanced strategy involves using related-key differentials with probability 1. Specifically, for a block cipher $E = E^1 \circ E^0$, we control round key differences to eliminate state differences before all nonlinear operations in E^0. Firstly, we search for an r_0-round related-key differential (α, γ) with probability 1 under key difference κ, i.e., $P_{E^0,\kappa}(\alpha, \gamma) = 1$. During the search process, it is crucial to ensure that the key difference does not undergo the nonlinear operation of KS. Next, we search for (β, β') such that $(\gamma, \gamma, \beta, \beta')$ is an r_1-round RT_i^2-IBD of E^1 for $i = 0, 1, 3$ under the key differences $(\kappa, \kappa, 0, 0)$. Consequently, $(\alpha, \alpha, \beta, \beta')$ is an $(r_0 + r_1)$-round RK-IBD under the key differences $(\kappa, \kappa, 0, 0)$.

In four-related-keys setting. The four-related-keys setting is applicable to block ciphers with linear key schedules. When the key schedule is linear, the key differences $k_0 \oplus k_1 = \kappa_0$, $k_2 \oplus k_3 = \kappa_1$ and $k_1 \oplus k_2 = \kappa_2$ imply that for any round index i and j:

$$\begin{cases} \mathrm{KS}_i(k_0) \oplus \mathrm{KS}_i(k_1) = \mathrm{KS}_i(\kappa_0), \mathrm{KS}_i(k_2) \oplus \mathrm{KS}_i(k_3) = \mathrm{KS}_i(\kappa_1), \\ \mathrm{KS}_j(k_1) \oplus \mathrm{KS}_j(k_2) = \mathrm{KS}_j(\kappa_2), \mathrm{KS}_j(k_0) \oplus \mathrm{KS}_j(k_3) = \mathrm{KS}_j(\kappa_0 \oplus \kappa_1 \oplus \kappa_2), \end{cases}$$

indicating that the differences of round keys are fully determined.

For a block cipher $E = E^2 \circ E^1 \circ E^0$, the search strategy for RT_i^4-IBD for $i = 0, 1, 3$ under the key differences κ_0, κ_1, κ_2 and $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$ is as follows. Firstly, we search for two r_0-round related-key differentials (α, γ) under

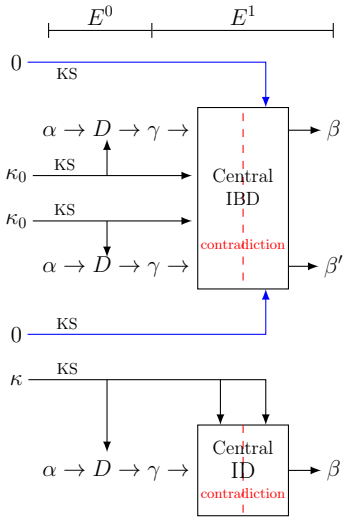

Fig. 4: Schematic for two-related-keys setting

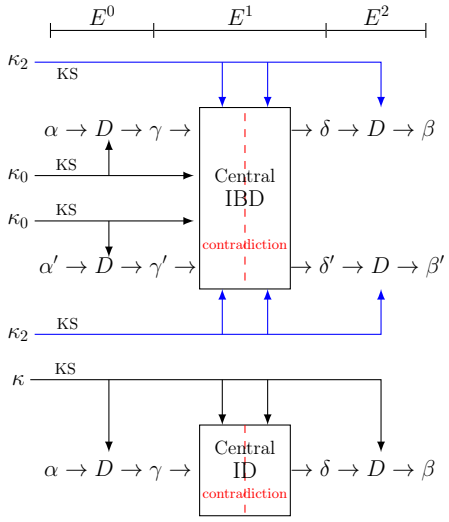

Fig. 5: Schematic for four-related-keys setting

†Here D denotes a related-key differential with probability 1.

key difference κ_0 and (α', γ') under key difference κ_1 of E^0 and two r_2-round related-key differentials (β, δ) under key difference κ_2 and (β', δ') under key difference κ_3 of $(E^2)^{-1}$, all with probability 1:

$$P_{E^0,\kappa_0}(\alpha,\gamma) = 1, P_{E^0,\kappa_1}(\alpha',\gamma') = 1, P_{(E^2)^{-1},\kappa_2}(\beta,\delta) = 1, P_{(E^2)^{-1},\kappa_3}(\beta',\delta') = 1 \quad (4)$$

Next, we verify whether $(\gamma, \gamma', \delta, \delta')$ forms an r_1-round RT_i^4-IBD for $i = 0, 1, 3$ under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$. If so, then $(\alpha, \alpha', \beta, \beta')$ is an $(r_0 + r_1 + r_2)$-round RT_i^4-IBD for $i = 0, 1, 3$. However, directly searching for these differentials while satisfying the relation $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$ poses practical challenges. To address this issue, we focus exclusively on block ciphers with linear key schedules, since the definitive round key differences $\mathrm{KS}_i(\kappa_j)$ for $i = 0, \ldots, r-1$ and $j = 0, 1, 2, 3$ provided by the linear key schedule facilitates managing the elimination of state differences in E^2.

Comparison of RK-IBDs and RK-IDs. A related-key ID (RK-ID) is a pair of difference (α, β) such that the input difference α cannot propagate to the output difference β under the key difference κ. Compared to RK-IDs, RK-IBDs present advantages in two folds.

Distinguisher Search. Considering the two-related-keys setting illustrated in Fig 4, we focus on searching for the central distinguisher. After the contradiction in RK-IBDs, the key difference for RK-IBDs is 0. In contrast, the key difference for RK-IDs undergoes extensive diffusion through KS algorithm, especially with respect to nonlinear KS, making it difficult to control or eliminate. This allows RK-IBDs to accommodate more rounds than RK-IDs.

Distinguisher Extension. Considering the four-related-keys setting illustrated in Fig 5, we focus on the extension of central distinguisher. For block ciphers with linear key schedules, extending both E^0 and E^2 under the same key difference is necessary for RK-IDs, limiting flexibility. In contrast, RK-IBDs allow greater flexibility in key differences, requiring only that $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$. This enables RK-IBDs to extend more rounds compared to RK-IDs.

5 New Automatic Search Methods for (RK-)IBDs

In this section, we present our search method for (RK-)IBDs from the aspects of differential, state propagation and generalized BTs, and propose several key search strategies.

5.1 Searching for (RK-)IBDs from the aspect of differential propagation

To search for $(R)T_0$-IBDs and $(R)T_1$-IBDs, we propose a SAT-based method to model the differential propagation through common operations such as `Xor`, `Copy`, `KeyAdd`, `MatrixMultiply` and `S-box`, as well as arbitrary S-box (`AS`) proposed in [36] that treats the S-box as only a permutation.

Model 1. *The modeling method of differential propagation through* `Xor`*,* `Copy`*, and* `KeyAdd` *is detailed as follows*

Operation	*Input Diff*	*Output Diff*	*Modeling Method*
`Copy`	$\alpha \in \mathbb{F}_2$	$\beta_0, \beta_1 \in \mathbb{F}_2$	$\beta_0 = \alpha, \beta_1 = \alpha$
`Xor`	$\alpha_0, \alpha_1 \in \mathbb{F}_2$	$\beta \in \mathbb{F}_2$	$\beta = \alpha_0 \oplus \alpha_1$
`KeyAdd`	$\alpha \in \mathbb{F}_2$	$\beta \in \mathbb{F}_2$	$\beta = \alpha$

Model 2. *For* `MatrixMultiply` *operation* $M = (m_{i,j})_{u \times v}$*, let* $\{\alpha_0, \ldots, \alpha_{v-1}\}$ *and* $\{\beta_0, \ldots, \beta_{v-1}\}$ *be the input and output differences of* M*, then* $\beta_i = \oplus_{j=0}^{v-1} m_{i,j}\alpha_j$*. Thus, the differential propagation through* `MatrixMultiply` *can be expressed according to* `Xor`*.*

Model 3. *For* `S-box` *operation of a* $v \times u$*-bit* S*, let* $\{\alpha_0, \ldots, \alpha_{v-1}\}$ *and* $\{\beta_0, \ldots, \beta_{u-1}\}$ *be the input and output differences of* S*, then they are restricted by the* DDT *of* S*. Thus, the differential propagation through* S *can be expressed as a set of logic expressions with the help of some third-party tools like Logic Friday [37].*

Model 4. *For* `AS` *operation of a bijective* S*, let* $\{\alpha_0, \ldots, \alpha_{v-1}\}$ *and* $\{\beta_0, \ldots, \beta_{v-1}\}$ *be the input and output differences, then the following transitions are impossible:* $(0,1), \cdots, (0, 2^v - 1)$*,* $(1,0), \cdots, (2^v - 1, 0)$*, and can be removed by the boolean expressions:* $\alpha_{v-1}|| \cdots ||\alpha_0||\neg\beta_i = 1$*,* $\neg\alpha_i||\beta_{v-1}|| \cdots ||\beta_0 = 1$ *for* $0 \leq i \leq v-1$*.*

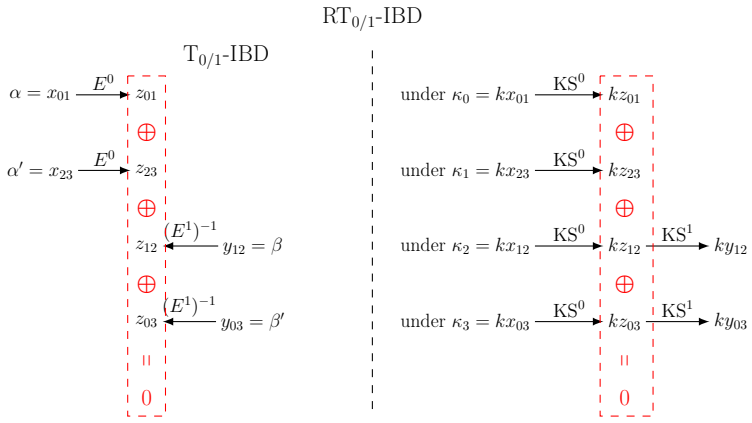

Fig. 6: The search model for $(R)T_0$-IBD or $(R)T_1$-IBD
†Here $(k)x_{ij} = (k)x_i \oplus (k)x_j$ denoting the difference variable of $(k)x_i$ and $(k)x_j$.

Based on above modeling method, for an r-round block cipher $E = E^1 \circ E^0$ with its key schedule $\mathrm{KS} = \mathrm{KS}^1 \circ \mathrm{KS}^0$, we can construct a model to determine whether a given $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ qualifies as an r-round $(R)T_0$-IBD or $(R)T_1$-IBD under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ by describing the constraints of differential propagation as shown in Fig 6. If the SAT-solver returns "no solution", then $(\alpha, \alpha', \beta, \beta')$ is an (RK-)IBD. Particularly in single-key setting, all four key differences are set to zero. The corresponding algorithm is provided in Algorithm 2 in Appendix C.

5.2 Searching for (RK-)IBDs from the aspect of state propagation

To search for T_2-IBDs and $(R)T_3$-IBDs from the aspect of state propagation, we revisit the SAT-based method to model the state propagation through common operations proposed in [20].

Model 5. *The state propagation through* `Xor`, `Copy` *and* `MatrixMultiply` *follows the methods described in Model 1 and Model 2. The state propagation through* `KeyAdd` *is modeled identically to* `Xor`.

Model 6. *For* `S-box` *operation of a* $v \times u$ *S, let* $\{\alpha_0, \ldots, \alpha_{v-1}\}$ *and* $\{\beta_0, \ldots, \beta_{u-1}\}$ *be the input and output states of S, then they are restricted by the truth table of S. Thus, the state propagation through S can be expressed as a set of logic expressions with the help of Logic Friday.*

Based on above modeling method, for an r-round block cipher E with its key schedule, we can construct a model to determine whether a given $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ qualifies as an r-round T_2-IBD or $(R)T_3$-IBD under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ by describing the constraints of differential propagation as shown

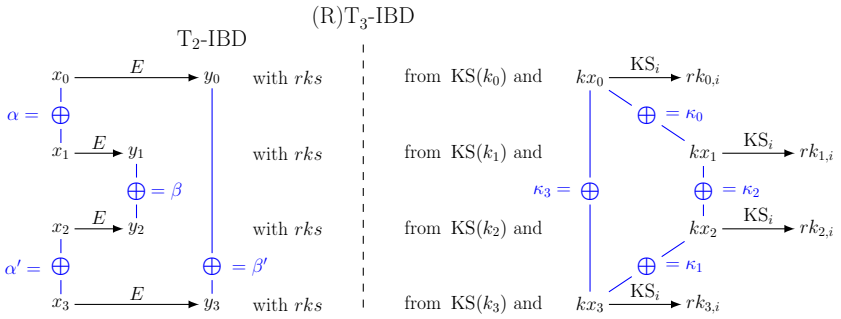

Fig. 7: The search model for T_2-IBD or $(R)T_3$-IBD

in Fig 7. If the SAT-solver returns "no solution", then $(\alpha, \alpha', \beta, \beta')$ is an (RK-)IBD. Particularly in single-key setting, all four key differences are set to zero. The corresponding algorithm is provided in Algorithm 3 in Appendix C.

5.3 Searching for (RK-)IBDs from the aspect of generalized BTs

To search for $(R)T_P$-IBDs and $(R)T_C$-IBDs from the aspect of generalized BTs, we are faced with two challenges: First, although GBCT/GFBCT-based method is relatively straightforward to model, however, GBCT can only detect contradictions within a single round; Second, to detect contradictions within more round, we need to search for IBDs based on continuous generalized BTs of GUBCT, GLBCT for SPN-like ciphers or GFUBCT, GFLBCT for Feistel-like ciphers or GEBCT, the number of parameters for input and output differences in these tables is no less than six, making it extremely difficult to model these tables directly by some third-party tools, even for 4-bit S-boxes. Therefore, currently there is no modeling method for generalized BT beyond GBCT. Fortunately, based on our state propagation method proposed in Section 5.2, we can provide a searchable model for these generalized BTs.

For simplicity, in this section, we model GEBCT as an example; other generalized BTs follow a similar modeling method. We introduce intermediate state variables $(x_0, x_1, x_2, x_3, y_0, y_1, y_2, y_3)$ and use the following modeling method to simulate GEBCT:

$$\begin{cases} y_i = S(x_i) \text{ for } i = 0, 1, 2, 3, \\ x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', \\ y_0 \oplus y_1 = \theta, y_2 \oplus y_3 = \theta', y_1 \oplus y_2 = \varphi, y_0 \oplus y_3 = \varphi', \end{cases}$$

where $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi'$ are eight input and output differences of GEBCT. Except SL, other operations (denoted as Λ), such as linear layer and round key addition, still employ the method of differential propagation for their modeling. Indeed, this method is a hybrid of iterative cross-use of differential propagation and state propagation. Based on above modeling method, for an r-round block

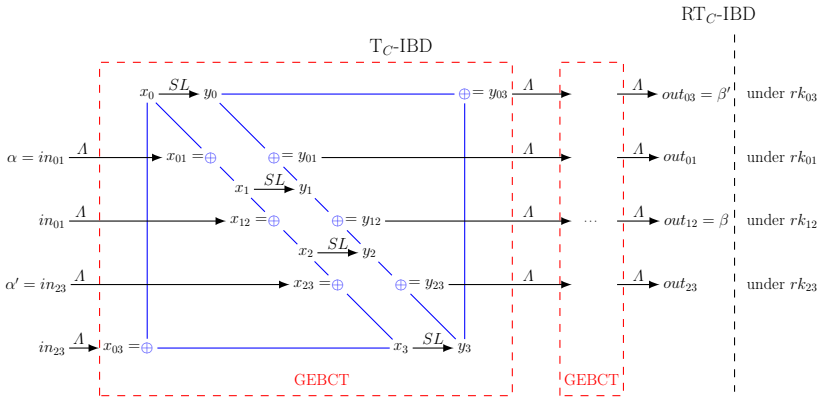

Fig. 8: The search model for $(R)T_C$-IBD

cipher E with its key schedule, we can construct a model to determine whether a given $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ qualifies as an r-round $(R)T_C$-IBD under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ by describing the constraints of differential propagation as shown in Fig 8. If the SAT-solver returns "no solution", then $(\alpha, \alpha', \beta, \beta')$ is an (RK-)IBD. Additionally, in single-key setting, all four key differences are set to zero. In related-key setting, the modeling for key schedules can be set based on differential propagation if the key schedule is linear, or based on state propagation as shown in the right part of Fig 7. The corresponding algorithm is provided in Algorithm 4 in Appendix C.

5.4 The search strategies for (RK-)IBDs

Search space. Based on Algorithm 2, 3, 4, we can efficiently search for (RK-) IBDs within a given search space. Let n be the block size and s be the S-box size, then there are $t = n/s$ S-boxes in SL. For $x = (x_{n-1}, \ldots, x_0) \in \mathbb{F}_2^n$, $wt(x) = \oplus_{i=0}^{n} x_i$ and $\overline{x} = (x_{n-1}|\cdots|x_{s\times(t-1)}, \cdots, x_{s-1}|\cdots|x_0)$, where $|$ denotes bitwise-OR. Similar to ID, we mainly focus on searching for the three following types of (RK-)IBDs.

Type-1. l_i input and l_o output active bits IBDs: an (RK-)IBD $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ with $wt_2(\alpha) = wt_2(\alpha') = l_i$ and $wt_2(\beta) = wt_2(\beta') = l_o$. Particularly, when $l_i = l_o$, $\mathcal{D}$ is called an l_i-active-bits (RK-)IBD.

Type-2. l_i input and l_o output active nibbles (resp. bytes) IBDs: an (RK-)IBD $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ with $wt(\overline{\alpha}) = wt(\overline{\alpha'}) = l_i$ and $wt(\overline{\beta}) = wt(\overline{\beta'}) = l_o$. Particularly, when $l_i = l_o$, $\mathcal{D}$ is called an l_i-active-nibbles (resp. bytes) (RK-) IBD.

Type-3. l_i input and l_o output active nibbles (resp. bytes) truncated IBDs: Given $a, a', b, b' \in \mathbb{F}_2^t$ with $wt(a) = wt(a') = l_i$ and $wt(b) = wt(b') = l_o$, if $\mathcal{D}$ is an (RK-)IBD for $\forall \mathcal{D} \in \{(\alpha, \alpha', \beta, \beta') \mid \overline{\alpha} = a, \overline{\alpha'} = a', \overline{\beta} = b, \overline{\beta'} = b'\}$, then (a, a', b, b') is called an l_i input and l_o output active nibbles (resp. bytes)

truncated IBD. Particularly, when $l_i = l_o$, (a, a', b, b') is called an l_i-active-nibbles (resp. bytes) truncated (RK-)IBD.

To search for Type-3 (RK-)IBD, we can simply modify Algorithm 2, 3, 4 by adding the relations between the bitwise input differences and output differences with (a, a') and (b, b').

Search strategy. When the search for (RK-)IBDs relying solely on state propagation is extremely time-consuming, or relying solely on differential propagation or generalized BTs is not optimized enough, we further consider a hybrid search method that use distinct propagation approaches and modeling methods for different operations or cipher stages. The selection of hybrid search can be integrated into the scheduling algorithm given in Algorithm 1.

Besides, we also propose a miss-in-the-middle approach for $E = E^1 \circ E^m \circ E^0$ that removes the current constraint of probability-1 differentials for E^0 and E^1. Instead, as shown in Fig 9, we propagate the input difference (α, α') forward and output differences (β, β') backward based on differential propagation or generalized BTs-based methods to derive all possible output differences of E^0 and all possible input differences of E^1. Subsequently, we verify these combinations of the intermediate differences all result in IBDs for E^m using these pure or hybrid modeling methods. Thus, we get an (RK-)IBD for E.

Verify strategy. To compensate for the limitations of manual verification, we employ some techniques of computer-aided verification of (RK-)IBDs given in Appendix F, where the computer can play the roles of detecting the contradiction location, traversing and disproving all plausible trails, and implementing cross-validation. Firstly, we can modify the bit-conditions in Algorithm 2, 3, 4 to filter out unrelated positions, and limit the possible location of the contradiction to a very small range, and then derive it further. Secondly, we can traverse all plausible trails from two ends and then disprove them by generalized BTs similarly as shown in Fig 9. Thirdly, we can call for different search methods for the same result or refer to BTs or generalized BTs to verify the result.

Besides, the correctness of the code is crucial for successfully searching for the correct distinguisher. In traditional methods, verifying the correctness of all distinguishers obtained through automatic search typically involves validating a subset of them. If issues arise, re-checking and re-running the search program can be extremely time-consuming. Therefore, an automatic verification technique that can minimize the occurrence of code implementation errors is essential. We propose a code self-feedback verification technique. The idea is that when the input and output differences are not an (RK-)IBD, the third-party solver will return a solution. The solution must correspond to trails that propagate input differences to output differences using state propagation, DDT, GEBCT, etc. By verifying these trails, we can detect erroneous trails generated by incorrect code.

6 Applications of (RK-)IBDs

In this section, we apply our method to search for (RK-)IBDs on 10 block ciphers with three main structures:

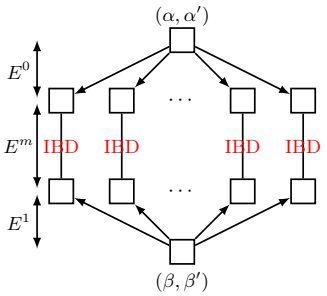

Fig. 9: The search strategy of (RK-)IBDs

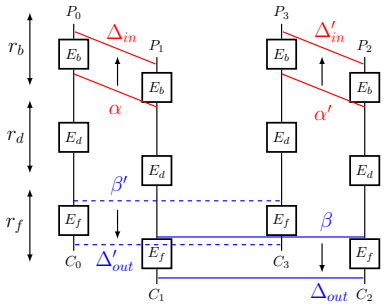

Fig. 10: The overview of IBA

SPN: `AES` (large S-box, MDS), `PRESENT-80` and `GIFT` (bit-permutation), `PRINTcipher48` (key-dependent permutation), `SKINNY-64/192` and `SKINNYee` (tweakey);
Feistel: `DES` (non-injective and non-quadratic S-box), `GOST`;
ARX: `SPECK`, `CHAM`.

Specifically, we search for IBDs on `AES`, `PRESENT-80`, `PRINTcipher48` and `DES`, as well as RK-IBDs on `AES-128`, `SPECK` with nonlinear KS, and `SKINNY-64/192`, `SKINNYee`, `GIFT`, `CHAM` and `GOST` with linear KS. For most of these ciphers, we are capable of conducting IBD search for the first time, as previous methods were ineffective on them. We present brief cipher descriptions in Appendix D, examples of searched IBDs in Appendix E, and their verifications in Appendix F.

To demonstrate the efficacy of our IBDs, we use `SKINNYee` as a case study and present its corresponding 31-round IBA. This constitutes the first 31-round attack on `SKINNYee` and represents the best result to date.

Experiments were conducted on an AMD(R) @2.60GHz platform with 80.00G RAM running a 64-bit Ubuntu18.04 system. For fair comparison, all timing is attributed to the spent on searching for a distinguisher using a single core.

6.1 Applications of IBDs in single-key setting

AES

-Configurations. As `AES` is built with 8-bit S-boxes with excellent cryptographic properties, searching for T_i-IBDs ($i = 1, 2, 3$) is extremely time-consuming. Therefore, our focus shifts to search for T_0-IBDs that treat the S-boxes as a permutation by Algorithm 2, allowing us to evaluate only truncated IBDs (Type-3). Consequently, we search for 1-active-byte truncated T_0-IBDs in a space of size $16^4 = 65536$.

-Results. We prove no 5-round 1-active-byte T_0-IBDs exist in 203.44 hours; and obtain all 61440 4-round 1-active-byte T_0-IBDs in 149.04 hours.

-Comparison. (i) Currently, the only result for IBDs on AES is some 4-round IBDs derived manually [4]. In contrast, our method first enables automatic search and result in a large number of 4-round IBDs, as our approach is the **first to enable the search for IBDs of block ciphers with large S-boxes considering the details of linear layers**. (ii) On AES, the maximum round number of IDs is 4 [17], with no 5-round ID existing [18]. Thus, the IBDs we found cover the same number of rounds as the optimized IDs.

DES

-Configurations. Our experimental results show that searching for T_2-IBDs or T_3-IBDs beyond 7 rounds is excessively time-consuming. Therefore, we focus on searching for 1-active-bit T_1-IBDs (Type-1) using Algorithm 2. Given DES's Feistel network, we restrict input active differences to the left branch and output active differences to the right branch, allowing us to propagate differences forward and backward through one round with probability 1. Additionally, either the two input differences or two output differences must be identical. The search space size is $2 \times 32^3 = 2^{16}$.

-Results. We prove no 8-round 1-active-bit T_1-IBDs within such search space exist in 372.38 hours; and obtain 1904 7-round 1-active-bit T_1-IBDs in 327.64 hours.

-Comparison. (i) Currently, $\mathcal{BCL}$-method [15] is the only approach searching for IBDs on Feistel ciphers, but it is limited to quadratic round functions and does not apply to DES. Our approach is the **first to enable searching for IBDs of Feistel ciphers with arbitrary round functions.** (ii) To compare with IDs, we used ST-method [36], constraining input and output differences to 1 active bit. This yielded 7-round IDs and confirmed no 8-round ID existing within this search space. Thus, the IBDs we found cover the same number of rounds as the optimized IDs.

PRESENT-80

-Configurations. We search for 1-active-nibble T_3-IBDs (Type-2) using Algorithm 3, with only the first S-box of the input and output differences being active. The search space size is $15^4 = 50625$.

-Results. We prove no 7-round T_3-IBDs exist within such search space in 24.52 hours; and obtain 58 6-round 1-active-nibble T_3-IBDs in 7.13 hours. To investigate the proper inclusion relationships in Summary 1, we tested whether these T_3-IBDs are also T_i-IBDs ($0 \leq i \leq 2$). The results show that all these T_3-IBDs are T_2-IBDs but not T_1-IBDs.

-Comparison. (i) Due to the use of bit permutation in PRESENT, existing methods based on truncated difference search for IBDs are not applicable. Our approach is the **first to enable the search for IBDs of block ciphers using bit permutation.** (ii) To our knowledge, the maximum round number of IDs for PRESENT-80 is 6 [19], with no 7-round 1-active-nibble ID existing [20]. Thus, the IBDs we found cover the same number of rounds as the optimized IDs.

PRINTcipher48

-Configurations. Since `PRINTcipher48` uses a key-dependent permutation, we directly search for 1-active-bit T_3-IBDs (Type-1) using Algorithm 3. Specifically, either the two input differences or the two output differences must be identical. The search space size is $2 \times 48^3 = 221184$.

-Results. We prove no 6-round T_3-IBDs exist within such search space in 40.07 hours; and obtain 2 5-round 1-active-bit T_3-IBDs in 14.75 hours. To investigate the proper inclusion relationships given in Summary 1 academically, we remove the relationship between round keys to test whether those T_3-IBDs are T_2-IBDs, and the result shows they are not.

-Comparison. (i) Due to the lack of consideration for state, existing search methods for IBDs cannot accurately model the key-dependent permutation in `PRINTcipher48`. Our approach is the **first to enable the search for IBDs of block ciphers using key-dependent permutation**. (ii) To our knowledge, the maximum round number of IDs for `PRINTcipher48` is 4, with no 5-round ID existing [20] even considering the key schedule details. Thus, the IBDs we found cover **one more round than the optimized IDs**.

6.2 Applications of RK-IBDs in related-key setting

AES-128

-Configurations. Since `AES-128` uses a non-linear KS, we search for RT_0^2-IBDs using Algorithm 2. Similar to the single-key setting, we focus on 1-active-byte RT_0^2-IBDs (Type-3) with input-output differences $(\alpha, \alpha, \beta, \beta')$ where α, β, β' are 1-active-byte truncated differences under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\alpha, \alpha, 0, 0)$. The search space size is $16^3 = 2^{12}$.

-Results. We prove no 6-round RT_0^2-IBDs exist within such search space in 18.68 hours; and obtain 768 5-round 1-active-byte RT_0^2-IBDs in 14.44 hours.

-Comparison. (i) 6-round RK-IBDs for `AES-192` and `AES-256` are manually derived in [4]. However, no results have been reported for `AES-128`. Our method presents the **first 5-round RK-IBDs** for `AES-128`, for the advantage discussed in in single-key setting. (ii) To compare with RK-IDs, we used the AS mode in ST-method [36] to search for r-round RK-IDs with input-output differences (α, β), where α and β are 1-active-byte truncated differences under key difference α. This yielded 3-round RK-IDs and confirmed no 4-round RK-IDs existing within this search space. Thus, the RK-IBDs we found cover **two more rounds than the optimized RK-IDs**.

SPECK

-Configurations. Since `SPECK` uses a non-linear KS, we search for the RT_3^2-IBDs using the advanced strategy described in Section 4. First, we search for the r_0-round related-key differential with probability 1, i.e. (α, γ) under key difference κ. Next, we search for β and β' such that $(\gamma, \gamma, \beta, \beta')$ forms

an r_1-round RT_3^2-IBDs under key differences $(\kappa, \kappa, 0, 0)$. Here, $\beta, \beta' \in \Lambda = \{(\mu, \mu), (0, \mu \ggg_b), (\mu, \mu \oplus (\mu \ggg_b))\}$, where $\ggg_b$ denotes a circular right shift by b bits ($b = 2$ for SPECK-32, $b = 3$ otherwise), and μ is $n/2$-bit, with only its most significant bit set to 1. This allows us to go backward through the modulo addition operation in the last round with probability 1. By connecting the central RK-IBD with the probability-1 related-key differentials at the beginning, we obtain $r = r_0 + r_1$ rounds. The search space size is 9.

-Results. First, we find that SPECK-$2w/4w$ (for $w = 16, 24, 32, 64$) has a 4-round related-key differential characteristic with probability 1, and SPECK-$2w/3w$ (for $w = 24, 32, 48, 64$) has a 3-round related-key differential characteristic with probability 1, in a very short time. Then, we use Algorithm 3 to search for RT_3^2-IBDs. The results are as follows.

Block cipher	Round (r)	Number	Time (hours)	Compared with IDs
SPECK-32/64	8/9	6/none	0.18/0.97	1 round more
SPECK-48/72	7/8	6/none	0.06/0.26	1 round more
SPECK-48/96	8/9	6/none	0.09/0.60	1 round more
SPECK-64/96	8/9	4/none	0.29/0.60	2 rounds more
SPECK-64/128	9/10	4/none	0.28/0.99	2 rounds more
SPECK-96/144	8/9	4/none	0.22/0.65	2 rounds more
SPECK-128/192	8/9	4/none	0.33/1.18	2 rounds more
SPECK-128/256	9/10	4/none	0.41/1.78	2 rounds more

-Comparison. (i) Since SPECK using the modular addition operation, all existing IBD search methods are not applicable to describe it. Our approach is the **first to enable the search for IBDs of block ciphers using modular additions**. (ii) To our knowledge, the maximum round number of RK-IDs on SPECK-$2w/4w(w = 16, 24, 32, 64)$ is 7, as well as that on SPECK-$2w/3w(w = 24, 32, 48, 64)$ is 6 [21]. Thus, the RK-IBDs we found cover **one or two more rounds than the optimized RK-IDs** to date for each version of SPECK.

SKINNY-64/192

-Configurations. Since SKINNY-64/192 uses a linear tweakable key schedule, we search for r-round RK_3^4-IBDs using the strategy that connects related-tweakey differentials with probability 1 by Algorithm 3. Let rb denote the beginning round of the distinguisher and set $rb = 0$ without loss of generality. From Property 4 in Appendix G, there exists the difference of four tweakeys ΔMTK such that $\Delta TK_{rb+j} = 0$ for $1 \leq j \leq 5$, $\Delta TK_{rb} \neq 0$ and $\Delta TK_{rb+6} \neq 0$, and the difference of four tweakeys $\Delta MTK'$ such that $\Delta TK'_{rb+r-6} \neq 0$, $\Delta TK'_{rb+r} \neq 0$ and $\Delta TK'_{rb+r-j} = 0$ for $1 \leq j \leq 5$. We then search for r-round RK_3^4-IBDs without the first SC operation $(\alpha, \alpha', \beta, \beta') = (\Delta\overline{TK}_{rb}, \Delta\overline{TK}_{rb}, MC \circ SR(\Delta\overline{TK}'_{rb+r}), MC \circ SR(\Delta\overline{TK}'_{rb+r})$ under tweakey differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\Delta MTK, \Delta MTK, \Delta MTK', \Delta MTK')$, where the notation $\overline{x}$ represents the operation of extending the 32-bit value x to a 64-bit value by appending zeros at the end of x. This choice allows the input

differences to propagate 6 rounds forward and output differences propagate 6 rounds backward with probability 1. Besides, we set only one cell of ΔTK_{rb} and ΔTK_{rb+r} active, the search space size is $(8 \times 15)^2 = 14400$.

-Results. We prove no 20-round RT_3^4-IBDs in above search space exist in 267.21 hours; and obtain 3 19-round 1-active-nibble RT_3^4-IBDs in 136.12 hours.

-Comparison. For `SKINNY-64/192`, $\mathcal{ZWT}$-method [12] proposed 18-round RK-IBDs, while our method achieves 19-round RK-IBDs. Thus, the RK-IBDs we found cover **one more round than the previous best RK-IBDs**.

SKINNYee

-Configurations. Since `SKINNYee` uses a linear tweak schedule, we search for r-round RK_3^4-IBDs using the strategy that connects related-tweak differentials with probability 1 by Algorithm 3. Let rb denote the beginning round of the distinguisher and set it as $rb = 0$ without loss of generality. From Property 5 in Appendix G, there exists the difference of four tweaks ΔMTK such that $\Delta TK_{rb+j} = 0 (1 \leq j \leq 7)$, $\Delta TK_{rb} \neq 0$ and $\Delta TK_{rb+8} \neq 0$, and the difference of four tweaks $\Delta MTK'$ such that $\Delta TK'_{rb+r-8} \neq 0$, $\Delta TK'_{rb+r} \neq 0$ and $\Delta TK'_{rb+r-j} = 0 (1 \leq j \leq 7)$. We then search for r-round RK_3^4-IBDs without the first SC operation $(\alpha, \alpha', \beta, \beta') = (\Delta\overline{TK}_{rb}, \Delta\overline{TK}_{rb}, MC \circ SR(\Delta\overline{TK}'_{rb+r}), MC \circ SR(\Delta\overline{TK}'_{rb+r})$ under tweaks differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\Delta MTK, \Delta MTK, \Delta MTK', \Delta MTK')$, where the notation $\overline{x}$ represents the operation of extending the 32-bit value x to a 64-bit value by appending zeros at the end of x. This choice allows the input differences to propagate 8 rounds forward and the output differences to propagate 8 rounds backward with probability 1. Besides, we set only one cell of ΔTK_{rb} and ΔTK_{rb+r} active, the search space size is $(8 \times 15)^2 = 14400$.

-Results. We prove no 24-round RT_3^4-IBDs in above search space exist in 488.89 hours; and obtain 5 23-round 1-active-nibble RT_3^4-IBDs in 209.78 hours.

-Comparison. $\mathcal{BCL}$-method [15] proposed 22-round RK-IBDs for `SKINNYee`, while our method achieves 23-round RK-IBDs. Thus, the RK-IBDs we found cover **one more round than the previous best RK-IBDs**.

GIFT

-Configurations. Since `GIFT` uses a linear KS, we search for r-round RK_3^4-IBDs using the strategy that connects related-key differentials with probability 1 by Algorithm 3. First, for `GIFT-64`, we search for RK_3^4-IBDs under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\lambda, \lambda, \eta, \eta)$ by setting input differences $(\alpha, \alpha') = (KS_{rb}(\lambda), KS_{rb}(\lambda))$ and output differences $(\beta, \beta') = (KS_{rb+r}(\eta), KS_{rb+r}(\eta))$, where rb denotes the beginning round of the distinguisher and is set as $rb = 0$ without loss of generality. This choice allows the input differences to propagate 4 rounds forward and the output differences to propagate 4 rounds backward with probability 1. Thus, we only need to verify whether $(0, 0, 0, 0)$ is an $(r - 8)$-round RK-IBD under key differences $(\lambda, \lambda, \eta, \eta)$. Moreover, we

set the value of λ such that only one S-box is active in round $(rb+5)$, and the value of η such that only one S-box is active in round $(rb+r-5)$, which facilitates the search for distinguishers covering a large number of rounds. The search space size is $(16 \times 3)^2 = 2034$. We apply the same method to GIFT-128 and the search space size is $(32 \times 3)^2 = 9216$.

-Results. The results obtained are as follows.

Block cipher	Round(r)	Number	Time(hours)	Compared with IDs
GIFT-64	13/14	48/none	0.51/1.91	1 round more
GIFT-128	10/11	373/none	3.71/32.15	same rounds more distinguishers

-Comparison. (i) Similar to PRESENT, GIFT uses bit permutation, making it impossible to search for RK-IBDs of GIFT using previous methods. (ii) To our knowledge, the optimized RK-IDs are 48 12-round RK-ID for GIFT-64 and 96 10-round RK-IDs for GIFT-128 with no more round distinguisher existing [20]. Thus, the RK-IBDs we found cover **one more round than the optimized RK-IDs** for GIFT-64, and same as the maximum round number of RK-IDs for GIFT-128. But there are more instances of RK-IBDs compared to those of RK-IDs.

CHAM

-Configurations. Since CHAM uses a linear KS, we search for RT_3^4-IBDs using a strategy that connects related-key differentials with probability 1. Without loss of generality, we assume that the distinguisher for $E = E^0 \circ E^m \circ E^1$ starts from round 0. We search for (α, γ), an r_0-round related-key differential with probability 1 for E^0 under the key difference μ, and (β, δ), an r_2-round related-key differential with probability 1 for $(E^1)^{-1}$ under the key difference ν. Subsequently, we verify whether $(\gamma, \gamma, \delta, \delta)$ is an r_1-round RT_3^4-IBD under the key differences $\kappa_0 = \kappa_1 = \mu, \kappa_2 = \kappa_3 = \nu$ by Algorithm 3; if so, then $(\alpha, \alpha', \beta, \beta')$ is an $r_0 + r_1 + r_2$-round RK-IBD under the key difference.

-Results. The results obtained are as follows:

Block cipher	Round(r)	Number	Time(hours)	Compared with IDs
CHAM-64/128	30/31-32	3/none	0.15/0.22	4 round more
CHAM-128/256	28/29-30	4/none	0.48/0.63	2 round more

-Comparison. (i) To date, no existing methods have been able to search for RK-IBDs on CHAM due to its use of modular additions. (ii) To our knowledge, the maximum round number of RK-IDs for CHAM-64/128 and CHAM-128/256 are both 26 [21]. Thus, the RK-IBDs we found cover **4 and 2 more rounds than the optimized RK-IDs** for CHAM-64/128 and CHAM-128/256, respectively.

GOST

-Configurations. Since GOST uses a linear KS, we search for RT_3^4-IBDs using the strategy that connects related-key differentials with probability 1 by Algorithm 3. Specifically, according to KS, the key difference κ_i can be written

as $\kappa_i = \kappa_{i,7}||\ldots||\kappa_{i,0}$, where $\kappa_{i,j}$ for $0 \leq j \leq 7$ is a 32-bit value. As shown in Fig 40, 41, 42 and 43, GOST has 24-round related-key differential and 7-round related-key differential with probability 1. To make good use of this property, we position the distinguisher in round s from 23 to 25 and search for the value of $\kappa_{2,7}$ with $\kappa_{0,7} = 0x800000000, \kappa_{1,7} = 0x000000000, \kappa_{3,7} = \kappa_{0,7} \oplus \kappa_{1,7} \oplus \kappa_{2,7}$, such that $((0x00000000, 0x80000000), (0x80000000, 0x00000000), (0x80000000, 0x00000000), (0x00000000, 0x80000000))$ is an 2-round RT_3^4-IBDs. This enables us to extend the distinguisher to the full rounds. Specifically, we set only 1 bit active in $\kappa_{2,7}$. The search space size is 32.

-Results. We found two 2-round RT_3^4-IBDs for both GOST-FB and GOST-PS within 5 minutes. Specifically, it requires $\kappa_{2,7} = 0x40000000$ or $\kappa_{2,7} = 0x20000000$. These two RT_3^4-IBDs can be extended to full-round distinguishers for both GOST-FB and GOST-PS.

-Comparison. (i) To date, no existing methods have been able to search for RK-IBDs on GOST due to its use of modular additions. Our results present the **first full-round RK-IBDs** on GOST. As our distinguisher is full-round, we do not compare it with other distinguishers anymore.

6.3 An applied case of derived IBDs for an RK-IBA on SKINNYee

The overview of the IBA is shown in Fig 10. Specifically, assume we have an r_d-round (RK)-IBD $(\alpha, \alpha', \beta, \beta')$, then we propagate α, α' r_b-round backward and propagate β, β' r_f-round forward, to get the sets Δ_{in}, Δ'_{in}, Δ_{out}, Δ'_{out} of plaintext or ciphertext differences. To mount an attack, we construct the plaintext-ciphertext quartets $Q = \{(P_0, C_0), (P_1, C_1), (P_2, C_2), (P_3, C_3)\}$ that satisfy the differeces within Δ_{in}, Δ'_{in}, Δ_{out}, Δ'_{out}. Then, by guessing necessary key and partially encrypting the plaintext or decrypting the ciphertext, we can filter out incorrect keys by IBDs, thus achieving the goal of recovering the key.

In [15], the authors proposed 22-round RK-IBDs of SKINNYee, and a 29-round on SKINNYee based on the 22-round RK-IBD. As we find 23-round RK-IBDs, we choose the Distinguisher 8, one of the 23-round RK-IBDs, and add 4 rounds before the distinguisher and 4 rounds after the distinguisher to mount a 31-round IBA on SKINNYee, where the data complexity is 2^{66}, the time complexity is $2^{126.6}$, and the memory complexity is 2^{104}. Moreover, in the beyond full-codebook setting [12], the data complexity, time complexity and memory complexity of 31-round IBA on SKINNYee is $2^{67.01}$, $2^{123.68}$, and 2^{104} respectively. The details of the attack is given in Appendix H. As far as we known, this is the **first 31-round attack on SKINNYee and the optimized attack up to date**.

7 Conclusion and Future Work

In this paper, we explore complete construct theory and hierarchical apply strategy of IBDs in both single-key and related-key settings from aspects of differential, state and generalized BTs. Additionally, we develop a SAT-based tool

with novel strategies to automatically search for IBDs on various block ciphers, including SPN, Feistel, and ARX designs. The results obtained for the first time demonstrate that our approach overcomes all limitations of current search methods for IBDs and further reveals that IBDs can cover more rounds than IDs in many block ciphers. Consequently, resistance against IBA becomes a crucial consideration in block cipher designs.

It should be noted that our work only focuses on the search method for basic IBDs with two input differences and two output differences. A more generalized boomerang distinguisher can involve multiple input differences and output differences; however, this aspect remains to be explored in future research. This study also proposes to delve deeper into the integration of existing IBD construct methods with the key recovery framework.

References

1. Eli Biham, Alex Biryukov, and Adi Shamir. Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In Jacques Stern, editor, *Advances in Cryptology - EUROCRYPT '99, International Conference on the Theory and Application of Cryptographic Techniques, Prague, Czech Republic, May 2-6, 1999, Proceeding*, volume 1592 of *Lecture Notes in Computer Science*, pages 12–23. Springer, 1999.
2. Lars R. Knudsen. Deal - a 128-bit block cipher. *Complexity*, 1998.
3. David A. Wagner. The boomerang attack. In Lars R. Knudsen, editor, *Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings*, volume 1636 of *Lecture Notes in Computer Science*, pages 156–170. Springer, 1999.
4. Jiqiang Lu. Cryptanalysis of block ciphers. *PhD thesis, University of London UK, 2008*.
5. Jiqiang Lu. The (related-key) impossible boomerang attack and its application to the AES block cipher. *Des. Codes Cryptogr.*, 60(2):123–143, 2011.
6. Jiali Choy and Huihui Yap. Impossible boomerang attack for block cipher structures. In Tsuyoshi Takagi and Masahiro Mambo, editors, *Advances in Information and Computer Security, 4th International Workshop on Security, IWSEC 2009, Toyama, Japan, October 28-30, 2009, Proceedings*, volume 5824 of *Lecture Notes in Computer Science*, pages 22–37. Springer, 2009.
7. Sean Murphy. The return of the cryptographic boomerang. *IEEE Trans. Inf. Theory*, 57(4):2517–2521, 2011.
8. Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. In Tal Rabin, editor, *Advances in Cryptology - CRYPTO 2010, 30th Annual Cryptology Conference, Santa Barbara, CA, USA, August 15-19, 2010. Proceedings*, volume 6223 of *Lecture Notes in Computer Science*, pages 393–410. Springer, 2010.
9. Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. *J. Cryptol.*, 27(4):824–849, 2014.
10. Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. Boomerang connectivity table: A new cryptanalysis tool. In Jesper Buus Nielsen and Vincent Rijmen, editors, *Advances in Cryptology - EUROCRYPT 2018 - 37th Annual*

International Conference on the Theory and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II, volume 10821 of *Lecture Notes in Computer Science*, pages 683–714. Springer, 2018.
11. Haoyang Wang and Thomas Peyrin. Boomerang switch in multiple rounds. application to AES variants and deoxys. *IACR Trans. Symmetric Cryptol.*, 2019(1):142–169, 2019.
12. Jianing Zhang, Haoyang Wang, and Deng Tang. Impossible boomerang attacks revisited applications to deoxys-bc, joltik-bc and SKINNY. *IACR Trans. Symmetric Cryptol.*, 2024(2):254–295, 2024.
13. Hosein Hadipour, Nasour Bagheri, and Ling Song. Improved rectangle attacks on skinny and craft. *IACR Transactions on Symmetric Cryptology*, 2021(2):140–198, Jun. 2021.
14. Qianqian Yang, Ling Song, Siwei Sun, Danping Shi, and Lei Hu. New properties of the double boomerang connectivity table. *IACR Transactions on Symmetric Cryptology*, 2022(4):208–242, Dec. 2022.
15. 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.
16. Hamid Boukerrou, Paul Huynh, Virginie Lallemand, Bimal Mandal, and Marine Minier. On the feistel counterpart of the boomerang connectivity table introduction and analysis of the FBCT. *IACR Trans. Symmetric Cryptol.*, 2020(1):331–362, 2020.
17. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
18. Qian Wang and Chenhui Jin. Upper bound of the length of truncated impossible differentials for AES. *Des. Codes Cryptogr.*, 86(7):1541–1552, 2018.
19. Hosein Hadipour, Simon Gerhalter, Sadegh Sadeghi, and Maria Eichlseder. Improved search for integral, impossible differential and zero-correlation attacks application to ascon, forkskinny, skinny, mantis, PRESENT and qarmav2. *IACR Trans. Symmetric Cryptol.*, 2024(1):234–325, 2024.
20. Xichao Hu, Yongqiang Li, Lin Jiao, Shizhu Tian, and Mingsheng Wang. Mind the propagation of states - new automatic search tool for impossible differentials and impossible polytopic transitions. In Shiho Moriai and Huaxiong Wang, editors, *Advances in Cryptology - ASIACRYPT 2020 - 26th International Conference on the Theory and Application of Cryptology and Information Security, Daejeon, South Korea, December 7-11, 2020, Proceedings, Part I*, volume 12491 of *Lecture Notes in Computer Science*, pages 415–445. Springer, 2020.
21. Xichao Hu, Yongqiang Li, Lin Jiao, Shizhu Tian, and Mingsheng Wang. Mind the propagation of states new automatic search tool for impossible differentials and impossible polytopic transitions (full version). *IACR Cryptol. ePrint Arch.*, page 1093, 2020.
22. Hosein Hadipour, Sadegh Sadeghi, and Maria Eichlseder. Finding the impossible: Automated search for full impossible-differential, zero-correlation, and integral attacks. In Carmit Hazay and Martijn Stam, editors, *Advances in Cryptology - EUROCRYPT 2023 - 42nd Annual International Conference on the Theory and Applications of Cryptographic Techniques, Lyon, France, April 23-27, 2023, Proceedings, Part IV*, volume 14007 of *Lecture Notes in Computer Science*, pages 128–157. Springer, 2023.
23. U.S. Department of Commerce National Bureau of Standards. Data encryption standard. *Federal Information Processing Standards Publication 46*, 1977.

24. Andrey Bogdanov, Lars R. Knudsen, Gregor Leander, Christof Paar, Axel Poschmann, Matthew J. B. Robshaw, Yannick Seurin, and C. Vikkelsoe. PRESENT: an ultra-lightweight block cipher. In Pascal Paillier and Ingrid Verbauwhede, editors, *Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings*, volume 4727 of *Lecture Notes in Computer Science*, pages 450–466. Springer, 2007.
25. Lars R. Knudsen, Gregor Leander, Axel Poschmann, and Matthew J. B. Robshaw. Printcipher: A block cipher for ic-printing. In Stefan Mangard and François-Xavier Standaert, editors, *Cryptographic Hardware and Embedded Systems, CHES 2010, 12th International Workshop, Santa Barbara, CA, USA, August 17-20, 2010. Proceedings*, volume 6225 of *Lecture Notes in Computer Science*, pages 16–32. Springer, 2010.
26. Ray Beaulieu, Douglas Shors, Jason Smith, Stefan Treatman-Clark, Bryan Weeks, and Louis Wingers. The SIMON and SPECK lightweight block ciphers. In *Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015*, pages 175:1–175:6. ACM, 2015.
27. Christof Beierle, Jérémy Jean, Stefan Kölbl, Gregor Leander, Amir Moradi, Thomas Peyrin, Yu Sasaki, Pascal Sasdrich, and Siang Meng Sim. The SKINNY family of block ciphers and its low-latency variant MANTIS. In Matthew Robshaw and Jonathan Katz, editors, *Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II*, volume 9815 of *Lecture Notes in Computer Science*, pages 123–153. Springer, 2016.
28. Yusuke Naito, Yu Sasaki, and Takeshi Sugawara. Secret can be public: Low-memory AEAD mode for high-order masking. In Yevgeniy Dodis and Thomas Shrimpton, editors, *Advances in Cryptology - CRYPTO 2022 - 42nd Annual International Cryptology Conference, CRYPTO 2022, Santa Barbara, CA, USA, August 15-18, 2022, Proceedings, Part III*, volume 13509 of *Lecture Notes in Computer Science*, pages 315–345. Springer, 2022.
29. Subhadeep Banik, Sumit Kumar Pandey, Thomas Peyrin, Yu Sasaki, Siang Meng Sim, and Yosuke Todo. GIFT: A small present - towards reaching the limit of lightweight encryption. In Wieland Fischer and Naofumi Homma, editors, *Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings*, volume 10529 of *Lecture Notes in Computer Science*, pages 321–345. Springer, 2017.
30. Dongyoung Roh, Bonwook Koo, Younghoon Jung, Ilwoong Jeong, Donggeon Lee, Daesung Kwon, and Woo-Hwan Kim. Revised version of block cipher CHAM. In Jae Hong Seo, editor, *Information Security and Cryptology - ICISC 2019 - 22nd International Conference, Seoul, South Korea, December 4-6, 2019, Revised Selected Papers*, volume 11975 of *Lecture Notes in Computer Science*, pages 1–19. Springer, 2019.
31. Vasily Dolmatov. GOST 28147-89: Encryption, Decryption, and Message Authentication Code (MAC) Algorithms. RFC 5830, March 2010.
32. Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the fastest boomerangs application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.
33. Chenmeng Li, Baofeng Wu, and Dongdai Lin. Generalized boomerang connectivity table and improved cryptanalysis of GIFT. In Yi Deng and Moti Yung, editors, *Information Security and Cryptology - 18th International Conference, Inscrypt 2022,*

Beijing, China, December 11-13, 2022, Revised Selected Papers, volume 13837 of *Lecture Notes in Computer Science*, pages 213–233. Springer, 2022.
34. Ling Song, Xianrui Qin, and Lei Hu. Boomerang connectivity table revisited. application to SKINNY and AES. *IACR Trans. Symmetric Cryptol.*, 2019(1):118–141, 2019.
35. Dachao Wang, Baocang Wang, and Siwei Sun. Sat-aided automatic search of boomerang distinguishers for ARX ciphers. *IACR Trans. Symmetric Cryptol.*, 2023(1):152–191, 2023.
36. Yu Sasaki and Yosuke Todo. New impossible differential search tool from design and cryptanalysist aspects - revealing structural properties of several ciphers. In Jean-Sébastien Coron and Jesper Buus Nielsen, editors, *Advances in Cryptology - EUROCRYPT 2017 - 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30 - May 4, 2017, Proceedings, Part III*, volume 10212 of *Lecture Notes in Computer Science*, pages 185–215, 2017.
37. Ahmed Abdelkhalek, Yu Sasaki, Yosuke Todo, Mohamed Tolba, and Amr M. Youssef. MILP modeling for (large) s-boxes to optimize probability of differential characteristics. *IACR Trans. Symmetric Cryptol.*, 2017(4):99–129, 2017.
38. Jiqiang Lu, Jongsung Kim, Nathan Keller, and Orr Dunkelman. Improving the efficiency of impossible differential cryptanalysis of reduced camellia and MISTY1. In Tal Malkin, editor, *Topics in Cryptology - CT-RSA 2008, The Cryptographers' Track at the RSA Conference 2008, San Francisco, CA, USA, April 8-11, 2008. Proceedings*, volume 4964 of *Lecture Notes in Computer Science*, pages 370–386. Springer, 2008.

Table of Contents

A Constructing IBDs from the Aspect of generalized BTs for Other Block Cipher Networks

A.1 Constructing IBDs based on generalized BTs for Feistel ciphers

To apply boomerang switch for Feistel ciphers, boomerang tables are proposed accordingly.

Definition 14 ([16]). *Given four differences $\gamma, \theta, \lambda, \delta \in \mathbb{F}_2^n$, the FBCT, BDT and FBET for an n-bit S-box are defined as*

$$\mathrm{FBCT}(\gamma,\delta) = \#\left\{ x \in \mathbb{F}_2^n \middle| S(x) \oplus S(x\oplus\gamma) \oplus S(x\oplus\delta) \oplus S(x\oplus\gamma\oplus\delta) = 0 \right\},$$

$$\mathrm{BDT}(\gamma,\theta,\delta) = \#\left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x\oplus\gamma) \oplus S(x\oplus\delta) \oplus S(x\oplus\gamma\oplus\delta) = 0, \\ S(x) \oplus S(x\oplus\gamma) = \theta \end{array} \right\},$$

$$\mathrm{FBET}(\gamma,\theta,\delta,\lambda) = \#\left\{ x \in \mathbb{F}_2^n \middle| \begin{array}{l} S(x) \oplus S(x\oplus\gamma) \oplus S(x\oplus\delta) \oplus S(x\oplus\gamma\oplus\delta) = 0, \\ S(x) \oplus S(x\oplus\gamma) = \theta, \\ S(x\oplus\gamma) \oplus S(x\oplus\gamma\oplus\delta) = \lambda \end{array} \right\}.$$

The incompatibility of BDs resulting from the dependence in Feistel ciphers was observed by Boukerrou et al. [16]. To address this problem, they extended the BCT and BDT to Feistel and introduced FBCT, FBDT, and FBET. To construct IBDs, we generalize FBCT and FBDT to define GFBCT, GFUBCT, and GFLBCT. Additionally, the generalized table of FEBCT is defined identically to that of GEBCT.

Definition 15. *Given nine differences $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi' \in \mathbb{F}_2^n$ where $\rho' = \mu \oplus \mu' \oplus \rho$, $\varphi' = \theta \oplus \theta' \oplus \varphi$), the GFBCT, GFUBCT and GFLBCT for an n-bit S-box is defined as*

$\mathrm{GFBCT}(\mu,\mu',\rho,\rho',\eta) = \#\{(x_0,x_1,x_2,x_3) \in \{0,1\}^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_2 \oplus x_3 = \rho', \oplus_{i=0}^{3} S(x_i) = \eta\}$,

$\mathrm{GFUBCT}(\mu,\mu',\rho,\rho',\theta,\theta',\eta) = \#\{(x_0,x_1,x_2,x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_2 \oplus x_3 = \rho', \oplus_{i=0}^{3} S(x_i) = \eta, S(x_0) \oplus S(x_1) = \theta, S(x_2) \oplus S(x_3) = \theta'\}$,

$\mathrm{GFLBCT}(\mu,\mu',\rho,\rho',\varphi,\varphi',\eta) = \#\{(x_0,x_1,x_2,x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_2 \oplus x_3 = \rho', \oplus_{i=0}^{3} S(x_i) = \eta, S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.

A schematic diagram for these generalized BTs is shown in Fig 11.

To illustrate the relationships among these generalized tables, we introduce the following notations.

Notation 1. *For an n-bit Feistel cipher E, let $x = (x_L \| x_R) \in \mathbb{F}_2^n$ be the state of E, where x_L and x_R denote the left and right branches, respectively. Let $\gamma, \gamma', \eta, \eta' \in \mathbb{F}_2^n$ be four input differences and $\omega, \omega', \delta, \delta', \in \mathbb{F}_2^n$ be four output differences of E. Then,*

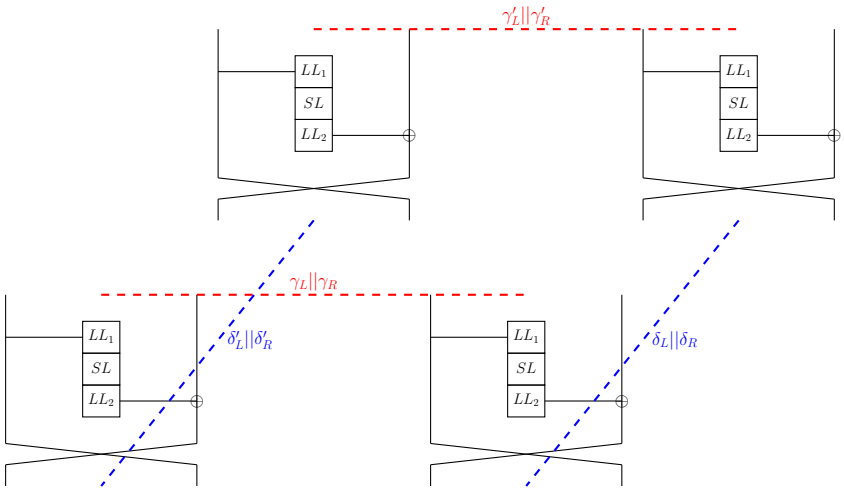

Fig. 11: The illustration of differential propagation rule through S-boxes in E based on GFBCT

1. $(\gamma, \gamma') \xrightarrow{GFBCT} (\delta, \delta')$ *indicates that the propagation rule through S-boxes follows GFBCT. Specifically, for an* E *as shown in Fig 11, let* $\gamma = \gamma_L||\gamma_R$, $\gamma' = \gamma'_L||\gamma'_R$, $\delta = \delta_L||\delta_R$, $\delta' = \delta'_L||\delta'_R$, *let* $LL_1(\gamma_L) = (a_{t-1}, \ldots, a_0)$, $LL_1(\gamma_{L'}) = (a'_{t-1}, \ldots, a'_0)$, $LL_1(\delta_R) = (b_{t-1}, \ldots, b_0)$, $LL_1(\delta'_R) = (b'_{t-1}, \ldots, b'_0)$, *and let* $LL_2^{-1}(\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L) = (c_{t-1}, \ldots, c_0)$, *where* LL_i *for* $i = 1, 2$ *are linear layers and there are* t *S-boxes in* SL. *Thus,* a_i, a'_i, b_i, b'_i *and* c_i *are the differences corresponding to GFBCT of the* i*-th S-box for* $i = 0, \ldots, t-1$. *Then,* $(\gamma, \gamma') \xrightarrow{GFBCT} (\delta, \delta')$ *is equivalent to that there exists* $i \in \{0, \ldots, t-1\}$ *such that* $\mathrm{GFBCT}(a_i, a'_i, b_i, b'_i, c_i) \neq 0$.
2. $(\gamma, \gamma') \xrightarrow{GFUBCT} (\omega, \omega', \delta, \delta')$ *indicates that the propagation rule through S-boxes follows GFUBCT, equivalent to* $(\gamma, \gamma') \xrightarrow{GFBCT} (\delta, \delta')$ *and* $(\gamma, \gamma') \xrightarrow{UDDT} (\omega, \omega')$.
3. $(\gamma, \gamma', \eta, \eta') \xrightarrow{GFLBCT} (\delta, \delta')$ *indicates that the propagation rule through S-boxes follows GFLBCT, equivalent to* $(\gamma, \gamma') \xrightarrow{GFBCT} (\delta, \delta')$ *and* $(\eta, \eta') \xrightarrow{LDDT} (\delta, \delta')$.
4. $(\gamma, \gamma', \eta, \eta') \xrightarrow{GEBCT} (\omega, \omega', \delta, \delta')$ *indicates that the propagation rule through S-boxes follows GEBCT, equivalent to* $(\gamma, \gamma') \xrightarrow{GFUBCT} (\omega, \omega', \delta, \delta')$ *and* $(\gamma, \gamma', \eta, \eta') \xrightarrow{GFLBCT} (\delta, \delta')$.

Proposition 3. UDDT, LDDT, GFBCT, GFUBCT, GFLBCT, GEBCT *have the following relations:*

1. *If* $(\gamma, \gamma') \xrightarrow{GFBCT} (\delta, \delta')$, *then* $\exists \eta, \eta', \omega, \omega'$, *s.t.* $(\gamma, \gamma') \xrightarrow{UDDT} (\omega, \omega')$ *and* $(\eta, \eta') \xrightarrow{LDDT} (\delta, \delta')$.

2. *If* $(\gamma,\gamma') \xrightarrow{GFUBCT} (\omega,\omega',\delta,\delta')$, *then* $\exists\eta,\eta'$, *s.t.* $(\gamma,\gamma') \xrightarrow{UDDT} (\omega,\omega')$ *and* $(\eta,\eta') \xrightarrow{LDDT} (\delta,\delta')$.
3. *If* $(\gamma,\gamma',\eta,\eta') \xrightarrow{GFLBCT} (\delta,\delta')$, *then* $\exists\omega,\omega'$, *s.t.* $(\gamma,\gamma') \xrightarrow{UDDT} (\omega,\omega')$ *and* $(\eta,\eta') \xrightarrow{LDDT} (\delta,\delta')$.
4. *If* $(\gamma,\gamma',\eta,\eta') \xrightarrow{GEBCT} (\omega,\omega',\delta,\delta')$, *then* $(\gamma,\gamma') \xrightarrow{UDDT} (\omega,\omega')$, $(\eta,\eta') \xrightarrow{LDDT} (\delta,\delta')$, $(\gamma,\gamma') \xrightarrow{GFUBCT} (\omega,\omega',\delta,\delta')$ *and* $(\gamma,\gamma',\eta,\eta') \xrightarrow{GFLBCT} (\delta,\delta')$.

We also consider a hybrid use of UDDT, LDDT, GFBCT, GFUBCT, GFLBCT and GEBCT to construct IBDs. At first, we define a set of propagation rules for a Feistel cipher E with t S-boxes $(S_0,\ldots,S_{t-1})$ in total as $\mathcal{AP}_E^F = \{(p_0,\ldots,p_{t-1})|\ p_i \in \{\mathrm{UDDT},\mathrm{LDDT},\mathrm{GFBCT},\mathrm{GFUBCT},\ \mathrm{GFLBCT},\ \mathrm{GEBCT}\}\}$. Then $P = (p_0,\ldots,p_{t-1}) \in \mathcal{AP}_E^F$, denotes that the propagation rule through the i-th S-box follows p_i.

Definition 16. *Let* $E = E_{r-1,rk_{r-1}} \circ \cdots \circ E_{0,rk_0}(x)$ *be an* r*-round Feistel cipher. Let* $P = (P_0,\ldots,P_{r-1})$ *be a predefined propagation rule of* E*, where* $P_i \in \mathcal{AP}^F_{E_{i,rk_i}}$ *denotes a propagation rule of* E_{i,rk_i} *for* $i \in \{0,\ldots,r-1\}$*. Let* $\epsilon_0^i,\epsilon_1^i,\epsilon_2^i,\epsilon_3^i$ *be the four input differences and* $\epsilon_0^{i+1},\epsilon_1^{i+1},\epsilon_2^{i+1},\epsilon_3^{i+1}$ *be the four output differences of the round function* E_{i,rk_i} *for* $i \in \{0,\ldots,r-1\}$*. For two input differences* α,α' *and two output differences* β,β' *of the block cipher* E*, if there exists a trail*

$$(\epsilon_0^0=\alpha,\epsilon_1^0,\epsilon_2^0=\alpha',\epsilon_3^0) \xrightarrow{P_0} \cdots \xrightarrow{P_{r-1}} (\epsilon_0^r,\epsilon_1^r=\beta,\epsilon_2^r,\epsilon_3^r=\beta'),$$

then it is called an r*-round* T_P^F *boomerang trail. Here,* $\xrightarrow{P_i}$ *represents that the propagation rule through S-boxes in* E_{i,rk_i} *follows* P_i*.*

Accordingly, we have the following construction.

Construction 9 (T_P^F-IBD). *Given an* r*-round Feistel cipher* E *and a predefined rule* $P \in \mathcal{AP}_E^F$*, for two input differences* α,α' *and two output differences* β,β'*, if no* r*-round* T_P^F *boomerang trail exists, then* $(\alpha,\alpha',\beta,\beta')$ *is an IBD, called an* r*-round* T_P^F*-IBD.*

T_1-IBD is a special example of T_P^S-IBD.

Theorem 7. *For any predefined rule* $P \in \mathcal{AP}_E^F$*, an* r*-round* T_1*-IBD* $(\alpha,\alpha',\beta,\beta')$ *is an* r*-round* T_P^F*-IBD.*

Proposition 3 demonstrates that GEBCT in Feistel block ciphers has a similar status in SPN ciphers. Additionally, the definition of T_C-IBD is also applicable to Feistel ciphers. Thus, we have the following inclusion relationship.

Theorem 8. *For any predefined rule* $P \in \mathcal{AP}_E^F$*, an* r*-round* T_P^F*-IBD* $(\alpha,\alpha',\beta,\beta')$ *is an* r*-round* T_C*-IBD.*

In addition to the relationship given in Theorem 6, we can prove that the definition of T_C-IBD is equivalent to that of T_2-IBD for Feistel ciphers with full-left-branch-state round-key-addition operations. A schematic diagram is shown in Fig 17.

Proposition 4. *Given an Feistel cipher with full-left-branch-state round-key-addition operations,* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* T_C*-IBD if and only if it is an* r*-round* T_2*-IBD.*

A.2 Constructing IBDs based on generalized BTs for ARX ciphers

For ARX ciphers, instead of considering the modular addition as a $2n$-bit to n-bit mapping, we regard it as an $2n \times 2n$-bit S-box, inspired by [35]. As Illustrated in Fig 12(a), this S-box is defined as $S(x\|y) = (x \boxplus y, y)$. Thus, the definitions of these S-box based generalized BTs still work for modular addition. For example, $\mathrm{BCT}_{\boxplus}(\alpha_l, \alpha_r, \beta_l, \beta_r) = \#\{(x, y) \in \mathbb{F}_2^n \times \mathbb{F}_2^n \mid (((x \boxplus y) \oplus \beta_l) \boxminus (y \oplus \beta_r)) \oplus ((((x \oplus \alpha_l) \boxplus (y \oplus \alpha_r)) \oplus \beta_l) \boxminus (y \oplus \alpha_r \oplus \beta_r)) = \alpha_l\}$. In doing so, the IBD construction methods based on our generalized BTs apply to ARX ciphers as well. For example, the `SPECK` cipher can be viewed as a SPN cipher when modular addition is treated as an S-box, with round keys XORed with part of the state, as shown in Fig 12(b).

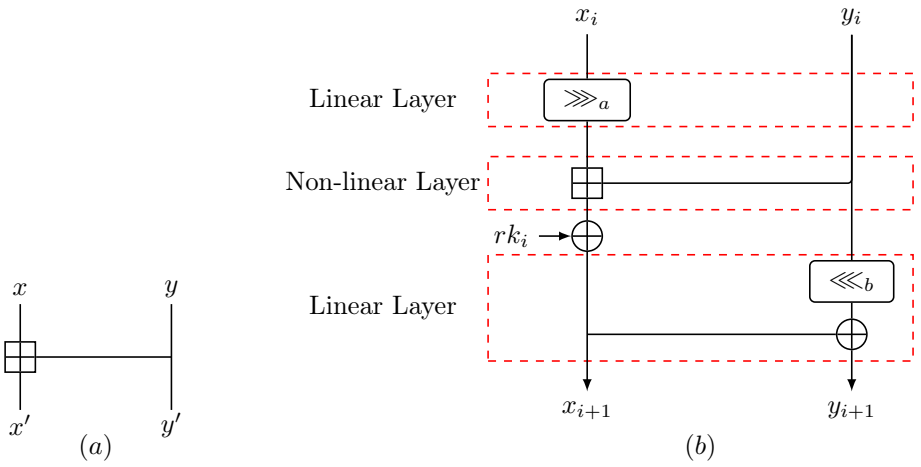

Fig. 12: The SPN cipher perspective of ARX cipher (`SPECK`)

Additionally, for non-typical SPN or Feistel like ARX ciphers, we can use equivalent transformations to convert them into SPN or Feistel ciphers, such as `CHAM`-like cipher shown in Fig 13 and `GOST`-like cipher shown in Fig 14. Thus, the IBD construction methods based on our generalized BTs are still valid. Similar to SPN and Feistel ciphers, we denote the IBD constructed based on a mixed use of generalized BTs as T_P^A-IBD, and the IBD constructed based on GEBCT as T_C-IBD for ARX ciphers.

Therefore, the relationships among these construction methods are naturally generalized to ARX ciphers as well.

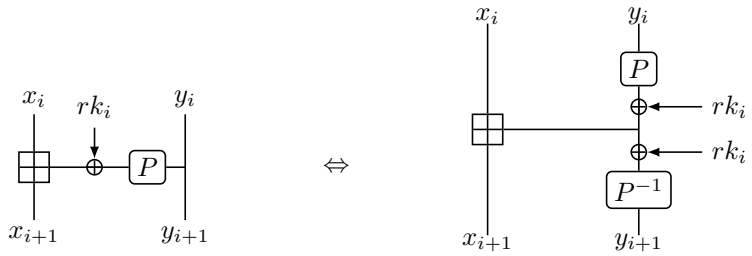

Fig. 13: The SPN cipher perspective of modular addition (CHAM)

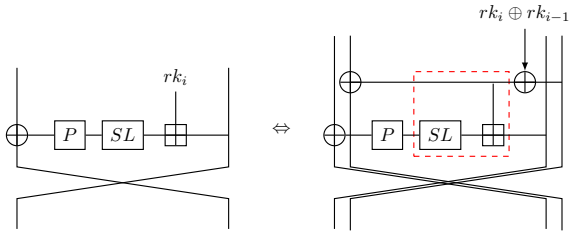

Fig. 14: The Feistel cipher perspective of modular addition (GOST)

B Proofs

B.1 Proof of Theorem 1

Theorem 1

Proof (proof by contradiction). If an r-round T_0-IBD $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_1-*IBD*, there must exist one r-round T_1 boomerang trail:

$$(\alpha, \alpha') \to \cdots \to \underbrace{(\gamma, \gamma')(\delta, \delta')}_{\gamma \oplus \gamma' \oplus \delta \oplus \delta' = 0} \to \cdots \to (\beta, \beta').$$

But, an r-round T_1 boomerang trail is an r-round T_0 boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_0-*IBD*. □

B.2 Proof of Theorem 2

Theorem 2

Proof (proof by contradiction). If an r-round T_2-IBD $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_3-*IBD*, there must exist one r-round T_3 boomerang trail: $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$, where $x_j^{i+1} = E_{i,\mathrm{KS}_i(k)}(x_j^i)$ for $0 \le i \le r-1, 0 \le j \le 4$. But, an r-round T_3 boomerang trail is an r-round T_2 boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_2-*IBD*. □

B.3 Proof of Theorem 3

Theorem 3

Proof. (Definition 3⇒ Construction 4) Let $(\alpha, \alpha', \beta, \beta')$ be an r-round IBD, then any pair of plaintexts (x_0, x_3) cannot simultaneously satisfy $E_k(x_0) \oplus E_k(x_3) = \beta$ and $E_k(x_0 \oplus \alpha) \oplus E_k(x_3 \oplus \alpha') = \beta'$. If $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_3-IBD. Let $x_0^0 = x_0$, $x_1^0 = x_0 \oplus \alpha$, $x_3^0 = x_3$, $x_2^0 = x_3 \oplus \alpha'$, there exist an r-round T_3 boomerang trail $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$, where $x_1^r \oplus x_2^r = \beta$ and $x_0^r \oplus x_3^r = \beta'$. Thus $E_k(x_0) \oplus E_k(x_3) = \beta$ and $E_k(x_0 \oplus \alpha) \oplus E_k(x_3 \oplus \alpha') = \beta'$, which is a contradiction.
(Construction 4⇒ Definition 3) Let $(\alpha, \alpha', \beta, \beta')$ be an r-round T_3-IBD. then there is not any r-round T_3 boomerang trail $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$. Thus, any pair of (x_0^0, x_3^0) cannot simultaneously meet $E_k(x_0^0) \oplus E_k(x_3^0) = \beta$ and $E_k(x_0^0 \oplus \alpha) \oplus E_k(x_3^0 \oplus \alpha') = \beta'$, which is according with Definition 3. □

B.4 Proof of Theorem 4

Theorem 4

Proof (proof by contradiction). If $\exists P \in \mathcal{AP}_E^S$ and an r-round T_1-IBD $(\alpha, \alpha', \beta, \beta')$ such that it is not an r-round T_P^S-IBD, there must exist at least one r-round T_P^S boomerang trail: $(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{P_0} \cdots \xrightarrow{P_{r-1}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta')$. Based on the relations of tables in Proposition 1, it is also an r-round T_1-IBD boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_P^S-*IBD*. □

B.5 Proof of Theorem 5

Theorem 5

Proof (proof by contradiction). If an r-round T_P^S-IBD $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_C-*IBD*, there must exist at least one r-round T_C boomerang trail: $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \xrightarrow{GEBCT} \cdots \xrightarrow{GEBCT} (\epsilon_0^{r_0}, \epsilon_1^{r_0}, \epsilon_2^{r_0}, \epsilon_3^{r_0}) \xrightarrow{GEBCT} (\epsilon_0^{r_0+1}, \epsilon_1^{r_0+1}, \epsilon_2^{r_0+1}, \epsilon_3^{r_0+1}) \xrightarrow{GEBCT} \cdots \xrightarrow{GEBCT} (\epsilon_0^r, \epsilon_1^r, \epsilon_2^r, \epsilon_3^r)$. Based on the relations of tables in Proposition 1, it is also an r-round T_P^S boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_C-*IBD*. □

B.6 Proof of Theorem 6

Theorem 6

Proof (proof by contradiction). If an r-round T_C-IBD $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_2-*IBD*, there must exist one r-round T_2 boomerang trail: $(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r)$, where $x_0 \oplus x_1 = \alpha, x_2 \oplus x_3 = \alpha', y_1 \oplus y_2 = \beta, y_0 \oplus y_3 = \beta'$ and $x_j^{i+1} = E_{i,rk_i}(x_j^i)$ for $0 \le i \le r-1, 0 \le j \le 4$ and $(rk_0, \cdots, rk_{r-1}) \in (\mathbb{F}_2^l)^r$. For each operation in E_{i,rk_i}, we can obtain a trail that conform to its differential propagation rule by XORing the input and output states. Especially for

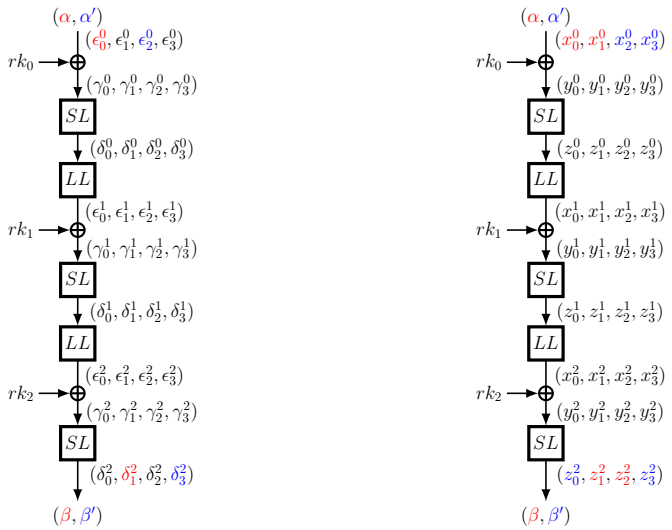

Fig. 15: The equivalence between T_C-IBD and T_2-IBD in SPN ciphers with full-state round-key-addition operations

the S-box operation, we can derive the four input differences and four output differences that are consistent with GEBCT accordingly. Thus, this r-round T_2 boomerang trail is also an r-round T_C boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_C-*IBD*. □

B.7 Proof of Proposition 2

Proposition 2

Proof. The schematic diagram is given in Fig 15. This is equivalent to prove that $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \overset{\text{AK}}{\to} (\gamma_0^0, \gamma_1^0, \gamma_2^0, \gamma_3^0) \overset{\text{GEBCT}}{\to} (\delta_0^0, \delta_1^0, \delta_2^0, \delta_3^0) \overset{\text{LL}}{\to} (\epsilon_0^1, \epsilon_1^1, \epsilon_2^1, \epsilon_3^1) \overset{\text{AK}}{\to} \cdots \overset{\text{GEBCT}}{\to} (\delta_0^{r-1}, \delta_1^{r-1}, \delta_2^{r-1}, \delta_3^{r-1})$ is an r-round T_C boomerang trail if and only if $(x_0^0, x_1^0, x_2^0, x_3^0) \overset{\text{AK}}{\to} (y_0^0, y_1^0, y_2^0, y_3^0) \overset{\text{SL}}{\to} (z_0^0, z_1^0, z_2^0, z_3^0) \overset{\text{LL}}{\to} (x_0^1, x_1^1, x_2^1, x_3^1) \overset{\text{AK}}{\to} \cdots \overset{\text{SL}}{\to} (z_0^{r-1}, z_1^{r-1}, z_2^{r-1}, z_3^{r-1})$ is an r-round T_2 boomerang trail, where $\alpha = \epsilon_0^0, \alpha' = \epsilon_2^0, \beta = \epsilon_1^{r-1}$, $\beta' = \epsilon_3^{r-1}$, and $\alpha = x_0^0 \oplus x_1^0, \alpha' = x_2^0 \oplus x_3^0$, $\beta = z_1^{r-1} \oplus z_2^{r-1}$ and $\beta' = z_0^{r-1} \oplus z_3^{r-1}$. In particular, we prove this in the case of $r = 3$. The other cases can be proved analogously. Suppose $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \overset{\text{AK}}{\to} (\gamma_0^0, \gamma_1^0, \gamma_2^0, \gamma_3^0) \overset{\text{GEBCT}}{\to} (\delta_0^0, \delta_1^0, \delta_2^0, \delta_3^0) \overset{\text{LL}}{\to} (\epsilon_0^1, \epsilon_1^1, \epsilon_2^1, \epsilon_3^1) \overset{\text{AK}}{\to} \cdots \overset{\text{GEBCT}}{\to} (\delta_0^2, \delta_1^2, \delta_2^2, \delta_3^2)$ is an 3-round T_C boomerang trail. Since $(\gamma_0^i, \gamma_1^i, \gamma_2^i, \gamma_3^i) \overset{\text{SL, GEBCT}}{\to} (\delta_0^i, \delta_1^i, \delta_2^i, \delta_3^i)$, there exists $(y_0^i, y_1^i, y_2^i, y_3^i)$ and $(z_0^i, z_1^i, z_2^i, z_3^i)$, such that

$$y_0^i \oplus y_1^i = \gamma_0^i, y_1^i \oplus y_2^i = \gamma_1^i, y_2^i \oplus y_3^i = \gamma_2^i, y_0^i \oplus y_3^i = \gamma_3^i,$$
$$z_0^i \oplus z_1^i = \delta_0^i, z_1^i \oplus z_2^i = \delta_1^i, z_2^i \oplus z_3^i = \delta_2^i, z_0^i \oplus z_3^i = \delta_3^i.$$

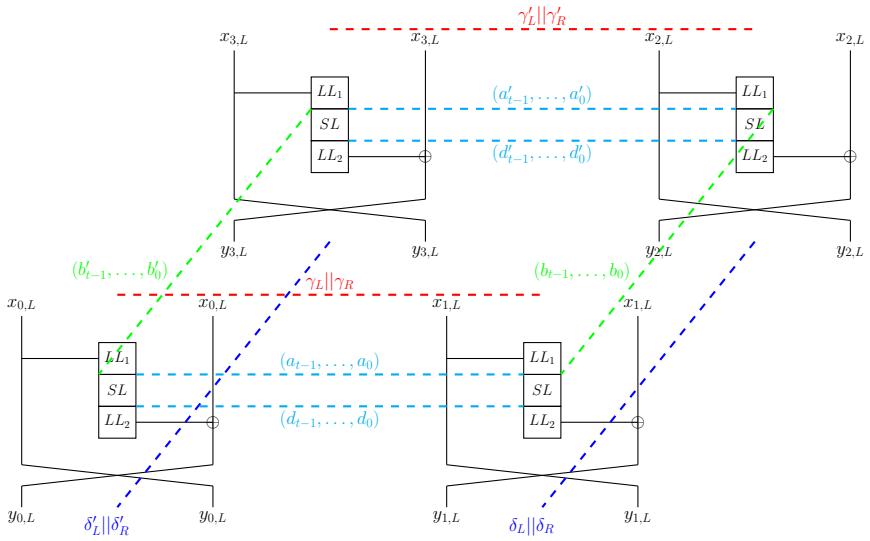

Fig. 16: The relation between GFBCT and UDDT

Let $x_i^0 = y_i^0 \oplus rk_0$ $(0 \le i \le 3)$ and $rk_j = LL(z_x^{j-1}) \oplus y_i^j$ $(0 \le i \le 3, j = 1, 2)$, then $(x_0^0, x_1^0, x_2^0, x_3^0) \xrightarrow{\text{AK}} (y_0^0, y_1^0, y_2^0, y_3^0) \xrightarrow{\text{SL}} (z_0^0, z_1^0, z_2^0, z_3^0) \xrightarrow{\text{LL}} (x_0^1, x_1^1, x_2^1, x_3^1) \xrightarrow{\text{AK}} \cdots \xrightarrow{\text{SL}} (z_0^{r-1}, z_1^{r-1}, z_2^{r-1}, z_3^{r-1})$ is an 3-round T_2 boomerang trail. The above process is invertible. □

B.8 Proof of Proposition 3

Proposition 3

Proof. As shown in Fig 16, let l_i be the linear function of $LL_i (i = 1, 2)$, $\gamma = \gamma_L||\gamma_R$, $\gamma' = \gamma'_L||\gamma'_R$, $\delta = \delta_L||\delta_R$, and $\delta' = \delta'_L||\delta'_R$. $(a_{t-1}, \ldots, a_0) = l_1(\gamma_L)$, $(a'_{t-1}, \ldots, a'_0) = l_1(\gamma_{L'})$, $(b_{t-1}, \ldots, b_0) = l_1(\delta_R)$, $(b'_{t-1}, \ldots, b'_0) = l_1(\delta'_R)$, and $(c_{t-1}, \ldots, c_0) = l_2^{-1}(\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L)$, where a_i, a'_i, b_i, b'_i and c_i are the input or the output difference of the S-box in SL and t is the number of the S-boxes in SL. For the input differences a_i and a'_i, according to the definition of GFBCT, there exist output differences d_i and d'_i, such that a_i propagates to d_i and a_i propagates to d_i, and $d_i \oplus d'_i = c_i (0 \le i \le t-1)$. Let $\lambda = l_2((d_{t-1}, \ldots, d_0))$ and $\lambda' = l_2((d'_{t-1}, \ldots, d'_0))$, then $\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L = \lambda \oplus \lambda'$. Therefore, there exist λ and λ' with $\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L = \lambda \oplus \lambda'$, such that the difference γ can propagate to the difference $(\gamma_R \oplus \lambda)||\gamma_L$, and the difference γ' can propagate to the difference $(\gamma'_R \oplus \lambda')||\gamma'_L$. That is, $(\gamma, \gamma') \xrightarrow{UDDT} ((\gamma_R \oplus \lambda)||\gamma_L, \gamma'_R \oplus \lambda')||\gamma'_L)$. For other relations, we can prove them similarly.

B.9 Proof of Theorem 7

Theorem 7

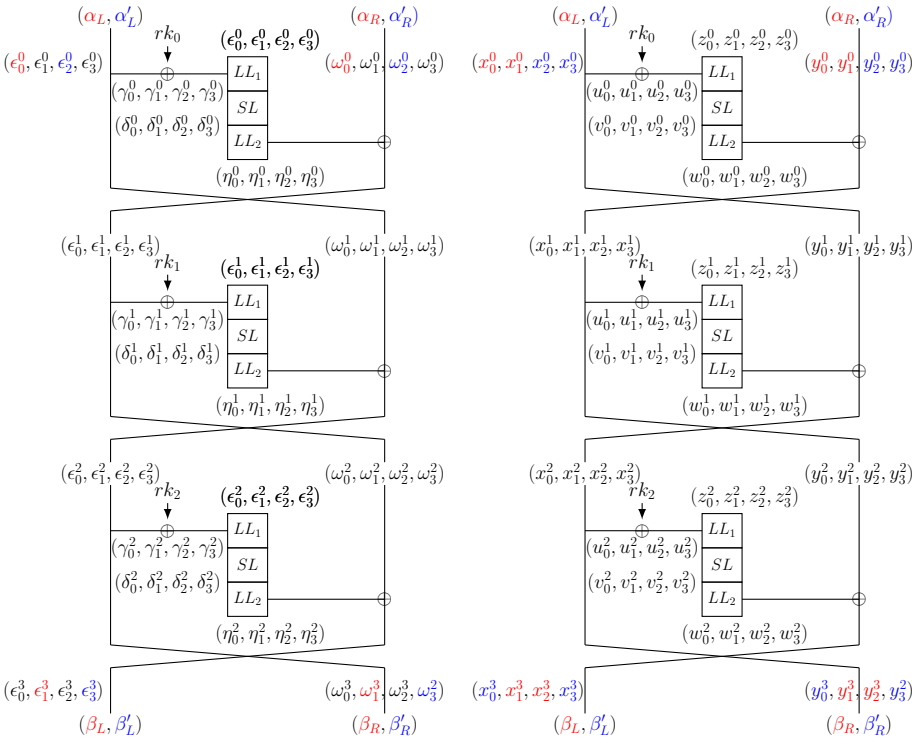

Fig. 17: The equivalence between T_C-IBD and T_2-IBD in Feistel cipher with full-left-branch-state round- key-addition operations

Proof (proof by contradiction). If $\exists P \in \mathcal{AP}_E^F$ and an r-round T_1-IBD $((\alpha, \alpha'), (\beta, \beta'))$ such that it is not an r-round T_P^F-IBD, there must exist at least one r-round T_P^F boomerang trail: $(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{P_0} \cdots \xrightarrow{P_{r-1}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta')$. Based on the Proposition 3, the T_P^F boomerang trail is also an r-round T_1-IBD boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_P^F-*IBD*. □

B.10 Proof of Theorem 8

Theorem 8

Proof (proof by contradiction). If an r-round T_P^F-IBD $(\alpha, \alpha', \beta, \beta')$ is not an r-round T_C-*IBD*, there must exist at least one r-round T_C boomerang trail: $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \xrightarrow{GEBCT} \cdots \xrightarrow{GEBCT} (\epsilon_0^{r_0}, \epsilon_1^{r_0}, \epsilon_2^{r_0}, \epsilon_3^{r_0}) \xrightarrow{GEBCT} (\epsilon_0^{r_0+1}, \epsilon_1^{r_0+1}, \epsilon_2^{r_0+1}, \epsilon_3^{r_0+1}) \xrightarrow{GEBCT} \cdots \xrightarrow{GEBCT} (\epsilon_0^r, \epsilon_1^r, \epsilon_2^r, \epsilon_3^r)$. Based on the Proposition 3, it is also an r-round T_P^S boomerang trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is neither an r-round T_C-*IBD*. □

B.11 Proof of Proposition 4

Proposition 4

Proof. As shown in Fig 17, this is equivalent to prove that $((\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0), (\omega_0^0, \omega_1^0, \omega_2^0, \omega_3^0)) \overset{\text{GEBCT}}{\rightarrow} \cdots \overset{\text{GEBCT}}{\rightarrow} \cdots \overset{\text{GEBCT}}{\rightarrow} ((\epsilon_0^r, \epsilon_1^r, \epsilon_2^r, \epsilon_3^r), (\omega_0^r, \omega_1^r, \omega_2^r, \omega_3^r))$ is an r-round T_C boomerang trail if and only if $((x_0^0, x_1^0, x_2^0, x_3^0), (y_0^0, y_1^0, y_2^0, y_3^0)) \rightarrow \cdots \rightarrow \cdots \rightarrow ((x_0^r, x_1^r, x_2^r, x_3^r), (y_0^r, y_1^r, y_2^r, y_3^r))$ is an r-round T_2 boomerang trail, where $(\alpha_L, \alpha_R) = (\epsilon_0^0, \omega_0^0), (\alpha'_L, \alpha'_R) = (\epsilon_2^0, \omega_2^0), (\beta_L, \beta_R) = (\epsilon_1^r, \omega_1^r)$, and $(\beta'_L, \beta'_R) = (\epsilon_3^r, \omega_3^r)$, and $(\alpha_L, \alpha_R) = (x_0^0 \oplus x_1^0, y_0^0 \oplus y_1^0), (\alpha'_L, \alpha'_R) = (x_2^0 \oplus x_3^0, y_2^0 \oplus y_3^0), (\beta_L, \beta_R) = (x_0^r \oplus x_2^r, y_0^r \oplus y_2^r)$, and $(\beta'_L, \beta'_R) = (x_1^r \oplus x_3^r, y_1^r \oplus y_3^r)$. In particular, we prove this in the case of $r = 3$. The other cases can be proved analogously. Suppose $((\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0), (\omega_0^0, \omega_1^0, \omega_2^0, \omega_3^0)) \overset{\text{GEBCT}}{\rightarrow} \cdots \overset{\text{GEBCT}}{\rightarrow} \cdots \overset{\text{GEBCT}}{\rightarrow} ((\epsilon_0^3, \epsilon_1^3, \epsilon_2^3, \epsilon_3^3), (\omega_0^3, \omega_1^3, \omega_2^3, \omega_3^3))$ is an 3-round T_C boomerang trail. For $0 \leq i \leq 2$, since $(\gamma_0^i, \gamma_1^i, \gamma_2^i, \gamma_3^i) \overset{\text{SL, GEBCT}}{\rightarrow} (\delta_0^i, \delta_1^i, \delta_2^i, \delta_3^i)$, there exists $(u_0^i, u_1^i, u_2^i, u_3^i)$ and $(v_0^i, v_1^i, v_2^i, v_3^i)$, such that

$$u_0^i \oplus u_1^i = \gamma_0^i, u_1^i \oplus u_2^i = \gamma_1^i, u_2^i \oplus u_3^i = \gamma_2^i, u_0^i \oplus u_3^i = \gamma_3^i,$$
$$v_0^i \oplus v_1^i = \delta_0^i, v_1^i \oplus v_2^i = \delta_1^i, v_2^i \oplus v_3^i = \delta_2^i, v_0^i \oplus v_3^i = \delta_3^i.$$

Let l_j be the linear function of $LL_j (j = 1, 2)$, $(z_0^i, z_1^i, z_2^i, z_3^i) = l_1^{-1}(u_0^i, u_1^i, u_2^i, u_3^i)$, and $(w_0^i, w_1^i, w_2^i, w_3^i) = l_2(v_0^i, v_1^i, v_2^i, v_3^i)$. Then $z_0^i \oplus z_1^i = x_0^i \oplus x_1^i$, $z_1^i \oplus z_2^i = x_1^i \oplus x_2^i$, $z_2^i \oplus z_3^i = x_2^i \oplus x_3^i$, and $z_0^i \oplus z_3^i = x_0^i \oplus x_3^i$. Let $rk_i = x_0^i \oplus z_0^i$, then $rk_i = x_j^i \oplus z_j^i (0 \leq j \leq 3)$. Let $x_j^{i+1} = w_j^i \oplus y_j^i (0 \leq j \leq 3)$, then $((x_0^0, x_1^0, x_2^0, x_3^0), (y_0^0, y_1^0, y_2^0, y_3^0)) \rightarrow \cdots \rightarrow \cdots \rightarrow ((x_0^3, x_1^3, x_2^3, x_3^3), (y_0^3, y_1^3, y_2^3, y_3^3))$ is an r-round T_2 boomerang trail. The above process is invertible. □

C The Algorithm of Searching for (RK-)IBDs

C.1 Scheduling algorithm for searching (RK-)IBDs based on hierarchical apply strategy

A brief illustration is as follows.

- For the input, the set $\mathcal{SD}$ consists of the (RK-)IBDs that are permitted to be searched. This set can include all types or a subset of (RK-)IBDs.
- Line 4: We employ multi-process technology to measure the time for each (RK-)IBD type within $\mathcal{SD}$. If the size of $\mathcal{SD}$ exceeds $\mathcal{N}_{max}$, we process these types in batches accordingly.
- Line 6: We select IBD types whose corresponding processes have completed the determination of all sampling points within the time limit, and record them as set $\mathcal{SD}_C$. Then, based on the IBD construction theory we have established in Summary 1 and Summary 2, we will select the IBD types with the highest level of coverage from set $\mathcal{SD}_C$ as the final search type of (RK-)IBDs. Here, we use the local execution results to identify the target (RK-)IBD type, which is justified by our observation that the running times of individual elements show minimal variation throughout the search space.

Finally, Algorithm 1 returns the target type of (RK-)IBDs for further automatic search.

Algorithm 1: Scheduling algorithm for searching r-round (RK-)IBD

Input: set of (RK-)IBD types $\mathcal{SD}$, search space of input and output differences $\mathcal{SP}$, time limit $\mathcal{T}_{max}$, maximum number of processes $\mathcal{N}_{max}$, number of sampling points $\mathcal{N}_p$, number of rounds r

Output: search type of (RK-)IBDs

1 Construct models for each type of (RK-)IBD in $\mathcal{SD}$
2 Randomly select $\mathcal{N}_p$ elements from $\mathcal{SP}$ as sampling points.
3 Invoke $\mathcal{N}_{max}$ processes simultaneously, with each process evaluating one type of (RK-)IBD in $\mathcal{SD}$ within the time limit $\mathcal{T}_{max}$ by determining whether the sampling points are r-round (RK-)IBD.
4 Create a table Ta to record the running time for each process. Determine the search type of (RK-)IBDs according to Ta.
5 **return** *determine (RK-)IBD*

C.2 The algorithm of searching for (RK-)IBDs from the aspect of differential propagation

Without ambiguity, we simplify the state full-round-update function and round key generation function in i-round of key schedules to KS and KS_i in this paper.

A brief illustration of Algorithm 2 is as follows.

- Due to potential contradictions in IBDs at different rounds, r_u is included as an input parameter in Algorithm 2.
- Line 2-5: *BuildUpDP*(r, x, z, kx, kz) establishes the relations of upper trial by differential propagation for $x \xrightarrow{E^0, kx} z$ and $kx \xrightarrow{\mathrm{KS}^0} kz$.
  BuildLowDP(r, y, z, kx, kz, ky) establishes the relations of lower trail by differential propagation for $y \xrightarrow{(E^1)^{-1}, kx} z$, $kx \xrightarrow{\mathrm{KS}^0} kz$ and $kx \xrightarrow{\mathrm{KS}} ky$ according to the modeling methods provided in Section 5.1. Here, x, z, y denote the variables for differences in E, and kx, kz, ky denote the variables for differences in KS.
- Line 6: *DiffConnectBD*$(z_{01}, z_{23}, z_{12}, z_{03})$ sets $z_{01} \oplus z_{23} \oplus z_{12} \oplus z_{03} = 0$ based on BD definition.
- Line 7: *DiffConnectKey*$(kz_{01}, kz_{23}, kz_{12}, kz_{03})$ sets $kz_{01} \oplus kz_{23} \oplus kz_{12} \oplus kz_{03} = 0$ according to a trivial two-round XOR elimination of kz_0, kz_1, kz_2, kz_3.
- Line 8-10: *SetDiffIn*, *SetDiffOut* and *SetDiffKey* assign the latter part's parameter values to the former part's variables of input differences, output differences and key differences.

Algorithm 2: Search Model for $(R)T_0$-IBD or $(R)T_1$-IBD

Input: input differences (α, α'), output differences (β, β'), key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ with $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, E's round number r, E_0's round number r_u

Output: Model C

1 Declare six variables of input differences $x_{01}, x_{23}, kx_{01}, kx_{23}, kx_{12}, kx_{03}$ and four variables of output differences $y_{12}, y_{03}, ky_{12}, ky_{03}$; Declare eight intermediate variables of differences $z_{01}, z_{23}, z_{12}, z_{03}$ and $kz_{01}, kz_{23}, kz_{12}, kz_{03}$

2 $C_0 \leftarrow BuildUpDP(r_u, x_{01}, z_{01}, kx_{01}, kz_{01})$

3 $C_1 \leftarrow BuildUpDP(r_u, x_{23}, z_{23}, kx_{23}, kz_{23})$

4 $C_2 \leftarrow BuildLowDP(r - r_u, y_{12}, z_{12}, kx_{12}, kz_{12}, ky_{12})$

5 $C_2 \leftarrow BuildLowDP(r - r_u, y_{03}, z_{03}, kx_{03}, kz_{03}, ky_{03})$

6 $C_4 \leftarrow DiffConnectBD(z_{01}, z_{23}, z_{12}, z_{03})$

7 $C_5 \leftarrow DiffConnectKey(kz_{01}, kz_{23}, kz_{12}, kz_{03})$

8 $C_6 \leftarrow SetDiffIn(x_{01}, x_{23}, \alpha, \alpha')$

9 $C_7 \leftarrow SetDiffOut(y_{12}, y_{03}, \beta, \beta')$

10 $C_8 \leftarrow SetDiffKey(kx_{01}, kx_{23}, kx_{12}, kx_{03}, \kappa_0, \kappa_1, \kappa_2, \kappa_3)$

11 $C \leftarrow [C_0, C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8, C_9]$

12 **return** C

Finally, Algorithm 2 returns Model C to the SAT solver and identifies an IBD if no solution exists.

C.3 The algorithm of searching for (RK-)IBDs from the aspect of state propagation

A brief illustration of Algorithm 3 is as follows.

- Line 2: $BuildRK(r, kx_0, kx_1, kx_2, kx_3)$ generates the round keys and establishes their relations: For T_2-IBD, $rk_{0,i} = rk_{1,i} = rk_{2,i} = rk_{3,i}$ for $i = 0, \ldots, r-1$. For RT_3-IBD, $rk_{j,i} = \mathrm{KS}_i(kx_j)$ for $j = 0, 1, 2, 3$, $i = 0, \ldots, r-1$.
- Line 3-6: $BuildSP(r, x_j, y_j, rk_j)$ establishes the relations of state propagation of $E_{r-1,rk_{j,r-1}} \circ \cdots \circ E_{0,rk_{j,0}}(x_j) = y_j$ according to the modeling methods provided in Section 5.2. Here, x, y denote the variables for state in E.
- Line 7: $SetStateIn(x_0, x_1, x_2, x_3, \alpha, \alpha')$ sets $x_0 \oplus x_1 = \alpha, x_2 \oplus x_3 = \alpha'$.
- Line 8: $SetStateOut(y_0, y_1, y_2, y_3, \beta, \beta')$ sets $y_1 \oplus y_2 = \beta, y_0 \oplus y_3 = \beta'$.
- Line 9: $SetStateKey(kx_0, kx_1, kx_2, kx_3, \kappa_0, \kappa_1, \kappa_2, \kappa_3)$ sets $kx_0 \oplus kx_1 = \kappa_0, kx_2 \oplus kx_3 = \kappa_1, kx_1 \oplus kx_2 = \kappa_2, kx_0 \oplus kx_3 = \kappa_3$.

Finally, Algorithm 3 returns Model C to SAT solver and identifies an IBD if no solution exists.

Algorithm 3: Search Model for T_2-IBD and $(R)T_3$-IBD

Input: input differences (α, α'), output differences (β, β'), key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ with $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, E's round number r

Output: Model C

1 Declare eight input variables $x_0, x_1, x_2, x_3, kx_0, kx_1, kx_2, kx_3$, four output variables y_0, y_1, y_2, y_3, and $4r$ immediate variables $rk_{0,i}, rk_{1,i}, rk_{2,i}, rk_{3,i}$ for round keys

2 $C_0, rk_0, rk_1, rk_2, rk_3 \leftarrow \mathit{BuildRK}(r, kx_0, kx_1, kx_2, kx_3, ky_0, ky_1, ky_2, ky_3)$

3 $C_1 \leftarrow \mathit{BuildSP}(r, x_0, y_0, rk_0)$

4 $C_2 \leftarrow \mathit{BuildSP}(r, x_1, y_1, rk_1)$

5 $C_3 \leftarrow \mathit{BuildSP}(r, x_2, y_2, rk_2)$

6 $C_4 \leftarrow \mathit{BuildSP}(r, x_3, y_3, rk_3)$

7 $C_5 \leftarrow \mathit{SetStateIn}(x_0, x_1, x_2, x_3, \alpha, \alpha')$

8 $C_6 \leftarrow \mathit{SetStateOut}(y_0, y_1, y_2, y_3, \beta, \beta')$

9 $C_7 \leftarrow \mathit{SetStateKey}(kx_0, kx_1, kx_2, kx_3, \kappa_0, \kappa_1, \kappa_2, \kappa_3)$

10 $C \leftarrow [C_0, C_1, C_2, C_3, C_4, C_5, C_6, C_7, C_8]$

11 **return** C

C.4 The algorithm of searching for (RK-)IBDs from the aspect of generalized BTs

A brief illustration of Algorithm 3 is as follows.

- Line 2: $\mathit{BuildRK}(r)$ generates the round key differences for $i = 0, \ldots, r-1$ and establishes their relations: For T_C-IBD, $rk_{01,i} = rk_{23,i} = rk_{12,i} = rk_{03,i} = 0$. For RT_C-IBD, if the round key differences are generated by differential propagation, declare four variables of input differences $kx_{01}, kx_{23}, kx_{12}, kx_{03}$ of the key schedule, and set $C \leftarrow \mathit{SetDiffKey}(kx_{01}, kx_{23}, kx_{12}, kx_{03}, \kappa_0, \kappa_1, \kappa_2, \kappa_3)$ and $kx_{01} \xrightarrow{\mathrm{KS}_i} rk_{01,i}$, $kx_{23} \xrightarrow{\mathrm{KS}_i} rk_{23,i}$, $kx_{12} \xrightarrow{\mathrm{KS}_i} rk_{12,i}$, $kx_{03} \xrightarrow{\mathrm{KS}_i} rk_{03,i}$; if the round key differences are generated by state propagation, declare four variables of input states kx_0, kx_1, kx_2, kx_3 of the key schedule, and set $C \leftarrow \mathit{SetStateKey}(kx_0, kx_1, kx_2, kx_3, \kappa_0, \kappa_1, \kappa_2, \kappa_3)$ and $rk_{01,i} = \mathrm{KS}_i(kx_0) \oplus \mathrm{KS}_i(kx_1)$, $rk_{23,i} = \mathrm{KS}_i(kx_2) \oplus \mathrm{KS}_i(kx_3)$, $rk_{12,i} = \mathrm{KS}_i(kx_1) \oplus \mathrm{KS}_i(kx_2)$, $rk_{03,i} = \mathrm{KS}_i(kx_0) \oplus \mathrm{KS}_i(kx_3)$.
- Line 3-6, 9-12: $\mathit{BuildDP}_{\Lambda}(y, x, rk)$ establishes the relations of Λ operation, operations except SL, by differential propagation of $y \xrightarrow{\Lambda, rk} x$ according to the modeling methods provided in Section 5.1.
- Line 8:$C_{i0} \leftarrow \mathit{BuildSP}_{SL}(x_{01,i}, x_{23,i}, x_{12,i}, x_{03,i}, y_{01,i}, y_{23,i}, y_{12,i}, y_{03,i})$ establishes the relations of SL operation based on GEBCT for S-boxes, which is simulated by state propagation as follows: Declare 4 immediate variables of input states of SL operations $x_{0,i}, x_{1,i}, x_{2,i}, x_{3,i}$ and 4 immediate variables

Algorithm 4: Search Model for $(R)T_C$-IBD

Input: input differences (α, α'), output differences (β, β'), key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ with $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, E's round number r

Output: Model C

1 Declare two variables of input differences $in_{01}, in_{23}, in_{12}, in_{03}$ and two variables of output differences $out_{01}, out_{23}, out_{12}, out_{03}$. Declare $4(r+1)$ immediate variables of round key differences $rk_{01,i}, rk_{23,i}, rk_{12,i}, rk_{03,i}$. Declare $4r$ immediate variables of input differences of SL operations $x_{01,i}, x_{23,i}, x_{12,i}, x_{03,i}$ and $4r$ immediate variables of output differences of SL operations $y_{01,i}, y_{23,i}, y_{12,i}, y_{03,i}$

2 $C_{r0}, rk_{01}, rk_{23}, rk_{12}, rk_{03} \leftarrow BuildRK(r)$

3 $C_{r1} \leftarrow BuildDP_{\Lambda}(in_{01}, x_{01,i}, rk_{01,0})$

4 $C_{r2} \leftarrow BuildDP_{\Lambda}(in_{23}, x_{23,i}, rk_{01,0})$

5 $C_{r3} \leftarrow BuildDP_{\Lambda}(in_{12}, x_{12,i}, rk_{12,0})$

6 $C_{r4} \leftarrow BuildDP_{\Lambda}(in_{03}, x_{03,i}, rk_{03,0})$

7 **for** *i=0,. . . ,r-1* **do**

8 $\quad C_{i0} \leftarrow BuildSP_{SL}(x_{01,i}, x_{23,i}, x_{12,i}, x_{03,i}, y_{01,i}, y_{23,i}, y_{12,i}, y_{03,i})$

9 $\quad C_{i1} \leftarrow BuildDP_{\Lambda}(y_{01,i}, x_{01,i+1}, rk_{01,i+1})$

10 $\quad C_{i2} \leftarrow BuildDP_{\Lambda}(y_{23,i}, x_{23,i+1}, rk_{23,i+1})$

11 $\quad C_{i3} \leftarrow BuildDP_{\Lambda}(y_{12,i}, x_{12,i+1}, rk_{12,i+1})$

12 $\quad C_{i4} \leftarrow BuildDP_{\Lambda}(y_{03,i}, x_{03,i+1}, rk_{03,i+1})$

13 **end**

14 $C_{p0} \leftarrow SetDiffIn(x_{01}, x_{23}, \alpha, \alpha')$

15 $C_{p1} \leftarrow SetDiffOut(y_{12}, y_{03}, \beta, \beta')$

16 $C \leftarrow [C_{p0}, C_{p1}, C_{i1}, C_{i2}, C_{i3}, C_{i4}$ for $i = 0, \ldots, r]$

17 **return** C

of output states of SL operations $y_{0,i}, y_{1,i}, y_{2,i}, y_{3,i}$, and set

$$\begin{cases} y_{j,i} = SL(x_{j,i}) \text{ for } j = 0, 1, 2, 3, \\ x_{0,i} \oplus x_{1,i} = x_{01,i}, x_{2,i} \oplus x_{3,i} = x_{23,i}, x_{1,i} \oplus x_{2,i} = x_{12,i}, x_{0,i} \oplus x_{3,i} = x_{03,i}, \\ y_{0,i} \oplus y_{1,i} = y_{01,i}, y_{2,i} \oplus y_{3,i} = y_{23,i}, y_{1,i} \oplus y_{2,i} = y_{12,i}, y_{0,i} \oplus y_{3,i} = y_{03,i} \end{cases}$$

- Line 14-15: *SetDiffIn* and *SetDiffOut* assign the latter part's parameter values to the former part's variables of input differences and output differences.

Finally, Algorithm 4 returns Model C to SAT solver and identifies an IBD if no solution exists. Search Model for $(R)T_P$-IBDs is similar.

D Specifications of Block Ciphers

Only brief descriptions of block ciphers for applications are given here. For more details, please refer to their corresponding references.

D.1 Specifications of AES-128

AES [17] is one of the most renowned block ciphers worldwide. Its design philosophy has had a profound impact on block ciphers. AES is a 128-bit block cipher that supports key sizes of 128, 192, and 256 bits, and the S-box size is 8 bits. It is an SPN cipher that employs the MDS matrix to achieve excellent diffusivity. The internal state is regarded as a square array of bytes as follows, where $s_i \in \mathbb{F}_2^8$ $(0 \leq i \leq 15)$.

$$S = \begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix}.$$

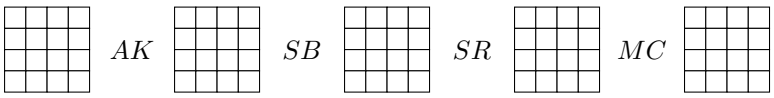

Fig. 18: One round of block cipher AES

One encryption round of AES is depicted in Fig 18, and it consists of the following four operations:

- AddRoundKey(AK): The round key derived from the key schedule is XORed with the state.
- SubBytes(SB): Applying the 8-bit S-box to each byte in parallel to the cipher's internal state.
- ShiftRows(SR): The i-th rows $(0 \leq i \leq 3)$ of the internal state is rotated by i bytes form right to left.
- Mix-Column(MC): Each column of the internal state is multiplied by the MDS matrix.

AES-128 is AES block cipher with 128-bit key. The key schedule of AES-128 is shown as Fig 19. The function g is a 32-bit to 32-bit function which consists of:

– Perform a right rotation of the input by 1 byte.
– Process all four bytes of this rotated input using the AES S-box.
– Add a fixed round coefficient to the output of the first S-box.

D.2 Specifications of DES

DES [23] is one of the earliest block ciphers to gain widespread adoption. It was standardized for use in a variety of applications, thereby becoming a pioneer

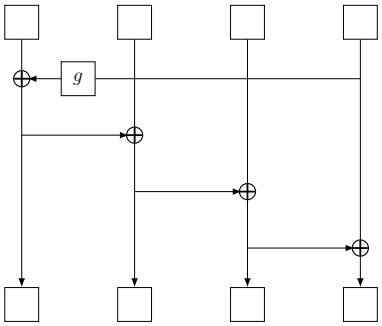

Fig. 19: The key schedule of `AES-128`

in bringing encryption to a broader range of users, including those in commercial and civilian sectors. `DES` is a 64-bit block cipher with a real key size of 56 bits. It employs eight distinct non-bijective S-boxes where the input of each S-box is 6 bits, and the output is 4 bits. `DES` adopts the Feistel network.

One round of `DES` is depicted in Fig 20, the round function acts on a 32-bit branch at a time and is composed of four stages:

- Expansion (EX): The 32-bit half-block is expanded to 48 bits through the expansion permutation by duplicating half of its bits.
- Key mixing: The result is XORed with a round key. Sixteen 48-bit round keys, one per round, are derived from the main key via the key schedule.
- Substitution (SL): The eight 6-bit chunks of the state are non-linearly transformed by eight distinct S-boxes. The output of each S-box is only 4 bits in length.
- Permutation (P): This is a fixed permutation of the output of the substitution layer, which guarantees diffusion.

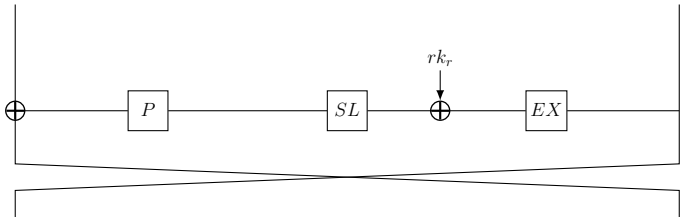

Fig. 20: One round of block cipher `DES`

The key schedule divides the 56 effective bits of the key into two 28-bit halves. The function responsible for partitioning the bits is named PC1. Each of these two halves is cyclically rotated by a fixed amount in each round. The amount of rotation is either one or two bits depending on the round. The sequence of

rotation amounts is irregular. Specifically, in round s 1, 2, 9, and 16, the rotation amount is one bit, while in all other rounds, it is two bits. From the two rotated 28-bit halves, 48 bits are selected, with 24 bits from each half, by using a fixed function called PC2 to form the round key.

D.3 Specifications of PRESENT-80

PRESENT [24] is a notable lightweight block cipher. It is extremely crucial for resource-constrained devices such as RFID tags and sensor nodes in the Internet of Things (IoT). As of now, it acts as a benchmark for new lightweight ciphers in terms of security and efficiency. PRESENT-80 is one version of PRESENT. It has a block size of 64 bits, a key size of 80 bits, and a S-box size of 4 bits. It is an SPN cipher that makes use of the operation of bit permutation.

Table 2: The S-box of PRESENT.

x	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$S(x)$	12	5	6	11	9	0	10	13	3	14	15	8	4	7	1	2

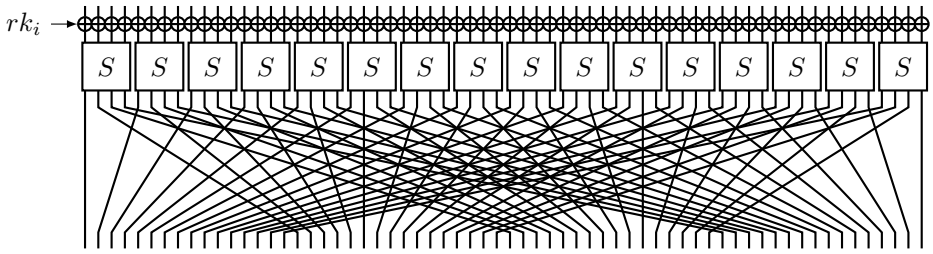

Fig. 21: One round of block cipher PRESENT

One round of PRESENT is depicted in Fig 21, the round function of it involves an XOR with the round key, the application of a 4-bit S-box (as shown in Table 2) in parallel to the state and a bit permutation.

For the key schedule of PRESENT-80, the master key is stored in a register K and is represented as $\mathrm{k}_{79}\mathrm{k}_{78}\cdots\mathrm{k}_0$. At round i, the round key K_i consists of the 64 leftmost bits of the current content of the register $K_i = \mathrm{k}_{79}\mathrm{k}_{78}\ldots\mathrm{k}_{16}$. Once

the round key is extracted, the register K is updated in the following way:

$$[k_{79}k_{78}\dots k_1k_0] = [k_{18}k_{17}\dots k_{20}k_{19}]$$
$$[k_{79}k_{78}k_{77}k_{76}] = S\,[k_{79}k_{78}k_{77}k_{76}]$$
$$[k_{19}k_{18}k_{17}k_{16}k_{15}] = [k_{19}k_{18}k_{17}k_{16}k_{15}] \oplus \text{round\_counter}.$$

D.4 Specifications of PRINTcipher48

PRINTcipher [25] enjoys a prominent status in the realm of lightweight cryptography. It is elaborately designed for settings with intense resource constraints. To date, it has rendered substantial contributions to the exploration and development of security solutions specifically targeted at low-power devices. PRINTcipher48 is one version of PRINTcipher. It has a block size of 48 bits, a key size of 80 bits, and an S-box size of 3 bits. It is an SPN cipher that makes use of the operation of key-dependent bit permutation.

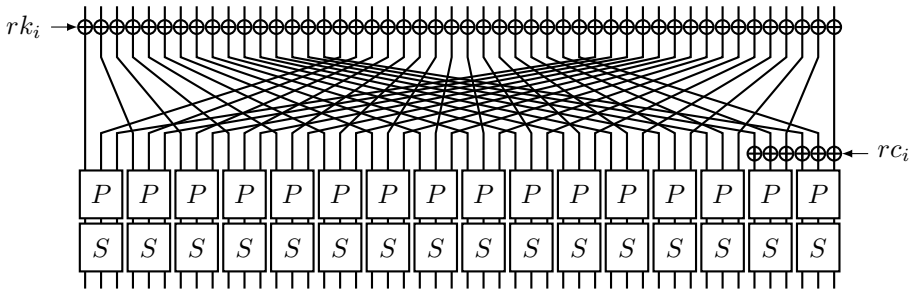

Fig. 22: One round of block cipher PRINTcipher48

One round of PRINTcipher48 is shown in Fig 22, the round function of it involves an XOR with the round key, a bit permutation, an XOR with the round constant, the key-dependent bit permutation and the application of a 3-bit S-box in parallel to the state.

The key schedule of PRINTcipher48 is rather simple, it uses the same key for all rounds.

D.5 Specifications of SPECK

SPECK [26] is an important player in the field of lightweight cryptography. It has emerged as a notable algorithm in the family of block ciphers. It has been recognized for its suitability for use in resource-constrained environments, which

has given it a distinct place in modern cryptographic research and development. The SPECK is usually denoted as SPECK-n/m where n, m are block size and key size respectively in bits, and SPECK-n if the key length does not need to be specified, where the parameters n and m are shown in Table 3. SPECK is an add-rotate-xor (ARX) cipher with operations modular addition and so on.

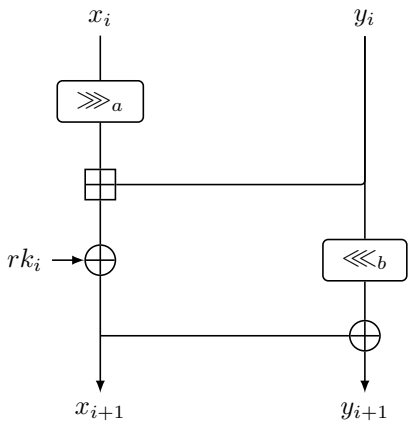

Fig. 23: One round of SPECK

block size n	key size m
32	64
48	72
	96
64	96
	128
96	96
	144
128	128
	192
	256

Table 3: Parameters n, m of SPECK.

One round of SPECK is shown in Fig 23, the round function of it involves a modular addition $\boxplus$, bitwise-xor $\oplus$, left circular shift $\lll$, and right circular shift $\ggg$, where $(a, b) = (7, 2)$ for SPECK-32 and $(a, b) = (8, 3)$ for other versions.

For the key schedule, the master key k is written as $k = (l_{t-2}, \ldots, l_0, k_0)$, where $t = 2m/n$. The k_i and l_i are defined by

$$l_{i+m-1} = (k_i + (l_i \ggg_a) \oplus i,$$
$$k_{i+1} = k_i \lll_b \oplus l_{i+m-1}.$$

The value k_i is the i-th round key.

D.6 Specifications of SKINNY

SKINNY [27] is a family of tweakable block ciphers designed within TWEAKEY framework. The SKINNY family encompasses 6 distinct versions, which are designated as SKINNY-n-$NT \cdot n$. Here, $n \in \{64, 128\}$ represents the block size, and $NT \in \{1, 2, 3\}$ indicates the number of n-bit tweakey state For SKINNY-64, the cell size c is 4, while for SKINNY-128, the cell size c is 8. The ordering of the

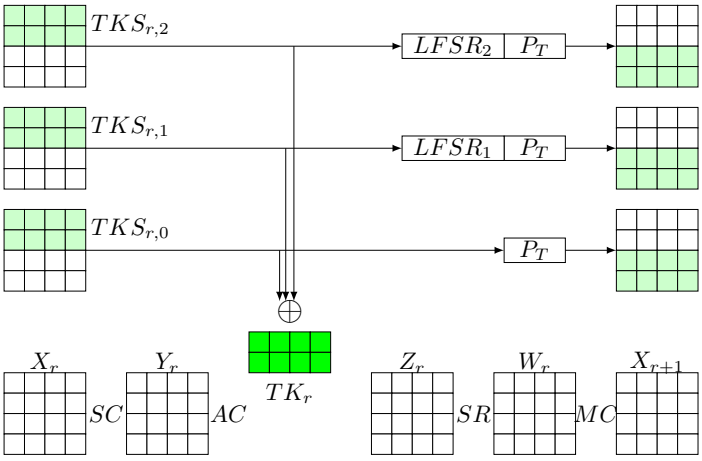

Fig. 24: One round of block cipher SKINNY

internal state and the tweakey state is represented by a 4×4 matrix:

$$S = \begin{pmatrix} s_0 & s_1 & s_2 & s_3 \\ s_4 & s_5 & s_6 & s_7 \\ s_8 & s_9 & s_{10} & s_{11} \\ s_{12} & s_{13} & s_{14} & s_{15} \end{pmatrix}.$$

One round of SKINNY-n-$3n$ is shown in Fig 24. It uses three tweaks $TKS_{0,0}$, $TKS_{0,1}$ and $TKS_{0,2}$ to derive the tweakey. Besides, SKINNY-n-n uses $TKS_{0,0}$ to derive the tweakey, and SKINNY-n-$2n$ uses $TKS_{0,0}$ and $TKS_{0,1}$ to derive the tweakey. One round of SKINNY consists of the following five operations:

- SubCell(SC): Applying the 4-bit S-box to each byte in parallel to the cipher's internal state.
- AddConstant(AC): The constant is XORed with the state.
- AddRoundTweakey(ART): There are NT tweak states $TKS_{r,i}(0 \leq i \leq NT-1)$, the first and second row of $TKS_{r,0} \oplus \cdots \oplus TKS_{r,NT-1}$ is the tweakey TK_r that XORed with the first and second row of the state. At round r, the first and second row of tweak states $TKS_{r,i}(1 \leq i \leq NT-1)$ are updated by $LFSR_i$ cell by cell first, then the tweak states $TKS_{r,i}(0 \leq i \leq NT)$ are updated by a cell permutation P_T.
- ShiftRow(SR): The i-th rows $(0 \leq i \leq 3)$ of the internal state is rotated by i bytes form left to right.
- MixColumn(MC): Each column of the internal state is multiplied with the matrix.

D.7 Specifications of SKINNYee

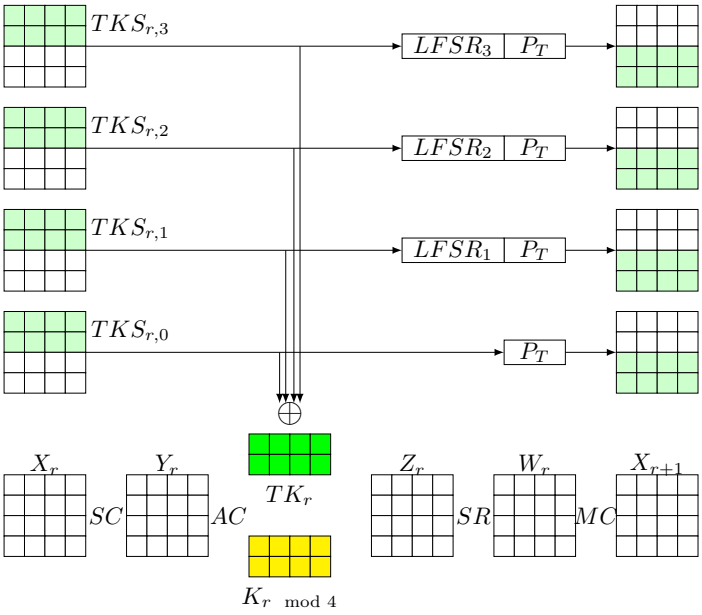

Fig. 25: One round of block cipher `SKINNYee`

`SKINNYee` [28] is tweakable block cipher, its block size is 64-bit and the key size is 128-bit. One round of `SKINNYee` is shown in Fig 25, and it consists of the following six operations:

- SubCell(SC): Applying the 4-bit S-box to each byte in parallel to the cipher's internal state.
- AddConstant(AC): The constant is XORed with the state.
- AddRoundTweak(ART): There are 4 tweak states $TKS_{r,i}(0 \leq i \leq 3)$, the first and second row of $TKS_{r,0} \oplus TKS_{r,1} \oplus TKS_{r,2} \oplus TKS_{r,3}$ is the tweak TK_r that XORed with the first and second row of the state. At round r, the first and second row of tweak states $TKS_{r,i}(1 \leq i \leq 3)$ are updated by $LFSR_i$ cell by cell first, then, the tweak states $TKS_{r,i}(0 \leq i \leq 3)$ are updated by a cell permutation P_T.
- AddRoundKey(ARK): The 128 bits master key MK is divided into four 32-bit round keys K_0, K_1, K_2 and K_3, and XORed with the third and fourth row of the state in turn.
- ShiftRow(SR): The i-th rows $(0 \leq i \leq 3)$ of the internal state is rotated by i bytes form left to right.
- MixColumn(MC): Each column of the internal state is multiplied with the matrix.

D.8 Specifications of `GIFT`

GIFT [29] has solidly positioned itself as a significant constituent in the realm of lightweight cryptography. With its design focused on efficient resource usage, it stands apart from a plethora of cryptographic algorithms. It has garnered acclaim as a contemporary and superbly crafted block cipher and is being seriously considered for applications where both security and efficiency hold paramount significance. GIFT comes in two versions: GIFT-64 and GIFT-128. GIFT-64 is a 64-bit block cipher with a 128-bit master key. GIFT-128 is a 128-bit block cipher also with a 128-bit master key. Both of these versions are SPN ciphers that utilize the operation of bit permutation.

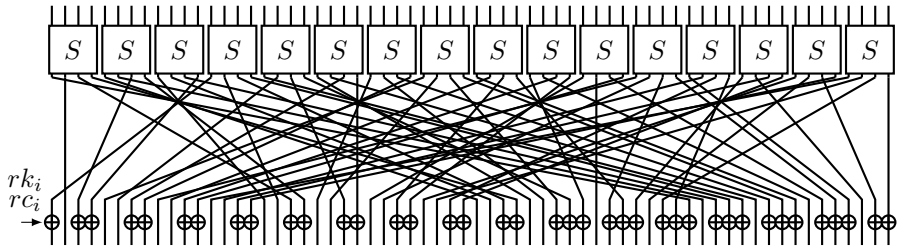

Fig. 26: One round of block cipher GIFT-64

One round of GIFT-64 is shown in Fig 26, the round function of it involves the application of a 4-bit S-box in parallel to the state, a bit permutation, and an XOR with the round key. In particular, in i-th round, for the 64-bit state $s_j (0 \leq j \leq 63)$, the 32-bit round key $rk_i = u||v = u_{15} \dots u_0 || v_{15} \dots v_0$ is XORed to the state as $b_{4j+1} \leftarrow b_{4j+1} \oplus u_j, b_{4j} \leftarrow b_{4j} \oplus v_j, (0 \leq j \leq 15)$.

One round of GIFT-128 is similar to GIFT-64, the round function of it involves the application of a 4-bit S-box in parallel to the state, a bit permutation, and an XOR with the round key. In particular, in i-th round, for the 128-bit state $s_j (0 \leq j \leq 128)$, the 64-bit round key $rk_i = u||v = u_{31} \dots u_0 || v_{31} \dots v_0$ is XORed to the state as $b_{4j+1} \leftarrow b_{4j+1} \oplus u_j, b_{4j+2} \leftarrow b_{4j+2} \oplus v_j, (0 \leq j \leq 31)$.

For both versions of GIFT, the 128-bit master key k is denoted as $k = k_7||k_6|| \dots ||k_1||k_0$, the key is updated as follows,

$$k_7 \, \|k_6\| \dots \|k_1\| \, k_0 \leftarrow k_1 \ggg_2 \, \|k_0 \ggg_2 \| \dots \|k_3\| \, k_2,$$

where $\ggg_i$ is an i bits right rotation within a 16-bit word. For GIFT-64, the 32-bit round key $rk_i = u||v$ is derived as $u \leftarrow k_1$ and $v \leftarrow k_0$. For GIFT-128, the 64-bit round key $rk_i = u||v$ is derived as $u \leftarrow k_5||k_4$ and $v \leftarrow k_1||k_0$.

D.9 Specifications of CHAM

CHAM [30] is a family of block ciphers that is suitable for devices with limited resources, such as Internet of Things (IoT) devices and embedded systems. Its design has been optimized to have relatively low requirements for computational power, memory, and energy consumption. Each cipher in this family is denoted by CHAM-n/m, where n represents the block size and m represents the key size. Table 4 presents the list of ciphers within the family along with their parameters. Here, w denotes the bit length of a branch, and r represents the new number of rounds, respectively. CHAM adopts the 4-branch generalized Feistel with the operation modular addition.

Table 4: List of CHAM ciphers and their parameters.

Cipher	n	m	w	r
CHAM-64/128	64	128	16	88
CHAM-128/128	128	128	32	112
CHAM-128/256	128	256	32	120

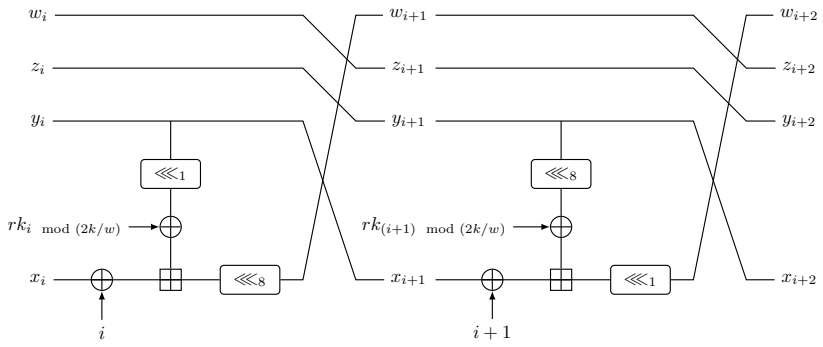

Fig. 27: Two consecutive rounds of block cipher CHAM beginning with the even i-th round

As shown in Fig 27, CHAM-n/k encrypts four w-bit words (x_0, y_0, z_0, w_0) to four w-bit words (x_r, y_r, z_r, w_r). To be more specific, in the i-th round $(0 \leq i < r)$

$$(x_{i+1}, y_{i+1}, z_{i+1}, w_{i+1}) \leftarrow \left(y_i, z_i, w_i, \left((x_i \oplus i) \boxplus \left((y_i \lll \alpha_i) \oplus rk_{i \bmod 2k/w}\right)\right) \lll \beta_i\right),$$

where $\alpha_i = 1$ and $\beta_i = 8$ when $i \bmod 2 = 0$ and $\alpha_i = 8$ and $\beta_i = 1$ when $i \bmod 2 = 1$, and $rk_{i \bmod 2k/w}$ is the round key.

The key schedule of CHAM-n/k takes k/w secret keys $K[0], K[1], \cdots, K[k/w-1]$ and generates $2k/w$ w-bit round keys $rk_0, rk_1, \cdots, rk_{2k/w-1}$. The round keys

are generated in the following way:

$$rk_i \longleftarrow K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 8),$$
$$rk_{(i+k/w)\oplus 1} \longleftarrow K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 11),$$

where $0 \leq i < k/w$.

D.10 Specifications of GOST

GOST 28147-89 has been used in a variety of applications in Russia and some other regions with historical or technological ties to Russia. It has been implemented in government and military communication systems, as well as in some financial and industrial applications where data security is of great importance. GOST is a block cipher with a 64-bit block size and a 256-bit key size. It consists of 32 Feistel rounds and adopts the operation modular addition. As depicted in Fig 28, the i-th round is defined as follows:

$$F_{K_i}(X_L, X_R) = (X_R, X_L \oplus \lll_{11} (S(X_R \boxplus K_i))),$$

where $\oplus$ denotes bit-wise XOR and $\boxplus$ denotes modular addition modulo 2^{32}, $\lll_{11}$ (A) denotes cyclic left-rotation of A by 11 bits for 32-bit word A, K_i denotes the round key, and S is an S-box layer of eight 4 bits S-boxes, these S-boxes can be either public or secret and are not necessarily permutations.

In our work, we employ public S-boxes for automatic search. Particularly, we search for IBDs in two of the most renowned versions, namely GOST-FB and GOST-PS. GOST-FB indicates GOST that uses eight different S-boxes as employed by the Central Bank of the Russian Federation (as following S_0-S_7), and GOST-PS represents GOST with only the PRESENT S-box.

$$S_0 = \{4, 10, 9, 2, 13, 8, 0, 14, 6, 11, 1, 12, 7, 15, 5, 3\}$$
$$S_1 = \{14, 11, 4, 12, 6, 13, 15, 10, 2, 3, 8, 1, 0, 7, 5, 9\}$$
$$S_2 = \{5, 8, 1, 13, 10, 3, 4, 2, 14, 15, 12, 7, 6, 0, 9, 11\}$$
$$S_3 = \{7, 13, 10, 1, 0, 8, 9, 15, 14, 4, 6, 12, 11, 2, 5, 3\}$$
$$S_4 = \{6, 12, 7, 1, 5, 15, 13, 8, 4, 10, 9, 14, 0, 3, 11, 2\}$$
$$S_5 = \{4, 11, 10, 0, 7, 2, 1, 13, 3, 6, 8, 5, 9, 12, 15, 14\}$$
$$S_6 = \{13, 11, 4, 1, 3, 15, 5, 9, 0, 10, 14, 7, 6, 8, 2, 12\}$$
$$S_7 = \{1, 15, 13, 0, 5, 7, 10, 4, 9, 2, 3, 14, 6, 11, 8, 12\}$$

The key schedule is extremely simple. The 256-bit key is divided into eight 32-bit subkeys $K_0, \ldots, K_7$. These subkeys are employed in this particular order three times during rounds $1 - 24$. In the last 8 rounds $25 - 32$, they are used in the reversed order of $K_7, \ldots, K_0$.

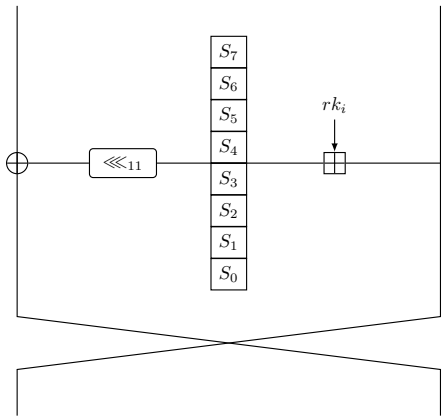

Fig. 28: One round of block cipher GOST

E Example of (RK-)IBDs

In this section, we provide some examples of derived (RK-)IBDs. Unless otherwise specified, all input differences of (RK-)IBDs are assumed to be placed in the first round.

E.1 The 4-round IBD of AES

Distinguisher 1. $(\alpha, \alpha', \beta, \beta')$ *is an IBD of* 4 *rounds* ***AES*** *without the last SR and MC layer, where*

$$\begin{cases} \alpha &= 0xuv000000000000000000000000000000, \quad (0xuv \in \mathbb{F}_2^{8*}), \\ \alpha' &= 0xu'v'000000000000000000000000000000, (0xu'v' \in \mathbb{F}_2^{8*}), \\ \beta &= 0xpq000000000000000000000000000000, \quad (0xpq \in \mathbb{F}_2^{8*}), \\ \beta' &= 0x00000000p'q'00000000000000000000, (0xp'q' \in \mathbb{F}_2^{8*}). \end{cases}$$

E.2 The 7-round IBD of DES

Distinguisher 2. $(\alpha, \alpha', \beta, \beta')$ *is an IBD of* 7*-round* ***DES****, where*

$$\begin{cases} \alpha = 0x4000000000000000, \alpha' = 0x4000000000000000, \\ \beta = 0x0000000040000000, \beta' = 0x0000000010000000. \end{cases}$$

E.3 The 6-round IBD of PRESENT-80

Distinguisher 3. $(\alpha, \alpha', \beta, \beta')$ *is an IBD of* 6*-round* ***PRESENT-80*** *without the last bit permutation, where*

$$\begin{cases} \alpha = 0x0000000000000001, \alpha' = 0x0000000000000001, \\ \beta = 0x0000000000000001, \beta' = 0x0000000000000005. \end{cases}$$

E.4 The 5-round IBD of PRINTcipher48

Distinguisher 4. *$(\alpha, \alpha', \beta, \beta')$ is an IBD of 5-round PRINTcipher48, where*

$$\begin{cases} \alpha = 0x000080000000, \alpha' = 0x400000000000, \\ \beta = 0x000100000000, \beta' = 0x000100000000. \end{cases}$$

E.5 The 5-round RK-IBD of AES-128

Distinguisher 5. *$(\alpha, \alpha', \beta, \beta')$ is an RK-IBD of 5-round AES-128 without the last MC layer under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\alpha, \alpha', 0, 0)$, where*

$$\begin{cases} \alpha &= 0x00000000000000uv00000000000000, & (0xuv \in \mathbb{F}_2^{8*}), \\ \alpha' &= 0x00000000000000u'v'00000000000000, & (0xu'v' \in \mathbb{F}_2^{8*}), \\ \beta &= 0x0000pq000000000000000000000000, & (0xpq \in \mathbb{F}_2^{8*}), \\ \beta' &= 0xp'q'000000000000000000000000000000, & (0xp'q' \in \mathbb{F}_2^{8*}). \end{cases}$$

E.6 The RK-IBDs of SPECK versions

Distinguisher 6. *$(\alpha, \alpha, \beta, \beta')$ is an RK-IBD of r-round SPECK under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\kappa, \kappa, 0, 0)$, where r, κ, α, β and β' are shown in Table 5.*

E.7 The 19-round RK-IBD of SKINNY-64/192

Distinguisher 7. *$(\alpha, \alpha, \beta, \beta)$ is an RK-IBD of 19-round SKINNY-64/192 without the first SC operation under the tweakey differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\eta, \eta, \theta, \theta)$, where $\eta = (\eta_0, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5)$ and $\theta = (\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5)$ are two tweakey differences from round 1 to round 6, and*

$$\begin{cases} \alpha = 0xc0000000000000000, \\ \beta = 0x0000000000400000, \\ \eta_0 = 0xc0000000, \\ \eta_1 = \eta_2 = \eta_3 = \eta_4 = \eta_5 = 0x00000000, \\ \theta_0 = 0x00000070, \theta_2 = 0x00000a00, \theta_4 = 0x000b0000, \\ \theta_1 = \theta_3 = \theta_5 = 0x00000000. \end{cases}$$

E.8 The 23-round RK-IBD of SKINNYee

Distinguisher 8. *$(\alpha, \alpha, \beta, \beta)$ is an RK-IBD of 23-round SKINNYee without the first SC operation under the tweak differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\eta, \eta, \theta, \theta)$, where α is the input difference in round 4, β is the output difference in round*

Table 5: The example of RK-IBDs of SPECK in two-related-keys setting.

Block cipher	params	value
SPECK-32/64	r	8
	κ	$0x0040000000000000$
	α	$0x00000000$
	β	$0x80008002$
	β'	$0x80008000$
SPECK-48/72	r	7
	κ	$0x000080000000000000$
	α	$0x000000000000$
	β	$0x800000800004$
	β'	$0x800000800000$
SPECK-48/96	r	8
	κ	$0x000080000000000000000000$
	α	$0x000000000000$
	β	$0x800000800004$
	β'	$0x800000800000$
SPECK-64/96	r	8
	κ	$0x000000800000000000000000$
	α	$0x0000000000000000$
	β	$0x8000000080000004$
	β'	$0x8000000080000000$
SPECK-64/128	r	9
	κ	$0x000000800000000000000000000000000$
	α	$0x0000000000000000$
	β	$0x8000000080000004$
	β'	$0x8000000080000000$
SPECK-64/96	r	8
	κ	$0x000000000080000000000000000000000000$
	α	$0x000000000000000000000000$
	β	$0x800000000000800000000004$
	β'	$0x800000000000800000000000$
SPECK-128/192	r	8
	κ	$0x000000000000008000000000$
		$0x000000000000000000000000$
	α	$0x00000000000000000000000000000000$
	β	$0x80000000000000008000000000000004$
	β'	$0x80000000000000008000000000000000$
SPECK-128/256	r	9
	κ	$0x00000000000000800000000000000000$
		$0x00000000000000000000000000000000$
	α	$0x00000000000000000000000000000000$
	β	$0x80000000000000008000000000000004$
	β'	$0x80000000000000008000000000000000$

27, $\eta = (\eta_0, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5, \eta_6, \eta_7)$ *and* $\theta = (\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7)$ *are two tweak differences from round* 1 *to round* 8*, and*

$$\begin{cases} \alpha = 0x0b00000000000000, \\ \beta = 0x0005000500000005, \\ \eta_0 = 0x00080000, \eta_2 = 0x00000008, \eta_4 = 0x0b000000, \\ \eta_1 = \eta_3 = \eta_5 = \eta_6 = \eta_7 = 0x00000000, \\ \theta_0 = 0x40000000, \theta_2 = 0x00c00000, \theta_4 = 0x0000e000, \theta_6 = 0x000000b0, \\ \theta_1 = \theta_3 = \theta_5 = \theta_7 = 0x00000000. \end{cases}$$

In the view of the difference of four tweaks, let $\Delta TK1$, $\Delta TK2$, $\Delta TK3$, and $\Delta TK4$ be the differences of four tweaks in the upper trail, and $\nabla TK1$, $\nabla TK2$, $\nabla TK3$, and $\nabla TK4$ be the differences of four tweaks in the lower trail. Then, η and θ can be derived from $(\Delta TK1, \Delta TK2, \Delta TK3, \Delta TK4)$ and $\nabla TK1, \nabla TK2$, $\nabla TK3, \nabla TK4)$, where

$$\begin{cases} \Delta TK1 = 0x000c000000000000, \Delta TK2 = 0x0006000000000000, \\ \Delta TK3 = 0x0009000000000000, \Delta TK4 = 0x000b000000000000, \\ \nabla TK1 = 0x5000000000000000, \nabla TK2 = 0x2000000000000000, \\ \nabla TK3 = 0xe000000000000000, \nabla TK4 = 0xd000000000000000. \end{cases}$$

The process of deriving the difference of tweaks at round $0-27$ (including η and θ) from $(\Delta TK1, \Delta TK2, \Delta TK3, \Delta TK4)$ and $\nabla TK1, \nabla TK2, \nabla TK3, \nabla TK4)$ is shown in Figure 29 and Figure 30.

E.9 The RK-IBDs of GIFT versions

Table 6: The example of RK-IBDs of GIFT in four-related-keys setting.

Block cipher	params	value
GIFT-64	r	13
	α	$0x0001000000000000$
	β	$0x0000000000020000$
	η	$0x00000000000000000000000000001000$
	θ	$0x00000000000000001000000000000000$
GIFT-128	r	10
	α	$0x00000000000000000002000000000004$
	β	$0x00000000000000000400000000000000$
	η	$0x00000000000000010000000000001000$
	θ	$0x00000000000000000000000000008000$

Distinguisher 9. $(\alpha, \alpha, \beta, \beta)$ *is an RK-IBD of* r*-round* GIFT *under the key differences* $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\eta, \eta, \theta, \theta)$*, where* r, α, β, η *and* θ *are shown in Table 6.*

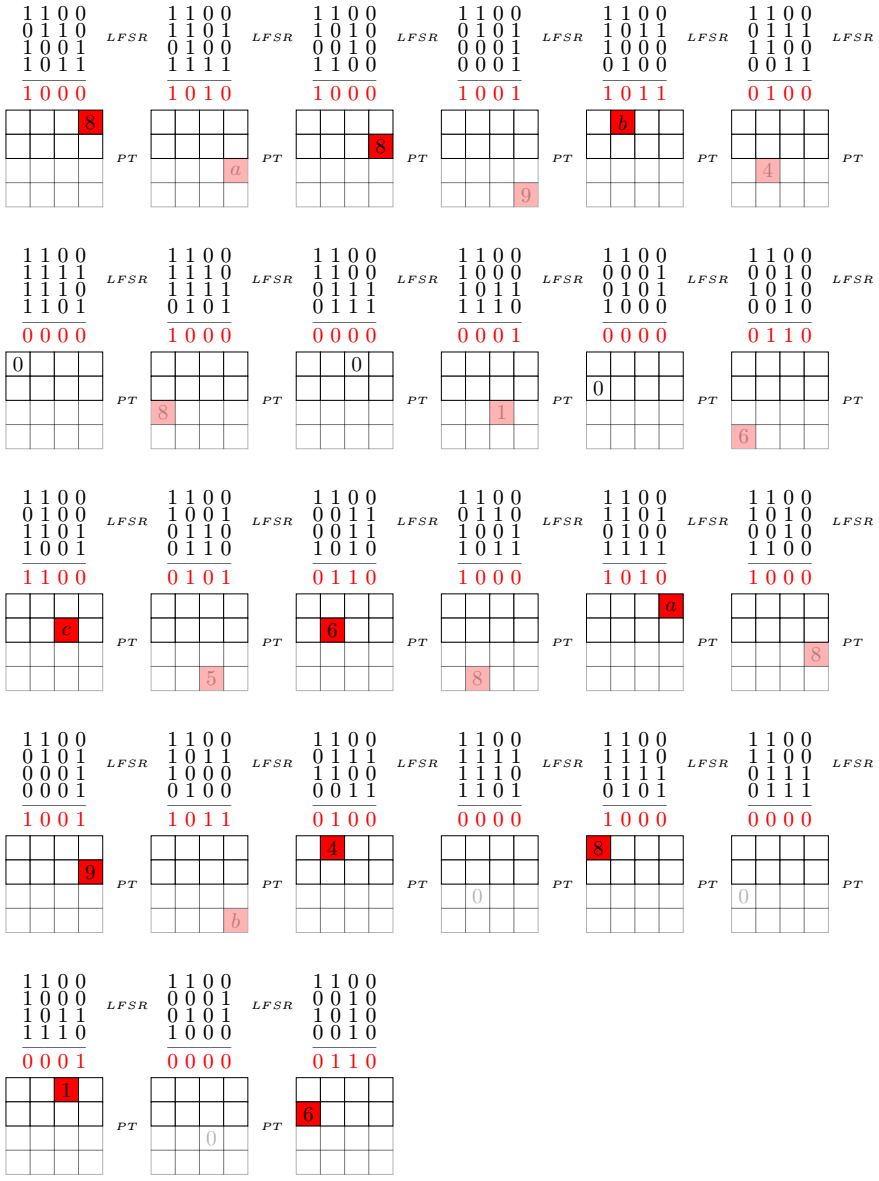

Fig. 29: Derive the difference of tweaks at round 0-27 in the upper trail

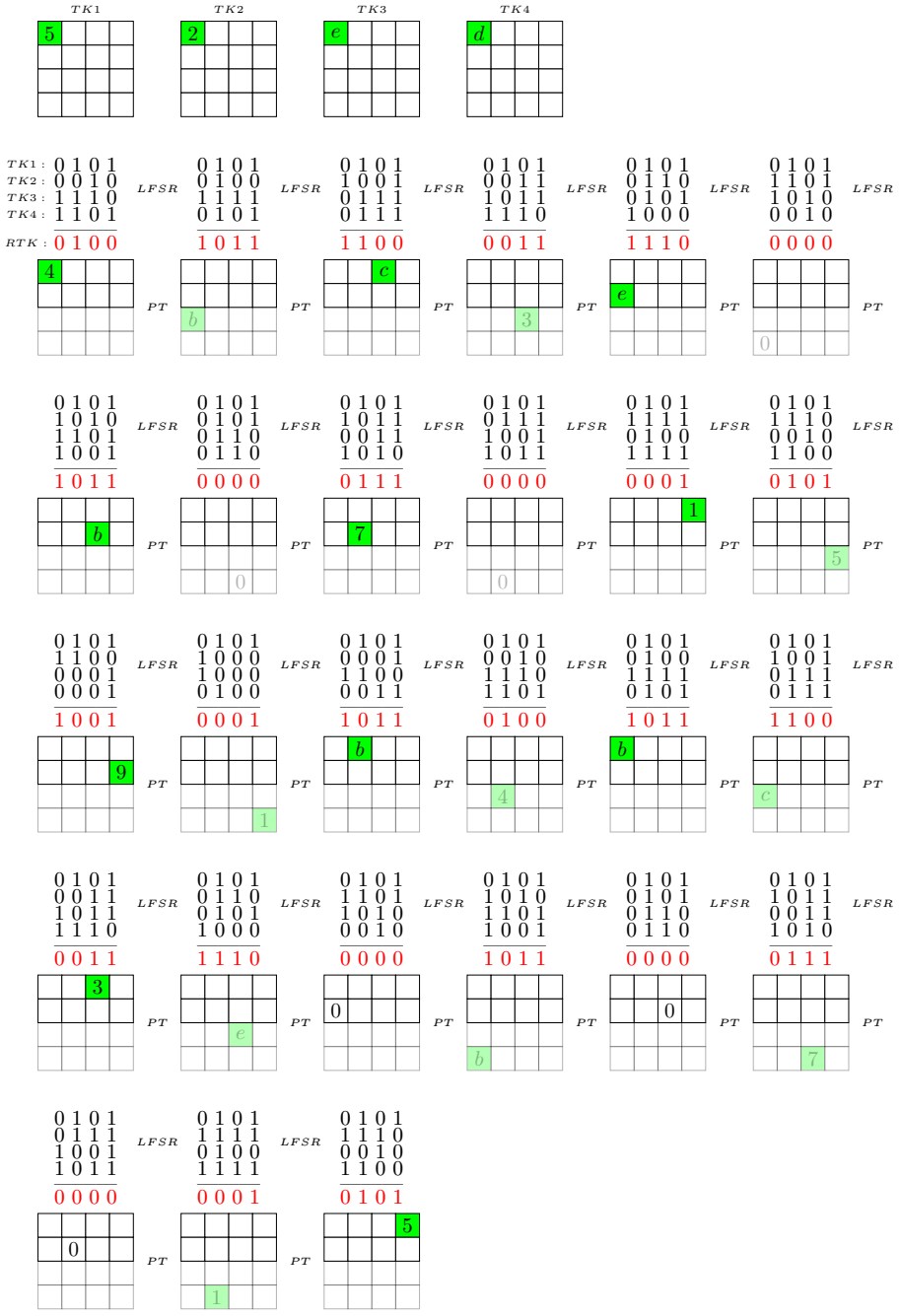

Fig. 30: Derive the difference of tweaks at round 0-27 in the lower trail

E.10 The RK-IBDs of CHAM versions

Distinguisher 10. *$(\alpha, \alpha, \beta, \beta)$ is an RK-IBD of r-round CHAM under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\eta, \eta, \theta, \theta)$, where r, α, β, η and θ are shown in Table 7.*

Table 7: The example of RK-IBDs of CHAM in four-related-keys setting.

Block cipher	params	value
CHAM-64/128	r	30
	α	$0x0000000000000000$
	β	$0x0000000000000000$
	η	$0x0000602000000000000000000000000000$
	θ	$0x6020c040000000000000000000000000$
CHAM-128/256	r	28
	α	$0x00000000000000000000000000000000$
	β	$0x00000000000000000000000000000000$
	η	$0x78081828000000000000000000000000$
		$0x00000000000000000000000000000000$
	θ	$0x00000000000000078081828f0103050$
		$0x00000000000000000000000000000000$

E.11 The full-round RK-IBD of GOST

Distinguisher 11. *$(\alpha, \alpha', \beta, \beta')$ is an RK-IBD of full-round GOST under the key differences $\kappa_{i,j} (0 \leq i \leq 3, 0 \leq j \leq 7)$, where*

$$
\begin{cases}
\alpha = 0x8000000000000000, \alpha' = 0x0000000080000000, \\
\beta = 0x0000000080000000, \beta' = 0x8000000000000000, \\
\kappa_{0,0} = 0x00000000, \kappa_{1,0} = 0x80000000, \kappa_{2,0} = 0x00000000, \kappa_{3,0} = 0x80000000, \\
\kappa_{0,1} = 0x80000000, \kappa_{1,1} = 0x00000000, \kappa_{2,1} = 0x80000000, \kappa_{3,1} = 0x00000000, \\
\kappa_{0,2} = 0x00000000, \kappa_{1,2} = 0x80000000, \kappa_{2,2} = 0x00000000, \kappa_{3,2} = 0x80000000, \\
\kappa_{0,3} = 0x80000000, \kappa_{1,3} = 0x00000000, \kappa_{2,3} = 0x80000000, \kappa_{3,3} = 0x00000000, \\
\kappa_{0,4} = 0x00000000, \kappa_{1,4} = 0x80000000, \kappa_{2,4} = 0x00000000, \kappa_{3,4} = 0x80000000, \\
\kappa_{0,5} = 0x80000000, \kappa_{1,5} = 0x00000000, \kappa_{2,5} = 0x80000000, \kappa_{3,5} = 0x00000000, \\
\kappa_{0,6} = 0x00000000, \kappa_{1,6} = 0x80000000, \kappa_{2,6} = 0x00000000, \kappa_{3,6} = 0x80000000, \\
\kappa_{0,7} = 0x80000000, \kappa_{1,7} = 0x00000000, \kappa_{2,7} = 0x40000000, \kappa_{3,7} = 0xc0000000.
\end{cases}
$$

F Verification of Examples of (RK-)IBDs

F.1 General verify strategies: computer-aided verification

For an automatic method, the correctness of the results depends on two factors: the accuracy of the modeling approach and the precision of the code implementation. In our work, we use the same modeling method and call the same set of code interfaces to automatically search for distinguishers. Therefore, verifying a subset of the distinguishers is sufficient to demonstrate the overall correctness of our results.

Manual derivation is commonly used to verify the correctness of automatic results, but it can be difficult and time-consuming. In such cases, computer-aided verification can be employed. Here, the computer can serve three roles:

-Detect the location where the contradiction occurs. Take Algorithm 2 as an example. Assume $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ is an IBD or an RK-IBD under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$. Use $\mathcal{D}$ as the input of Algorithm 2 for the input and output differences. In Line 6 of Algorithm 2, we modify the function $\mathit{DiffConnectBD}(z_{01}, z_{23}, z_{12}, z_{03})$ that sets $z_{01} \oplus z_{23} \oplus z_{12} \oplus z_{03} = 0$, equivalent to $z_{01,i} \oplus z_{23,i} \oplus z_{12,i} \oplus z_{03,i} = 0$ for $0 \leq i \leq n-1$ where n represents the block size of the cipher, into a new function. The new function is defined as $\mathit{DiffConnectD}(z_{01}, z_{23}, z_{12}, z_{03}, p)$ for $0 \leq p \leq n-1$ that sets $z_{01,i} \oplus z_{23,i} \oplus z_{12,i} \oplus z_{03,i} = 0$ for $0 \leq i \leq n-1, i \neq p$. Then, if $\mathcal{D} = (\alpha, \alpha', \beta, \beta')$ is still an (RK-)IBD under the modified algorithm when $p = j$, j is a position unrelated to the contradiction. Using this method, we can filter out all the positions unrelated to the contradiction and then derive the contradiction from the remaining positions.

-Traverse all plausible trails and disprove them. When the computing resources permit, propagate the differences or states from the input to the middle round forward and from the output to the middle round backward. Then, use generalized BTs like GBCT and GEBCT to disprove all these combined trails.

-Cross verification. There are multiple methods to search for (RK)-IBDs. Using two or more distinct approaches to search for distinguishers can validate the findings. The recommended methods are state propagation and GEBCT-based verification. Besides, for example, we can look for GEBCT and other generalized BTs to verify the (RK)-IBDs derived by state propagation.

F.2 General verify strategies: code self-feedback verification

To verify the correctness of the code, we propose a code self-feedback verification technique. This technique examines the code by verifying the correctness of the solution returned by the solver when the input and output differences are not an (RK-)IBD. The overall algorithm is shown in Algorithm 8. It automatically parses the solution to create a dictionary of the values of all variables used in the search for the distinguisher, and then checks the propagation of input and output values through each operation. Examples of check algorithms are provided in Algorithm 6, 5, 7. Similar checks can be applied to other operations.

Algorithm 5: Check the propagation of difference through DDT

Input: input difference μ, output difference θ, the S-box S
Output: **True** or **False** to indicate whether μ can propagate to θ

flag = **False**
for $x \in \{0, \ldots, 2^n - 1\}$ **do**
  if $S(x) \oplus S(x \oplus \mu) = \theta$ **then**
    flag = **True**
    break
  end
end
return *flag*

Algorithm 6: Check the propagation of state through S-box

Input: input value x, output value y, the S-box S
Output: **True** or **False** to indicate whether x can propagate to y

flag = **False**
if $S(x) = y$ **then**
  flag = **True**
end
return *flag*

Algorithm 7: Check the propagation of difference through GEBCT

Input: input differences (μ, μ', ρ, ρ'), output differences $(\theta, \theta', \varphi, \varphi')$, the S-box S
Output: **True** or **False** to indicate whether (μ, μ', ρ, ρ') can propagate to $(\theta, \theta', \varphi, \varphi')$

flag = **False**
for $x \in \{0, \ldots, 2^n - 1\}$ **do**
  if $(S(x) \oplus S(x \oplus \mu) = \theta)$ *and* $(S(x \oplus \mu) \oplus S(x \oplus \mu \oplus \rho) = \varphi)$ *and* $(S(x \oplus \mu \oplus \rho) \oplus S(x \oplus \mu \oplus \rho \oplus \mu') = \theta')$ *and* $(S(x) \oplus S(x \oplus \rho') = \varphi')$ **then**
    flag = **True**
    break
  end
end
return *flag*

Algorithm 8: Code self-feedback verification

Input: a solution returned from a third-party solver.
Output: **True** or **False** to denote the correctness of the solution

```
flag = True
Parse the solution to obtain a dictionary that holds the values of all
  variables used in the search for the distinguisher
for each operation do
    Get the input and output values of the operation
    Check the propagation of the input and output value through the
    operation
    if check fails then
        flag = False
        break
    end
end
return flag
```

Now we present our verification of some examples of IBDs and RK-IBDs in Appendix E as follows. Apart from the following examples, all the (RK-)IBDs we obtained have passed both the computer-aided verification and code self-feedback verification.

F.3 Verification of the 4-round IBD of AES

Distinguisher 1

Verification (verify by contradiction). Assume (α, α') can propagate to (β, β'), as shown in Figure 31. For $X_0, X_1 = X_0 \oplus \alpha, X_2, X_3 = X_2 \oplus \alpha'$, and $Y_0, Y_1, Y_2 = Y_1 \oplus \beta, Y_3 = Y_0 \oplus \beta'$, let Z_i be the value obtained by encrypting X_i after 2 rounds without the last MC layer, and W_i be the value obtained by decrypting Y_i after 2 rounds. Then $Z_0 \oplus Z_1 = \gamma$, $Z_2 \oplus Z_3 = \gamma'$, $W_1 \oplus W_2 = \delta$, and $W_0 \oplus W_3 = \delta'$. On the one hand, $W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} = \delta_0 \neq 0$ and $W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} = 0$, since

$$W_{1,0} \oplus W_{2,0} = \delta_0 \neq 0, W_{0,0} \oplus W_{3,0} = 0,$$
$$W_{1,1} \oplus W_{2,1} = 0, W_{0,1} \oplus W_{3,1} = 0.$$

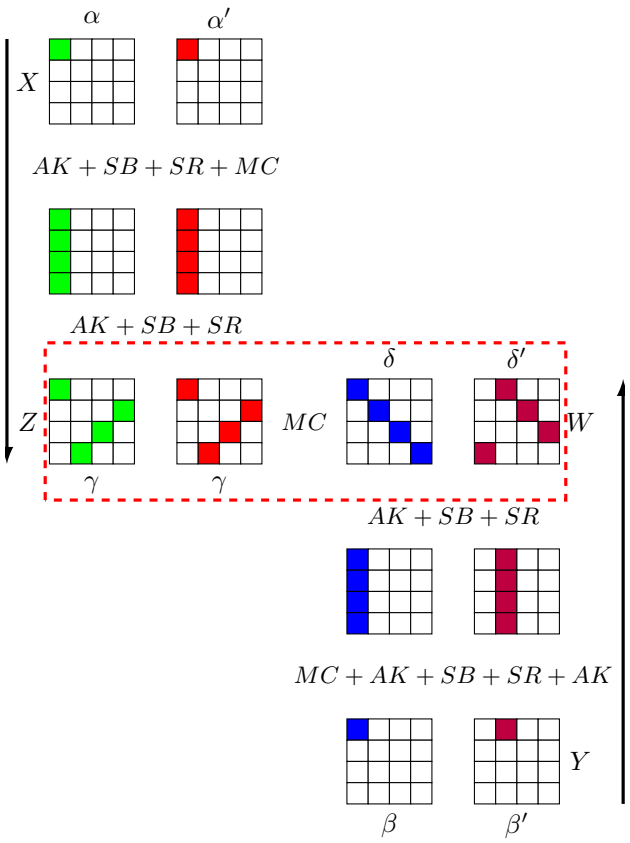

Fig. 31: One of 4-round IBDs of `AES`

On the other hand, since

$$\begin{pmatrix} W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} \\ W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} \\ W_{0,2} \oplus W_{1,2} \oplus W_{2,2} \oplus W_{3,2} \\ W_{0,3} \oplus W_{1,3} \oplus W_{2,3} \oplus W_{3,3} \end{pmatrix} = M \cdot \begin{pmatrix} Z_{0,0} \oplus Z_{1,0} \oplus Z_{2,0} \oplus Z_{3,0} \\ Z_{0,1} \oplus Z_{1,1} \oplus Z_{2,1} \oplus Z_{3,1} \\ Z_{0,2} \oplus Z_{1,2} \oplus Z_{2,2} \oplus Z_{3,2} \\ Z_{0,3} \oplus Z_{1,3} \oplus Z_{2,3} \oplus Z_{3,3} \end{pmatrix}$$
$$= M \cdot \begin{pmatrix} \gamma_0 \oplus \gamma_0' \\ 0 \\ 0 \\ 0 \end{pmatrix},$$

$W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} = 0$ and $W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} = 0$, or $W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} \neq 0$ and $W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} \neq 0$. Thus there is a contradiction. □

F.4 Verification of the 7-round IBD of DES

Distinguisher 2

Verification (verify by contradiction). We propagate the input differences 3 rounds in the forward direction, and the output differences 4 rounds in the backward direction. The middle 5 rounds of the distinguisher are shown in Figure 32, where a and a' are the two differences propagated from the input differences, b and b' are the two differences propagated from the output differences, and (x_0, x_1, x_2, x_3) is the states of the right branch in round 4. Then $x_0 \oplus x_1 = P(a) \oplus 1$, $x_2 \oplus x_3 = P(a') \oplus 1$, and $x_1 \oplus x_2 = P(b) \oplus 1$, $x_0 \oplus x_3 = P(b') \oplus 3$, where P denotes the permutation of DES. Thus $b \oplus b' = a \oplus a' \oplus P^{-1}(3)$. Let $b' = (b'_0, \ldots, b'_7)$ where $(b'_i \in \mathbb{F}_2^4, 0 \leq i \leq 7)$. Then according to the permutation P and the values of a, a' and b, we have $b'_6 = b'_0 = 0$, which implies the bit 0 and bit 3 of the output of the second S-box in round 6 must be zero. However, the input difference of the S-box is $0x20$, the first and fourth bits of the output cannot be zero simultaneously. There is a contradiction.

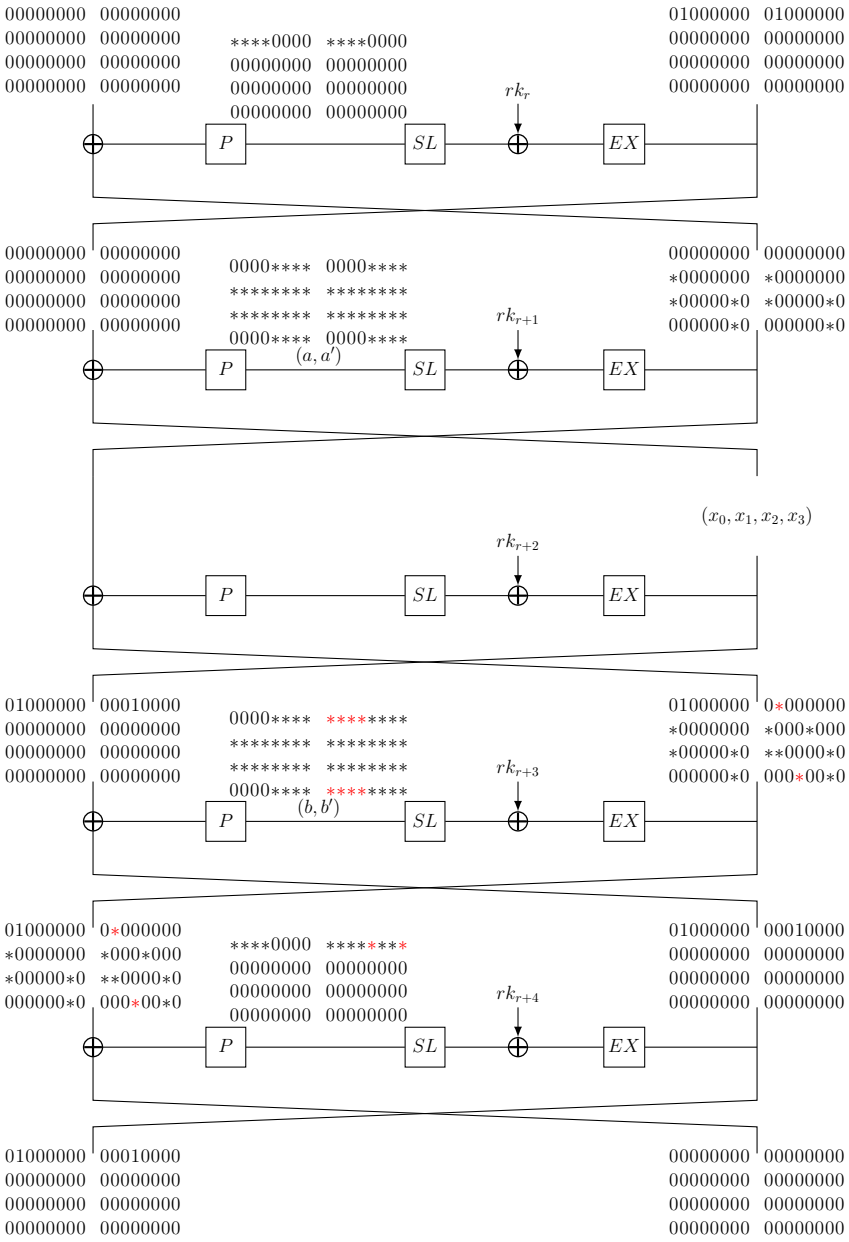

Fig. 32: The core of one 7-round IBDs of `DES`

F.5 Verification of the 6-round IBD of PRESENT-80

Distinguisher 3

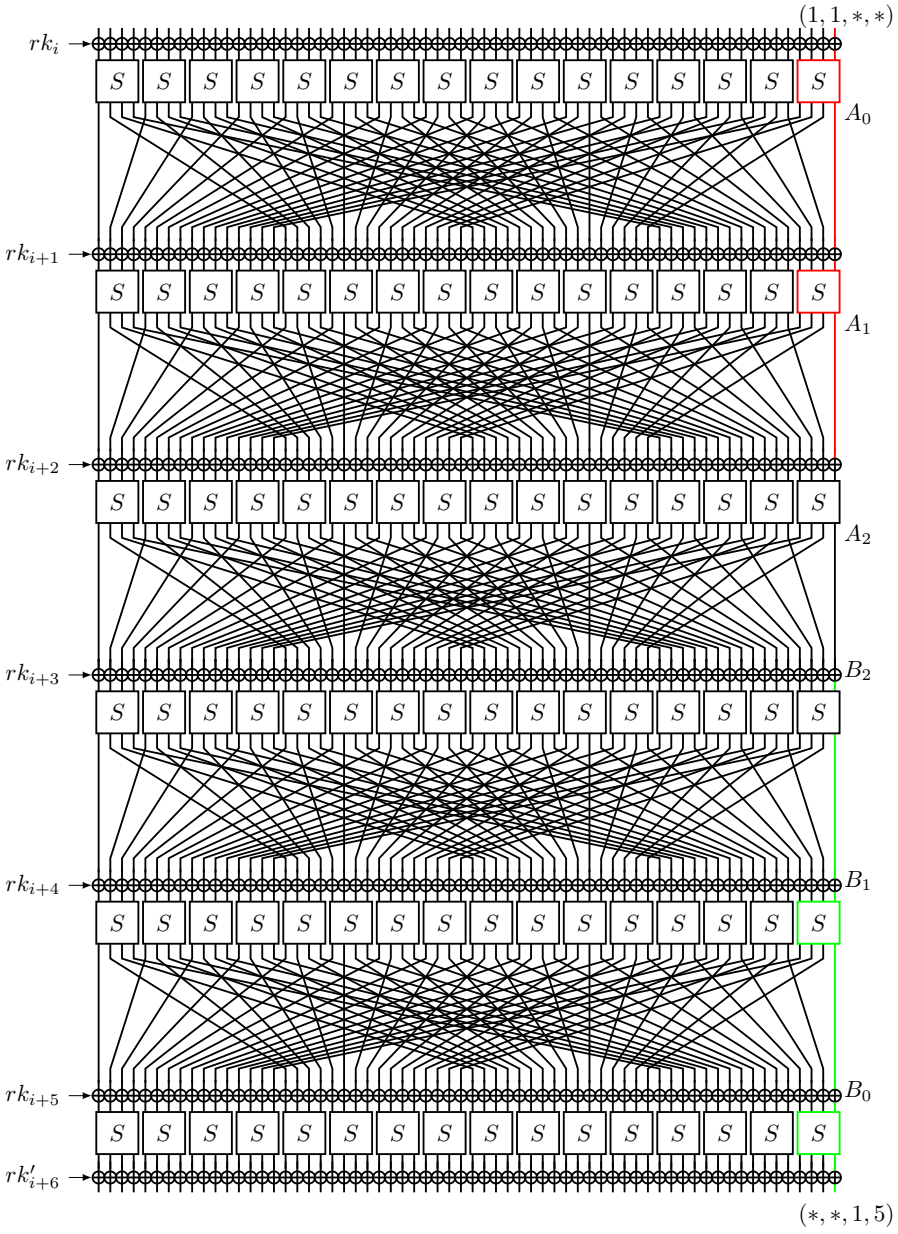

Fig. 33: One of 6-round IBDs of PRESENT-80

Our verification makes use of the definition of GEBCT. Thus we demonstrate some basic properties of the S-box of PRESENT in the view of such table. The analysis reveals some new properties of the S-box of PRESENT.

Property 1 (GEBCT). *Let $\mathcal{T}$ and $\mathcal{T}_{inv}$ be the* GEBCT *of S-box and the invertible S-box of PRESENT, then*

$$(1,1,*,*) \xrightarrow{\mathcal{T}} (1,1,0,0),(1,1,1,1),$$
$$(*,*,1,5) \xrightarrow{\mathcal{T}_{inv}} (1,0,1,0),(0,1,1,0),$$
$$(1,0,1,0),(0,1,1,0) \xrightarrow{\mathcal{T}_{inv}} (1,0,1,0),(0,1,1,0),$$

where $(\mu,\mu',\rho,\rho') \xrightarrow{T} (\theta,\theta',\varphi,\varphi')$ means $T(\mu,\mu',\rho,\rho',x||\theta,x'||\theta',y||\varphi,y'||\varphi') \neq 0$ for $T \in \{\mathcal{T},\mathcal{T}_{inv}\}$, $\mu,\mu',\rho,\rho' \in \mathbb{F}_2^4$, $\theta,\theta',\varphi,\varphi' \in \mathbb{F}_2$, $x,x',y,y' \in \mathbb{F}_2^3$, $''||''$ means the concatenation of two variables, and '$*$' represents arbitrary 4-bit value.

Verification (verify by contradiction). As shown in Figure 33, let $(\mu_i,\mu_i',\rho_i,\rho_i')$ be the input differences of the 0-th S-box (least signification) at round i $(i=0,1,2)$, $(\theta_{i,0},\theta_{i,0}',\varphi_{i,0},\varphi_{i,0}')$ be the 0-th bit (least signification) of the output difference of the 0-th S-box at round i $(i=0,1,2)$, $A_i(i=0,1,2)$ be the set of all possible output difference of the 0-th bit of 0-th S-box at round i.

- Round 0: $(\mu_0,\mu_0',\rho_0,\rho_0') = (1,1,*,*)$, according to Property 1, we have $(\theta_{0,0}, \theta_{0,0}',\varphi_{0,0},\varphi_{0,0}') \in \{(1,1,0,0),(1,1,1,1)\}$. Thus, $A_0 = \{(1,1,0,0),(1,1,1,1)\}$.
- Round 1: for the 0-th S-box, the 0-th bit of the input difference is indeed $(\theta_{0,0},\theta_{0,0}',\varphi_{0,0},\varphi_{0,0}')$. While the values of the other bits of the input difference of 0-th S-box are all zero (they are not affected by the difference of the 0-th S-box at round 0), we have $(\mu_1,\mu_1',\rho_1,\rho_1') \in \{(1,1,0,0),(1,1,1,1)\}$. According to Property 1, we have $(\theta_{1,0},\theta_{1,0}',\varphi_{1,0},\varphi_{1,0}') \in \{(1,1,0,0),(1,1,1,1)\}$. Thus, $A_1 = \{(1,1,0,0),(1,1,1,1)\}$.
- Round 2: for the 0-th S-box, the 0-th bit of the input difference is indeed $(\theta_{1,0},\theta_{1,0}',\varphi_{1,0},\varphi_{1,0}')$. While the values of the other bits of the input difference of 0-th S-box are all zero (they are not affected by the difference of the 0-th S-box at round 0), we have $(\mu_2,\mu_2',\rho_2,\rho_2') \in \{(1,1,0,0),(1,1,1,1)\}$. According to Property 1, we have $(\theta_{2,0},\theta_{2,0}',\varphi_{2,0},\varphi_{2,0}') \in \{(1,1,0,0),(1,1,1,1)\}$. Thus, $A_1 = \{(1,1,0,0),(1,1,1,1)\}$.

All in all, it holds that $A_0 = A_1 = A_2 = \{(1,1,0,0),(1,1,1,1)\}$, and $(\theta_{2,0},\theta_{2,0}', \varphi_{2,0},\varphi_{2,0}') \in A_2$.

Let $(\mu_i,\mu_i',\rho_i,\rho_i')$ be the output differences of the 0-th S-box at round i $(i=3,4,5)$, $(\theta_{i,0},\theta_{i,0}',\varphi_{i,0},\varphi_{i,0}')$ be the 0-th bit of the input difference of the 0-th S-box at round i $(i=3,4,5)$, $B_i(i=0,1,2)$ be the set of all possible input difference of the 0-th bit of 0-th S-box at round $(5-i)$.

- Round 5: $(\mu_5,\mu_5',\rho_5,\rho_5') = (*,*,1,5)$, according to Property 1, we have $(\theta_{5,0}, \theta_{5,0}',\varphi_{5,0},\varphi_{5,0}') \in \{(1,0,1,0),(0,1,1,0)\}$. Thus, $B_0 = \{(1,0,1,0),(0,1,1,0)\}$.

- Round 4: for the 0-th S-box, the 0-th bit of the output difference is indeed $(\theta_{5,0}, \theta'_{5,0}, \varphi_{5,0}, \varphi'_{5,0})$. While the values of the other bits of the output difference of 0-th S-box are all zero (they are not affected by the difference of the 0-th S-box at round 5), we have $(\mu_4, \mu'_4, \rho_4, \rho'_4) \in \{(1,0,1,0),(0,1,1,0)\}$. According to Property 1, we have $(\theta_{4,0}, \theta'_{4,0}, \varphi_{4,0}, \varphi'_{4,0}) \in \{(1,0,1,0),(0,1,1,0)\}$. Thus, $B_1 = \{(1,0,1,0),(0,1,1,0)\}$.
- Round 3: for the 0-th S-box, the 0-th bit of the input difference is indeed $(\theta_{4,0}, \theta'_{4,0}, \varphi_{4,0}, \varphi'_{4,0})$. While the values of the other bits of the input difference of 0-th S-box are all zero (they are not affected by the difference of the 0-th S-box at round 5), we have $(\mu_3, \mu'_3, \rho_3, \rho'_3) \in \{(1,0,1,0),(0,1,1,0)\}$. According to Property 1, we have $(\theta_{3,0}, \theta'_{3,0}, \varphi_{3,0}, \varphi'_{3,0}) \in \{(1,0,1,0),(0,1,1,0)\}$. Thus, $B_2 = \{(1,0,1,0),(0,1,1,0)\}$.

All in all, it holds that $B_0 = B_1 = B_2 = \{(1,0,1,0),(0,1,1,0)\}$, and $(\mu_{3,0}, \mu'_{3,0}, \rho_{3,0}, \rho'_{3,0}) \in B_2$.

There is a contradiction, since $(\theta_{2,0}, \theta'_{2,0}, \varphi_{2,0}, \varphi'_{2,0}) = (\mu_{3,0}, \mu'_{3,0}, \rho_{3,0}, \rho'_{3,0})$, $\theta_{2,0} = 1, \theta'_{2,0} = 1$, and one value of $\mu_{3,0}$ and $\mu'_{3,0}$ is 0. □

F.6 Verification of the 5-round IBD of PRINTcipher48

Distinguisher 4

To conduct this verification, we first give some properties of the S-box from the perspectives of DDT and GBCT.

Property 2 (DDT). *Let $\mathcal{T}$ and $\mathcal{T}_{inv}$ be the* DDT *of S-box and the invertible S-box of PRINTcipher48, then*

$$001 \xrightarrow{\mathcal{T}} **1, 010 \xrightarrow{\mathcal{T}} *1*, 100 \xrightarrow{\mathcal{T}} 1**,$$
$$001 \xrightarrow{\mathcal{T}_{inv}} **1, 010 \xrightarrow{\mathcal{T}_{inv}} *1*, 100 \xrightarrow{\mathcal{T}_{inv}} 1**,$$

where '$*$' can be 0 or 1, and $abc \xrightarrow{T} a'b'c'$ means $T(abc, a'b'c') \neq 0$ for $T \in \{\mathcal{T}, \mathcal{T}_{inv}\}$ and 3-bit values abc and $a'b'c'$.

Property 3 (GBCT). *Let $\mathcal{T}$ be the* GBCT *of S-box of PRINTcipher48, then*

$$(\alpha, 0) \xrightarrow{\mathcal{T}} (\beta, \beta),$$
$$(\gamma, 0)) \xrightarrow{\mathcal{T}} (0,0), (\gamma, \delta)) \xrightarrow{\mathcal{T}} (0,0),$$
$$(1,2) \xrightarrow{\mathcal{T}} (4,4), (1,4) \xrightarrow{\mathcal{T}} (2,2), (2,4) \xrightarrow{\mathcal{T}} (1,1),$$

where the weight of both α and β is 1, $\gamma \neq 0$, and $\delta \neq \gamma$.

Verification (verify by contradiction). We prove it as shown in Figure 34. In which, blocks of the same color and the same symbol indicate that they are affected by the same S-box. Let $(x_i^0, x_i^1, x_i^2, x_i^3)$ and $(y_i^0, y_i^1, y_i^2, y_i^3)$ be the four states before and after the key-dependent layer at the round i, and $(z_i^0, z_i^1, z_i^2, z_i^3)$

be the four states after the S-box layer at the round i, $x^t_{i,j}$, $y^t_{i,j}$ and $z^t_{i,j}$ be the value of j-th nibble of x^t_i, y^t_i and z^t_i, $x^t_{i,j,k}$, $y^t_{i,j,k}$ and $z^t_{i,j,k}$ be the k-th bit value of j-th nibble of x^t_i, y^t_i and $z^t_i (0 \le k \le 2, 0 \le j \le 15, 0 \le t \le 3)$.

We propagate the input two differences 3 rounds in the forward direction, and the output two differences 2 rounds in the backward direction. According to the Property 3, the input differences and output differences of S-box i ($i \in \{0, 3, 5, 7, 9, 12, 14\}$) in round 3 must be 0. Since $z^1_{5,10,2} \oplus z^2_{5,10,2} = 1$, and $z^0_{5,10,2} \oplus z^3_{5,10,2} = 1$, according the Property 2, $\exists k \in \{0, 1, 2\}$, such that $x^1_{5,10,j} \oplus x^2_{5,10,j} = 1$, and $x^0_{5,10,j} \oplus x^3_{5,10,j} = 1$, Similarly, $\exists (j, k) \in \{(1, 0), (2, 2), (4, 2), (6, 1), (8, 0), (10, 0), (11, 2), (13, 1), (15, 1)\}$, such that $z^1_{3,j,k} \oplus z^2_{3,j,k} = 1$, and $z^0_{3,j,k} \oplus z^3_{3,j,k} = 1$.

Now we continue to discuss different cases. When the key-dependent permutation 15 transforms the bit 1 to bit 2 and the key-dependent permutation 14 transforms the bit 2 to bit 2. As shown in Figure 35, $\exists (j, k) \in \{(4, 2), (6, 1), (13, 1), (15, 1)\}$, such that $z^1_{3,j,k} \oplus z^2_{3,j,k} = 1$, and $z^0_{3,j,k} \oplus z^3_{3,j,k} = 1$. If only $z^1_{3,15,1} \oplus z^2_{3,15,1} = 1$, and $z^0_{3,15,1} \oplus z^3_{3,15,1} = 1$, then the S-box 14 in round 4 is active, and the key-dependent permutation 14 transforms the bit 2 to bit 0. This leads to a contradiction. Meanwhile, $x^0_{3,15,2} \oplus x^1_{3,15,2} = 0$, and $x^2_{3,15,2} \oplus x^3_{3,15,2} = 0$, since the key-dependent permutation 15 transforms the bit 1 to bit 2 already and cannot transform the bit 2 to bit 2. Thus, $x^0_{3,13,2} \oplus x^1_{3,13,2} = 1$, and $x^2_{3,13,2} \oplus x^3_{3,13,2} = 0$ or $x^0_{3,13,2} \oplus x^1_{3,13,2} = 0$, and $x^2_{3,13,2} \oplus x^3_{3,13,2} = 1$. According to the Property 3, $z^1_{3,13,1} \oplus z^2_{3,13,1} = 1$, and $z^0_{3,13,1} \oplus z^3_{3,13,1} = 1$ cannot hold.

Now, only $\exists (j, k) \in \{(4, 2), (6, 1)\}$, such that $z^1_{3,j,k} \oplus z^2_{3,j,k} = 1$, and $z^0_{3,j,k} \oplus z^3_{3,j,k} = 1$. Assume $z^1_{3,4,2} \oplus z^2_{3,4,2} = 1$, and $z^0_{3,4,2} \oplus z^3_{3,4,2} = 1$, then

$$\begin{cases} z^1_{3,4,2} \oplus z^2_{3,4,2} = 1, z^0_{3,4,2} \oplus z^3_{3,4,2} = 1, \\ y^0_{3,4,2} \oplus y^1_{3,4,2} = 0, y^2_{3,4,2} \oplus y^3_{3,4,2} = 0, \\ z^1_{3,6,1} \oplus z^2_{3,6,1} = a, z^0_{3,6,1} \oplus z^3_{3,6,1} = a \oplus 1, \\ y^0_{3,6,1} \oplus y^1_{3,6,1} = 0, y^0_{3,6,2} \oplus y^3_{3,6,3} = 1, \\ z^1_{3,13,1} \oplus z^2_{3,13,1} = b, z^0_{3,13,1} \oplus z^3_{3,13,1} = b \oplus 1, \\ y^0_{3,13,1} \oplus y^1_{3,13,1} = 1, y^2_{3,13,1} \oplus y^3_{3,13,1} = 0, \end{cases}$$

must be hold, where a, b can be 0 or 1. Thus,

$$\begin{cases} x^1_{5,10} \oplus x^2_{5,10} = 1||a||b, x^0_{5,10} \oplus x^3_{5,10} = 1||a \oplus 1||b \oplus 1, \\ x^0_{5,10} \oplus x^1_{5,10} = c||0||1, x^2_{5,10} \oplus x^3_{5,10} = c||1||0, \end{cases}$$

where c can be 0 or 1. For any permutation of key-dependent permutation 10 in round 5, this is impossible according to the $GEBCT$ of S-box and $z^1_{5,10,2} \oplus z^2_{5,10,2} = 1$, and $z^0_{5,10,2} \oplus z^3_{5,10,2} = 1$. For other cases, we can verify similarly.

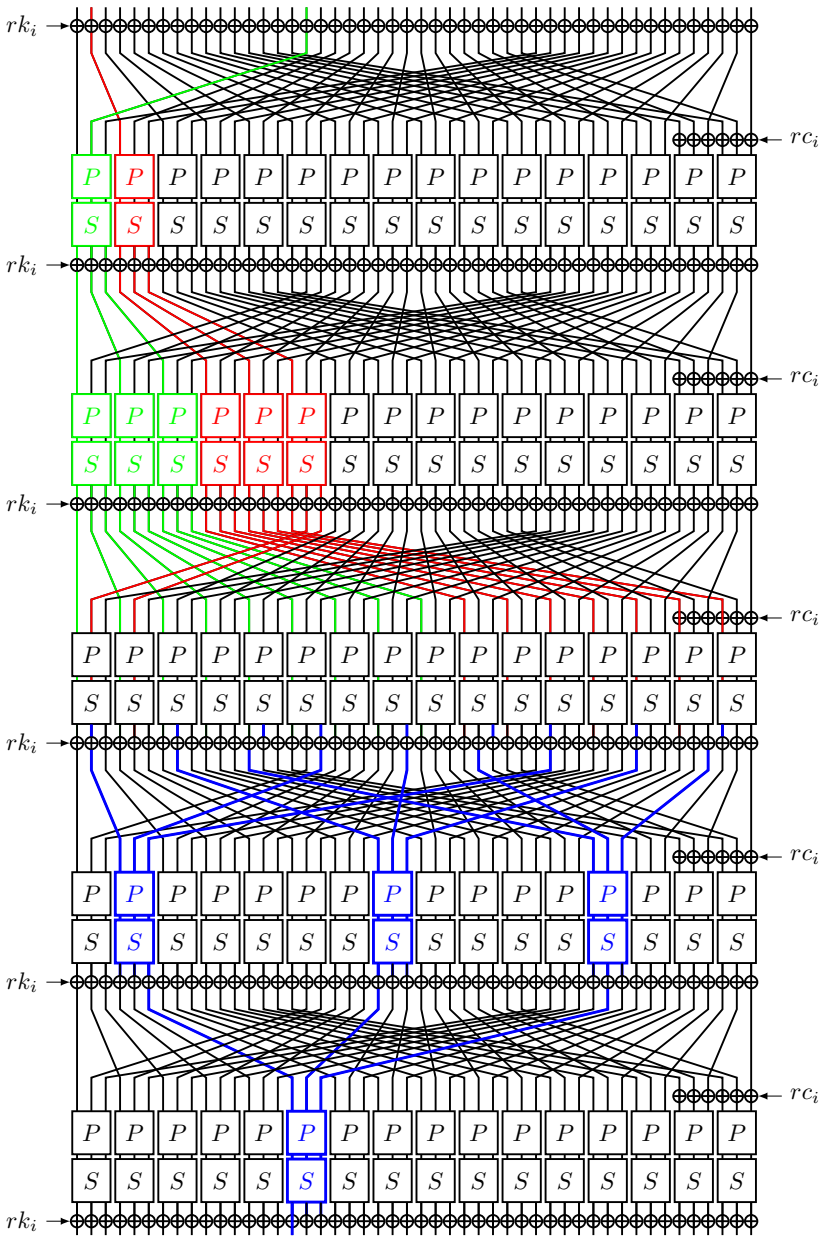

Fig. 34: One of 5-round IBDs of `PRINTcipher48`

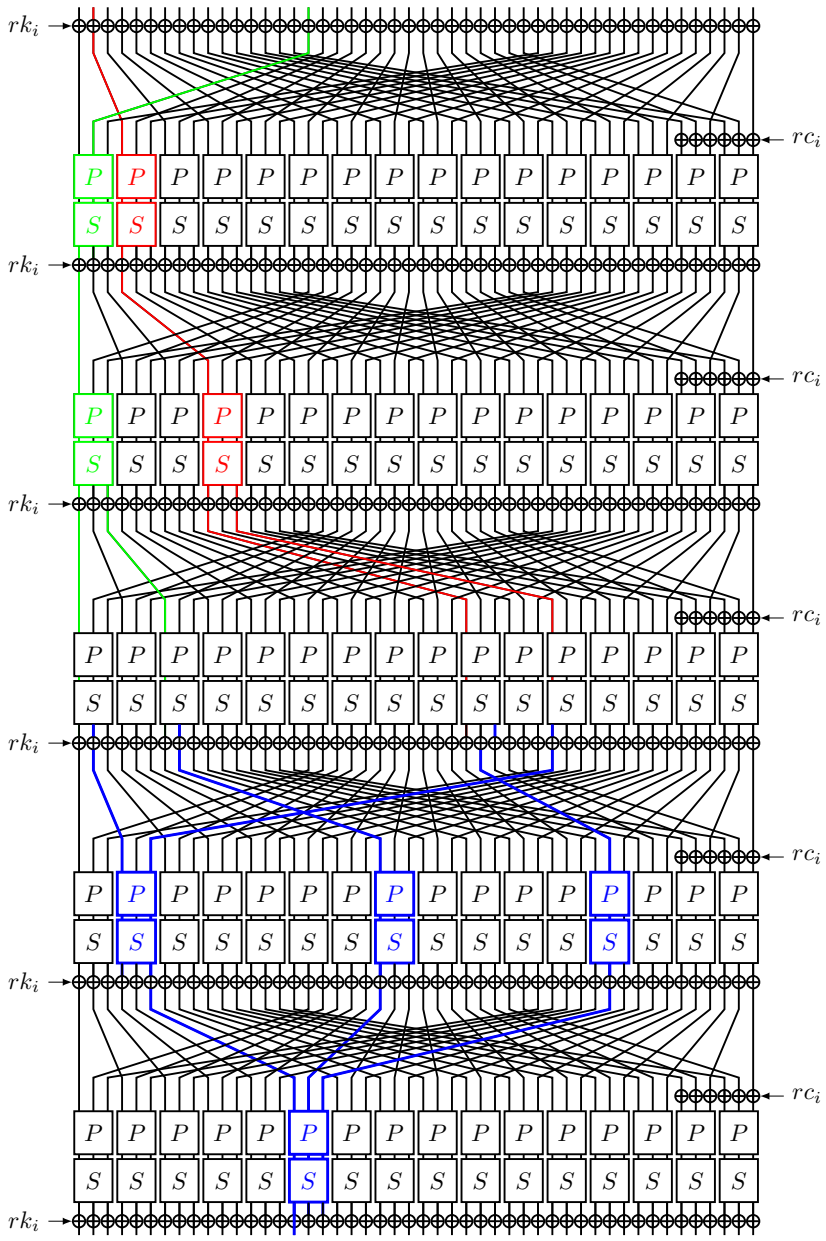

Fig. 35: One impossible propagation trail of 5-round IBD of `PRINTcipher48`

F.7 Verification of the 5-round RK-IBD of AES-128

Distinguisher 5

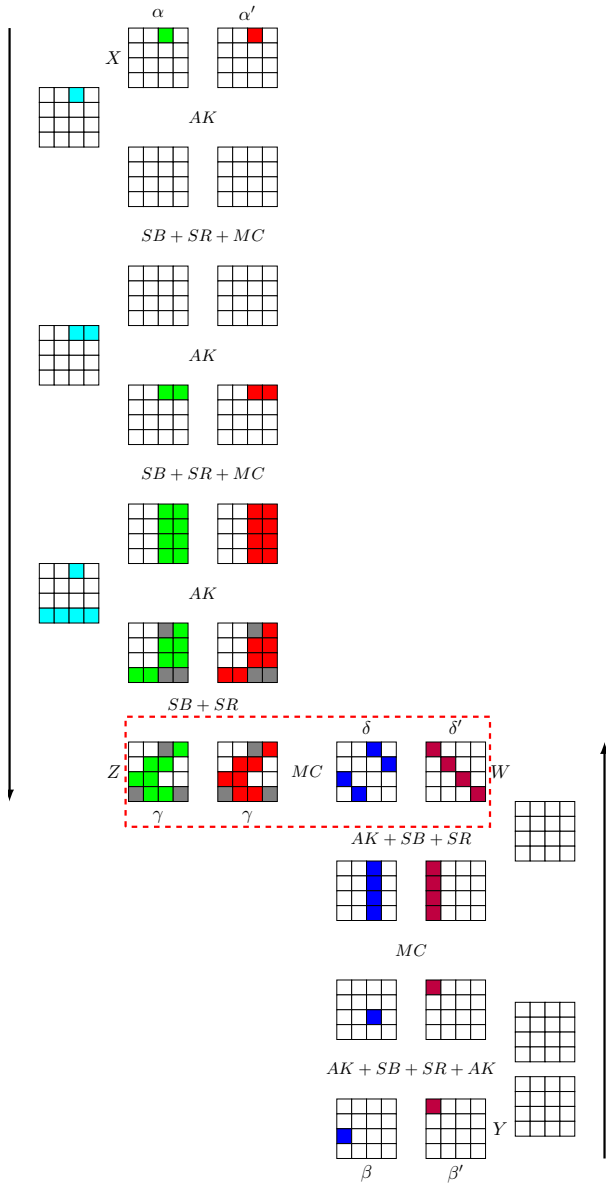

Fig. 36: One of 5-round RT_0^2-IBDs of AES-128

Verification (verify by contradiction). Assume (α, α') can propagate to (β, β'), as shown in Figure 36, for $X_0, X_1 = X_0 \oplus \alpha, X_2, X_3 = X_2 \oplus \alpha'$, and $Y_0, Y_1, Y_2 = Y_1 \oplus \beta, Y_3 = Y_0 \oplus \beta'$, let Z_i be the value obtained by encrypting X_i after 3 rounds without the last MC layer, and W_i be the value obtained by decrypting Y_i after 2 rounds. Then $Z_0 \oplus Z_1 = \gamma$, $Z_2 \oplus Z_3 = \gamma'$, $W_1 \oplus W_2 = \delta$, and $W_0 \oplus W_3 = \delta'$.

On the one hand, since

$$\begin{aligned}
&W_{1,0} \oplus W_{2,0} = 0, W_{0,0} \oplus W_{3,0} = \delta'_0 \neq 0,\\
&W_{1,1} \oplus W_{2,1} = 0, W_{0,1} \oplus W_{3,1} = 0,\\
&W_{1,2} \oplus W_{2,2} = \delta_2 \neq 0, W_{0,2} \oplus W_{3,2} = 0,\\
&W_{1,3} \oplus W_{2,3} = 0, W_{0,3} \oplus W_{3,3} = 0.
\end{aligned}$$

we have $W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} = \delta'_0 \neq 0$, $W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} = 0$, $W_{0,2} \oplus W_{1,2} \oplus W_{2,2} \oplus W_{3,2} = \delta_2 \neq 0$ and $W_{0,3} \oplus W_{1,3} \oplus W_{2,3} \oplus W_{3,3} = 0$.

On the other hand,

$$\begin{aligned}
\begin{pmatrix} W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} \\ W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} \\ W_{0,2} \oplus W_{1,2} \oplus W_{2,2} \oplus W_{3,2} \\ W_{0,3} \oplus W_{1,3} \oplus W_{2,3} \oplus W_{3,3} \end{pmatrix} &= M \cdot \begin{pmatrix} Z_{0,0} \oplus Z_{1,0} \oplus Z_{2,0} \oplus Z_{3,0} \\ Z_{0,1} \oplus Z_{1,1} \oplus Z_{2,1} \oplus Z_{3,1} \\ Z_{0,2} \oplus Z_{1,2} \oplus Z_{2,2} \oplus Z_{3,2} \\ Z_{0,3} \oplus Z_{1,3} \oplus Z_{2,3} \oplus Z_{3,3} \end{pmatrix}\\
&= M \cdot \begin{pmatrix} 0 \\ 0 \\ \gamma_2 \oplus \gamma'_2 \\ \gamma_3 \oplus \gamma'_3 \end{pmatrix}.
\end{aligned}$$

If $\gamma_2 \oplus \gamma'_2 = \gamma_3 \oplus \gamma'_3 = 0$, then $W_{0,i} \oplus W_{1,i} \oplus W_{2,i} \oplus W_{3,i} = 0 (0 \leq i \leq 3)$. There exists a contradiction. If $\gamma_2 \oplus \gamma'_2 \neq 0$ or $\gamma_3 \oplus \gamma'_3 \neq 0$, then according to the property of the MDS matrix, at least three of $i (0 \leq i \leq 3)$ such that $W_{0,i} \oplus W_{1,i} \oplus W_{2,i} \oplus W_{3,i} = 0$. There exists a contradiction. □

F.8 Verification of the 8-round RK-IBD of SPECK-32/64

Distinguisher 6

We choose the example of RK-IBD of SPECK-32/64 to verify.

Verification (verify by contradiction). We propagate the input differences 5 rounds in the forward direction, and the output differences 3 rounds in the backward direction. The differential propagation of the last 4-round is shown in Figure 37, where the $a_i, a'_i, b_i, b'_i, c_i, c'_i$ can be 0 or 1 and $b'_{11} = b'_8 \oplus 1$.

On the one hand, since $x_{0,9} \oplus x_{1,9} = a_0 = y_{0,9} \oplus y_{1,9}$ and $x_{2,9} \oplus x_{3,9} = a'_0 = y_{2,9} \oplus y_{3,9}$, $x_{1,9} \oplus x_{2,9} = 1$, $y_{1,9} \oplus y_{2,9} = 0$, $x_{0,9} \oplus x_{3,9} = c'_2$, $y_{0,9} \oplus y_{3,9} = 1$, we have $c'_2 = 0$. Meanwhile, since $x_{0,8} \oplus x_{1,8} = 1$ and $x_{2,8} \oplus x_{3,8} = 1$, $x_{1,8} \oplus x_{2,8} = 0$, we have $x_{0,8} \oplus x_{0,8} = c'_1 = 0$. Similarly, it holds that $c'_0 = 0, b'_{11} = 1, b'_7 = b'_6 = 0$.

On the other hand, for the second modular addition in the right of Fig 37, the least 3 bits of input of the modular addition is $(000, 100)$, thus $b'_6 = 1$. This is a contradiction.

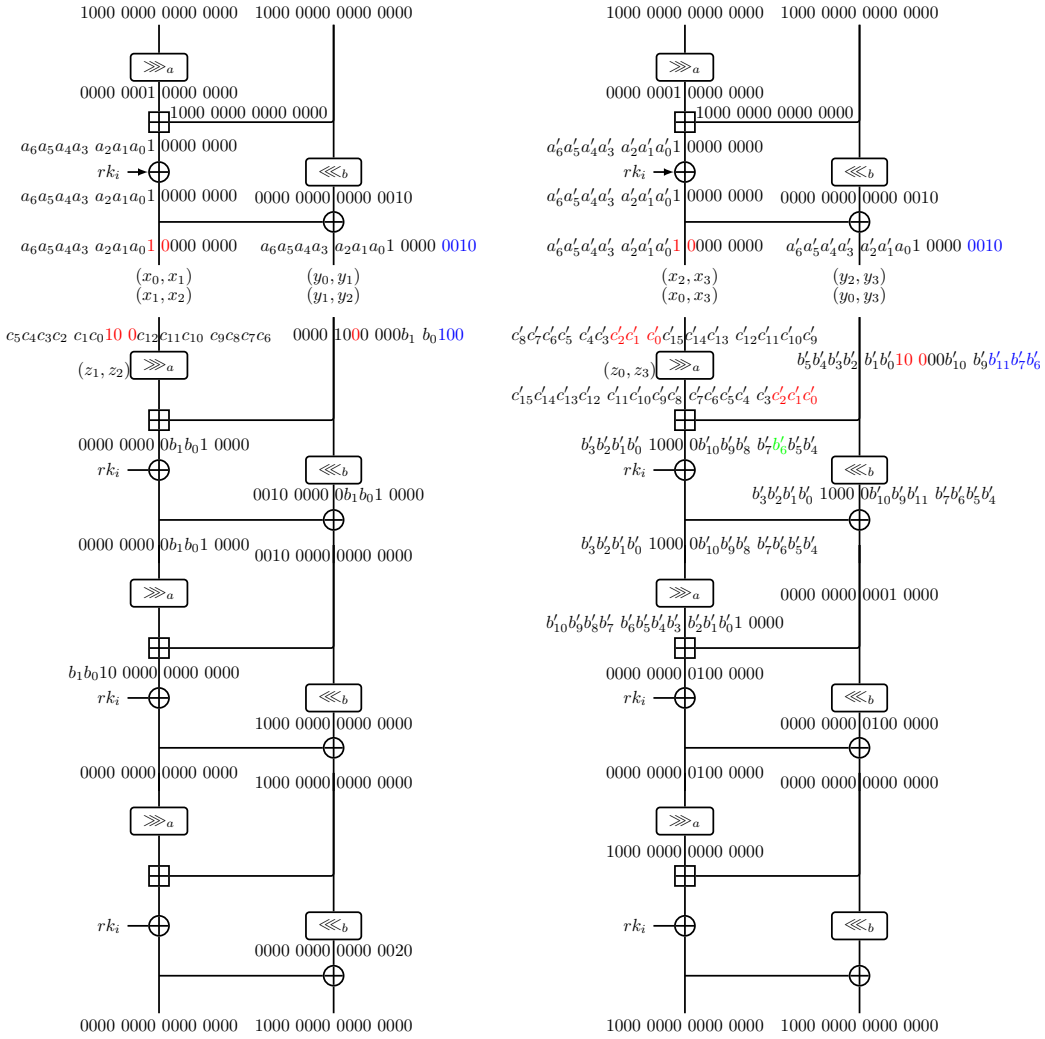

Fig. 37: The core of one 8-round RT_3^2-IBDs of SPECK-32/64

F.9 Verification of the 23-round RK-IBD of SKINNYee

Distinguisher 8

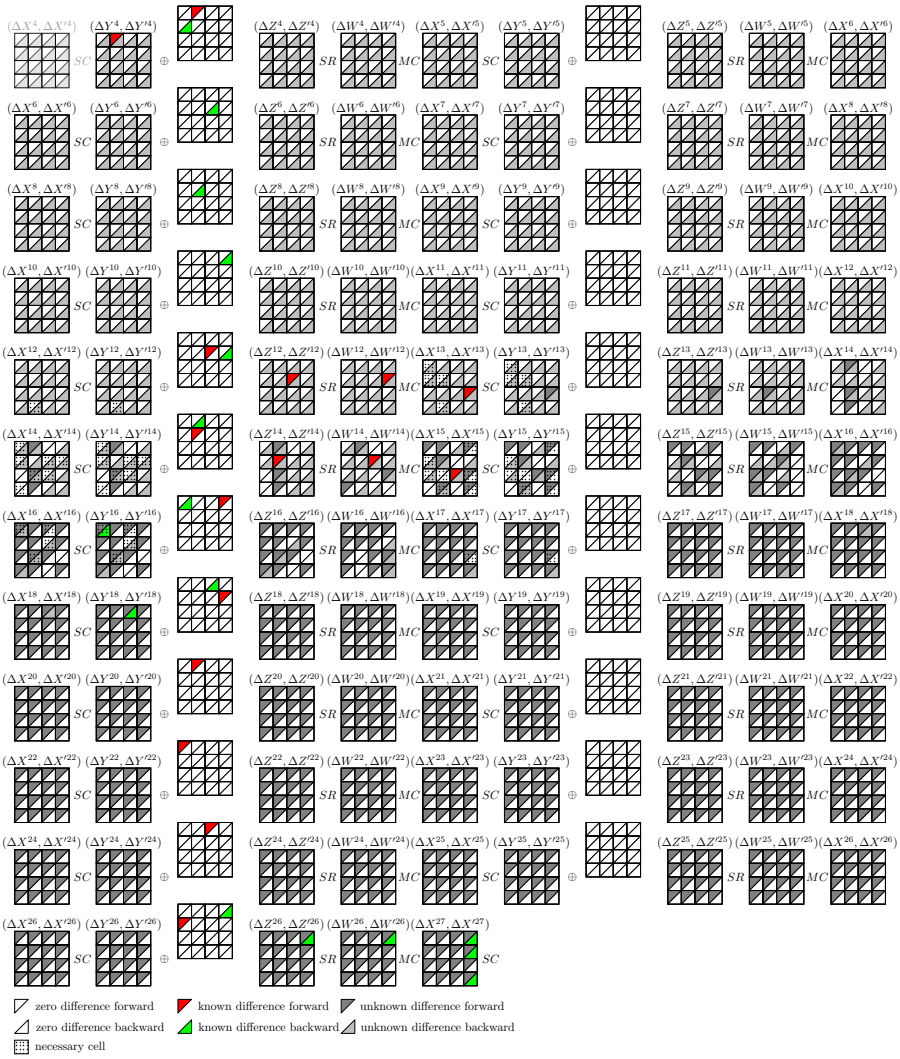

Fig. 38: One 23-round RT_3^2-IBD of SKINNYee

Verification. We employ the cross-validation method to verify this distinguisher. Since the distinguisher is detected via the propagation of the state, we refer to the GEBCTs for its verification. Specifically, we use the GEBCT to depict the propagation of the difference through the S-box. This approach enables us to

verify the distinguisher from the perspective of difference propagation. Eventually, we complete the verification process. Moreover, we remove the constraints of GEBCT of S-boxes as many as possible, and get the necessary cells as shown in Figure 38. As shown in the Figure 38, the positional pattern of contradictions is not obvious. Therefore, it is hard to construct this distinguisher by using GBCT or other tables alone in a certain round.

F.10 Verification of the 13-round RK-IBD of GIFT-64

Distinguisher 9

We choose the example of RK-IBD of GIFT-64 to verify.

Verification (verify by contradiction). Let (γ, γ') be the difference that propagates (α, α') 4-round in the forward direction, and (δ, δ') be the difference that propagates (β, β') 4-round in the backward direction. Then, according the key schedule, (α, α') propagates to $(\gamma, \gamma') = (0, 0)$ under key differences (κ_0, κ_1) with probability 1, and (β, β') propagates to $(\delta, \delta') = (0, 0)$ under key differences (κ_2, κ_3) with a probability of 1. Now, we show that (γ, γ') cannot propagate to the output differences (δ, δ') under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ after 5 rounds of GIFT-64.

Let $(x_i^0, x_i^1, x_i^2, x_i^3)$ and $(y_i^0, y_i^1, y_i^2, y_i^3)$ be the four states before and after the S-box layer at the round i. Let $x_{i,j}^t$ and $y_{i,j}^t$ be the j-th nibble value of x_i^t and $y_i^t (0 \leq j \leq 15, 0 \leq t \leq 3)$. First, we remove the constraints nibble by nibble in round 6 in our SAT model to detect the necessary nibble for generating contradictions. The contradiction occurs in the nibble i ($i \in \{2, 3, 7, 15\}$). Then, we propagate the input two differences 2 rounds in the forward direction, and propagate the output two differences 2 rounds in the backward direction, and whether those differences can be connected according to the GBCT of the S-box, i.e. we check whether $(x_{6,j}^0 \oplus x_{6,j}^1, x_{6,j}^2 \oplus x_{6,j}^3) \xrightarrow{GBCT} (y_{6,j}^1 \oplus y_{6,j}^2, y_{6,j}^0 \oplus y_{6,j}^3)$ ($j \in \{2, 3, 7, 15\}$). Through a simple Python program, we can remove most of the differential propagations.

The remaining differential propagation is $((y_4^0 \oplus y_4^1, y_4^2 \oplus y_4^3), (x_8^1 \oplus x_8^2, x_8^0 \oplus x_8^3)$, where $y_{4,0}^0 \oplus y_{4,0}^1 \in \{8, 9, 10, 11\}, y_{4,0}^2 \oplus y_{4,0}^3 \in \{8, 9, 10, 11\}$, and $x_{8,8}^1 \oplus x_{8,8}^2 \in \{5, 6, 7, 12, 13, 15\}$, $x_{8,8}^0 \oplus x_{8,8}^3 \in \{5, 6, 7, 12, 13, 15\}$. Thus, the bit 3 of $y_4^0 \oplus y_4^1$ and $y_4^2 \oplus y_4^3$ must be 1, and the bit 34 of $x_8^1 \oplus x_8^2$ and $x_8^0 \oplus x_8^3$ must be 1. That is, the differential propagation in the Figure 39 must be hold, where κ_i^t denotes the key difference of κ_t in round i. For the S-box 3 of round 6, if $(x_{6,3}^0 \oplus x_{6,3}^1, x_{6,3}^2 \oplus x_{6,3}^3) = (1, 1) \xrightarrow{GBCT} (y_{6,3}^1 \oplus y_{6,3}^2, y_{6,3}^0 \oplus y_{6,3}^3) = (8, 8)$, then $x_{6,3}^1 \oplus x_{6,3}^2, x_{6,3}^0 \oplus x_{6,3}^3) = (1, 1)$. Thus, for the S-box 12 of round 5, it must hold as follows:

$$\begin{cases} (x_{5,12}^0 \oplus x_{5,12}^1, x_{5,12}^2 \oplus x_{5,12}^3) = (8, 8) \xrightarrow{GBCT} (y_{5,12}^1 \oplus y_{5,12}^2, y_{5,12}^0 \oplus y_{5,12}^3) = (1, 1), \\ y_{5,12}^0 \oplus y_{5,12}^1 = 1, \\ y_{5,12}^2 \oplus y_{5,12}^3 = 1. \end{cases}$$

However, the above formula does not hold. Thus, for the remaining differential propagation, it still cannot hold. In conclusion, we have verified our distinguisher.

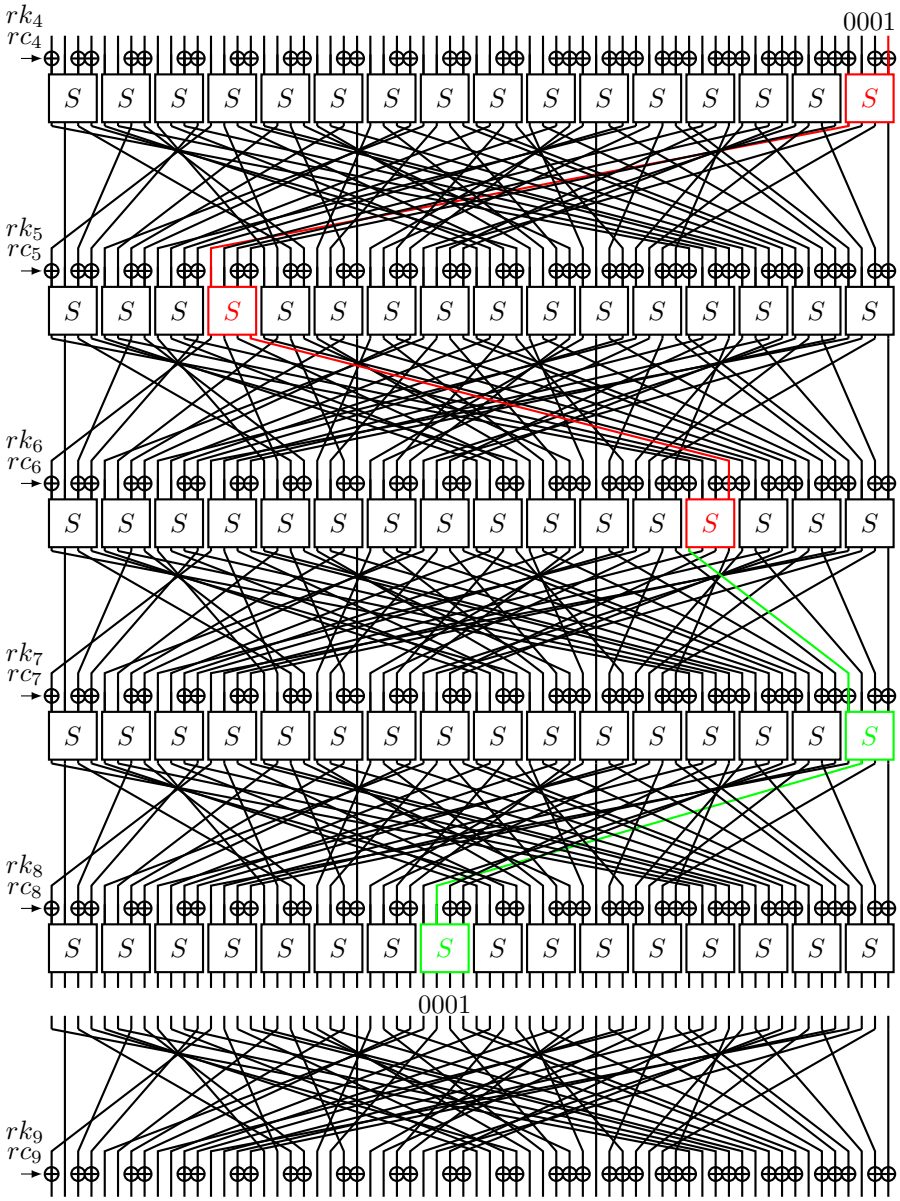

Fig. 39: The core of one 13-round RT_3^2-IBD of `GIFT-64`

F.11 Verification of the full-round RK-IBD of GOST

Distinguisher 11

Verification (verify by contradiction). As shown in Figure 40, Figure 41, Figure 42 and Figure 43. The input differences (α, α') can propagate to $((0x00000000, 0x80000000), (0x80000000, 0x00000000))$ with probability 1 after 23 rounds in the forward direction, and the output differences (β, β') can propagate to $((0x8000 0000, 0x00000000), (0x00000000, 0x80000000))$ with probability 1 after 7 rounds in the backward direction. Thus, we just need to verify $((0x00000000, 0x8000 0000), (0x80000000, 0x00000000)$ cannot propagate to $((0x80000000, 0x00000000), (0x00000000, 0x80000000)$ after 2 rounds under the key differences $\kappa_{0,7}$, $\kappa_{1,7}$, $\kappa_{2,7}$, and $\kappa_{3,7}$.

As shown in Figure 44, we have $x_1 = x_0, x_2 = x_0 \oplus 0x80000000$ and $x_3 = x_0$. On the one hand, since $y_0 \oplus y_1 \oplus y_2 \oplus y_3 = 0x80000000$, with the values of $L_{2,25}$ and $L_{3,25}$, we have $z_0 \oplus z_1 \oplus z_2 \oplus z_3 = 0$. On the other hand, we have $k_{1,7} = k_{0,7} \oplus 0x80000000$, $k_{2,7} = k_{3,7} = k_{0,7} \oplus 0xc0000000$. For a 32-bit value v, let v' be the most significant 4-bit, and S represents the S-box operating the most significant 4-bit. Then, we have

$$\begin{cases} z'_0 = S(x'_0 + k'_0), z'_1 = S(x'_0 + k'_0 \oplus 0x8), \\ z'_2 = S(x'_0 \oplus 0x8 + k'_0 \oplus 0xc), z'_3 = S(x'_0 + (k'_0 \oplus 0xc). \end{cases}$$

Then, $z'_0 \oplus z'_1 \oplus z'_2 \oplus z'_3 = 0$ cannot hold for S-box of both GOST-FB and GOST-PS. This is a contradiction.

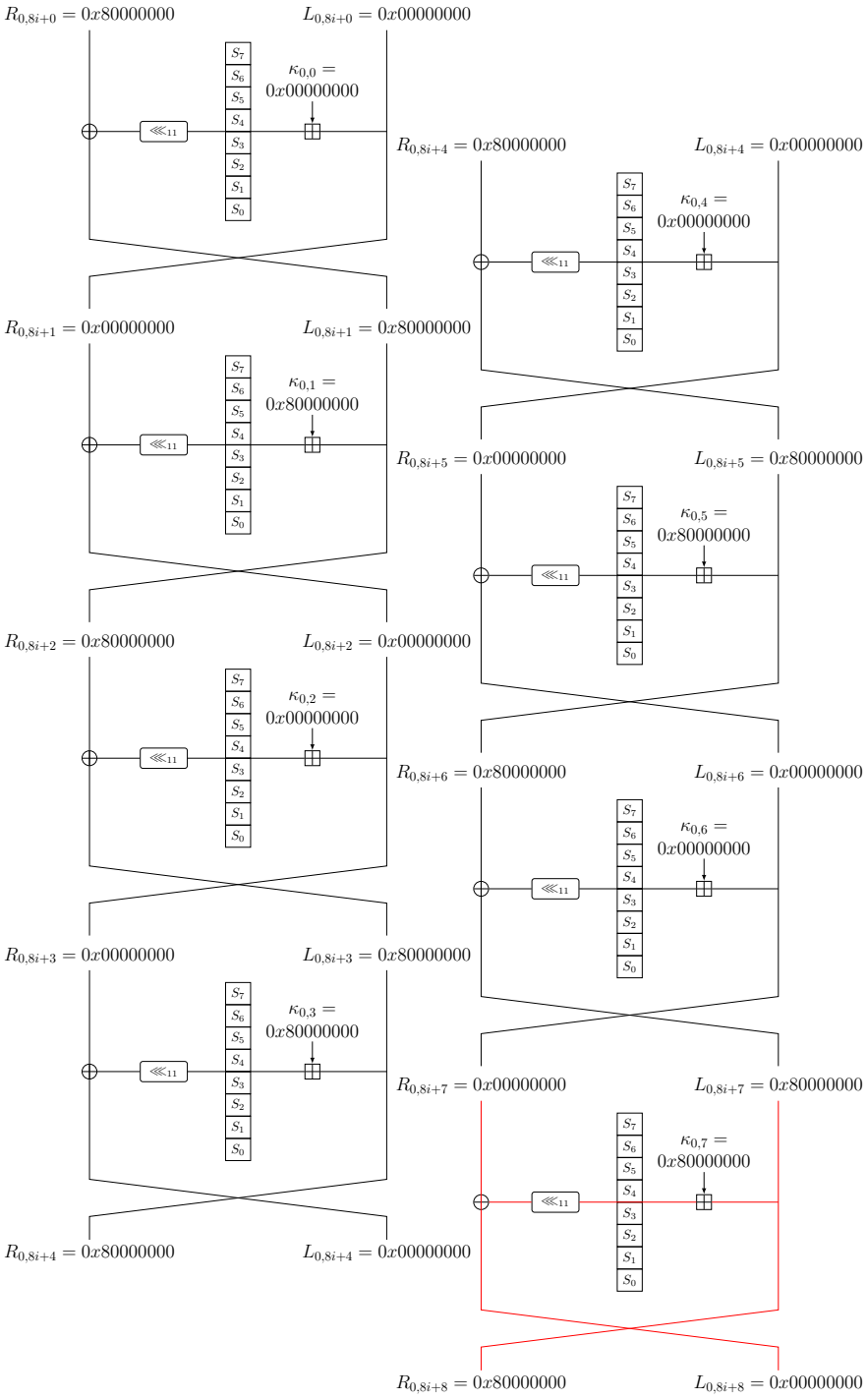

Fig. 40: The 24-round related-key differential of `GOST` by iterating above 3 times

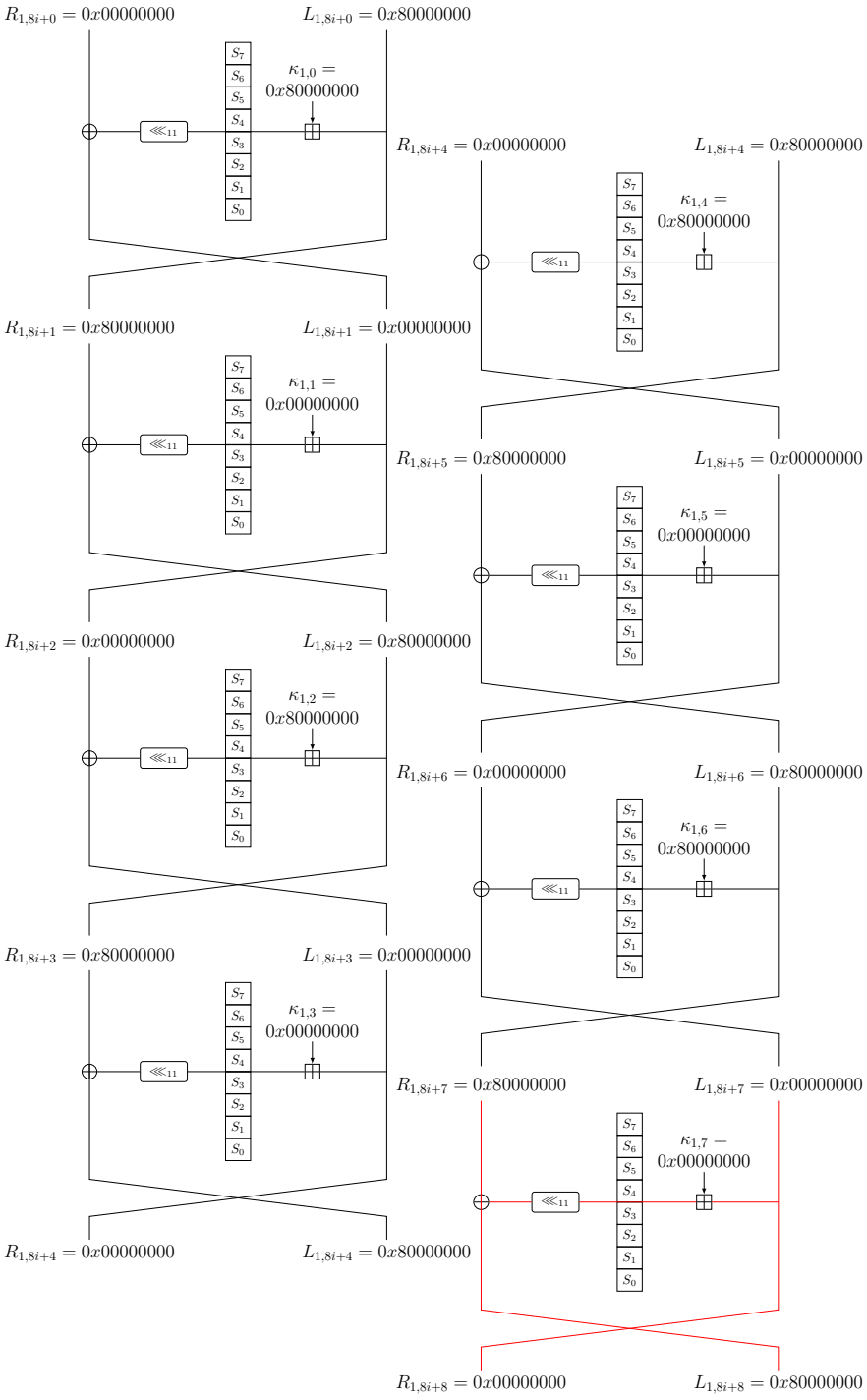

Fig. 41: The 24-round related-key differential of `GOST` by iterating above 3 times

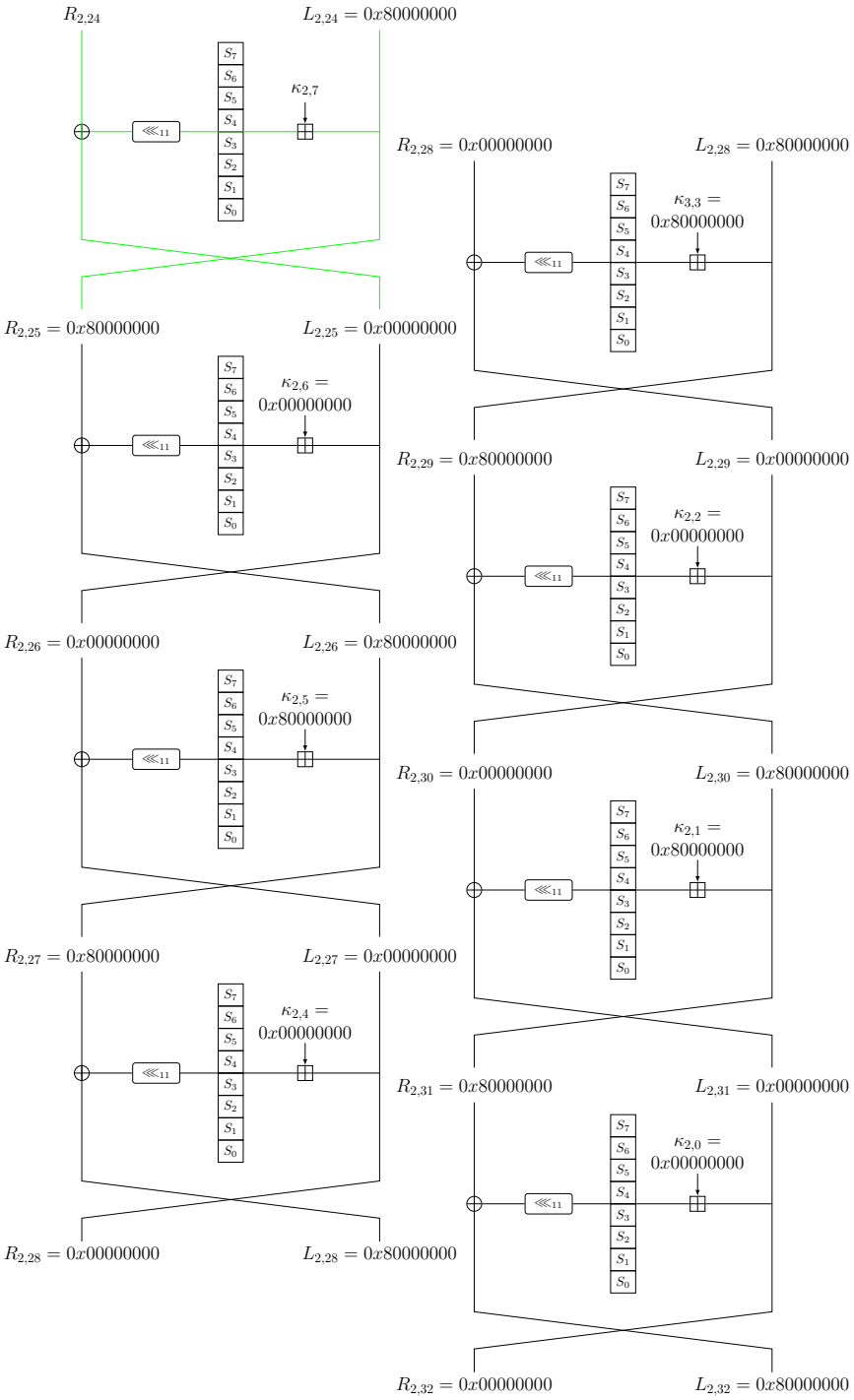

Fig. 42: The 7-round related-key differential of `GOST`

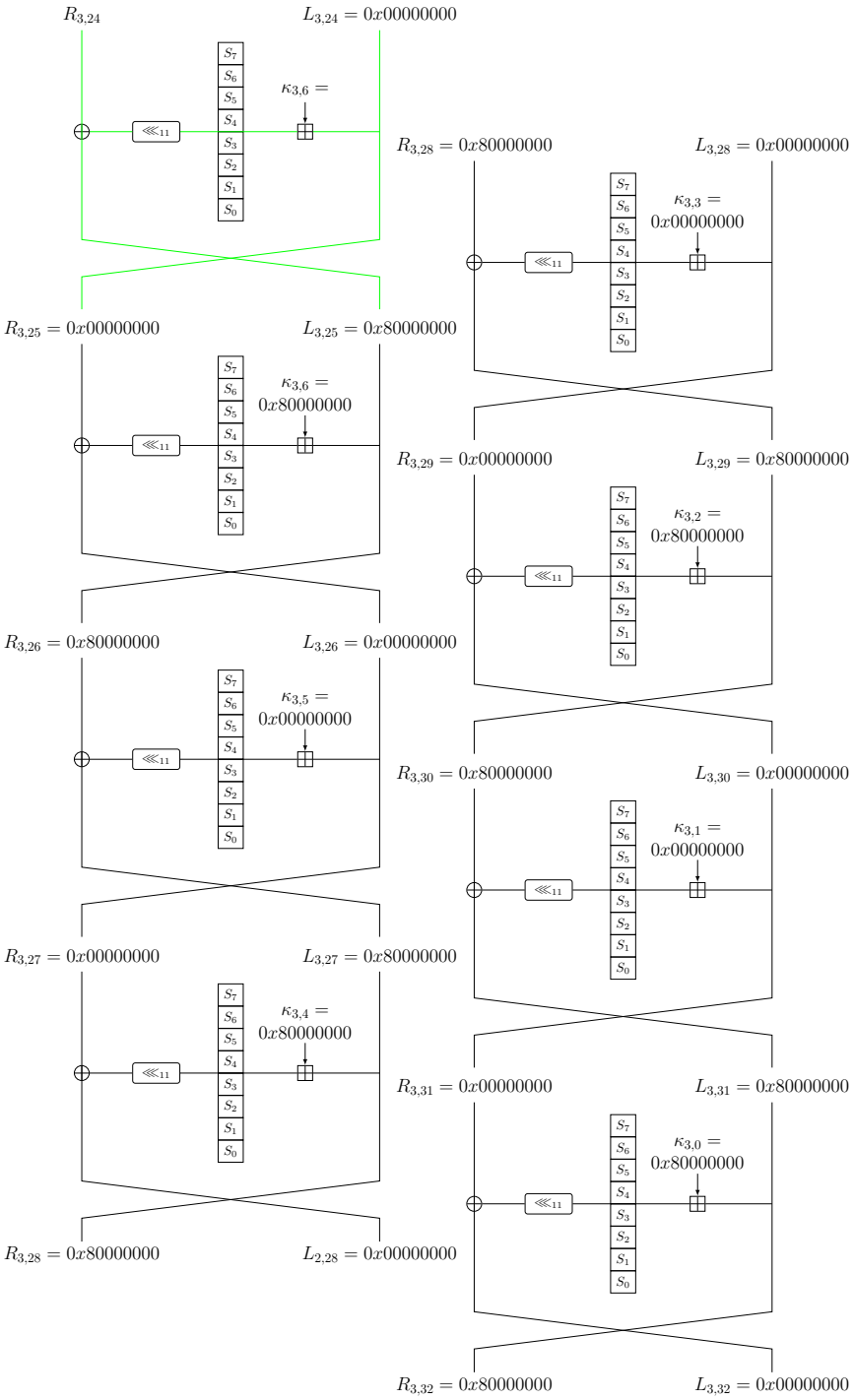

Fig. 43: The 7-round related-key differential of GOST

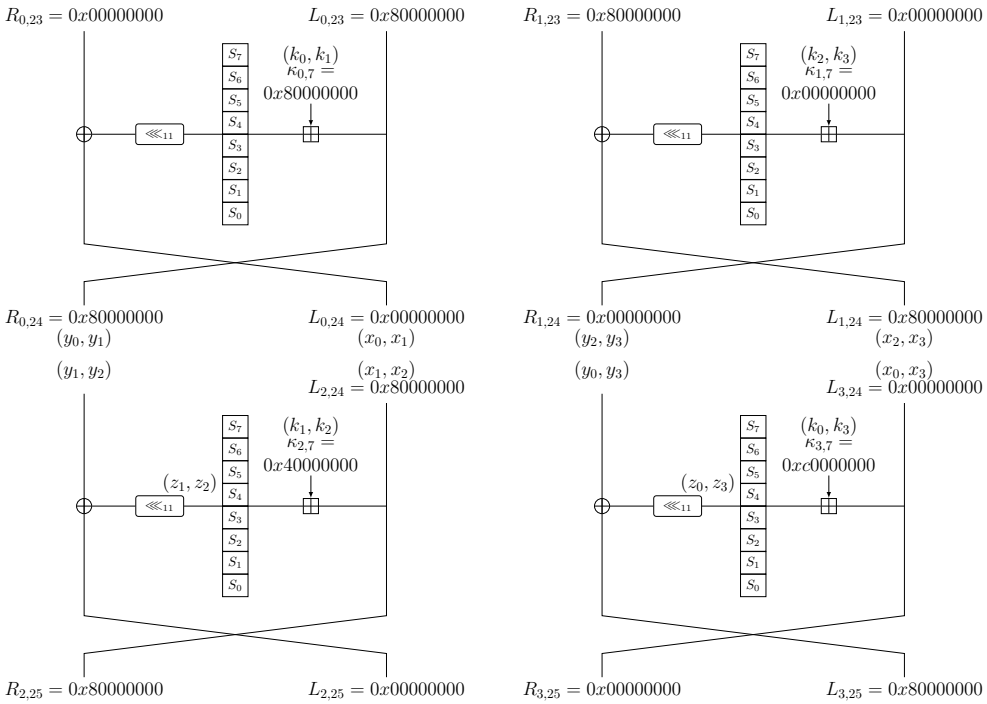

Fig. 44: The core of one full-round RK-IBDs of GOST

G Subtweakey Difference Cancellation for Tweakable Block Ciphers

In this section, we study the properties of subtweakey difference cancellation for `SKINNY` and `SKINNYee`. Those properties allow us to construct the 6-round and 8-round difference characteristics with a probability of 1. for `SKINNY` and `SKINNYee`, respectively. Here, we only give the proof of the property of `SKINNY`, as the other properties can be proved analogously.

Property 4 (SKINNY). *For `SKINNY`-n-3n, let $TKS_{r,i}(i = 0, 1, 2)$ be the tweakey states in round r, and TK_r be the tweakey that derived from $TKS_{r,i}(i = 0, 1, 2)$. For the round r_0, if only a single cell of ΔTK_{r_0} is active, and the difference of this cell is u, then there exist difference of $\Delta TKS_{r_0,i}(i = 0, 1, 2)$ and a value v, such that $\Delta TK_{r_0+i}(1 \leq i \leq 5)$ is inactive, and only a single cell of ΔTK_{r_0+6} is active, and the difference of this cell is v. Besides, in this case, for $\forall j \notin \{r_0, \ldots, r_0 + 6\}$ and $(j - r_0) \mod 2 = 0$, ΔTK_j is active.*

Proof. Let a_0, a_1 and a_2 be differences of the active cell of $TKS_{r_0,i}(i = 0, 1, 2)$, respectively. Since $LFSR_2$ and $LFSR_3$ are two different linear operations, the equation

$$\begin{cases} a_0 \oplus a_1 \oplus a_2 = u, \\ a_0 \oplus LFSR_2^1(a_1) \oplus LFSR_3^1(a_2) = 0, \\ a_0 \oplus LFSR_2^2(a_1) \oplus LFSR_3^2(a_2) = 0, \end{cases}$$

has only one solution for (a_0, a_1, a_2), where $LFSR_j^i(j = 2, 3)$ denoted the operation that applying $LFSR_j(j = 2, 3)$ $i(i > 0)$ times and applying the inverse of $LFSR_j(j = 2, 3)$ $i(i < 0)$ times. Then, it holds that $a_0 \oplus LFSR_2^i(a_1) \oplus LFSR_3^i(a_2) \neq 0(i \neq \{0, 1, 2\})$. Otherwise, we will have three independent linear equations for (a_0, a_1, a_2) equal 0, thus $(a_0, a_1, a_2) = (0, 0, 0)$, therefore $u = a_0 \oplus a_1 \oplus a_2 = 0$. This is a contradiction. Let $v = a_0 \oplus LFSR_2^3(a_1) \oplus LFSR_3^3(a_2)$, then $v \neq 0$ and v is determined by u. Consequently, in accordance with the tweakey schedule, we conclude our proof. □

Property 5 (SKINNYee). *For `SKINNYee`, let $TKS_{r,i}(i = 0, 1, 2, 3)$ be the tweak states in round r, and TK_r be the tweak that derived from $TKS_{r,i}(i = 0, 1, 2, 3)$. For the round r_0, if only a single cell of ΔTK_{r_0} is active, and the difference of this cell is u, then there exist difference of $\Delta TKS_{r_0,i}(i = 0, 1, 2, 3)$ and a value v, such that $\Delta TK_{r_0+i}(1 \leq i \leq 7)$ is inactive, and only a single cell of ΔTK_{r_0+8} is active, and the difference of this cell is v. Besides, in this case, for $\forall j \notin \{r_0, \ldots, r_0 + 8\}$ and $(j - r_0) \mod 2 = 0$, ΔTK_j is active.*

H Impossible Boomerang Attacks of 31-round SKINNYee

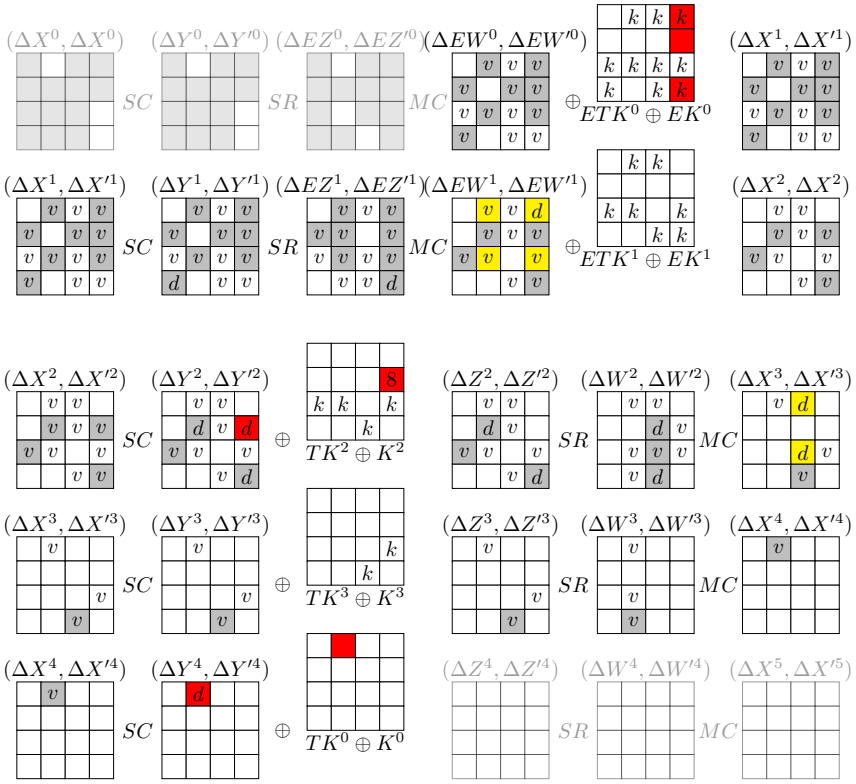

Fig. 45: Top 4 rounds added for key recovery in 31-round attack on SKINNYee

In this part, we propose a 31-round RK-IBA on SKINNYee, this is also the first 31-round attack. In details, we employ the 23-round IBD as Distinguisher 8, and add 4 additional rounds at the upper part and 4 additional rounds at the lower part. The details of these procedures are respectively illustrated in Fig 45 and Fig 46. In Fig 45 and Fig 46, a cell marked with "v" means that we need to know its value, while a cell marked with "d" means that we need to know its corresponding difference. In Fig 45, cells colored red and yellow indicate that the differences of these cells are utilized in the process of key sieving. Similarly, cells colored green and yellow in Fig 46 also signify that the differences of these cells are employed for key sieving. The notations used in this part are detailed as follows.

- $c = 4$ be the cell size.
- ΔK and ∇K: the tweak differences that correspond to Distinguisher 8.

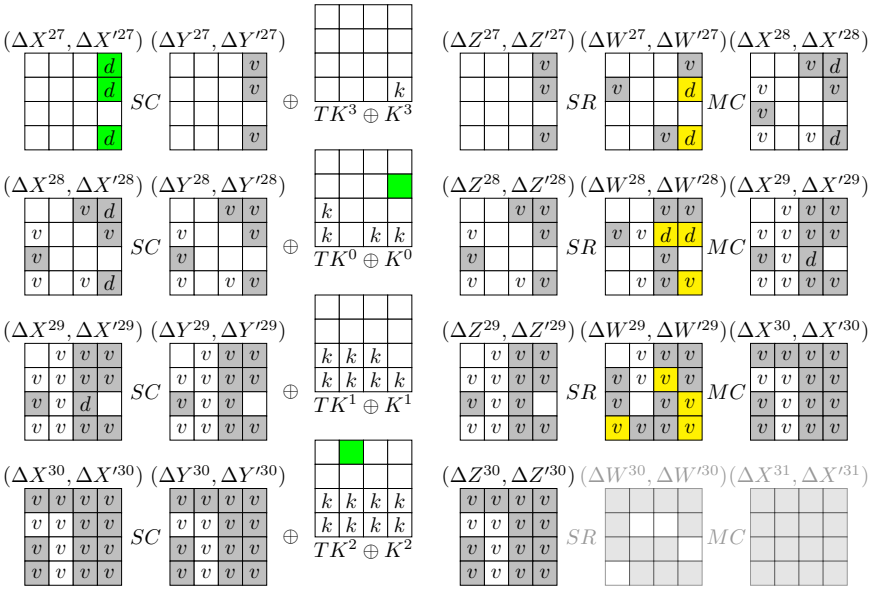

Fig. 46: Bottom 4 rounds added for key recovery in our 31-round attack on `SKINNYee`

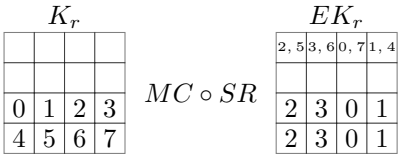

Fig. 47: The relation of K_r and its equivalent version EK_r of `SKINNYee`

Table 8: The known differences and the keys that need guessed of SKINNYee.

Difference	Keys that need guessed
$\Delta X_2^3, \Delta X_2'^3$	$K_{0,0}^0, K_{0,3}^0, K_{0,1}^0 \oplus K_{0,4}^0, K_{0,1}^1, K_{0,2}^1$
$\Delta X_{10}^3, \Delta X_{10}'^3$	$K_{0,0}^0, K_{0,3}^0 \oplus K_{0,6}^0, K_{0,2}^1$
$\Delta Y_1^4, \Delta Y_1'^4$	$K_{0,3} \oplus K_{0,6}, K_{0,0} \oplus K_{0,7}, K_{0,0}, K_{0,1}$ $K_{0,2}^0, K_{0,3}^0, K_{0,3}^1 \oplus K_{0,6}^1, K_{0,0}^1 \oplus K_{0,7}^1$ $K_{0,0}^1, K_{0,1}^1, K_{0,2}^1, K_{0,3}^1, K_{0,0}^2, K_{0,1}^2$ $K_{0,3}^2, K_{0,6}^2, K_{0,3}^3, K_{0,6}^3$
$\Delta W_6^{28}, \Delta W_6'^{28}$	$K_{0,2}^1, K_{0,6}^1, K_{0,3}^2, K_{0,4}^2, K_{0,5}^2, K_{0,7}^2$
$\Delta W_7^{27}, \Delta W_7'^{27}$	$K_{0,7}^0, K_{0,0}^1, K_{0,4}^1, K_{0,6}^1, K_{0,1}^2, K_{0,5}^2, K_{0,6}^2, K_{0,7}^2$
$\Delta W_{15}^{27}, \Delta W_{15}'^{27}$	$K_{0,7}^0, K_{0,6}^1, K_{0,0}^2, K_{0,4}^2, K_{0,5}^2$
$\Delta X_3^{27}, \Delta X_3'^{27}$	$K_{0,0}^1, K_{0,4}^1, K_{0,1}^2, K_{0,5}^2, K_{0,6}^2, K_{0,7}^2$
$\Delta X_7^{27}, \Delta X_7'^{27}$	$K_{0,0}^0, K_{0,4}^0, K_{0,1}^1, K_{0,5}^1, K_{0,6}^1, K_{0,7}^1, K_{0,2}^2, K_{0,3}^2, K_{0,4}^2, K_{0,5}^2, K_{0,6}^2, K_{0,7}^2$
$\Delta X_{15}^{27}, \Delta X_{15}'^{27}$	$K_{0,6}^0, K_{0,5}^1, K_{0,3}^2, K_{0,4}^2, K_{0,7}^2, K_{0,7}^3$

Table 9: The key recovery steps of the early abort technique of SKINNYee.

Step	Difference	Guessed keys	Time complexity	Remained quartets
1	$\Delta X_{10}^3, \Delta X_{10}'^3$	$K_{0,0}^0, K_{0,2}^1$	$2^{11c+2c} \cdot 2^{18c} \cdot (2 \cdot 4)/(16 \cdot 31)$	$2^{18c}/2^{2c} = 2^{16c}$
2	$\Delta X_2^3, \Delta X_2'^3$	$K_{0,1}^1$	$2^{13c+c} \cdot 2^{16c} \cdot 4/(16 \cdot 31)$	$2^{16c}/2^{2c} = 2^{14c}$
3	$\Delta W_6^{28}, \Delta W_6'^{28}$	$K_{0,6}^1, K_{0,3}^2, K_{0,5}^2$	$2^{14c+3c} \cdot 2^{14c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{14c}/2^{2c} = 2^{12c}$
4	$\Delta W_{15}^{27}, \Delta W_{15}'^{27}$	$K_{0,7}^0$	$2^{17c+c} \cdot 2^{12c} \cdot 4/(16 \cdot 31)$	$2^{12c}/2^{2c} = 2^{10c}$
5	$\Delta W_7^{27}, \Delta W_7'^{27}$ $\Delta X_3^{27}, \Delta X_3'^{27}$	$K_{0,0}^1, K_{0,4}^1, K_{0,1}^2$	$2^{18c+3c} \cdot 2^{10c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{10c}/2^{2 \cdot 2c} = 2^{6c}$
6	$\Delta X_7^{27}, \Delta X_7'^{27}$	$K_{0,5}^1$	$2^{21c+c} \cdot 2^{6c} \cdot 4/(16 \cdot 31)$	$2^{6c}/2^{2c} = 2^{4c}$
7	$\Delta X_{15}^{27}, \Delta X_{15}'^{27}$	$K_{0,7}^3$	$2^{22c+c} \cdot 2^{4c} \cdot 4/(16 \cdot 31)$	$2^{4c}/2^{2c} = 2^{2c}$
8	$\Delta Y_1^4, \Delta Y_1'^4$	$K_{0,3}^1, K_{0,3}^3, K_{0,6}^3$	$2^{23c+3c} \cdot 2^{2c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{2c}/2^{2c} = 2^0$

- $(X_0^r, X_1^r, X_2^r, X_3^r)$: the input quartet of SC in round r.
- $(Y_0^r, Y_1^r, Y_2^r, Y_3^r)$: the output quartet of SC in round r.
- $(Z_0^r, Z_1^r, Z_2^r, Z_3^r)$: the input quartet of SR in round r.
- $(W_0^r, W_1^r, W_2^r, W_3^r)$: the input quartet of MC in round r.
- $(\Delta X_0^r, \Delta X_0'^r)$: the two differences in the upper trails or the lower trails. In the upper trails, it holds that $X_0^r \oplus X_1^r = \Delta X_0^r$, $X_2^r \oplus X_3^r = \Delta X_0'^r$. In the lower trails, it holds that $X_1^r \oplus X_2^r = \Delta X_0^r$, $X_0^r \oplus X_3^r = \Delta X_0'^r$. Similar symbol hold for $(\Delta Y_0^r, \Delta Y_0'^r)$, $(\Delta Z_0^r, \Delta Z_0'^r)$ and $(\Delta W_0^r, \Delta W_0'^r)$.
- $(K_0^{r \bmod 4}, K_1^{r \bmod 4}, K_2^{r \bmod 4}, K_3^{r \bmod 4})$: the quartet of the round keys in round r.
- $X_{i,j}^r$: the j-th cell of X_i^r. Similar symbol hold for $Y_i^r, Z_i^r, W_i^r, K_i^{r \bmod 4} (i = 0, 1, 2, 3)$, and $\Delta X_i^r, \Delta Y_i^r, \Delta Z_i^r, \Delta W_i^r (i = 0, 1)$.

Furthermore, during the first two rounds of the attack, we utilize an equivalent round key (and analogously, an equivalent round tweak) corresponding to $MC(SR(K_r))$. Specifically, the states corresponding to the key K_r and its equivalent counterpart EK_r are presented as shown in Fig 47. Meanwhile, the input quartet and difference of MC in round r is denoted as $(EZ_0^r, EZ_1^r, EZ_2^r, EZ_3^r)$ and $(\Delta EZ^r, \Delta EZ'^r)$, and the output quartet and difference of MC in round r is denoted as $(EW_0^r, EW_1^r, EW_2^r, EW_3^r)$ and $(\Delta EW^r, \Delta EW'^r)$ $(r = 0, 1)$. The entire attack process is as follows.

1. For all 2^{16c} plaintexts, query the ciphertexts under four related tweaks: $(MK_0, MK_1, MK_2, MK_3) = (K, K \oplus \Delta K, K \oplus \Delta K \oplus \nabla K, K \oplus \nabla K)$. Denote T_i be the plaintext-ciphertext sets encrypted by $MK_i (i \in \{0, 1, 2, 3\})$.
2. Guess $K_{0,1}^0$, $K_{0,2}^0$, $K_{0,3}^0$, $K_{0,4}^0$, $K_{0,6}^0$, $K_{0,7}^1$, $K_{0,0}^2$, $K_{0,2}^2$, $K_{0,4}^2$, $K_{0,6}^2$, and $K_{0,7}^2$. Then,
   (a) For each $(EW_i^0, Z_i^{30}) \in T_i (i \in \{0, 1, 2, 3\})$, partial decrypt Z_i^{30} under the keys $K_{i,7}^1$, $K_{i,0}^2$, $K_{i,2}^2$, $K_{i,4}^2$, $K_{i,6}^2$, and $K_{i,7}^2$ to get the values $W_{i,6}^{29}, W_{i,11}^{29}, W_{i,12}^{29}, W_{i,15}^{29}, W_{i,7}^{28}$ and $W_{i,15}^{28}$. and store $C_i = (EW_i^0, Z_i^{30}, W_{i,6}^{29}, W_{i,11}^{29}, W_{i,12}^{29}, W_{i,15}^{29}, W_{i,7}^{28}, W_{i,15}^{28})$ into a new table T_i'.
   (b) Let $RK_i^0 = ETK_i^0 \oplus EK_i^0 (i = 0, 1, 2, 3)$, then $RK_{i,1}^0, RK_{i,3}^0, RK_{i,9}^0, RK_{i,11}^0$ and $RK_{i,12}^0$ are known as the keys $K_{0,1}^0$, $K_{0,2}^0$, $K_{0,3}^0$, $K_{0,4}^0$, $K_{0,6}^0$ are known. For $x^0 = (x_1^0, x_3^0, x_4^0, x_6^0, x_9^0, x_{11}^0, x_{12}^0) \in (\mathbb{F}_2^4)^7$ and $x^1 = (x_1^1, x_3^1, x_4^1, x_6^1, x_9^1, x_{11}^1, x_{12}^1) \in (\mathbb{F}_2^4)^7$, find all (x^0, x^1) such that

$$\begin{cases} (S(x_1^0 \oplus RK_{0,1}^0) \oplus S(x_1^1 \oplus RK_{1,1}^0)) \oplus (S(x_{11}^0 \oplus RK_{0,11}^0) \\ \qquad \oplus S(x_{11}^1 \oplus RK_{1,11}^0)) = 0, \\ (S(x_3^0 \oplus RK_{0,3}^0) \oplus S(x_3^1 \oplus RK_{1,3}^0)) \oplus (S(x_9^0 \oplus RK_{0,9}^0) \\ \qquad \oplus S(x_9^1 \oplus RK_{1,3}^0)) \oplus (S(x_{12}^0 \oplus RK_{0,12}^0) \oplus S(x_{12}^1 \oplus RK_{1,12}^0)) = 0, \\ (S(x_4^0) \oplus S(x_4^1)) \oplus (S(x_{11}^0 \oplus RK_{0,11}^0) \oplus S(x_{11}^1 \oplus RK_{1,11}^0)) = 0, \\ (S(x_6^0) \oplus S(x_6^1)) \oplus (S(x_9^0 \oplus RK_{0,9}^0) \oplus S(x_9^1 \oplus RK_{1,9}^1)) = 0, \\ S(S(x_3^0 \oplus RK_{0,3}^0)) \oplus S(S(x_3^1 \oplus RK_{1,3}^0)) = 0x8, \end{cases}$$

   and store them into a table T_p^0. As a result, for a fix x^0, there are 2^{2c} corresponding x^1. Adopt the same steps to make table T_p^1 under the keys RK_2^0 and RK_3^0.

(c) For each plaintext EW_0^0, construct EW_1^0 such that $((EW_{0,1}^0, EW_{0,3}^0, EW_{0,4}^0, EW_{0,6}^0, EW_{0,9}^0, EW_{0,11}^0, EW_{0,12}^0), (EW_{1,1}^0, EW_{1,3}^0, EW_{1,4}^0, EW_{1,6}^0, EW_{0,9}^1, EW_{1,11}^0, EW_{1,12}^0)) \in T_p^0$, $EW_{1,7}^0 \in \mathbb{F}_2^4$, and the remained cell of EW_1^0 are the same as EW_0^0. Similarly, conduct the same operation to get (EW_2^0, EW_3^0). Finally, two lists of size $2^{16c+3c} = 2^{19c}$ are constructed.

$$L_0 = \{(EW_0^0, C_0, EW_1^0, C_1)\},$$
$$L_1 = \{(EW_2^0, C_2, EW_3^0, C_3)\}.$$

(d) Insert L_0 into a hash table H indexed by the $20c$ bits in the cells $Z_{0,4}^{30}, Z_{0,5}^{30}, Z_{0,9}^{30}, Z_{0,13}^{30}, W_{0,6}^{29}, W_{0,11}^{29}, W_{0,12}^{29}, W_{0,15}^{29}, W_{0,7}^{28}, W_{0,15}^{28}$ and $Z_{1,4}^{30}, Z_{1,5}^{30}, Z_{1,9}^{30}, Z_{1,13}^{30}, W_{1,6}^{29}, W_{1,11}^{29}, W_{1,12}^{29}, W_{1,15}^{29}, W_{1,7}^{28}, W_{1,15}^{28}$. For each $(EW_2^0, C_2, EW_3^0, C_3) \in L_1$, lookup H to find the the corresponding $(EW_0^0, C_0, EW_1^0, C_1)$ such that

$$\begin{cases} Z_{1,j}^{30} \oplus Z_{2,j}^{30} = 0, Z_{0,j}^{30} \oplus Z_{3,j}^{30} = 0, (j = 4, 5, 9, 13), \\ W_{1,j}^{29} \oplus W_{2,j}^{29} = 0, W_{0,j}^{29} \oplus W_{3,j}^{29} = 0, (j = 6, 11, 12, 15), \\ W_{1,j}^{28} \oplus W_{2,j}^{28} = 0, W_{0,j}^{28} \oplus W_{3,j}^{28} = 0, (j = 7, 15). \end{cases}$$

Finally, there are $Q = 2^{2 \cdot 19c - 2 \cdot 10c} = 2^{18c}$ quartets can be constructed.

(e) Guess the keys and employ the early abort technique [38] to eliminate incorrect keys. In detail, with the early abort technique, we use the known differences in Table 8 to recovery the key step by step. In Table 8, we also summarize the keys that need guessed to get the known differences. Finally, the key recovery steps is detailed in Table 9.

3. Exhaustively search the remaining key.

Complexity. The number of all keys that need guessed is 2^{26c}. After the early abort technique, the probability of not discarding a key is $p = (1-2^{-18c})^Q = e^{-1}$. The data complexity is $2^2 \cdot 2^{16c} = 2^{66}$. For the time complexity:

- Cost of step 2(a), for each $(EW_i^0, Z_i^{30})(i = 0, 1, 2, 3)$, 14 S-boxes are used in partial decryption. Thus, the time complexity is $2^2 \cdot 2^{11c} \cdot 2^{16c} \cdot 14/(16 \cdot 31) \approx 2^{104.86}$.
- Cost of step 2(b), 22 S-boxes are employed to process the $(2^{7c})^2$ data, the time complexity is $2 \times 2^{11c} \times 2^{14c} \cdot 22/(16 \cdot 31) \approx 2^{97.51}$.
- Cost of step 2(c), the time complexity is $2 \times 2^{11c} \times 2^{19c} = 2^{121}$.
- Cost of step 2(d), the time complexity is $2^{11c} \times 2^{18c} = 2^{116}$.
- Cost of step 2(e), as shown in Table 9, the time complexity is mainly consumed in step 1, 3, and 5. Thus, the time complexity is $2^{31c} \cdot (2 \cdot 4)/(16 \cdot 31) + 2 \cdot 2^{31c} \cdot (3 \cdot 4)/(16 \cdot 31) \approx 2^{120.1}$.
- Cost of step 4, the time complexity is $2^{128}/e \approx 2^{126.6}$.

During the entire process, the largest memory consumption is for storing the keys in step 2(e). As we guess 2^{26c} keys in step 2(e), the memory complexity is 2^{104}.

All in all, we successfully attacked 31-round SKINNYee, and the data complexity is 2^{66}, the time complexity is $2^{126.6}$, and the memory complexity is 2^{104}.

Beyond full-codebook attack. As SKINNYee takes an $16c$-bit plaintext as input and a $(4 \times 16c)$-bit tweak, it is reasonable to assume that an attacker may have available an amount of data with $D > 2^{16c}$ to carry out an attack, as long as $D \leq (2^{16c+4\times16c})$. The attack in such a setting is called the beyond full-codebook attack [12]. If we choose $2^{16c+\tau}$ plaintexts, then the data complexity is $2^2 \cdot 2^{16c+\tau} = 2^{66+\tau}$. For the time complexity:

- Cost of step 2(a), the time complexity is $2^2 \cdot 2^{11c} \cdot 2^{16c+\tau} \cdot 14/(16 \cdot 31) \approx 2^{104.86+\tau}$.
- Cost of step 2(b), 22 S-boxes are employed to process the $(2^{7c})^2$ data, the time complexity is $2 \times 2^{11c} \times 2^{14c} \cdot 22/(16 \cdot 31) \approx 2^{97.51}$.
- Cost of step 2(c). In this step, two lists of size $2^{16c+\tau+3c} = 2^{19c+\tau}$ are constructed. Thus, the time complexity is $2 \times 2^{11c} \times 2^{19c+\tau} = 2^{121+\tau}$.
- Cost of step 2(d). In this step, there are $Q' = 2^{2\cdot(19c+\tau)-2\cdot10c} = 2^{18c+2\tau}$ quartets are constructed. Thus, the time complexity is $2^{11c} \times 2^{18c+2\tau} = 2^{116+2\tau}$.
- Cost of step 2(e), as $Q'/Q = 2^{2\tau}$, the time complexity is $2^{120.1+2\tau}$. After this step, the probability of not discarding a key is $p' = (1 - 2^{-18c})^{Q'} = e^{-2^{2\tau}}$.
- Cost of step 4, the time complexity is $2^{128}/e^{2^{2\tau}}$.

During the entire process, the largest memory consumption is for storing the keys in step 2(e). As we guess 2^{26c} keys in step 2(e), the memory complexity is 2^{104}.

Finally, we choose $\tau = 1.01$ to attack 31-round SKINNYee, the data complexity is $2^2 \cdot 2^{16c+1.01} = 2^{67.01}$, the time complexity is $2^{121+1.01} + 2^{120.1+2\cdot1.01} + 2^{128}/e^{2^{2\cdot1.01}} \approx 2^{123.68}$, and the memory complexity is 2^{104}.