Multi-round Dependency Identification: Theoretical Construction and Automatic Search of Impossible Boomerang Distinguishers

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

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

Abstract. The impossible boomerang (IB) attack is one of the most powerful cryptanalytic techniques for block ciphers, with notable successes against AES. However, constructing IB distinguishers—the core component of IB attack—remains a fundamental challenge: existing methods are restricted to specific cipher structures, cannot detect contradictions spanning more than two rounds, and lack a unified framework across attack settings. To solve these issues, this paper introduces three hierarchically complementary methods for constructing IB distinguishers: (1) a GEBCT-based method breaking the 2-round contradiction barrier; (2) a state-based method proven exactly equivalent to the definition of IB distinguishers (the most precise available); and (3) a structural method for fast security evaluation. We prove their inclusion relations, unify all prior methods, and connect the state-based method to the Quasidifferential Framework within IB distinguishers.

Furthermore, we develop a unified SAT/SMT-based framework that automatically searches distinguishers under single-key, two-related-key, and four-related-key settings. We evaluate on seven representative block ciphers covering almost major design paradigms: We present the first IB distinguishers for ARX cipher SPECK (ISO/IEC standard) and CHAM, bit-permutation cipher GIFT-64 (CHES'17, an example with 5-round contradictions), and key-dependent Sbox cipher PRINTcipher48 (CHES'20). We obtain the first 5-round related-key IB distinguishers for AES-128 (NIST's standard with nonlinear key schedule, first automatically searched), and 1-round improvements over prior IB distinguishers for tweakable cipher SKINNY-64/192 (CRYPTO'16) and SKINNYee (CRYPTO'22). All IB distinguishers outperform the best impossible differentials by 1–4 rounds, including a PRINTcipher48 result that disproves the long-standing claim about single-key IB distinguisher limitations. Finally, we present a 31-round related-key IB attack on SKINNYee, the best result to date.

Keywords: Impossible Boomerang· Single/related-key· Block cipher

1 Introduction

The Impossible Boomerang (IB) attack elaborately combines the ideas of the Impossible Differential (ID) attack [1,2] and the Boomerang attack [3], forming a powerful symmetric cryptanalysis method. At the core of an IB attack lies the IB distinguisher (denoted SK-IB for single-key and RK-IB for related-key settings), which is a boomerang distinguisher guaranteed to occur with probability zero. Once an optimal IB distinguisher is constructed, it can be extended at both ends to eliminate wrong key candidates. Consequently, the development of precise, general, and automatic methods for IB distinguisher construction is critical to advancing the state-of-the-art of IB attacks.

In 2008, Lu first proposed the concept of IB attack [4,5]. Lu's IB distinguisher construction method divides a block cipher into two subciphers and leverages four independent probability-1 differentials in forward and backward directions. A contradiction is generated when the XOR of the intermediate differences is non-zero. Lu applied the IB attack to AES, achieving attacks on 6-round AES-128, 7-round AES-192 and 7-round AES-256 in single-key scenarios, as well as 8-round AES-192 and 9-round AES-256 in related-key scenarios, based on manually obtained 4-round SK-IB distinguishers and 6-round RK-IB distinguishers. The debut of IB attacks immediately demonstrated its strength.

Then followed a decade-long period of limited progress in IB attack research, due to the lack of effective automatic search methods. Choy and Yap introduced $\mathcal{UB}$-method [6] for searching Lu's construction, which uses matrix manipulation criteria to characterize differential propagation roughly. However, Lu's construction suffers from a fundamental limitation: it assumes independence between the two subciphers, leading to the omission of valid IB distinguishers.

The independence assumption has long plagued conventional boomerang attacks as well [7]. This issue was resolved with the introduction of boomerang connective tables, including the Boomerang Connectivity Table (BCT), Upper BCT (UBCT), Lower BCT (LBCT), and Double BCT (DBCT) [8,8,9,8]. Due to this inspiration, two independent works by Bonnetain et al. and Zhang et al. [10,11] in 2024 proposed modern IB distinguisher construction methods following the sandwich framework [12,13]. These methods adopt a lightweight two-stage technique: First, derive definite upper and lower differential trails (differential propagation for the subciphers) to identify active Sboxes with known input-output differences in the middle rounds; Second, consult (G)BCT to detect contradictions within a single round or DBCT(*) across two rounds. Here, Bonnetain et al. used a CP-based approach to achieve a 29-round RK-IB attack on SKINNYee [14] based on an actual 22-round RK-IB distinguisher[5], while Zhang et al. employed a MIQCP-based approach to attack three tweakable block ciphers Deoxys-BC [15], Joltik-BC [16], and SKINNY [17]. Furthermore, Bonnetain et al. constructed a 23-round RK-IB distinguisher for SIMON-32/64 [18]—a lightweight Feistel cipher with quadratic round functions—using the Feistel Boomerang Connectivity Ta-

[5] The result in [10] demonstrated a 21-round RK-IB distinguisher, which can be extended by one additional round when excluding the first SC operation in SKINNYee.

ble (FBCT) to detect one-round contradictions and employ an SMT solver for search. These works mark a significant breakthrough in the study of IB attacks, and demonstrate the strong power of IB attack, especially in related-key scenarios for tweakable block ciphers. However, compared with the ID attack—a long-standing and highly effective cryptanalytic method that has been extensively studied—the IB attack demonstrates more advantages, but it receives less attention and has a more limited development.

Focus on the existing construction and automatic search methods for IB distinguisher, they share three critical inherent limitations:

- **2-Round contradiction span bottleneck.** They are exclusively limited to 2-round contradictions. Contradictions across 3 or more rounds remain undetectable, leaving significant room for improvement in IB distinguisher length.
- **Rough cipher modeling.** Existing methods use lightweight models that neglect critical details: the $\mathcal{UB}$-method ignores even Sbox and linear layer details; table-based methods only capture Sbox differential properties locally with known input-output differences outer the search process; all exclude key schedules and round constants. These simplifications cause many valid IB distinguishers to be overlooked.
- **Poor generality.** Table-based methods are incompatible with ciphers featuring nonlinear key schedules, as the output differences become unpredictable. DBCT(*)-based methods are not applicable to ciphers with bit permutation layers like GIFT or ARX ciphers, which are equivalent to using a super-Sbox.

Our contribution. Addressing these limitations, we present a unified construction and search framework for IB distinguisher.

Firstly, to break the 2-round barrier, we formally define the GEBCT for precise characterization of arbitrary-round quartet-differential propagation. We prove that all existing IB distinguisher construction methods are special cases of our GEBCT-based method, unifying the theoretical foundation of IB distinguisher construction. We resolve the GEBCT application bottleneck via on-the-fly state modeling, where constraint size no longer scales with GEBCT entries for 8 variables. We achieve several multi-round contradiction-span IB distinguishers, such as 5-round contradictions in GIFT, enabling longer attacks.

Secondly, to establish the tightest IB distinguisher construction method, we propose the state-based method via quartet-state propagation, and rigorously prove its exact equivalence to the original IB distinguisher definition, making it the most precise method available. This method can be effectively transformed into the first full-detail automatic IB distinguisher search model incorporating Sboxes, linear layers, key schedules, and round constants. We also establish its equivalence to the Quasi-Differential Framework (QDFM). This finding illustrates the importance of taking key schedules and constant settings into account apart from the differential properties of Sboxes from another perspective. This method serves as a sound foundation for proving resistance of a block cipher against IB distinguishers, and also a standard verification tool for distinguishers derived from other approaches.

Table 1: Comparison of IB distinguisher construction methods

Method	Contradiction	Application	Precision	Function
Lu's	0-round: $\delta \oplus \delta' \oplus \gamma \oplus \gamma' \neq 0$	General	Sbox details neglected, linear layer described word-oriented	Manual derivation
(G)BCT-based	1-round: (G)BCT$(\cdot) = 0$	Linear key schedule	Sbox differential properties described by outer table look-up with known differences	Lightweight search
DBCT(*)-based	2-round: DBCT$^*(\cdot) = 0$			
GEBCT-based	multi-round	SPN/ Feistel/ ARX/ And-RX/ ("Key-dependent" is exclusively for State-based)	Sbox and linear layer details described (modular addition treated as Sboxes)	Optimal efficiency-precision tradeoff
State-based	multi-round		All details described (key schedule/ round constants)	1) Exact search; 2) Security proof; 3) Automatic verification
Structural	multi-round	Bijective Sbox	Linear layer described bit-level	1) Efficient search 2) Structural evaluation

Thirdly, to maximize efficiency, we define the $\overline{\text{GEBCT}}$ based on *arbitrary Sbox* mode [19], and provide a structural method that accounts for only impossible differential properties through Sboxes and full linear layer details. In summary, we present three hierarchically complementary methods covering all use cases: structural method for fast structural security evaluation, GEBCT-based method for optimal precision-efficiency balance, and state-based method for optimal distinguishers and security proofs. We rigorously prove the inclusion relationship: "Structural/Existing $\subseteq$ GEBCT-based $\subseteq$ State-based = `IB` distinguisher definition", providing clear guidance for method selection. A comparison of these three new methods is presented in Table 1.

Furthermore, (1) we disprove the long-standing `IB-ID` distinguisher misconception that a `SK-IB` distinguisher implies a same-round `SK-ID` distinguisher (denoted `SK-ID` for single-key and `RK-ID` for related-key settings). Its proof only holds for Lu's original method. We present a concrete counterexample, for `PRINTcipher48`, the longest `SK-IB` distinguisher exceeds the longest `SK-ID` distinguisher by one round. (2) Since the construction of `RK-IB` distinguishers remains reliant on empirical heuristics rather than systematic guidance, we analyze two related-key scenarios (`two-related-keys` and `four-related-keys` settings), clarifying their applicability conditions and advantages for block ciphers with nonlinear or linear key schedules. We also detail their search procedures for the upper/lower trail and core `IB` distinguisher within the sandwich framework.

Based on our three new IB distinguisher construction methods, we propose a unified SAT/SMT-based automatic search framework, supporting all kinds of block ciphers including SPN, Feistel, and ARX ciphers, in three settings of single-key, two-related-keys, and four-related-keys. We propose a two-layer verification strategy with model automatic-verification and code self-feedback techniques that ensure the soundness of automatically found distinguishers.

We demonstrate the universal applicability of our three complementary IB distinguisher construction methods across seven representative block ciphers covering all major design paradigms, achieving unprecedented or better results (given in Table 2 with detailed runtime in Table 2):

- Our GEBCT-based method enables the **first** IB distinguisher search for **ARX** ciphers and **bit-permutation** ciphers, as well as block ciphers with **non-linear key schedules** that cannot be processed using current table-based methods. Applied to SPECK [18] (two-related-keys setting) and GIFT-64, CHAM [23], SKINNY-64/192 [17], SKINNYee [14] (four-related-keys setting), it yields RK-IB distinguishers that are **1–4 rounds longer** than the best known RK-ID distinguishers [20] for all these ciphers, for the flexibility in selecting key differences for the upper and lower trails. For SKINNY-64/192 and SKINNYee, our method produces RK-IB distinguishers that are **1 round longer** than the best known RK-IB distinguishers detected by the (G)BCT-based and DBCT(*)-based methods [12,13] for the ability to detect multiple-round contradictions. For SPECK, GIFT-64, and CHAM, our method yields the **first** RK-IB distinguishers for the ability to conduct bit-wise operations, including modular additions and bit permutation layers.
- Our state-based method is the **first** to support IB distinguisher search for ciphers with **key-dependent** Sboxes. Applied to PRINTcipher48 (single-key setting), it produces 5-round SK-IB distinguishers, which are one round longer than the best SK-ID distinguisher and provides a concrete **counterexample** to Lu's long-standing claim that SK-IB distinguishers cannot outperform SK-ID distinguishers. Furthermore, applying the state-based method to CHAM-64/128, we obtain 30-round RK-IB distinguishers—two rounds longer than the best-known results achieved by the GEBCT-based method, serve as an evidence that the state-based method is stronger than the GEBCT-based method. In addition, using the state-based method, we confirmed that the round coverage of the obtained IB distinguishers is indeed the upper limit within the given search space.
- Our structural method enables efficient multi-round RK-IB distinguisher search and structural security evaluation. We obtain the **first** 5-round RK-IB distinguishers for AES-128, which are two rounds longer than the best RK-ID distinguisher in the same *arbitrary Sbox* model.

Finally, we present a complete IB attack on SKINNYee based on our detected RK-IB distinguishers as an example to further validate our work. It is the **first** 31-round attack on SKINNYee, as well as the **state-of-the-art** result for SKINNYee.

Table 2: Summary of our main results

Cipher	Distinguisher	Rounds	Count	Time(h)	Method	Ref.
GEBCT-based Method (ARX, bit-permutation, and tweakable ciphers)						
SPECK-2w/4w	RK-ID	7	many	–	HLJ†	[20]
SPECK-2w/3w	RK-ID	6	many	–	HLJ†	[20]
SPECK-32/64	**RK-IB**	**8**	**6**	**0.18**	**GEBCT-based**	**Sec. 5.1**
SPECK-48/72		**7**	**6**	**0.06**		
SPECK-48/96		**8**	**6**	**0.09**		
SPECK-64/96		**8**	**4**	**0.29**		
SPECK-64/128		**9**	**4**	**0.28**		
SPECK-96/144		**8**	**4**	**0.22**		
SPECK-128/192		**8**	**4**	**0.33**		
SPECK-128/256 (ISO/IEC standard)		**9**	**4**	**0.41**		
CHAM-64/128 CHAM-128/256	RK-ID	26	many	–	HLJ†	[20]
CHAM-64/128	**RK-IB**	**28**	**8**	**0.02**	**GEBCT-based**	**App. F.7**
CHAM-128/256		**28**	**3**	**0.02**		
CHAM-64/128		**30**	**3**	**0.15**	**State-based**	
CHAM-128/256		**28**	**4**	**0.48**		
GIFT-64 (CHES 2017)	RK-ID	12	48	–	HLJ†	[20]
	RK-IB	**13**	**48**	**0.23‡**	**GEBCT-based**	**Sec. 5.2**
SKINNY-64/192 (CRYPTO 2016)	RK-ID	17	1	–	–	[21]
	RK-IB	18	1	–	DBCT-based	[11]
	RK-IB	**19**	**3**	**1.05‡**	**GEBCT-based**	**App. F.5**
SKINNYee (CRYPTO 2022)	RK-ID	19	1	–	–	[21]
	RK-IB	22	103	–	BCT-based	[10]
	RK-IB	**23**	**5**	**1.54‡**	**GEBCT-based**	**App. F.6**
State-based Method (ciphers with key-dependent operations)						
PRINTcipher48 (CHES 2010)	SK-ID	4	many	–	–	[20]
	SK-IB	**5**	**2**	**0.46‡**	**State-based**	**Sec. 5.3**
Structural Method (ciphers with nonlinear key schedules)						
AES-128 (NIST Standard)	RK-ID	3	64	0.39	ST†	[19]
	RK-IB	**5**	**768**	**0.45‡**	**Structural**	**Sec. 5.4**
RK-IB attack example based on our RK-IB distinguishers						
SKINNYee (CRYPTO 2022)	Key Recovery	29	$2^{67.2}$	$2^{119.2}$	–	[10]
		30	$2^{67.03}$	$2^{123.61}$	$2^{58.05}$	[22]
		31	$\mathbf{2^{67.2}}$	$\mathbf{2^{123.53}}$	$\mathbf{2^{104}}$	**Sec. 6**

(1) † denotes our reimplementation of the reference method. (2) All times are actual running times; those marked ‡ use 32 processes. (3) Search spaces are defined in Sec. 5. (4) The complexity of RK-IB attack on SKINNYee is one of multiple possible trade-offs.

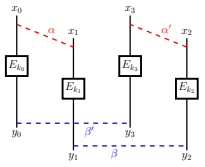


Fig. 1: The illustration of IB distinguisher

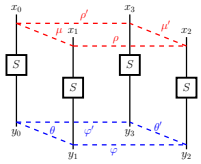


Fig. 2: Generalized BTs

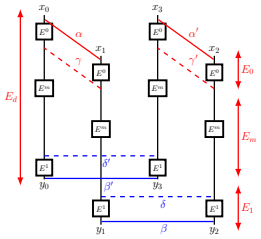


Fig. 3: The sandwich framework

The remainder of this paper is organized as follows. Sec. 2 covers preliminaries and related work. New construction theory for IB distinguishers and corresponding automatic search methods are presented in Sec. 3 and Sec. 4, respectively. Applications to numerous ciphers for IB distinguishers are given in Sec. 5, and an IB attack example is given in Sec. 6.

This work focuses on developing a comprehensive framework for constructing and searching IB distinguishers. Recent research has made advances in both distinguisher detection and key recovery. A detailed discussion of these complementary developments and their relationship to our work is provided in Sec. 2.2.

2 Preliminaries

2.1 Notations

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 $E_k(x)$ 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 $y = E_k(x) \in \mathbb{F}_2^n$. Let E_{i,rk_i} denote the i-th round of E_k with round key rk_i. Then $E_k(x) = E_{r-1,rk_{r-1}} \circ \cdots \circ E_{0,rk_0}(x)$. In clear contexts, E_k is abbreviated as E, and E_{i,rk_i} is abbreviated as E_i.
- Let S denote an s-bit Sbox and SL, LL, AK denote an Sbox layer, a linear layer and a round-key XOR operation, respectively.
- Let $\alpha \xrightarrow{f,\kappa} \beta$ denote a probability-1 differential where the input difference α propagates to the output difference β under the key difference κ for the function $f : \mathbb{F}_2^m \times \mathbb{F}_2^n \to \mathbb{F}_2^n$. In clear contexts, κ may be omitted.

2.2 Related work

The definition of IB distinguisher is as follows, given in Fig. 1.

Definition 1 ([4,5]). *For a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^m \to \mathbb{F}_2^n$ with key schedule* KS *under four keys $k_\ell \in \mathbb{F}_2^m$ ($\ell = 0, 1, 2, 3$), given key differences $\kappa_0 = k_0 \oplus k_1$, $\kappa_1 = k_2 \oplus k_3$, $\kappa_2 = k_1 \oplus k_2$, $\kappa_3 = k_0 \oplus k_3$ ($\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$), input differences*

α, α', *and output differences* β, β', *then* $(\alpha, \alpha', \beta, \beta')$ *is an* `RK-IB` *distinguisher of* E *under* $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ *if no plaintext pair* (x_1, x_2) *satisfies*

$$E_{k_1}(x_1) \oplus E_{k_2}(x_2) = \beta, \quad E_{k_0}(x_1 \oplus \alpha) \oplus E_{k_3}(x_2 \oplus \alpha') = \beta'. \tag{1}$$

Particularly, $(\alpha, \alpha', \beta, \beta')$ *is an* `SK-IB` *distinguisher of* E *when* $k = k_0 = k_1 = k_2 = k_3$.

We recall the definitions of boomerang connective tables[6].

Definition 2 ([24,8,9,25,26]). *Given differences* $\mu, \rho, \theta, \varphi \in \mathbb{F}_2^s$, *the* BCT, UBCT, LBCT *and* EBCT *for an* s*-bit Sbox are defined as follows*

$$\mathrm{BCT}(\mu, \varphi) = \#\left\{x \in \mathbb{F}_2^s \mid S^{-1}(S(x) \oplus \varphi) \oplus S^{-1}(S(x \oplus \mu) \oplus \varphi) = \mu\right\},$$

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

$$\mathrm{LBCT}(\mu, \rho, \varphi) = \#\left\{ x \in \mathbb{F}_2^s \middle| \begin{array}{l} S(x) \oplus S(x \oplus \rho) = \varphi, \\ S^{-1}(S(x) \oplus \varphi) \oplus S^{-1}(S(x \oplus \mu) \oplus \varphi) = \mu \end{array} \right\},$$

$$\mathrm{EBCT}(\mu, \rho, \theta, \varphi) = \#\left\{ x \in \mathbb{F}_2^s \middle| \begin{array}{l} S(x) \oplus S(x \oplus \mu) = \rho, S(x) \oplus S(x \oplus \theta) = \varphi, \\ S^{-1}(S(x) \oplus \varphi) \oplus S^{-1}(S(x \oplus \mu) \oplus \varphi) = \mu \end{array} \right\},$$

and the DBCT *is defined as* $\mathrm{DBCT}(\mu, \varphi) = \sum_{\upsilon_0, \upsilon_1} UBCT(\mu, \upsilon_0, \upsilon_1) \cdot LBCT(\upsilon_0, \upsilon_1, \varphi)$.

A variant of the DBCT, called the DBCT*, was proposed in [27]. These tables can be generalized to accommodate asymmetric boomerang, where facing-side differences differ. For example, the GBCT was defined as follows.

Definition 3. *Given differences* $\mu, \mu', \varphi, \varphi' \in \mathbb{F}_2^s$, *the* GBCT *for an* s*-bit Sbox is defined as follows (given in Fig. 2)*

$$\mathrm{GBCT}(\mu, \mu', \varphi, \varphi') = \#\left\{x \in \mathbb{F}_2^s \mid S^{-1}(S(x) \oplus \varphi) \oplus S^{-1}(S(x \oplus \mu) \oplus \varphi') = \mu'\right\}.$$

Next, we uniformly present the related work on `IB` distinguisher constructions in sandwich framework [12,13], as shown in Fig. 3. Let $E = E^1 \circ E^m \circ E^0$. Given key differences $\kappa_0, \kappa_1, \kappa_2, \kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, input and output differences $\alpha, \alpha', \beta, \beta'$, then $(\alpha, \alpha', \beta, \beta')$ is called an `IB` distinguisher if there exist four differentials $\alpha \xrightarrow{E^0, \kappa_0} \gamma$, $\alpha' \xrightarrow{E^0, \kappa_1} \gamma'$, $\beta \xrightarrow{(E^1)^{-1}, \kappa_2} \delta$, and $\beta' \xrightarrow{(E^1)^{-1}, \kappa_3} \delta'$ such that:
(Lu's Method [4,5]) $\delta \oplus \delta' \oplus \gamma \oplus \gamma' \neq 0$ (corresponding to the case of 0-round E^m, i.e., the contradiction occurs only at a certain moment).
(BCT/GBCT-based method [10,11]) at least one Sbox in E^m satisfies the following conditions[7]:

[6] In addition to the BCTs for SPN ciphers, boomerang connective tables are also defined for Feistel and ARX ciphers, given in App. A.

[7] Bonnetain et al. [10] also search for `IB` distinguishers of the block cipher `SIMON` based on FBCT [28]; this choice is motivated by the cipher's quadratic round function, which simplifies its modeling. Treating the one-round encryption of SIMON (excluding the key XOR operation) as a whole, this method can also be regarded as a BCT-based method.

– its input and output differences $\mu, \mu', \varphi, \varphi' \in \mathbb{F}_2^s$ are known ($\mu = \mu'$ and $\varphi = \varphi'$ are additionally required for BCT-based method);
– the outer table-lookup yields $\mathrm{GBCT}(\mu, \mu', \varphi, \varphi') = 0$ guided by $\mu, \mu', \varphi, \varphi'$

(corresponding to the case of 1-round contradiction that occurs at a certain Sbox).

(DBCT/DBCT*-based method [10,11]) analogous conditions hold, but verified exclusively via outer DBCT(*)-lookup (corresponding to the case of 2-round contradiction that spans two consecutive Sboxes, only applied to block ciphers with binary linear layers).

Recent developments in IB attacks. Since the initial submission of this work, several important advances have been made in `IB` attacks, addressing the distinguisher construction based on two-round contradictions and, more substantially, strengthening the key recovery methodologies.

- Wang et al. [22] made two key contributions: first, they presented iUBCT/iLBCT tools for detecting two-round contradictions in `IB` distinguisher; second, they modernized the key recovery framework by adapting techniques from rectangle cryptanalysis. Their work achieved the first 30-round attack on SKINNYee.
- Chen et al [29]. presented a holistic key recovery framework tailored for optimal `IB` attacks, integrating meet-in-the-middle techniques into the key recovery process. They obtained the first 11-round `IB` attacks on Deoxys-BC-256 and Joltik-BC-128, as well as the first 33-round `IB` attack on SKINNYe v2.
- Jiao et al. [30] developed a unified key recovery framework for `IB` attacks with flexible partial pre-guessing, pre-sieving, and automatic key-guessing techniques based on directed graphs. They demonstrated the effectiveness of their framework by achieving a full-round attack on ARADI [31] and a 34-round `IB` attack on SKINNYe v2.

In summary, Lu's method does not account for the dependence between upper and lower trails [7]. Although the (G)BCT/DBCT(*)/iU(L)BCT-based methods capture such dependence over one or two rounds, the dependence can in fact significantly affect multiple rounds—e.g., up to six rounds according to `SKINNY`'s `Boomerang` distinguisher [32], allowing longer `IB` distinguishers that remain unidentified by current approaches. In this paper, we fill this gap in `IB` distinguisher construction. Moreover, research on key recovery in `IB` attacks is complementary across studies and collectively advances the `IB` cryptanalysis.

3 New Construction Methods for IB Distinguishers

As shown in Sec. 2, extending the dependency in more rounds is an effective means to increase the rounds of `IB` distinguishers. Moreover, enhancing the cipher model with a more precise description enables the identification of contradiction patterns not captured by existing construction methods and the application of cipher classes incompatible with those methods—thereby recovering `IB` distinguishers previously overlooked.

To achieve these targets, in this section, we introduce three complementary IB distinguisher construction methods generally for SPN, Feistel, and ARX ciphers (including And-RX): the GEBCT-based method, the state-based method, and the structural method. We establish their validity by proving consistency with the foundational IB distinguisher definition and relationship with related work.

3.1 GEBCT-based Method for IB Distinguisher Construction

How can we extend the contradiction span to three or more rounds? The direct answer is through more boomerang connective tables. We generalize the EBCT to handle asymmetric boomerang transitions, in order to describe the general multi-round quartet-differential propagation.

Definition 4. *Given eight differences* $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi' \in \mathbb{F}_2^s$ *where* $\rho' = \mu \oplus \mu' \oplus \rho$, $\varphi' = \theta \oplus \theta' \oplus \varphi$, *the* GEBCT *for an* s*-bit Sbox is defined as* $\mathrm{GEBCT}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_0) \oplus S(x_1) = \theta, S(x_2) \oplus S(x_3) = \theta', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.

To construct IB distinguisher based on GEBCT, currently, the technical routes that can be referred to fall into two categories—both of which prove inadequate for this setting:

1. The first adapts the lightweight "determine-then-lookup" strategy from [10,11], where upper and lower differential trails are fixed upfront and outer table lookup is performed, without incorporating any Sbox details into the search process. However, because GEBCT models variable-length, multi-round boomerang connections, intermediate-round differences are inherently unknown a priori, rendering local outer table lookup inapplicable.
2. The second attempts to encode GEBCT directly into automatic search constraints (e.g., MILP/SAT/CP formulations). Yet, bit-level modeling on eight s-bit inputs introduces an explosion in variable count and constraint complexity, making both precise characterization and efficient search intractable.

Since neither approach scales to GEBCT, we propose a new method that integrates GEBCT characterization "on-the-fly" into the IB distinguisher search by modeling the state propagation of the Sbox., and guiding the IB distinguisher search within a completely automatic framework without outer table lookups.

Specifically, we introduce two intermediate quartet-states $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^s)^4$ and $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^s)^4$, constrained by differences $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi' \in \mathbb{F}_2^s$ as in Definition 4. The propagation from x_ℓ to y_ℓ through each Sbox is modeled by a statement of "If-Then-Else", a built-in syntax of the SAT/SMT solver STP, according to the Sbox's truth table for $\ell \in \{0, 1, 2, 3\}$ (see Sec. 4 for technical details).

We give the GEBCT-based method for IB distinguisher construction.

Construction 1 (GEBCT-based Method). *For an* r*-round Sbox-based block cipher* $E_k = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ *with key schedule* KS *under four keys*

$k_\ell \in \mathbb{F}_2^m$ *($\ell = 0,1,2,3$), let* ϵ_ℓ^i *and* ϵ_ℓ^{i+1} *be the input and output differences of* $E_{i,\mathrm{KS}_i(k_\ell)}$ *for* $i \in \{0,\ldots,r-1\}$ *and* $\ell \in \{0,1,2,3\}$*. Given key differences* $\kappa_0 = k_0 \oplus k_1$*,* $\kappa_1 = k_2 \oplus k_3$*,* $\kappa_2 = k_1 \oplus k_2$*,* $\kappa_3 = k_0 \oplus k_3$ *($\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$), input differences* α, α'*, and output differences* β, β'*, then* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* `GEBCT-based RK-IB` *distinguisher under* $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ *if there exists no quartet-differential trail*

$$(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta'), \quad (2)$$

where $\xrightarrow{\text{GEBCT}}$ *denotes each Sbox differential propagation governed by GEBCT. Particularly,* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* `GEBCT-based SK-IB` *distinguisher when* $k = k_0 = k_1 = k_2 = k_3$*.*

Substituting E with E^m in Construction 1, Construction 1 can be applied to the sandwich framework.

The GEBCT-based method applies natively to SPN and Sbox-based Feistel ciphers. To extend it to ARX ciphers—which lack Sboxes and rely instead on modular addition or bitwise-and, we regard it as an $2n \times 2n$-bit Sbox instead of a $2n$-bit to n-bit mapping, inspired by [33]. This allows ARX ciphers to be converted into SPN or Sbox-based Feistel ciphers. For example, the `SPECK` cipher can be viewed as a SPN cipher when modular addition is treated as an Sbox, with round keys XORed with part of the state, as shown in Fig. 11. Thus, the GEBCT-based method is also applicable to ARX ciphers.

This construction method enables precise and efficient characterization of GEBCT and considers the propagation of quartet-differential across multiple rounds, while delivering three advantages: (1) Due to the "on-the-fly" GEBCT description method, it removes the Sbox size limitation and enables application to larger modular addition operations (e.g., 64-bit or wider); (2) Unlike technical route 1, it does not require pre-specified input–output differences—even when the method reduces to (G)BCT-based, it supports fully free differential selection, thereby uncovering significantly more `IB` distinguisher candidates; (3) In contrast to technical route 2, the constraint system size scales only with the entries in the Sbox's truth table (2^s)—not with that in the GEBCT (2^{8s}) when encoding GEBCT directly. This makes it practically feasible to incorporate the full differential properties provided by GEBCT while maintaining tractable and efficient automatic search.

3.2 State-based Method for IB Distinguisher Construction

In addition to the GEBCT-based method, we adopt a more precise construction method of `IB` distinguisher that models the full distinguisher using quartet-state propagation.

Construction 2 (State-based Method). *For an* r*-round block cipher* $E_k = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ *with key schedule* KS *under four keys* $k_\ell \in \mathbb{F}_2^m$ *($\ell = 0,1,2,3$), given key differences* $\kappa_0 = k_0 \oplus k_1$*,* $\kappa_1 = k_2 \oplus k_3$*,* $\kappa_2 = k_1 \oplus k_2$*,*

$\kappa_3 = k_0 \oplus k_3$ ($\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$), *input differences* α, α', *and output differences* β, β', *then* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* `state-based RK-IB` *distinguisher under* $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$ *if there exists no quartet-state trail*

$$(x_0^0, x_1^0, x_2^0, x_3^0) \to \cdots \to (x_0^r, x_1^r, x_2^r, x_3^r) \tag{3}$$

where $x_0^0 \oplus x_1^0 = \alpha$, $x_2^0 \oplus x_3^0 = \alpha'$, $x_1^r \oplus x_2^r = \beta$, $x_0^r \oplus x_3^r = \beta'$, *and* $x_\ell^{i+1} = E_{i,\mathrm{KS}_i(k_\ell)}(x_\ell^i)$ *for* $i \in \{0, 1, \ldots, r-1\}$, $\ell \in \{0, 1, 2, 3\}$. *Particularly,* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* `state-based SK-IB` *distinguisher when* $\kappa_0 = \kappa_1 = \kappa_2 = \kappa_3 = 0$.

Substituting E with E^m in Construction 2, Construction 2 can be applied to the sandwich framework. Beyond the Sbox, more modeling methods for state propagation is given in Sec. 4.

Next, we demonstrate that the state-based method is the tightest for constructing `IB` distinguisher.

Theorem 1. *For an* r*-round block cipher* $E_k = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ *with key schedule* KS, *a* `state-based IB` *distinguisher is equivalent to Definition 1.*

Proof. A `state-based IB` distinguisher is, by definition, an `IB` distinguisher. We prove, by contradiction, that every `IB` distinguisher $(\alpha, \alpha', \beta, \beta')$ is a `state-based IB` distinguisher. If not, there exists a quartet-state trail as in Eq. (3), where $x_0^0 \oplus x_1^0 = \alpha$, $x_3^0 \oplus x_2^0 = \alpha'$, $x_1^r \oplus x_2^r = \beta$, and $x_0^r \oplus x_3^r = \beta'$. Thus, (x_1^0, x_2^0) satisfies both $E_{k_1}(x_1^0) \oplus E_{k_2}(x_2^0) = \beta$ and $E_{k_0}(x_1^0 \oplus \alpha) \oplus E_{k_3}(x_2^0 \oplus \alpha') = \beta'$—consistent with Definition 1. □

The `state-based IB` distinguisher construction method applies universally to all block ciphers and fully incorporates their details, such as the key schedule, deriving optimal `IB` distinguishers. Furthermore, its equivalence to the formal `IB` distinguisher definition allows it to serve as a sound foundation and effective tool for proving resistance of a block cipher against `IB` distinguisher (i.e., no `IB` distinguisher exist within a given space of input and output differences).

3.3 Structural Method for IB Distinguisher Construction

Although the GEBCT-based method achieves a strong trade-off between search efficiency and description precision, we propose the structural method—a more efficient approach to `IB` distinguisher construction.

We incorporate the *arbitrary Sbox* from [19] into `IB` distinguisher—i.e., we assume an imaginary Sbox whose difference distribution table is defined as

$$\overline{\mathrm{DDT}}(a, b) = \begin{cases} 1, & a \text{ and } b \text{ are both zero or both nonzero;} \\ 0, & \text{otherwise.} \end{cases}$$

Let $\sigma_0, \sigma_1, \ldots, \sigma_{s-1}$ and $\phi_0, \phi_1, \ldots, \phi_{s-1}$ be binary variables representing the input and output differences, respectively. Then valid differential propagations of the s-bit *arbitrary Sbox* are characterized by CNF constraints as

$$\left(\bigwedge_{i=0}^{s-1} \left(\neg\phi_i \vee \bigvee_{j=0}^{s-1} \sigma_j\right)\right) \wedge \left(\bigwedge_{i=0}^{s-1} \left(\neg\sigma_i \vee \bigvee_{j=0}^{s-1} \phi_j\right)\right).$$

Similarly, we introduce $\overline{\text{GEBCT}}$, a weakened variant of GEBCT.

Definition 5. *For an s-bit Sbox, given input and output differences $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi' \in \mathbb{F}_2^s$ where $\rho' = \mu \oplus \mu' \oplus \rho$, $\varphi' = \theta \oplus \theta' \oplus \varphi$, then*

$$\overline{\text{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') = \overline{\text{DDT}}(\mu, \rho) \wedge \overline{\text{DDT}}(\mu', \rho') \wedge \overline{\text{DDT}}(\theta, \varphi) \wedge \overline{\text{DDT}}(\theta', \varphi').$$

Construction 3 (Structural Method). *The structural method follows the same framework as the GEBCT-based method for block ciphers with bijective Sboxes, except that it replaces the quartet-differential trail in Eq.* (2) *with the modified trail*

$$(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{\overline{\text{GEBCT}}} \cdots \xrightarrow{\overline{\text{GEBCT}}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta'). \quad (4)$$

Consequently, $(\alpha, \alpha', \beta, \beta')$ is an r-round `structural IB` *distinguisher.*

This method is more efficient than GEBCT-based method. It no longer exploits specific differential propagation within Sbox and requires fewer constraints to model quartet-differential propagation. This makes it well suited for evaluating the structural security of block ciphers (especially with large block sizes), which is an important research topic in block cipher design. `Structural IB` distinguishers are inherently impossible regardless of the Sbox or key schedule. Unlike purely truncated differential propagation, this method retains a bit-wise model, which fully accounts for the details of the linear layers. This advantage is often sufficient to uncover new `IB` distinguishers. Experimental results on `AES` validate the effectiveness of this method.

3.4 Relations among New Construction Methods and Related Work

We discuss the relations among the construction methods for `IB` distinguisher. First, we establish the relation between current methods and GEBCT-based method. The proof is provided in App. B.1.

Theorem 2. *The* `IB` *distinguisher constructed by Lu's method, (G)BCT-based method, and DBCT(*)-based method are all* `GEBCT-based IB` *distinguisher.*

Next, we discuss the relation between the GEBCT-based method and state-based method. The proof is given in App. B.2.

Theorem 3. *An r-round* `GEBCT-based IB` *distinguisher $(\alpha, \alpha', \beta, \beta')$ is an r-round* `state-based IB` *distinguisher. In particular, if the* `state-based SK-IB` *distinguisher is constructed by removing the key constraint from key schedule, then such* `SK-IB` *distinguisher is a* `GEBCT-based SK-IB` *distinguisher for*

- *SPN ciphers with full-state round key addition operations;*
- *Feistel ciphers with full-branch round key addition operations;*
- *ARX ciphers equivalent to the two cases above when their nonlinear operations are transformed as Sboxes.*

Since a GEBCT-based IB distinguisher is a state-based IB distinguisher, the state-based method can automatically verify all distinguishers found by the GEBCT-based method—significantly improving efficiency.

Then, we derive the relations among the new construction methods.

Summary. *For a block cipher E, let $S_{structural}$, S_{GEBCT} and S_{state} be the set of all IB distinguisher constructed by the structural method, GEBCT-based method and state-based method, respectively, and let S_{def} be the set corresponding to the definition of IB distinguisher, then*

$$(S_{structure} \subseteq) S_{GEBCT} \subseteq S_{state} = S_{def}.$$

The first inclusion holds only for block ciphers with bijective Sboxes; the rest hold generally.

A comparison of the new construction methods is presented in Table 1.

3.5 Relations among IB, ID and Boomerang distinguishers

With the development of cryptanalytic techniques, a rigorous re-examination of the relations among IB distinguisher, ID distinguisher and Boomerang distinguisher is warranted.

IB distinguisher v.s. ID distinguisher. In [5], Lu first established a relation between IB distinguisher and ID distinguisher, observing that "a SK-IB distinguisher implies a same-round SK-ID distinguisher; however, this implication does not extend to related-key scenario."

Crucially, Lu's argument applies only to SK-IB distinguishers constructed via his original method. Specifically, it relies on the condition $\delta \oplus \delta' \oplus \gamma \oplus \gamma' \neq 0$, from which $\delta \oplus \gamma \neq \delta' \oplus \gamma'$ follows, ensuring that at least one of $\delta \oplus \gamma$ or $\delta' \oplus \gamma'$ is nonzero and thus yielding an impossible differential $\alpha \nrightarrow \beta$ or $\alpha' \nrightarrow \beta'$ under a single key. As IB distinguisher construction techniques have advanced—enabling more-round contradiction spans—this derivation no longer holds universally. We provide a concrete counterexample: for PRINTcipher48, 5-round SK-IB distinguishers exist, whereas the longest known SK-ID distinguishers remain limited to 4 rounds. This breaks the inherent impression of the limitation of SK-IB distinguisher.

Furthermore, the flexibility in selecting key differences for the upper and lower trails in a RK-IB distinguisher—unavailable in RK-ID distinguisher— indeed offers a structural advantage that may enable breaking more rounds of the cipher (see Sec. 3.6 for more comparison).

IB distinguisher v.s. Boomerang distinguisher. In [5], Lu also stated that an IB distinguisher is more reasonable than the Boomerang distinguisher, since the latter usually relies on (related-key) differentials under heuristic assumptions. However, the Quasidifferential Framework (QDFM), proposed by Beyne and Rijmen at CRYPTO 2022 [34], solves this long-standing problem caused by the exact differential probability calculation under the fixed-key model [35]—without

heuristic hypotheses (e.g., the Markov assumption or key stochastic equivalence). Later, Boura et al. extended QDFM to 8-bit Sboxes [36]. Wang et al. [27] further extended QDFM to calculate the exact probability of `Boomerang` distinguishers. Therefore, in QDFM, an `IB` distinguisher occurs when the sum of correlations across all quasi-d-differential trails is identically zero for every key—that is, the total correlation vanishes universally, not merely on a subset of keys. (see App. C for more details about QDFM.)

Within the domain of `IB` distinguisher construction, the state-based method is equivalent to QDFM, as both methods can characterize `IB` distinguishers strictly in accordance with the original definition. This equivalence can also be inferred from the work of Beyne and Rijmen [34] "The probability of differentials can be precisely described by tracking the distribution of pairs of state values. However, directly tracking pairs is inconvenient because it does not provide a simple description of translations". Our stated-based method, developed for `IB` distinguisher construction, proposes an effective and precise modeling method for tracking state pairs or quartets that eliminates the need to calculate the sum of correlations across all quasi-d-differential trails.

Under QDFM, both round constants and keys govern the sign of each differential trace. Consequently, several key- or constant-dependent differentials have been identified [34]. Similar to QDFM, the stated-based method can also be applied to weak-key `IB` distinguishers by extending the definition of `stated-based IB` distinguisher, which imposes additional constraints—either on specific key values or on algebraic relation among keys—based on key partitioning. Crucially, this approach does not require knowledge of concrete key values; instead, it characterizes how the differential probability depends on the secret key, thereby enabling a structural partition of the key space into classes where the underlying quasidifferential behavior (excluding sign patterns) remains invariant. Attack strategies can then be tailored to each class to derive weak key `IB` distinguishers. We leave this for future work.

Overall, within the `IB` distinguisher (`ID` distinguisher) construction, the state-based method is equivalent to QDFM; however, while QDFM is grounded in a rigorous theoretical formulation—centered on correlation-based trace enumeration and sign-controlled superposition—the state-based approach offers a more intuitive, operationally streamlined, and search-friendly modeling paradigm.

3.6 Related-Key Settings for RK-IB Distinguisher Construction

Within sandwich framework, `RK-IB` distinguisher offer significant advantages, as attackers can exert more flexible control over differential relationships between related keys. The selection of key differences is technically demanding—a fact well-established in the cryptography community—yet no general formal selection rules have been established. This section clarifies the construction methods of `IB` distinguishers by analyzing two typical related-key settings.

Two-related-keys setting. This setting is especially well-suited to block ciphers with nonlinear key schedules, where `IB` distinguishers are constructed under the key differences $(k_0 \oplus k_1, k_2 \oplus k_3, k_1 \oplus k_2, k_0 \oplus k_3) = (\kappa, \kappa, 0, 0)$, since

the round key differences along the lower trails are difficult to control under nonlinear key schedule. Accordingly, we search RK-IB distinguisher $(\alpha, \alpha, \beta, \beta')$ for block cipher $E = E^1 \circ E^m \circ E^0$. Here, for the lower trail, the key difference is zero, so no probability-1 differential can pass through nonlinear operations. Therefore, E_1 is treated as the identity transformation. The specific construction is given below.

1. Search for r_0-round related-key differentials $\alpha \xrightarrow{E^0,\kappa} \gamma$, ensuring both the key and state difference do not pass through any nonlinear operations.
2. Search for β, β' such that $(\gamma, \gamma, \beta, \beta')$ forms an r_m-round core RK-IB distinguisher Concatenate these to obtain an $(r_0 + r_m)$-round RK-IB distinguisher.

Four-related-keys setting. This setting is especially well-suited to block ciphers with linear key schedules, where IB distinguishers are constructed under the key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_0 \oplus \kappa_1 \oplus \kappa_2)$, since all round key differences are uniquely determined by the master key differences. Accordingly, we search RK-IB distinguisher $(\alpha, \alpha, \beta, \beta')$ for block cipher $E = E^1 \circ E^m \circ E^0$. Specifically,

1. Search for four probability-1 related-key differentials: r_0-round $\alpha \xrightarrow{E^0,\kappa_0} \gamma$, $\alpha' \xrightarrow{E^0,\kappa_1} \gamma'$; r_1-round $\beta \xrightarrow{(E^1)^{-1},\kappa_2} \delta$, and $\beta' \xrightarrow{(E^1)^{-1},\kappa_3} \delta'$.
2. Verify whether $(\gamma, \gamma', \delta, \delta')$ forms an r_m-round RK-IB distinguisher. If it is, concatenate these to obtain an $(r_0 + r_m + r_1)$-round RK-IB distinguisher.

More comparison between RK-IB and RK-ID distinguishers in above related key settings. RK-IB distinguishers have two key advantages over RK-ID distinguishers: (1) Better controllability of key differences. In two-related-keys setting, RK-IB distinguishers have a known upper key difference and a fixed lower key difference 0. In contrast, the key difference for RK-ID distinguishers undergoes diffusion through the key schedule (especially for nonlinear ones), making it difficult to control or eliminate. (2) Higher flexibility in differential selection. In four-related-keys setting, RK-IB distinguishers only require $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$, allowing flexible selection of different key differences to extend differentials in E^0 and E^1. In contrast, RK-ID distinguishers require the same key difference for extending differentials in both E^0 and E^1. These advantages allow RK-IB distinguishers to cover more rounds.

4 United Automatic IB Distinguisher Search Framework

In this section, we present a unified SAT/SMT-based automatic search framework for these three new IB distinguisher construction methods in Sec. 3.

4.1 Modeling Techniques

SAT/SMT solvers can determine the satisfiability of Boolean formulas encoding cipher propagation rules for a distinguisher search problem efficiently. In this work, we use STP with CryptoMiniSat[8] as the backend solver. STP supports

[8] https://stp.github.io and https://github.com/msoos/cryptominisat

operations including Equal (=), Xor ($\oplus$), Modular-addition (+), Bitwise-and ($\cdot$), Bitwise-or ($\vee$), Bitwise-not ($\neg$), and conditional statements (If-Then-Else). These operations suffice to model the propagation of both quartet-differences and quartet-states through linear components (`Equal`, `Copy`, `Xor`, `Matrix-multiply`) and nonlinear components (`Modular-addition`, `Bitwise-and`), which are straightforward (See App. D for more details).

Modeling nonlinear components with GEBCT. The most challenging part of `IB` distinguisher modeling is handling Sboxes, whose quartet-difference propagation is governed by the GEBCT. As discussed in Sec. 3.1, directly encoding the full GEBCT is infeasible. Instead, we propose a state-propagation approach that introduces intermediate quartet-states to capture GEBCT constraints.

Model 1 (GEBCT Modeling). *Let $S : \mathbb{F}_2^{s_0} \to \mathbb{F}_2^{s_1}$ be an Sbox. For an input quartet-difference $(\mu, \mu', \rho, \rho') \in (\mathbb{F}_2^{s_0})^4$ and an output quartet-difference $(\theta, \theta', \varphi, \varphi') \in (\mathbb{F}_2^{s_1})^4$, we introduce two intermediate quartet-states $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^{s_0})^4$ and $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^{s_1})^4$. The* GEBCT *is then modeled as:*

$$\begin{cases} x_0 \oplus x_1 = \mu,\ x_1 \oplus x_2 = \rho,\ x_2 \oplus x_3 = \mu',\ x_0 \oplus x_3 = \rho', \\ y_0 \oplus y_1 = \theta,\ y_1 \oplus y_2 = \varphi,\ y_2 \oplus y_3 = \theta',\ y_0 \oplus y_3 = \varphi', \\ y_\ell = S(x_\ell) \text{ for } 0 \le \ell \le 3 \text{ according to Model 2.} \end{cases}$$

This approach naturally extends to ARX ciphers: treat composite nonlinear operations as a single Sbox and apply Model 1 (When the nonlinear operation is `Modular-addition` or `Bitwise-and`, the state propagation can be supported by STP directly, according to Model Model 7 and Model Model 8 in App. D).

Modeling nonlinear components with state propagation. Nonlinear components are modeled by direct state propagation, such as Example 1 in App. D.

Model 2. *Let $S : \mathbb{F}_2^{s_0} \to \mathbb{F}_2^{s_1}$ be an Sbox. For an input quartet-state $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^{s_0})^4$ and an output quartet-state $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^{s_1})^4$, the pair (x_ℓ, y_ℓ) $(0 \le \ell \le 3)$ is constrained by the truth table of S, modeling by a statement composed of If-Then-Else.*

Modeling bijective Sboxes with $\overline{\text{GEBCT}}$. When bijective Sboxes $S : \mathbb{F}_2^s \to \mathbb{F}_2^s$ is modeled by $\overline{\text{GEBCT}}$, we only enforce constraints on impossible input-output difference pairs. This significantly reduces clause count compared to the full GEBCT model.

Model 3 ($\overline{\text{GEBCT}}$ Modeling). *Let $S : \mathbb{F}_2^s \to \mathbb{F}_2^s$ be a bijective Sbox. For input quartet-difference (μ, μ', ρ, ρ') and output quartet-difference $(\theta, \theta', \varphi, \varphi')$, the $\overline{\text{GEBCT}}$ imposes the following constraints for each bit position $0 \le j \le s-1$:*

$$\begin{cases} \mu_j \oplus \mu'_j \oplus \rho_j \oplus \rho'_j = 0,\ \theta_j \oplus \theta'_j \oplus \varphi_j \oplus \varphi'_j = 0, \\ \mu_0 | \cdots | \mu_{p-1} | \neg\theta_j = 1,\ \neg\mu_j | \theta_0 | \cdots | \theta_{p-1} = 1,\ \mu'_0 | \cdots | \mu'_{p-1} | \neg\theta'_j = 1,\ \neg\mu'_j | \theta'_0 | \cdots | \theta'_{p-1} = 1, \\ \rho_0 | \cdots | \rho_{p-1} | \neg\varphi_j = 1,\ \neg\rho_j | \varphi_0 | \cdots | \varphi_{p-1} = 1,\ \rho'_0 | \cdots | \rho'_{p-1} | \neg\varphi'_j = 1,\ \neg\rho'_j | \varphi'_0 | \cdots | \varphi'_{p-1} = 1. \end{cases}$$

These constraints eliminate all impossible transitions where a zero input difference maps to a non-zero output difference, or vice versa.

Algorithm 1: SAT/SMT Model for IB Distinguishers

Input: input differences (α, α'), output differences (β, β'), key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$, r-round block cipher E_d

Output: SAT/SMT model $\mathcal{M}$

1 Declare an empty SAT/SMT model $\mathcal{M}$;

`// Declare variables`

2 $\mathcal{M}.var \leftarrow$ input quartet $\mathbb{Q}_I$ and output quartet $\mathbb{Q}_O$;

3 $\mathcal{M}.var \leftarrow$ input key quartet $\mathbb{K}_I$ and output key quartet $\mathbb{K}_O$;

4 $\mathcal{M}.var \leftarrow$ all intermediate quartets;

`// Model the key quartet propagation through key schedule`

5 **for** *each round in KS* **do**

6 $\quad \mathcal{M}.con \leftarrow$ constraints for key quartet propagation from $\mathbb{K}_I$ to $\mathbb{K}_O$ (according to Model* for nonlinear components);

`// Model the quartet propagation through the encryption`

7 **for** *each round in* E_d **do**

8 $\quad \mathcal{M}.con \leftarrow$ constraint for quartet propagation from $\mathbb{Q}_I$ to $\mathbb{Q}_O$ (according to Model* for nonlinear components);

`// Add constraints for the input and output quartets`

9 $\mathcal{M}.con \leftarrow$ constraints on $\mathbb{Q}_I$ according to $(\alpha, \alpha', *, *)$ and $\mathbb{Q}_O$ according to $(*, *, \beta, \beta')$;

10 $\mathcal{M}.con \leftarrow$ constraint on $\mathbb{K}_I$ according to $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$.

11 **return** $\mathcal{M}$

4.2 Unified Search Algorithm

We present a unified algorithm to verify whether $(\alpha, \alpha', \beta, \beta')$ forms an r-round `IB` distinguisher under key differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3)$, where $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$. The algorithm returns a SAT/SMT model encoding cipher propagation and distinguisher constraints. If the solver returns "UNSATISFIABLE", the tuple is a valid `IB` distinguisher.

Adaptations for new IB distinguisher construction methods. Algorithm 1 can be adapted to three different types of `IB` distinguishers by modifying variable interpretation and Sbox modeling:

1. `GEBCT-based IB` distinguisher:
   - $\mathbb{K}_I = (K_0^I, K_1^I, K_2^I, K_3^I), \mathbb{K}_O = (K_0^O, K_1^O, K_2^O, K_3^O), \mathbb{Q}_I = (Q_0^I, Q_1^I, Q_2^I, Q_3^I)$, $\mathbb{Q}_O = (Q_0^O, Q_1^O, Q_2^O, Q_3^O)$ represent quartet-differences (Line 2 and 3).
   - Model* is Model 1 (Line 6 and 8).
   - Input and output constraints (Line 9 and 10)[9]:

$$\begin{cases} Q_0^I = \alpha,\ Q_2^I = \alpha',\ Q_1^O = \beta,\ Q_3^O = \beta', \\ K_0^I = \kappa_0,\ K_1^I = \kappa_1,\ K_2^I = \kappa_2,\ K_3^I = \kappa_3. \end{cases} \tag{5}$$

[9] K_2^O and K_3^O are determined by κ_2 and κ_3, which is consistent with the code.

2. `State-based IB` distinguisher:
   - Variables $\mathbb{K}_I$, $\mathbb{K}_O$, $\mathbb{Q}_I$, $\mathbb{Q}_O$ represent quartet-states (Line 2 and 3).
   - Model* is Model 2 (Line 6 and 8).
   - Input and output constraints (Line 9 and 10)[10]:

$$\begin{cases} Q_{I,0} \oplus Q_{I,1} = \alpha,\ Q_{I,2} \oplus Q_{I,3} = \alpha',\ Q_{O,1} \oplus Q_{O,2} = \beta,\ Q_{O,0} \oplus Q_{O,3} = \beta', \\ K_{I,0} \oplus K_{I,1} = \kappa_0,\ K_{I,2} \oplus K_{I,3} = \kappa_1,\ K_{I,1} \oplus K_{I,2} = \kappa_2,\ K_{I,0} \oplus K_{I,3} = \kappa_3. \end{cases}$$

3. `Structural IB` distinguisher:
   - Variables $\mathbb{K}_I$, $\mathbb{K}_O$, $\mathbb{Q}_I$, $\mathbb{Q}_O$ represent quartet-differences (Line 2 and 3).
   - Model* is Model 3 for bijective Sboxes (Line 6 and 8).
   - Input and output constraints are identical to Eq. (5) (Line 9 and 10).

Search Space. The search space for `IB` distinguishers is enormous (exponential in block size). However, practical `IB` distinguishers almost always have low-weight input and output differences (few active bits, nibbles, or bytes). This is because low-weight input differences usually propagate more slowly through the cipher, thereby enabling longer `IB` distinguishers and improving the effectiveness of key recovery. We therefore restrict our search to input differences with Hamming weight below a predefined threshold.

4.3 Verification Strategy.

The correctness of automatically found distinguishers depends on both model accuracy and code implementation correctness. We use a two-layer verification strategy (see App. E for more details):

1. Model correctness verification: (1) Verifying a representative subset of the resulting `IB` distinguishers confirms the soundness of both the construction theory and modeling method. Crucially, because all `IB` distinguishers rely on the same underlying code interfaces, correctness established for this subset extends to the entire set. (2) In addition to traditional manual proof, we employ computer-assisted verification to: (2.1) locate the positions where contradictions occur; (2.2) verify the correctness by traversing all plausible trails and disproving them; (2.3) perform automatic verification of `GEBCT-based IB` distinguishers using the state-based method.
2. Code correctness verification: We propose a code self-feedback verification technique to minimizes the occurrence of code implementation errors.

5 Applications to Block Ciphers

We evaluate our three complementary `IB` distinguisher construction methods across six representative block ciphers, demonstrating their universal applicability to all major design paradigms and attack settings:

[10] $K_{O,1} \oplus K_{O,2}$ and $K_{O,0} \oplus K_{O,3}$ are determined by κ_2 and κ_3, consistent with the code.

- The GEBCT-based method is employed to construct RK-IB distinguishers in two related-key scenarios. (1) In two-related-keys setting, it is applied to SPECK, an ARX cipher, validating our ability to accurately model differential propagation for bit-wise operations. (2) In four-related-keys setting, it is applied to four ciphers with linear key/tweak/tweakey schedules in a similar manner: GIFT-64 (bit permutation layer based cipher), CHAM (ARX cipher), SKINNY-64/192 and SKINNYee (tweakable block ciphers). We elaborate on the results of GIFT-64 in this section, while the full results for the other three ciphers are provided in App. F.
- The state-based method for SK-IB distinguisher is applied to PRINTcipher48, which uses key-dependent Sboxes, showcasing its capability in considering full state propagation.
- The structural method for RK-IB distinguisher in the two-related-keys setting is applied to AES-128, which features a nonlinear key schedule, showing its utility in fast structural security evaluation.

Among them, the results of SPECK, CHAM, GIFT-64, and PRINTcipher48 yield the first IB distinguisher; previous methods cannot effectively be applied to these cipher structures. The RK-IB distinguisher for AES-128 constitutes its first such distinguisher—Lu established only a 4-round SK-IB distinguisher for AES-128 and 6-round RK-IB distinguishers for AES-192/256 [5]. Furthermore, our RK-IB distinguishers for SKINNY-64/192 and SKINNYee each extend the best-known results by one round [10,11].

All experiments were run on an AMD(R) @2.60GHz with 80G RAM running Ubuntu 18.04 (64-bit). Runtime statistics are summarized in Table 2. We provide only a brief introduction to the ciphers here; full descriptions appear in App. F.

5.1 GEBCT-based Method on SPECK in Two-Related-Key Setting

SPECK [18] is an NSA-designed ARX cipher family standardized in ISO/IEC 29167-22 for RFID air interface. It follows a typical ARX structure with modular addition, bitwise XOR, and circular shifts. Variants are denoted SPECK-n/m where n is the block size and m the key size, covering $(n, m) \in \{(32, 64), (48, 72), (48, 96), (64, 96), (64, 128), (96, 96), (96, 144), (128, 128), (128, 192), (128, 256)\}$.

Configurations and Results. SPECK's nonlinear key schedule and modular addition operations make it incompatible with all prior IB distinguisher search methods. We address this by treating modular addition as a large Sbox and applying our GEBCT-based method in two-related-keys setting setting $(\kappa, \kappa, 0, 0)$. Our search proceeds in two phases:

1. Find r_0-round related-key differentials $\alpha \xrightarrow{E^0,\kappa} \gamma$, and result in $r_0 = 4$ for SPECK-$2w/4w$ ($w = 16, 24, 32, 64$) and $r_0 = 3$ for SPECK-$2w/3w$ (for $w = 24, 32, 48, 64$).
2. Search for β, β' such that $(\gamma, \gamma, \beta, \beta')$ forms an r_1-round core RK-IB distinguisher.

Combine them to yield $(r_0 + r_1)$-round RK-IB distinguishers ranging over 7–9 rounds, summarized in Table 2 (full details in App. F.1). Here, we restrict β, β'

to the set $\Lambda = \{(\mu, \mu), (0, \mu \ggg_b), (\mu, \mu \oplus (\mu \ggg_b))\}$, where $\ggg_b$ denotes circular right shift by b bits ($b = 2$ for SPECK-32, $b = 3$ otherwise), and μ is an $n/2$-bit value with only the most significant bit set. This ensures probability-1 backward propagation through the final modular addition, yielding a search space of size 9. All results are automatically verified by the state-based method. We also run state-based searches and verify that these are indeed the upper bounds on RK-IB distinguisher coverage within the same search space.

Comparison.

- vs. prior RK-IB distinguishers: This work presents the first RK-IB distinguisher for ARX ciphers.
- vs. state-of-the-art RK-ID distinguishers: The best RK-ID distinguishers cover 7 rounds for SPECK-$2w/4w$ and 6 rounds for SPECK-$2w/3w$ [37]. Our RK-IB distinguishers cover 1–2 more rounds across all variants, consistent with our theoretical analysis in Sec. 3.6.

Example and Analysis. A representative RK-IB distinguisher of SPECK-32/64 is presented below. Other variants of SPECK are given in Table 6 within App. F.1. Apart from automatic verification using the state-based method, manual verification for Distinguisher 1 also appears in App. F.1.

Distinguisher 1. *$(\alpha, \alpha, \beta, \beta')$ is an 8-round RK-IB distinguisher for SPECK-32/64 under key differences $(\kappa, \kappa, 0, 0)$, where $\alpha = 0x00000000$, $\beta = 0x80008002$, $\beta' = 0x80008000$, and $\kappa = 0x0040000000000000$.*

Regarding the verification principle 1: (1) The master key of SPECK-32/64 is four times the length of the round key, resulting in 4-round probability-1 related-key differentials, which fixes most bit values after 5-round forward propagation. (2) The lower trail is propagated in the single-key setting, and certain bit values are fixed during the 2-round backward propagation. (3) A one-round contradiction arises from the differential properties of modular addition in the middle round.

5.2 GEBCT-based method on GIFT-64 in Four-related-key Setting

GIFT [38] is a lightweight SPN block cipher introduced at CHES 2017, widely used as the underlying primitive in authenticated encryption schemes and hash functions—including at least 10 NIST Lightweight Cryptography (LWC) competition candidates, such as GIFT-COFB. GIFT-64 features 64-bit block, 128-bit key, 4-bit Sbox, bit-permutation layer, and linear key schedule KS; and comprises 28 rounds of SubCells, PermBits, and AddRoundKey.

Configurations and Results. GIFT's bit-permutation linear layer makes it incompatible with all prior IB distinguisher search methods for 2- or more round contradictions. We address this by applying our GEBCT-based method in four-related-keys setting $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\lambda, \lambda, \eta, \eta)$. Our search proceeds in two phases:

1. Let $(\alpha, \alpha', \beta, \beta') = (KS_0(\lambda), KS_0(\lambda), KS_r(\eta), KS_r(\eta))$, where KS_i denotes the i-th round key difference derived from master key differences. This setting enables 4-round probability-1 related-key differentials on both ends.

2. Search for λ, η such that $(0,0,0,0)$ forms an $(r-8)$-round core RK-IB distinguisher.

Combine them to yield an r-round RK-IB distinguisher $(\alpha, \alpha', \beta, \beta')$. It results in 48 13-round RK-IB distinguishers in 0.23 hours. Here, we restrict λ and η to values that enforce exactly one active nibble in both Round 5 and Round $(r-5)$, yielding a search space of $(16 \times 3)^2 = 2304$. All results are automatically verified by the state-based method. We also confirm it is the upper bounds on RK-IB distinguisher coverage within the same search space in 1.91 hours.

Comparison.

- vs. prior RK-IB distinguishers: This work presents the first RK-IB distinguisher for bit-permutation ciphers.
- vs. state-of-the-art RK-ID distinguishers: The best RK-ID distinguishers cover 12 rounds for GIFT-64 [20]. Our RK-IB distinguishers cover one additional round, consistent with the flexibility of key difference selection in IB attacks.

Example and Analysis. A representative 13-round RK-IB distinguisher for GIFT-64 is shown below, which is also verified manually.

Distinguisher 2. *$(\alpha, \alpha, \beta, \beta)$ is a 13-round RK-IB distinguisher for GIFT-64 under key differences $(\lambda, \lambda, \eta, \eta)$, where $\alpha = 0x0001000000000000$, $\beta = 0x0000000000020000$, $\lambda = 0x00000000000000000000000000001000$, $\eta = 0x0000000000000000100000000000000$.*

Verification. (Proof by Contradiction) As 4-round probability-1 related-key differentials are appended to both ends of the distinguisher, proving the distinguisher is equivalent to proving that $(0,0,0,0)$ cannot propagate through the middle 5 rounds under key differences $(\lambda, \lambda, \eta, \eta)$. We first enumerate all differential characteristics that may propagate through the middle five rounds, and subsequently disprove them. Let $(\sigma_0^r, \sigma_1^r, \sigma_2^r, \sigma_3^r)$ and $(\phi_0^r, \phi_1^r, \phi_2^r, \phi_3^r)$ denote the input and output quartet-differences of round r, respectively. Let $x_{\ell,j}^r$ and $y_{\ell,j}^r$ represent the j-th nibble of x_ℓ^r and y_ℓ^r, where $0 \le j \le 15$ and $0 \le \ell \le 3$. We proceed in three steps:

1. We propagate the input differences forward for 2 rounds and the output differences backward for 2 rounds, then check whether these differences can be connected according to the GBCT of the j-th Sbox at Round 6 ($j \in \{2,3,7,15\}$[11]). Using a simple Python program, we eliminate most trails. For the remaining differences, we have $\phi_{0,0}^4, \phi_{2,0}^4 \in \{8,9,10,11\}$ and $\sigma_{1,8}^8, \sigma_{3,8}^8 \in \{5,6,7,12,13,15\}$. Accordingly, the third bit of $\phi_{0,0}^4, \phi_{2,0}^4$ and the second bit of $\sigma_{1,8}^8, \sigma_{3,8}^8$ are all equal to 1. Hence, all propagation trails from input differences to output differences are necessarily of the pattern shown in Figure 4.
2. The trail shown in Figure 4 implies $\sigma_{0,3}^6 = \sigma_{2,3}^6 = 1$ and $\phi_{1,3}^6 = \phi_{3,3}^6 = 8$. According to the GEBCT, $\phi_{0,3}^6 = \phi_{2,3}^6 = 8$ always holds for all feasible values of $\sigma_{1,3}^6$ and $\sigma_{3,3}^6$.
3. Since $\phi_{0,0}^4, \phi_{2,0}^4 \in \{8,9,10,11\}$, we derive $\sigma_{0,0}^7, \sigma_{2,0}^7 \in \{8,9,10,11\}$ via DDT lookup and differential propagation.

For all deduced feasible values of $\sigma_{0,0}^7, \sigma_{1,0}^7, \sigma_{2,0}^7, \sigma_{3,0}^7$, $\phi_{1,0}^7$ and $\phi_{3,0}^7$, it holds that GEBCT$(\sigma_{0,0}^7, \sigma_{1,0}^7, \sigma_{2,0}^7, \sigma_{3,0}^7, \phi_{0,0}^7, \phi_{1,0}^7, \phi_{2,0}^7, \phi_{3,0}^7) = 0$ for all possible values of $\phi_{0,0}^7$ and $\phi_{2,0}^7$, which leads to a contradiction. □

[11] This set consists of necessary nibbles obtained via our verification strategy.

It can be observed from the verification that the contradictions span 5 consecutive middle rounds. Current 1-round or 2-round contradiction detection methods fail to capture this multi-round dependency, demonstrating the unique advantage of our GEBCT-based approach.

5.3 State-based Method on PRINTcipher48 in Single-key Setting

PRINTcipher48 [39] is a lightweight SPN block cipher presented at CHES 2010, designed for printable circuits. It features 48-bit block, 80-bit key, 3-bit key-dependent Sbox (controlled by 2 bits of the master key); and comprises 48 rounds. The 80-bit key is reused each round: the 64-bit segment is XORed with the internal state, and the 32-bit segment controls the Sboxes (control keys).

Configurations and Results. PRINTcipher's key-dependent Sboxes make it incompatible with all prior IB distinguisher search methods, which cannot model state propagation or key-dependent components. We address this by applying our state-based method. Our search for SK-IB distinguishers proceeds as: Enumerate all 1-bit active input/output difference pairs with either $\alpha = \alpha'$ or $\beta = \beta'$, and verify each candidate using full state propagation modeling. This yields a search space of $2 \times 48^3 = 221184$ candidates, and results in 2 5-round SK-IB distinguishers in 0.46 hours. We also confirm it is the upper bounds on distinguisher coverage within the same search space in 1.25 hours[12].

Comparison.

- vs. prior SK-IB distinguishers: This work presents the first IB distinguisher distinguishers for any cipher with key-dependent Sboxes.
- vs. state-of-the-art SK-ID distinguishers: The best SK-ID distinguishers cover only 4 rounds for PRINTcipher48 [20]. Our 5-round SK-IB distinguishers provide a concrete counterexample to Lu's long-standing claim that SK-IB distinguishers cannot outperform SK-ID distinguishers in round coverage.

Example and Analysis. A representative 5-round SK-IB distinguisher for PRINTcipher48 is shown below. It is verified by computer-assisted analysis, and full details are given in App. F.3.

Distinguisher 3. $(\alpha, \alpha', \beta, \beta')$ *is a 5-round SK-IB distinguisher for PRINTcipher 48, where* $\alpha = 0x000080000000$, $\alpha' = 0x400000000000$, $\beta = 0x000100000000$, $\beta' = 0x000100000000$.

Its validity is verified through computer-assisted analysis in App. F.3. The verification principle is elaborated as follows: (1) For any valid quartet-differential trail, we can deduce the feasible set of control keys from the input/output differences of each key-dependent Sbox. (2) PRINTcipher48 uses the same master key in all rounds, meaning the same Sbox is controlled by identical key bits across rounds. (3) Comparing control key constraints across multiple rounds reveals that no single control key can satisfy all trail requirements, leading to a contradiction.

[12] The distinguisher is computed offline and reused online, so its computation time poses no overhead for the attacker.

5.4 Structural Method on AES-128 in Two-related-key Setting

AES-128 [40] is the NIST-standard 128-bit security-level block cipher and the most widely used symmetric cipher worldwide. It features a 10-round SPN structure with 128-bit block, 128-bit key, 8-bit Sbox, MDS linear layer, and nonlinear key schedule.

Configurations and Results. AES-128's nonlinear key schedule make it incompatible with all prior multi-round IB distinguisher search methods. We address this by applying our structural method in two-related-keys setting $(\kappa, \kappa, 0, 0)$, where setting $\kappa = \alpha$ cancels input differences with subkey differences. Our search proceeds as: Search for structural RK-IB distinguishers of the form $(\alpha, \alpha, \beta, \beta')$ under the above key difference setting with exactly 1-byte active differences in α, β, β'. This yields a total search space of $16^3 = 2^{12}$, and results in 768 5-round RK-IB distinguishers in 0.45 hours. We prove no 6-round structural RK-IB distinguishers exist in this space in 0.58 hours.

Comparison.

- vs. prior RK-IB distinguishers: This work presents the first RK-IB distinguishers for AES-128.
- vs. state-of-the-art RK-ID distinguishers: The best RK-ID distinguishers cover only 3 rounds in the same arbitrary Sbox model[13] [19]. Our RK-IB distinguishers cover two additional rounds.

Example and Analysis. A representative 5-round RK-IB distinguisher for AES-128 (without final MixColumns) is shown below.

Distinguisher 4. *$(\alpha, \alpha, \beta, \beta')$ is a 5-round RK-IB distinguisher for AES-128 under key differences $(\kappa, \kappa, 0, 0) = (\alpha, \alpha, 0, 0)$, where*

$$\begin{cases} \alpha &= (0,0,0,0,0,0,0,0,u,0,0,0,0,0,0,0) \in (\mathbb{F}_2^8)^{16}, \\ \beta &= (0,0,v,0,0,0,0,0,0,0,0,0,0,0,0,0) \in (\mathbb{F}_2^8)^{16}, \\ \beta' &= (v',0,0,0,0,0,0,0,0,0,0,0,0,0,0,0) \in (\mathbb{F}_2^8)^{16}, \end{cases}$$

for non-zero u, v, v'.

Verification. (Proof by Contradiction) Assume (α, α) can propagation to (β, β'). As shown in Figure 5, $X_1 \oplus X_0 = \alpha, X_3 \oplus X_2 = \alpha$, and $Y_2 \oplus Y_1 = \beta, Y_3 \oplus Y_0 = \beta'$. Let Z_i be the value by encrypting X_i after 3 rounds without the last MC layer, and W_i be the value 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, given that $W_{1,0} \oplus W_{2,0} = 0, W_{0,0} \oplus W_{3,0} = \delta'_0 \neq 0, W_{1,1} \oplus W_{2,1} = 0, W_{0,1} \oplus W_{3,1} = 0, W_{1,2} \oplus W_{2,2} = \delta_2 \neq 0, W_{0,2} \oplus W_{3,2} = 0, W_{1,3} \oplus W_{2,3} = 0, W_{0,3} \oplus W_{3,3} = 0$, it follows that $W_{0,0} \oplus W_{1,0} \oplus W_{2,0} \oplus W_{3,0} = \delta'_0 \neq 0$, $W_{0,1} \oplus W_{1,1} \oplus W_{2,1} \oplus W_{3,1} = 0$, $W_{0,2} \oplus W_{1,2} \oplus W_{2,2} \oplus W_{3,2} = \delta_2 \neq 0$ and $W_{0,3} \oplus W_{1,3} \oplus W_{2,3} \oplus W_{3,3} = 0$. On the other hand,

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

[13] RK-ID distinguisher (α, β) where α, β are required to be 1-byte active with key difference $\kappa = \alpha$

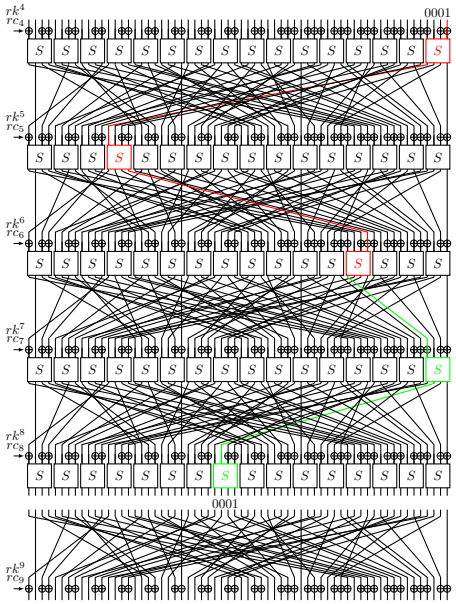


Fig. 4: The core of one 13-round RK-IB distinguisher for GIFT-64

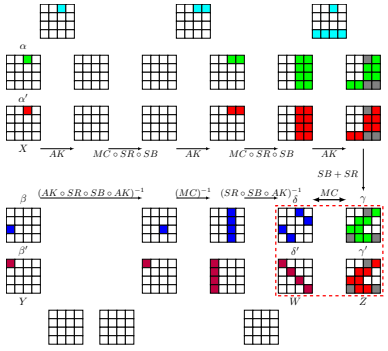


Fig. 5: The 5-round RK-IB distinguisher for AES-128

If $\gamma_2 \oplus \gamma_2' = \gamma_3 \oplus \gamma_3' = 0$, then $W_{0,i} \oplus W_{1,i} \oplus W_{2,i} \oplus W_{3,i} = 0$ for $0 \le i \le 3$, which leads to a contradiction. If $\gamma_2 \oplus \gamma_2' \neq 0$ or $\gamma_3 \oplus \gamma_3' \neq 0$, then by the properties of the MDS matrix, at least three values of i in $\{0, 1, 2, 3\}$ satisfy $W_{0,i} \oplus W_{1,i} \oplus W_{2,i} \oplus W_{3,i} = 0$, which again results in a contradiction. □

In summery, in the upper trail, related-key differences enable deterministic one-round differential propagation. Meanwhile, the slow diffusion of the key schedule ensures that certain words remain zero in the contradiction rounds—even when key differences are injected. In contrast, the lower trail propagates under the single-key model, forcing specific bytes to be non-zero at those same contradiction rounds. By jointly exploiting these two trails, the distinguisher can be well established by making use of the inherent properties of MDS matrices.

6 An RK-IB Attack Example on 31-round SKINNYee

SKINNYee (CRYPTO 2022) [14], is an extension of the tweakable block cipher SKINNY [17], specifically designed for the newly proposed operation mode HOMA. It features 64-bit block, 128-bit key, 256-bit tweak, and 4-bit S-box. Using our GEBCT-based method, we obtain 5 valid 23-round RK-IB distinguishers (verified by the state-based method, see App. F.6). A representative distinguisher is shown below.

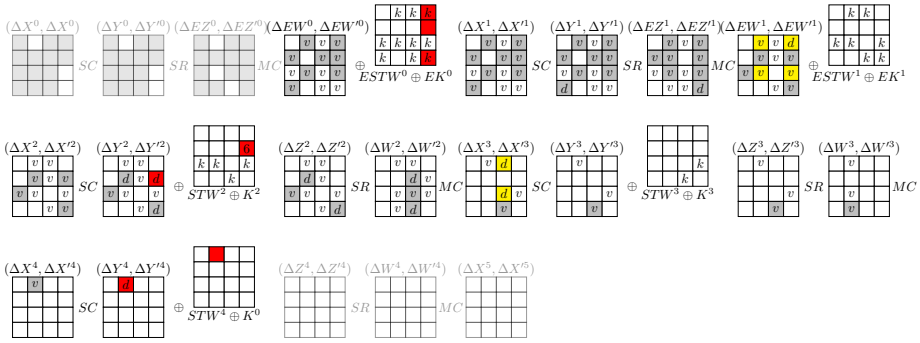


Fig. 6: Top 5 rounds added for key recovery in 31-round attack on SKINNYee

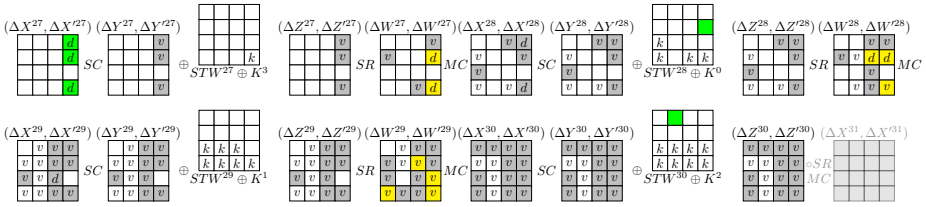


Fig. 7: Bottom 4 rounds added for key recovery in our 31-round attack on SKINNYee

Distinguisher 5. *$(\alpha, \alpha, \beta, \beta)$ is a 23-round RK-IB distinguisher for SKINNYee (without initial SC) under tweak differences $(\Delta TW, \Delta TW, \Delta TW', \Delta TW')$, where:*

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

Here α and β denote the input difference at Round 4 and the output difference at Round 27, respectively. Let $\eta = (\eta_0, \ldots, \eta_7)$ and $\theta = (\theta_0, \ldots, \theta_7)$ denote the upper and lower trail key differences from Round 0 to Round 7. Then ΔTW and $\Delta TW'$ are determined by η and θ, respectively.

We mount the first 31-round RK-IB distinguisher attack on SKINNYee by appending 5 rounds above and 4 rounds below this distinguisher (Figures 6 and 7) in the key recovery framework [22,29,30]. Red and yellow nibbles in Fig. 6, and green and yellow nibbles in Fig. 7 represent nibbles whose differences are used in key sieving by adopting the early abort technique [41]; the gray nibble denotes that the unknown difference. Let $c = 4$ denote the S-box size, other notations are defined as follows:

Table 3: The key recovery steps of the early abort technique of SKINNYee

Step	Difference	Guessed keys	Time complexity	Remained quartets
1	$\Delta X^3_{10}, \Delta X'^3_{10}$	$K^0_{0,0}, K^1_{0,2}$	$2^{11c+2c} \cdot 2^{18c} \cdot (2 \cdot 4)/(16 \cdot 31)$	$2^{18c}/2^{2c} = 2^{16c}$
2	$\Delta X^3_2, \Delta X'^3_2$	$K^1_{0,1}$	$2^{13c+c} \cdot 2^{16c} \cdot 4/(16 \cdot 31)$	$2^{16c}/2^{2c} = 2^{14c}$
3	$\Delta W^{28}_6, \Delta W'^{28}_6$	$K^1_{0,6}, K^2_{0,3}, K^2_{0,5}$	$2^{14c+3c} \cdot 2^{14c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{14c}/2^{2c} = 2^{12c}$
4	$\Delta W^{27}_{15}, \Delta W'^{27}_{15}$	$K^0_{0,7}$	$2^{17c+c} \cdot 2^{12c} \cdot 4/(16 \cdot 31)$	$2^{12c}/2^{2c} = 2^{10c}$
5	$\Delta W^{27}_7, \Delta W'^{27}_7$ $\Delta X^{27}_3, \Delta X'^{27}_3$	$K^1_{0,0}, K^1_{0,4}, K^2_{0,1}$	$2^{18c+3c} \cdot 2^{10c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{10c}/2^{2\cdot 2c} = 2^{6c}$
6	$\Delta X^{27}_7, \Delta X'^{27}_7$	$K^1_{0,5}$	$2^{21c+c} \cdot 2^{6c} \cdot 4/(16 \cdot 31)$	$2^{6c}/2^{2c} = 2^{4c}$
7	$\Delta X^{27}_{15}, \Delta X'^{27}_{15}$	$K^3_{0,7}$	$2^{22c+c} \cdot 2^{4c} \cdot 4/(16 \cdot 31)$	$2^{4c}/2^{2c} = 2^{2c}$
8	$\Delta Y^4_1, \Delta Y'^4_1$	$K^1_{0,3}, K^3_{0,3}, K^3_{0,6}$	$2^{23c+3c} \cdot 2^{2c} \cdot 3 \cdot 4/(16 \cdot 31)$	$2^{2c}/2^{2c} = 2^0$



Fig. 8: The relation of K^r_ℓ and its equivalent version $EK^r_\ell (0 \le \ell \le 3)$ of SKINNYee

- X^r_ℓ (input to SC), Y^r_ℓ (output of SC), Z^r_ℓ (output of SR), and W^r_ℓ (output of MC), for $\ell = 0, 1, 2, 3$.
- $(\Delta X^r_0, \Delta X'^r_0)$ (resp. $(\Delta X^r_1, \Delta X'^r_1)$): the two differences in the upper trails (resp. lower trails), where $X^r_0 \oplus X^r_1 = \Delta X^r_0$ and $X^r_2 \oplus X^r_3 = \Delta X'^r_0$ (resp. $X^r_1 \oplus X^r_2 = \Delta X^r_1$ and $X^r_0 \oplus X^r_3 = \Delta X'^r_1$). Analogous notations apply to Y, Z and W.
- $K^{\overline{r}}_\ell$ for $\ell = 0, 1, 2, 3$, where $\overline{r} = r \mod 4$ (round key in round r); STW^r: the subtweak in round r.
- $X^r_{\ell,j}$: the j-th nibble of X^r_ℓ, for $\ell = 0, 1, 2, 3$. Analogous notation applies to Y, Z, W, $K^{r \bmod 4}_\ell$, and all difference variables such as ΔX^r and $\Delta X'^r$.

Equivalent round keys and tweaks $EK^r_\ell = \mathrm{MC}(\mathrm{SR}(0 \| K^r_\ell))$ for $\ell = 0, 1, 2, 3$, and $ESTW^r = MC(SR(STW^r \| 0))$ are used for Round 0 and 1 (Figure 8). Meanwhile, the input and output of MC in round r are denoted as EZ^r_ℓ and EW^r_ℓ, and the corresponding differences are $(\Delta EZ^r, \Delta EW'^r)$ and $(\Delta EW^r, \Delta EW'^r)$.

The attack is detailed as follows.

1. For the four related tweaks $(T, T \oplus \Delta TW, T \oplus \Delta TW \oplus \Delta TW', T \oplus \Delta TW')$ defined in Distinguisher 5, query all 2^{16c} plaintexts and store the plaintext-ciphertext pairs in tables $\mathcal{T}_0$ to $\mathcal{T}_3$.
2. Guess $K^0_{0,j}$ (j=1, 2, 3, 4, 6), $K^1_{0,7}$, $K^2_{0,j}$ (j=0, 2, 4, 6, 7), then:
   (a) For each $(EW^0_\ell, Z^{30}_\ell) \in \mathcal{T}_\ell$ ($\ell = 0, 1, 2, 3$), we partially decrypt Z^{30}_ℓ using the guessed keys $K^1_{\ell,7}$ and $K^2_{\ell,j}$ ($j = 0, 2, 4, 6, 7$) to obtain $W^{29}_{\ell,j}$ ($j = 6, 11, 12, 15$) and $W^{28}_{\ell,j}$ ($j = 7, 15$). We then store (EW^0_ℓ, Z^{30}_ℓ) together with these partial decryption values in a new table $\mathcal{T}'_\ell$.

(b) Define $RK^0_\ell = EST K^0_\ell \oplus EK^0_\ell$ ($\ell = 0, 1, 2, 3$). Since $K^0_{0,j}$ ($j = 1, 2, 3, 4, 6$) are known, $RK^0_{\ell,j}$ ($j = 1, 3, 9, 11, 12$) can be determined according to Fig. 8. Let $x_i = (x_{i,1}, x_{i,3}, x_{i,4}, x_{i,6}, x_{i,9}, x_{i,11}, x_{i,12}) \in (\mathbb{F}_2^4)^7$ for $i = 0, 1$. Construct a table $\mathcal{U}_0$ storing all pairs (x_0, x_1) satisfying the constraints:

$$\begin{cases} \bigoplus_{\substack{i=0,1 \\ j=1,11}} S(x_{i,j} \oplus RK^0_{i,j}) = 0, \ \bigoplus_{\substack{i=0,1 \\ j=3,9,12}} S(x_{i,j} \oplus RK^0_{i,j}) = 0, \\ \bigoplus_{i=0,1} \big(S(x_{i,4}) \oplus S(x_{i,11} \oplus RK^0_{i,11})\big) = 0, \\ \bigoplus_{i=0,1} S\big(S(x_{i,3} \oplus RK^0_{i,3})\big) = 6, \ \bigoplus_{i=0,1} \big(S(x_{i,6}) \oplus S(x_{i,9} \oplus RK^0_{i,9})\big) = 0. \end{cases}$$

For a fixed x_0, there are 2^{2c} valid x_1 values. Similarly, construct a table $\mathcal{U}_1$ using RK^0_2 and RK^0_3.

(c) Let $\overline{EW}^0_\ell = (EW^0_{\ell,1}, EW^0_{\ell,3}, EW^0_{\ell,4}, EW^0_{\ell,6}, EW^0_{\ell,9}, EW^0_{\ell,11}, EW^0_{\ell,12})$. For each plaintext EW^0_0, construct the list $L_0 = \{(EW^0_0, C_0, EW^0_1, C_1)\}$, where $(EW^0_i, C_i) \in \mathcal{T}'_i$ for $i = 0, 1$. Moreover, EW^0_1 satisfies $(\overline{EW}^0_0, \overline{EW}^0_1) \in \mathcal{U}_0$, $EW^0_{1,7} \in \mathbb{F}_2^4$, and all other nibbles of EW^0_1 are identical to those of EW^0_0. The size of L_0 is $2^{16c+3c} = 2^{19c}$. Similarly, construct the list $L_1 = \{(EW^0_2, C_2, EW^0_3, C_3)\}$ of size 2^{19c}, where $(EW^0_i, C_i) \in \mathcal{T}'_i$ for $i = 2, 3$.

(d) Insert L_0 into a hash table $\mathcal{H}$, indexed by the $20c$ bits from the following nibbles: $Z^{30}_{i,j} (i = 0, 1; j = 4, 5, 9, 13)$, $W^{29}_{i,j} (i = 0, 1; j = 6, 11, 12, 15)$, and $W^{28}_{i,j} (i = 0, 1; j = 7, 15)$. For each $(EW^0_2, C_2, EW^0_3, C_3) \in L_1$, search $\mathcal{H}$ for matching entries $(EW^0_0, C_0, EW^0_1, C_1)$ such that:

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

This yields $Q = 2^{2 \cdot 19c - 2 \cdot 10c} = 2^{18c}$ valid quartets.

(e) Guess the necessary key bits and adopt the early abort technique [41] to rule out wrong key candidates. The detailed key recovery process is presented in Table 3.

3. Perform an exhaustive search over the remaining key bits.

Complexity. The total number of keys to be guessed is 2^{26c}. As shown in Table 3, a total of 18 c-bit conditions need to be satisfied. Thus, after adopting the early abort technique, the probability that a key candidate survives filtering is $p = \big(1 - 2^{-18c}\big)^Q = e^{-1}$. The data complexity is $2^2 \cdot 2^{16c} = 2^{66}$. The time complexity is dominated by Step 3. The time complexity of exhaustive search is fully discussed in [10,30]. It is given by $2^{\kappa-2}\big(1 - (1-p)^4\big)$, where κ denotes the key size. The time complexity of this attack is $2^{128-2}\big(1 - (1 - e^{-1})^4\big) \approx 2^{125.75}$. The maximum memory consumption occurs in Step 2(e), where 2^{26c} candidates are stored, resulting in a memory complexity of 2^{104}.

Additionally, under the beyond full-codebook setting [11], we achieve improved complexities: $2^{67.2}$ data, $2^{123.53}$ time, and 2^{104} memory. This matches the result in Table 2 and represents the best cryptanalytic result for `SKINNYee` to date. Full attack details are provided in App. G.

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*, 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. 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.
9. 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.
10. 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.
11. 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.
12. 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.
13. 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.
14. 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.

15. Jérémy Jean, Ivica Nikolić, Thomas Peyrin, and Yannick Seurin. Deoxys v1.41. Submitted to the CAESAR Competition, October 2016. Available at `https://competitions.cr.yp.to/round3/deoxysv141.pdf`.
16. Jérémy Jean, Ivica Nikolić, and Thomas Peyrin. Joltik v1.3. Submitted to the CAESAR Competition (Round 2), March 2015. Available at `https://competitions.cr.yp.to/round2/joltikv13.pdf`.
17. 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.
18. 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.
19. 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.
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. 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.
22. Haoyang Wang and Jianing Zhang. Optimizing key recovery in impossible cryptanalysis and its automated tool. *Des. Codes Cryptogr.*, 94(2):45, 2026.
23. 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.
24. 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 Tech-*

niques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II, volume 10821 of *Lecture Notes in Computer Science*, pages 683–714. Springer, 2018.
25. 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.
26. Stéphanie Delaune, Patrick Derbez, and Mathieu Vavrille. Catching the fastest boomerangs application to SKINNY. *IACR Trans. Symmetric Cryptol.*, 2020(4):104–129, 2020.
27. Libo Wang, Ling Song, Baofeng Wu, Mostafizar Rahman, and Takanori Isobe. Revisiting the boomerang attack from a perspective of 3-differential. *IEEE Trans. Inf. Theory*, 70(7):5343–5357, 2024.
28. 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.
29. Yincen Chen, Qinggan Fu, Ning Zhao, Jiahao Zhao, Ling Song, and Qianqian Yang. A holistic framework for impossible boomerang attacks. *IACR Communications in Cryptology*, 2(2), 2025.
30. Lin Jiao, Xichao Hu, Dengguo Feng, Yongqiang Li, Senpeng Wang, Yonglin Hao, and Xinxin Gong. A unified key recovery framework for impossible boomerang attacks: Applications to full-round-aradi and skinnye v2. *IEEE Transactions on Information Theory*, pages 1–1, 2026.
31. Patricia Greene, Mark Motley, and Bryan Weeks. ARADI and LLAMA: low-latency cryptography for memory encryption. *IACR Cryptol. ePrint Arch.*, page 1240, 2024.
32. 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.
33. 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.
34. Tim Beyne and Vincent Rijmen. Differential cryptanalysis in the fixed-key model. 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*, Lecture Notes in Computer Science, pages 687–716. Springer, 2022.
35. Eli Biham and Adi Shamir. Differential cryptanalysis of des-like cryptosystems. In Alfred Menezes and Scott A. Vanstone, editors, *Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings*, Lecture Notes in Computer Science, pages 2–21. Springer, 1990.
36. Christina Boura, Patrick Derbez, and Baptiste Germon. Extending the quasidifferential framework: From fixed-key to expected differential probability. *IACR Trans. Symmetric Cryptol.*, 2025(1):515–541, 2025.
37. 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.
38. 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.

39. 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.
40. Joan Daemen and Vincent Rijmen. *The Design of Rijndael: AES - The Advanced Encryption Standard*. Information Security and Cryptography. Springer, 2002.
41. 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.
42. Tyge Tiessen. Polytopic cryptanalysis. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology - EUROCRYPT 2016 - 35th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Vienna, Austria, May 8-12, 2016, Proceedings, Part I*, Lecture Notes in Computer Science, pages 214–239. Springer, 2016.
43. 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.
44. 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.
45. Mihir Bellare, Viet Tung Hoang, and Stefano Tessaro. Message-recovery attacks on feistel-based format preserving encryption. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, Vienna, Austria, October 24-28, 2016*, pages 444–455. ACM, 2016.
46. Ralph Ankele, Subhadeep Banik, Avik Chakraborti, Eik List, Florian Mendel, Siang Meng Sim, and Gaoli Wang. Related-key impossible-differential attack on reduced-round skinny. In Dieter Gollmann, Atsuko Miyaji, and Hiroaki Kikuchi, editors, *Applied Cryptography and Network Security - 15th International Conference, ACNS 2017, Kanazawa, Japan, July 10-12, 2017, Proceedings*, volume 10355 of *Lecture Notes in Computer Science*, pages 208–228. Springer, 2017.
47. Carlos Cid, Tao Huang, Thomas Peyrin, Yu Sasaki, and Ling Song. A security analysis of deoxys and its internal tweakable block ciphers. *IACR Trans. Symmetric Cryptol.*, 2017(3):73–107, 2017.
48. Rui Zong, Xiaoyang Dong, and Xiaoyun Wang. Related-tweakey impossible differential attack on reduced-round deoxys-bc-256. *Sci. China Inf. Sci.*, 62(3):32102:1–32102:12, 2019.

49. Xichao Hu, Lin Jiao, Dengguo Feng, Yongqiang Li, Senpeng Wang, Yonglin Hao, and Xinxin Gong. A unified key recovery framework for impossible boomerang attacks: Applications to full-round-ARADI and SKINNYe v2. Cryptology ePrint Archive, Paper 2025/056, 2025.

A Construction of IB distinguishers via Generalized BTs

A.1 Construction of IB Distinguishers via Generalized BTs for SPN Ciphers

We present the definitions of generalized BCT, UBCT and LBCT as follows.

Definition 6. *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 GBCT, GUBCT and GLBCT for an n-bit Sbox are defined as*
$\mathrm{GBCT}(\mu, \mu', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$,
$\mathrm{GUBCT}(\mu, \mu', \theta, \theta', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \theta, \theta', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', S(x_0) \oplus S(x_1) = \theta, S(x_2) \oplus S(x_3) = \theta', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$,
$\mathrm{GLBCT}(\mu, \mu', \rho, \rho', \varphi, \varphi') = \#\,\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi')$, *where*
$\mathrm{IN}_{\mathrm{GLBCT}}(\mu, \mu', \rho, \rho', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_0 \oplus x_1 = \mu, x_2 \oplus x_3 = \mu', x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$,

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
$\mathrm{IN}_{\mathrm{LDDT}}(\rho, \rho', \varphi, \varphi') = \{(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^{4n} \mid x_1 \oplus x_2 = \rho, x_0 \oplus x_3 = \rho', S(x_1) \oplus S(x_2) = \varphi, S(x_0) \oplus S(x_3) = \varphi'\}$.

These generalized tables satisfy the following relations.

Proposition 1. $\mathrm{UDDT}, \mathrm{LDDT}, \mathrm{GBCT}, \mathrm{GUBCT}, \mathrm{GLBCT}$ *and* GEBCT *satisfy 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. $\mathrm{IN}_{\mathrm{GEBCT}}(\mu,\mu',\rho,\rho',\theta,\theta',\varphi,\varphi') \subseteq \mathrm{IN}_{\mathrm{GLBCT}}(\mu,\mu',\rho,\rho',\varphi,\varphi')$.

Proof. The proof of such inclusion relations follows directly from the definitions of these generalized tables. □

With the definitions of these generalized BTs in place, we propose an approach that incorporates the mixed use of these tables to construct IB distinguishers. For SPN ciphers, we define a set of propagation rules for E_i with t parallel Sboxes $(S_0,\ldots,S_{t-1})$ as $\mathcal{AP}^S_{E_i} = \{(p_0,\ldots,p_{t-1}) \mid p_\iota \in \{\mathrm{UDDT},\mathrm{LDDT},\mathrm{GB}-\mathrm{CT},\mathrm{GUBCT},\mathrm{GLBCT},\mathrm{GEBCT}\}$ for $\iota \in \{0,\ldots,t-1\}\}$, where $P_i = (p_0,\ldots,p_{t-1}) \in \mathcal{AP}^S_{E_i}$ denotes that the propagation rule through the ι-th Sbox in E_i follows p_ι for $\iota \in \{0,\ldots,t-1\}$. Using generalized BTs as tools, we can construct IB distinguishers as follows.

Definition 7. *Given* $E = E_{r-1,\mathrm{KS}_{r-1}(k)} \circ \cdots \circ E_{0,\mathrm{KS}_0(k)}$ *as an* r*-round block cipher with the key schedule* KS*, let* $\epsilon_0^i,\epsilon_1^i,\epsilon_2^i,\epsilon_3^i$ *denote 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 6*

$$(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{\boldsymbol{P_0}} \cdots \xrightarrow{\boldsymbol{P_{r-1}}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta'), \tag{6}$$

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 related-key* $\boldsymbol{P}$*-based quartet-differential trail. Here,* $\boldsymbol{P} = (\boldsymbol{P_0},\ldots,\boldsymbol{P_{r-1}})$ *is a predefined propagation rule of* E*, where* $\boldsymbol{P_i} \in \mathcal{AP}^S_{E_i}$ *for* $i \in \{0,\ldots,r-1\}$*, and* $\xrightarrow{\boldsymbol{P_i}}$ *denotes the propagation rule through Sboxes in* E_i *follows* $\boldsymbol{P_i}$*. In particular, if* $\kappa_0 = \kappa_1 = \kappa_2 = \kappa_3 = 0$*, then Trail 6 is called an* r*-round single-key* $\boldsymbol{P}$*-based quartet-differential trail.*

Each propagation rule $\boldsymbol{P_i} \in \mathcal{AP}^S_{E_i}$ must be compatible. Accordingly, we present the following constructions.

Construction 4 ($\boldsymbol{P}$-based Method). *Let* $\boldsymbol{P} = (\boldsymbol{P_0},\ldots,\boldsymbol{P_{r-1}})$ *be a predefined propagation rule of* E*, where* $\boldsymbol{P_i} \in \mathcal{AP}^S_{E_i}$ *for* $i \in \{0,\ldots,r-1\}$*, Given an* r*-round block cipher* E *with the key schedule* KS*, for two input differences* α,α' *and two output differences* β,β'*, and* $\kappa_0,\kappa_1,\kappa_2,\kappa_3$ *four key differences satisfying* $\kappa_3 = \kappa_0 \oplus \kappa_1 \oplus \kappa_2$*. If there exists no* r*-round* $\boldsymbol{P}$*-based related-key (resp. single-key) quartet-differential trail, then* $(\alpha,\alpha',\beta,\beta')$ *is defined as an* r*-round* $\boldsymbol{P}$*-based RK-IB distinguisher (resp.* $\boldsymbol{P}$*-based SK-IB distinguisher).*

A.2 Construction of IB Distinguishers via Generalized BTs for Feistel Ciphers

To apply boomerang switches for Feistel ciphers, boomerang tables are thus proposed.

Definition 8 ([28]). *Given four differences $\mu, \rho, \theta, \phi \in \mathbb{F}_2^n$, the FBCT, BDT and FBET for an n-bit Sbox are defined as*

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

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

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

The incompatibility of boomerang distinguishers caused by internal dependencies in Feistel ciphers was first observed by Boukerrou et al. [28]. To address this issue, the authors extended the conventional BCT and BDT for Feistel structures and proposed the FBCT, FBDT, and FBET. To construct `IB` distinguishers, we generalize the FBCT and FBDT and further define three new generalized tables: GFBCT, GFUBCT, and GFLBCT. In addition, the generalized table of FEBCT is defined in the same manner as the GEBCT.

Definition 9. *Given nine differences $\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi', \eta \in \mathbb{F}_2^n$ where $\rho' = \mu \oplus \mu' \oplus \rho$, $\varphi' = \theta \oplus \theta' \oplus \varphi$, the GFBCT, GFUBCT and GFLBCT for an n-bit Sbox are 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 of these generalized BTs is depicted in Fig. 9.

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,*

1. *The notation $(\gamma,\gamma') \xrightarrow{\mathrm{GFBCT}} (\delta,\delta')$ means that the difference propagation rule over Sboxes satisfies the GFBCT constraint. Specifically, for the cipher E illustrated in Fig. 9, we denote $\gamma = \gamma_L \parallel \gamma_R, \gamma' = \gamma'_L \parallel \gamma'_R, \delta = \delta_L \parallel \delta_R, \delta' = \delta'_L \parallel \delta'_R$. Let $LL_1(\gamma_L) = (a_{t-1},\ldots,a_0), LL_1(\gamma'_L) = (a'_{t-1},\ldots,a'_0)$, $LL_1(\delta_R) = (b_{t-1},\ldots,b_0), LL_1(\delta'_R) = (b'_{t-1},\ldots,b'_0), LL_2^{-1}(\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L) = (c_{t-1},\ldots,c_0)$, where LL_1 and LL_2 represent two linear layers, and the substitution layer SL contains t Sboxes. Here, a_i, a'_i, b_i, b'_i and c_i stand for the related input/output differences involved in the* GFBCT *of the i-th Sbox with*

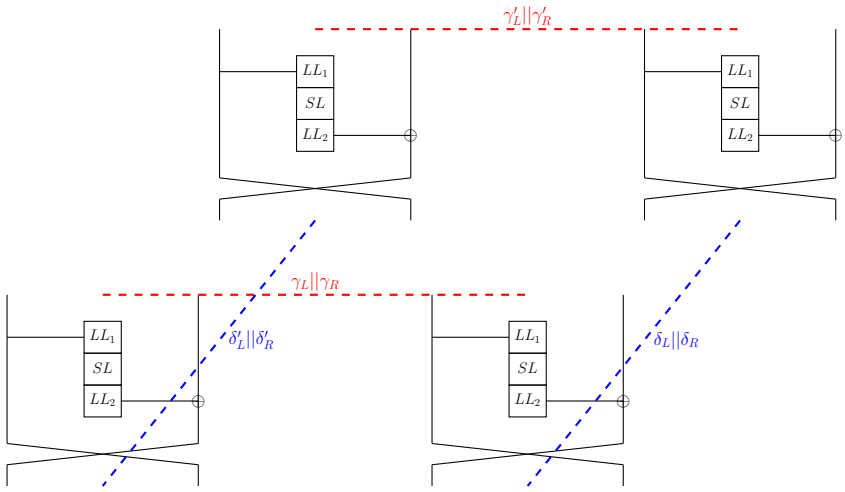


Fig. 9: The illustration of differential propagation rule through Sboxes in E based on GFBCT

$i = 0, 1, \ldots, t-1$. *Accordingly,* $(\gamma, \gamma') \xrightarrow{\text{GFBCT}} (\delta, \delta')$ *holds if and only if there exists some integer* $i \in \{0, 1, \ldots, t-1\}$ *such that* $\text{GFBCT}(a_i, a'_i, b_i, b'_i, c_i) \neq 0$.

2. *The notation* $(\gamma, \gamma') \xrightarrow{\text{GFUBCT}} (\omega, \omega', \delta, \delta')$ *indicates that the propagation rule through Sboxes follows* GFUBCT*, which is equivalent to* $(\gamma, \gamma') \xrightarrow{\text{GFUBCT}} (\delta, \delta')$ *and* $(\gamma, \gamma') \xrightarrow{\text{UDDT}} (\omega, \omega')$.
3. *The notation* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GFLBCT}} (\delta, \delta')$ *means that the difference propagation rule through Sboxes satisfies* GFLBCT*, which is equivalent to that both* $(\gamma, \gamma') \xrightarrow{\text{GFBCT}} (\delta, \delta')$ *and* $(\eta, \eta') \xrightarrow{\text{LDDT}} (\delta, \delta')$ *hold.*
4. *The notation* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GEBCT}} (\omega, \omega', \delta, \delta')$ *indicates that the propagation rule through Sboxes follows* GEBCT*, which is equivalent to that both* $(\gamma, \gamma') \xrightarrow{\text{GFUBCT}} (\omega, \omega', \delta, \delta')$ *and* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GFLBCT}} (\delta, \delta')$ *hold.*

Proposition 2. *The relations among* GFBCT, GFUBCT GFLBCT, GEBCT, UDDT *and* LDDT *are given as follows:*

1. *If* $(\gamma, \gamma') \xrightarrow{\text{GFBCT}} (\delta, \delta')$*, then there exist* $\eta, \eta', \omega, \omega'$ *such that* $(\gamma, \gamma') \xrightarrow{\text{UDDT}} (\omega, \omega')$ *and* $(\eta, \eta') \xrightarrow{\text{LDDT}} (\delta, \delta')$.
2. *If* $(\gamma, \gamma') \xrightarrow{\text{GFUBCT}} (\omega, \omega', \delta, \delta')$*, then there exist* η, η' *such that* $(\gamma, \gamma') \xrightarrow{\text{UDDT}} (\omega, \omega')$ *and* $(\eta, \eta') \xrightarrow{\text{LDDT}} (\delta, \delta')$.
3. *If* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GFLBCT}} (\delta, \delta')$*, then there exist* ω, ω' *such that* $(\gamma, \gamma') \xrightarrow{\text{UDDT}} (\omega, \omega')$ *and* $(\eta, \eta') \xrightarrow{\text{LDDT}} (\delta, \delta')$.
4. *If* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GEBCT}} (\omega, \omega', \delta, \delta')$*, then* $(\gamma, \gamma') \xrightarrow{\text{UDDT}} (\omega, \omega')$, $(\eta, \eta') \xrightarrow{\text{LDDT}} (\delta, \delta')$, $(\gamma, \gamma') \xrightarrow{\text{GFUBCT}} (\omega, \omega', \delta, \delta')$ *and* $(\gamma, \gamma', \eta, \eta') \xrightarrow{\text{GFLBCT}} (\delta, \delta')$.

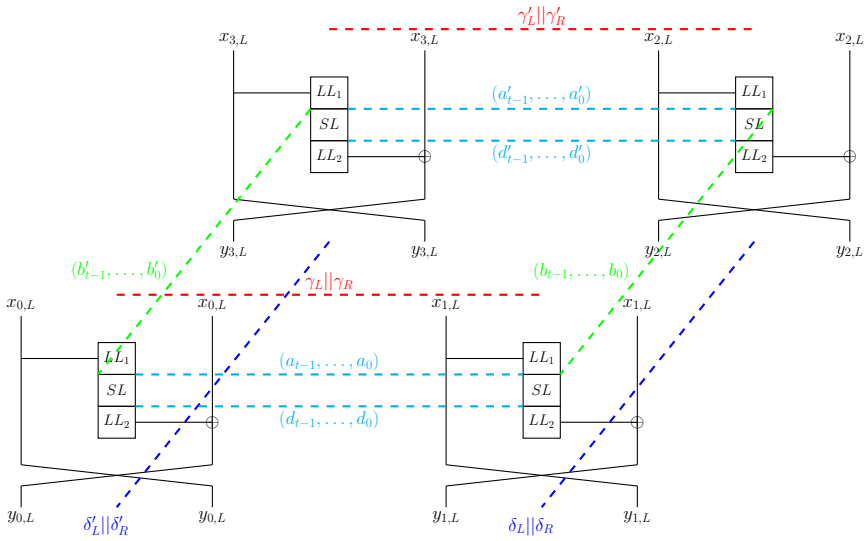


Fig. 10: The relation between GFBCT and UDDT

Proof. As illustrated in Fig. 10, let l_i denote the linear function corresponding to LL_i for $i = 1, 2$. We write $\gamma = \gamma_L\|\gamma_R$, $\gamma' = \gamma'_L\|\gamma'_R$, $\delta = \delta_L\|\delta_R$, and $\delta' = \delta'_L\|\delta'_R$. Define

$$\begin{aligned}(a_{t-1},\ldots,a_0) &= l_1(\gamma_L), \quad (a'_{t-1},\ldots,a'_0) = l_1(\gamma'_L),\\ (b_{t-1},\ldots,b_0) &= l_1(\delta_R), \quad (b'_{t-1},\ldots,b'_0) = l_1(\delta'_R),\\ (c_{t-1},\ldots,c_0) &= l_2^{-1}(\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_R),\end{aligned}$$

where a_i, a'_i, b_i, b'_i and c_i represent input and output differences of Sboxes in the substitution layer SL, and t denotes the total number of Sboxes contained in SL. For input differences a_i and a'_i, in accordance with the definition of GFBCT, there exist corresponding output differences d_i and d'_i such that a_i propagates to d_i and a'_i propagates to d'_i, satisfying $d_i \oplus d'_i = c_i$ for $0 \le i \le t-1$. Let $\lambda = l_2\big((d_{t-1},\ldots,d_0)\big)$ and $\lambda' = l_2\big((d'_{t-1},\ldots,d'_0)\big)$. It follows that $\gamma_R \oplus \gamma'_R \oplus \delta_L \oplus \delta'_L = \lambda \oplus \lambda'$. Hence, there exist λ and λ' satisfying the above equality, such that the difference γ can propagate to $(\gamma_R \oplus \lambda)\|\gamma_L$, and the difference γ' can propagate to $(\gamma'_R \oplus \lambda')\|\gamma'_L$. Consequently, we have

$$(\gamma, \gamma') \xrightarrow{\text{UDDT}} \big((\gamma_R \oplus \lambda)\|\gamma_L,\ (\gamma'_R \oplus \lambda')\|\gamma'_L\big).$$

The remaining relations can be proved by analogous arguments. □

We also consider a hybrid use of UDDT, LDDT, GFBCT, GFUBCT, GFLBCT and GEBCT to construct `IB` distinguishers. For Feistel ciphers, we define a set of propagation rules for E_i with t parallel Sboxes $(S_0,\ldots,S_{t-1})$ as $\mathcal{AP}^F_{E_i} = \{(p_0,\ldots,p_{t-1}) | p_\iota \in \{\text{UDDT}, \text{LDDT}, \text{GFBCT}, \text{GFUBCT}, \text{GFLBCT}, \text{GEBCT}\}$ for $\iota \in \{0,\ldots,t-1\}\}$, where $P_i = (p_0,\ldots,p_{t-1}) \in \mathcal{AP}^F_{E_i}$ denotes that the

propagation rule through the ι-th Sbox in E_i follows p_ι for $\iota \in \{0, \ldots, t-1\}$. By replacing $\mathcal{AP}^S_{E_i}$ with $\mathcal{AP}^F_{E_i}$, Definition 7 and Construction 4 also apply to Feistel ciphers.

A.3 Construction of IB Distinguishers via Generalized BTs for ARX Ciphers

For ARX ciphers, instead of regarding modular addition as a $2n$-bit to n-bit mapping, we treat it as a $2n \times 2n$-bit Sbox, inspired by [33]. As illustrated in Fig. 11(a), this Sbox is defined as $S(x\|y) = (x \boxplus y, y)$. Thus, the definitions of these Sbox-based generalized BTs are still valid 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 this way, the `IB` distinguisher construction methods based on our generalized BTs are also applicable to ARX ciphers. For instance, the `SPECK` cipher can be viewed as an SPN cipher when modular addition is regarded as an Sbox, where round keys are XORed with part of the internal state, as shown in Fig. 11(b).

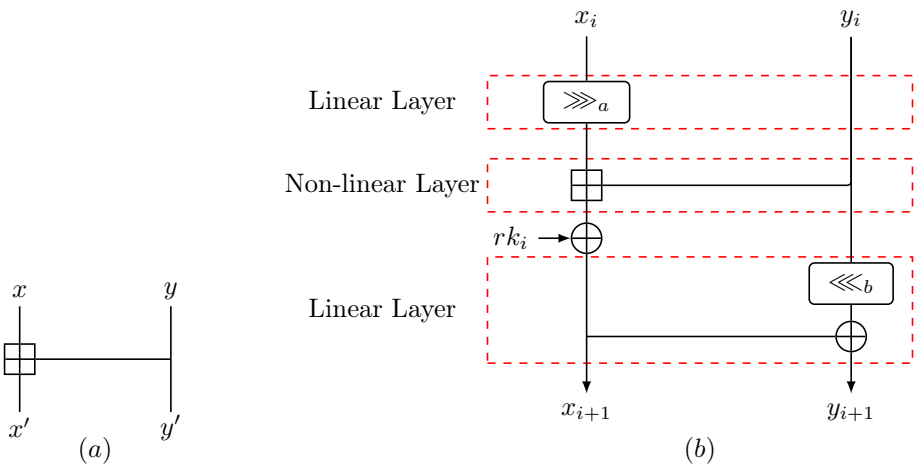


Fig. 11: The SPN cipher view of the ARX cipher `SPECK`

Additionally, for atypical SPN-like or Feistel-like ciphers, we can adopt equivalent transformations to convert them into standard SPN or Feistel structures, as demonstrated by `CHAM` in Fig. 12. Accordingly, the `IB` distinguisher construction methods based on our generalized BTs remain valid for such ciphers.

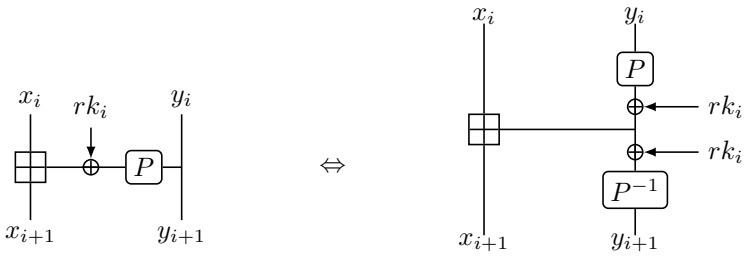


Fig. 12: The SPN view of the ARX cipher `CHAM`

A.4 Relationships Between the *P*-based Method and the GEBCT-based Method

It is obvious that the GEBCT-based method is a special case of the ***P***-based method. Furthermore, their concrete relationships are presented as follows.

Theorem 4. *For any predefined rule* ***P****, an* r*-round* ***P*****`-based IB`** *distinguisher* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* **`GEBCT-based IB`** *distinguisher.*

Proof. (proof by contradiction) If an r-round ***P*****`-based IB`** distinguisher $(\alpha, \alpha', \beta, \beta')$ is not an r-round `GEBCT-based IB` distinguisher, there must exist at least one r-round GEBCT-based related-key (resp. single-key) quartet-differential trail: $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^{r_0}, \epsilon_1^{r_0}, \epsilon_2^{r_0}, \epsilon_3^{r_0}) \xrightarrow{\text{GEBCT}} (\epsilon_0^{r_0+1}, \epsilon_1^{r_0+1}, \epsilon_2^{r_0+1}, \epsilon_3^{r_0+1}) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^r, \epsilon_1^r, \epsilon_2^r, \epsilon_3^r)$. Based on the relations between GEBCT and other tables in Proposition 1 and Proposition 2, it is also an r-round ***P***-based related-key (resp. single-key) quartet-differential trail. Thus, $(\alpha, \alpha', \beta, \beta')$ is not an r-round ***P*****`-based IB`** distinguisher. □

B Proofs of Relations Between the Newly Proposed Methods and Existing Methods

B.1 Proof of the Relationship Between State-based and GEBCT-based Methods

The proof of Theorem 3 relies on the following three lemmas.

Lemma 1. *An* r*-round* **`GEBCT-based IB`** *distinguisher* $(\alpha, \alpha', \beta, \beta')$ *is also an* r*-round* **`state-based IB`** *distinguisher.*

Proof. (proof by contradiction) If an r-round `GEBCT-based IB` distinguisher $(\alpha, \alpha', \beta, \beta')$ is not an r-round `state-based IB` distinguisher, there must exist an r-round quartet-state 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

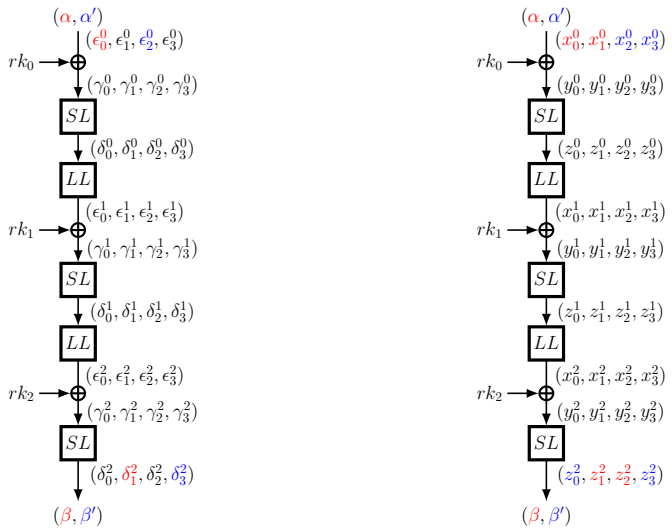


Fig. 13: The equivalence between `GEBCT-based IB` distinguishers and `state-based IB` distinguishers for SPN ciphers with full-state round-key-xor

$x_0^0 \oplus x_1^0 = \alpha$, $x_2^0 \oplus x_3^0 = \alpha'$, $x_1^r \oplus x_2^r = \beta$, $x_0^r \oplus x_3^r = \beta'$. Let $\epsilon_0^i = x_0^i \oplus x_1^i$, $\epsilon_1^i = x_1^i \oplus x_2^i$, $\epsilon_2^i = x_2^i \oplus x_3^i$, and $\epsilon_3^i = x_0^i \oplus x_3^i$ for $0 \leq i \leq r$. Then we can construct an r-round quartet-differential trail: $(\epsilon_0^0 = \alpha, \epsilon_1^0, \epsilon_2^0 = \alpha', \epsilon_3^0) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^r, \epsilon_1^r = \beta, \epsilon_2^r, \epsilon_3^r = \beta')$, which contradicts the assumption that $(\alpha, \alpha', \beta, \beta')$ is an r-round `GEBCT-based IB` distinguisher. □

Lemma 2. *Given an SPN cipher with full-state round-key-xor, $(\alpha, \alpha', \beta, \beta')$ is an r-round `state-based IB` distinguisher constructed under the independent round-key setting if and only if it is also an r-round `GEBCT-based IB` distinguisher.*

Proof. The schematic diagram is given in Fig. 13. This is equivalent to proving that $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \xrightarrow{\text{AK}} (\gamma_0^0, \gamma_1^0, \gamma_2^0, \gamma_3^0) \xrightarrow{\text{GEBCT}} (\delta_0^0, \delta_1^0, \delta_2^0, \delta_3^0) \xrightarrow{\text{LL}} (\epsilon_0^1, \epsilon_1^1, \epsilon_2^1, \epsilon_3^1) \xrightarrow{\text{AK}} \cdots \xrightarrow{\text{GEBCT}} (\delta_0^{r-1}, \delta_1^{r-1}, \delta_2^{r-1}, \delta_3^{r-1})$ is an r-round GEBCT-based boomerang trail if and only if $(x_0^0, x_1^0, x_2^0, x_3^0) \xrightarrow{\text{AK}} (y_0^0, y_1^0, y_2^0, y_3^0) \xrightarrow{\text{SL}} (z_0^0, z_1^0, z_2^0, z_3^0) \xrightarrow{\text{LL}} (x_0^1, x_1^1, x_2^1, x_3^1) \xrightarrow{\text{AK}} \cdots \xrightarrow{\text{SL}} (z_0^{r-1}, z_1^{r-1}, z_2^{r-1}, z_3^{r-1})$ is an r-round state-based boomerang trail under the independent round-key setting, 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 $r = 3$. Other cases can be proved analogously. Suppose $(\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0) \xrightarrow{\text{AK}} (\gamma_0^0, \gamma_1^0, \gamma_2^0, \gamma_3^0) \xrightarrow{\text{GEBCT}} (\delta_0^0, \delta_1^0, \delta_2^0, \delta_3^0) \xrightarrow{\text{LL}} (\epsilon_0^1, \epsilon_1^1, \epsilon_2^1, \epsilon_3^1) \xrightarrow{\text{AK}} \cdots \xrightarrow{\text{GEBCT}} (\delta_0^2, \delta_1^2, \delta_2^2, \delta_3^2)$ is a 3-round

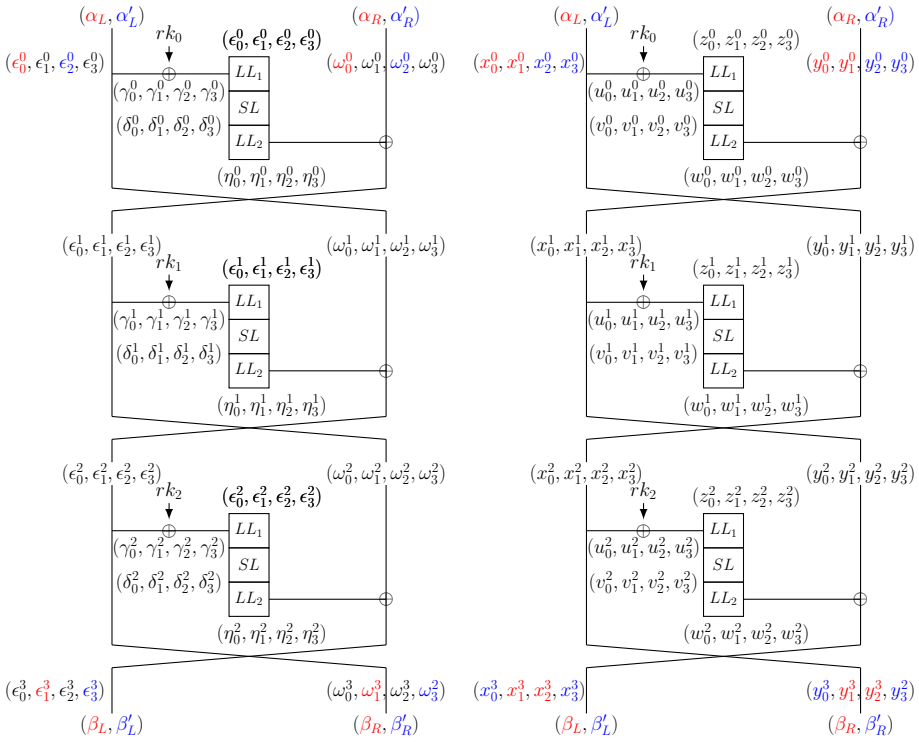


Fig. 14: The equivalence between `GEBCT-based IB` distinguishers and `state-based IB` distinguishers for Feistel ciphers with full-left-branch state round-key-xor

GEBCT-based boomerang trail. Since $(\gamma_0^i, \gamma_1^i, \gamma_2^i, \gamma_3^i) \xrightarrow{\text{SL}} (\delta_0^i, \delta_1^i, \delta_2^i, \delta_3^i)$, there exist $(y_0^i, y_1^i, y_2^i, y_3^i)$ and $(z_0^i, z_1^i, z_2^i, z_3^i)$ such that

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

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 a 3-round state-based boomerang trail under the independent round-key setting. The above process is invertible. □

Lemma 3. *Given a Feistel cipher with full-left-branch state round-key-xor,* $(\alpha, \alpha', \beta, \beta')$ *is an* r*-round* `state-based IB` *distinguisher constructed under the independent round-key setting if and only if it is also an* r*-round* `GEBCT-based IB` *distinguisher.*

Proof. As shown in Fig. 14, this is equivalent to proving that $((\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0), (\omega_0^0, \omega_1^0, \omega_2^0, \omega_3^0)) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} ((\epsilon_0^r, \epsilon_1^r, \epsilon_2^r, \epsilon_3^r), (\omega_0^r, \omega_1^r, \omega_2^r, \omega_3^r))$ is an r-round GEBCT-based boomerang trail if and only if $((x_0^0, x_1^0, x_2^0, x_3^0), (y_0^0, y_1^0, y_2^0, y_3^0)) \to \cdots \to ((x_0^r, x_1^r, x_2^r, x_3^r), (y_0^r, y_1^r, y_2^r, y_3^r))$ is an r-round state-based boomerang trail under the independent round-key setting, 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 for the case $r = 3$. Other cases can be proved analogously. Suppose $((\epsilon_0^0, \epsilon_1^0, \epsilon_2^0, \epsilon_3^0), (\omega_0^0, \omega_1^0, \omega_2^0, \omega_3^0)) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} ((\epsilon_0^3, \epsilon_1^3, \epsilon_2^3, \epsilon_3^3), (\omega_0^3, \omega_1^3, \omega_2^3, \omega_3^3))$ is a 3-round GEBCT-based boomerang trail. For $0 \le i \le 2$, since $(\gamma_0^i, \gamma_1^i, \gamma_2^i, \gamma_3^i) \xrightarrow{\text{SL, GEBCT}} (\delta_0^i, \delta_1^i, \delta_2^i, \delta_3^i)$, 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 $rk_i = x_j^i \oplus z_j^i (0 \le j \le 3)$. Let $x_j^{i+1} = w_j^i \oplus y_j^i (0 \le j \le 3)$, then $((x_0^0, x_1^0, x_2^0, x_3^0), (y_0^0, y_1^0, y_2^0, y_3^0)) \to \cdots \to \cdots \to ((x_0^3, x_1^3, x_2^3, x_3^3), (y_0^3, y_1^3, y_2^3, y_3^3))$ is a 3-round state-based boomerang trail under the independent round-key setting. The above process is invertible. □

B.2 Proofs of Relationships Between Existing and Newly Proposed Construction Methods

We prove Theorem 2 based on the following three lemmas. By Definition 4, the XOR of the input quartet-differences and the output quartet-differences equal zero. Thus, for a Trail (2)

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

it holds that

$$\epsilon_0^i \oplus \epsilon_1^i \oplus \epsilon_2^i \oplus \epsilon_3^i = 0 \ (0 \le i \le r),$$

which follows from the fact that the quartet-state is evaluated twice.

Lemma 4. *An `IB distinguisher` detected by Lu's method is indeed a `GEBCT-based IB` distinguisher.*

Proof. Assume that $(\alpha, \alpha', \beta, \beta')$ is an r-round `IB distinguisher` identified by Lu's method for $E = E^1 \circ E^0$, where E^0 covers r_0 rounds and E^1 covers r_1 rounds. Then there exist four (truncated) probability-1 differentials

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

such that the four intermediate differences satisfy $\delta \oplus \delta' \oplus \gamma \oplus \gamma' \neq 0$. Thus, in the upper trails, the quartet-differential

$$(\epsilon_0^0 = \alpha,\, \epsilon_1^0 = *,\, \epsilon_2^0 = \alpha',\, \epsilon_3^0 = *) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^{r_0} = \delta,\, \epsilon_1^{r_0} = *,\, \epsilon_2^{r_0} = \delta',\, \epsilon_3^{r_0} = *)$$

holds with probability-1 in the forward direction, and the quartet-differential

$$(\epsilon_0^r = *,\, \epsilon_1^r = \beta,\, \epsilon_2^r = *,\, \epsilon_3^r = \beta') \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\overline{\epsilon}_0^{r_0} = *,\, \overline{\epsilon}_1^{r_0} = \gamma,\, \overline{\epsilon}_2^{r_0} = *,\, \overline{\epsilon}_3^{r_0} = \gamma')$$

holds with probability-1 in the backward direction. Here, $*$ denotes any value. If $(\alpha, \alpha', \beta, \beta')$ is not an r-round `GEBCT-based IB` distinguisher, then $\epsilon_\ell^{r_0} = \overline{\epsilon}_\ell^{r_0}$ for $0 \leq \ell \leq 3$, and

$$\epsilon_0^{r_0} \oplus \epsilon_1^{r_0} \oplus \epsilon_2^{r_0} \oplus \epsilon_3^{r_0} = 0.$$

However, this leads to a contradiction, since $\delta \oplus \delta' \oplus \gamma \oplus \gamma' \neq 0$. Consequently, $(\alpha, \alpha', \beta, \beta')$ must be an r-round `GEBCT-based IB` distinguisher. □

By Proposition 1, it follows that

$$\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu', \varphi, \varphi').$$

In particular, we have $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu, \rho, \rho, \theta, \theta, \varphi, \varphi) \subseteq \mathrm{IN}_{\mathrm{GBCT}}(\mu, \mu, \varphi, \varphi)$. Thus, if $(\mu, \mu') \overset{\text{GBCT}}{\nrightarrow} (\varphi, \varphi')$ or $\mu \overset{\text{BCT}}{\nrightarrow} \varphi$, then $(\mu, \mu', \rho, \rho') \overset{\text{GEBCT}}{\nrightarrow} (\theta, \theta', \varphi, \varphi')$. This directly gives the following lemma.

Lemma 5. *An `IB distinguisher` detected by BCT/GBCT-based method is indeed a `GEBCT-based IB` distinguisher.*

Proof. Assume that $(\alpha, \alpha', \beta, \beta')$ is an r-round `IB distinguisher` identified by the BCT/GBCT-based method for $E = E^1 \circ E^m \circ E^0$, where E^0, E^m, and E^1 cover r_0, $r_m = 1$, and r_1 rounds, respectively. Then there exist four (truncated) probability-1 differentials

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

such that the difference pair (δ, δ') cannot propagate to (γ, γ') under the constraints of the BCT or GBCT. Thus, in the upper trails, the quartet-differential

$$(\epsilon_0^0 = \alpha,\, \epsilon_1^0 = *,\, \epsilon_2^0 = \alpha',\, \epsilon_3^0 = *) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^{r_0} = \delta,\, \epsilon_1^{r_0} = *,\, \epsilon_2^{r_0} = \delta',\, \epsilon_3^{r_0} = *)$$

holds with probability-1 in the forward direction, and the quartet-differential

$$(\epsilon_0^r = *,\, \epsilon_1^r = \beta, \epsilon_2^r = *,\, \epsilon_3^r = \beta') \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\overline{\epsilon}_0^{r_0+r_m} = *,\, \overline{\epsilon}_1^{r_0+r_m} = \gamma,\, \overline{\epsilon}_2^{r_0+r_m} = *,\, \overline{\epsilon}_3^{r_0+r_m} = \gamma')$$

holds with probability-1 in the backward direction. Here, $*$ denotes any value. If $(\alpha, \alpha', \beta, \beta')$ is not an r-round GEBCT-based `IB distinguisher`, then $(\delta, *, \delta', *) \xrightarrow{\text{GEBCT}} (*, \gamma, *, \gamma')$. However, according to the relationships between BCT and GEBCT, as well as between GBCT and GEBCT, it holds that

$$(\delta, *, \delta', *) \overset{\text{GEBCT}}{\nrightarrow} (*, \gamma, *, \gamma').$$

This yields a contradiction. Consequently, $(\alpha, \alpha', \beta, \beta')$ must be an r-round GEBCT-based `IB` distinguisher. □

By Proposition 1, it follows that

$$\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu', \rho, \rho', \nu, \nu', \phi, \phi') \subseteq \mathrm{IN}_{\mathrm{GUBCT}}(\mu, \mu', \nu, \nu', \phi, \phi'),$$

$$\mathrm{IN}_{\mathrm{GEBCT}}(\nu, \nu', \phi, \phi', \theta, \theta', \varphi, \varphi') \subseteq \mathrm{IN}_{\mathrm{GLBCT}}(\nu, \nu', \phi, \phi', \varphi, \varphi').$$

In particular, we have $\mathrm{IN}_{\mathrm{GEBCT}}(\mu, \mu, \rho, \rho, \nu, \nu, \phi, \phi) \subseteq \mathrm{IN}_{\mathrm{UBCT}}(\mu, \nu, \phi)$, and $\mathrm{IN}_{\mathrm{GEBCT}}(\nu, \nu, \phi, \phi, \theta, \theta, \varphi, \varphi) \subseteq \mathrm{IN}_{\mathrm{LBCT}}(\nu, \phi, \varphi)$. Thus, if $\mu \overset{\text{DBCT}}{\nrightarrow} \varphi$, then there exist ν and ϕ such that either $\mu \overset{\text{UBCT}}{\nrightarrow} (\nu, \phi)$ or $(\nu, \phi) \overset{\text{LBCT}}{\nrightarrow} \varphi$. This implies that $(\mu, \mu, \rho, \rho) \overset{\text{DBCT}}{\nrightarrow} (\nu, \nu, \phi, \phi)$ or $(\nu, \nu, \phi, \phi) \overset{\text{DBCT}}{\nrightarrow} (\theta, \theta, \varphi, \varphi)$.

In [27], DBCT* is defined as

$$\begin{aligned}
\mathrm{DBCT}^*(\mu, \varphi) = &\sum_{\nu, \phi, \phi' \in \mathbb{F}_2^n} \left(\sum_{\rho \in \mathbb{F}_2^n} \# \left\{ x \in \mathbb{F}_2^n \;\middle|\; (\mu, \rho, \mu \oplus \rho) \xrightarrow[x]{S} (\nu, \phi, \phi') \right\} \right) \\
&\times \left(\sum_{\theta \in \mathbb{F}_2^n} \# \left\{ x \in \mathbb{F}_2^n \;\middle|\; (\nu, \phi, \phi') \xrightarrow[x]{S} (\theta, \varphi, \theta \oplus \varphi) \right\} \right) \\
= &\sum_{\nu, \phi, \phi' \in \mathbb{F}_2^n} \mathrm{GUBCT}(\mu, \mu, \nu, \nu \oplus \phi \oplus \phi', \phi, \phi') \\
&\times \mathrm{GLBCT}(\nu, \nu \oplus \phi \oplus \phi', \phi, \phi', \varphi, \varphi).
\end{aligned}$$

Thus, if $\mu \overset{\text{DBCT}^*}{\nrightarrow} \varphi$, then $(\mu, \mu, \rho, \rho) \overset{\text{DBCT}}{\nrightarrow} (\nu, \nu, \phi, \phi)$ or $(\nu, \nu, \phi, \phi) \overset{\text{DBCT}}{\nrightarrow} (\theta, \theta, \varphi, \varphi)$.

Lemma 6. *An `IB distinguisher` detected by DBCT/DBCT*-based method is indeed a `GEBCT-based IB` distinguisher.*

Proof. Assume that $(\alpha, \alpha', \beta, \beta')$ is an r-round `IB distinguisher` identified by the DBCT/DBCT*-based method for $E = E^1 \circ E^m \circ E^0$, where E^0, E^m, and E^1 cover r_0, $r_m = 2$, and r_1 rounds, respectively. Then there exist four (truncated) probability-1 differentials

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

such that the difference pair (δ, δ') cannot propagate to (γ, γ') under the constraints of the BCT or GBCT. Thus, in the upper trails, the quartet-differential

$$(\epsilon_0^0 = \alpha,\, \epsilon_1^0 = *,\, \epsilon_2^0 = \alpha',\, \epsilon_3^0 = *) \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\epsilon_0^{r_0} = \delta,\, \epsilon_1^{r_0} = *,\, \epsilon_2^{r_0} = \delta',\, \epsilon_3^{r_0} = *)$$

holds with probability-1 in the forward direction, and the quartet-differential

$$(\epsilon_0^r = *,\, \epsilon_1^r = \beta, \epsilon_2^r = *,\, \epsilon_3^r = \beta') \xrightarrow{\text{GEBCT}} \cdots \xrightarrow{\text{GEBCT}} (\overline{\epsilon}_0^{r_0+r_m} = *,\, \overline{\epsilon}_1^{r_0+r_m} = \gamma,\, \overline{\epsilon}_2^{r_0+r_m} = *,\, \overline{\epsilon}_3^{r_0+r_m} = \gamma')$$

holds with probability-1 in the backward direction. Here, $*$ denotes any value. If $(\alpha, \alpha', \beta, \beta')$ is not an r-round `GEBCT-based IB` distinguisher, then $(\delta, *, \delta', *) \xrightarrow{\text{GEBCT}} (*, \gamma, *, \gamma')$ for E^m. However, according to the relationships between DBCT and GEBCT, as well as between DBCT* and GEBCT, it holds that

$$(\delta, *, \delta', *) \overset{\text{GEBCT}}{\nrightarrow} (*, \gamma, *, \gamma').$$

This yields a contradiction. Thus, $(\alpha, \alpha', \beta, \beta')$ must be an r-round `GEBCT-based IB` distinguisher. □

C A Brief Introduction to the Quasidifferential Framework

The Quasidifferential Framework (QDFM) is an innovative paradigm proposed by Beyne and Rijmen at CRYPTO 2022 [34]. This groundbreaking framework perfectly addresses the long-standing issue of exact differential probability calculation under the fixed-key model without relying on any extra heuristic assumptions.

Recently, Wang et al. [27] extended QDFM from classical differential cryptanalysis to `Boomerang` attacks [3] and `Polytopic` attacks [42]. They derived theoretically rigorous formulas to calculate the exact probabilities of d-differentials. In addition, from the perspective of 3-differentials, they established theoretical formulations for computing the exact probabilities of `Boomerang` distinguishers. In this section, we review the fundamental concepts of QDFM.

Definition 10 (d-difference [42]). *For a $(d+1)$-tuple $(m_0, m_1, \ldots, m_d)$, its d-difference is defined as the following d-tuple $(m_1 \oplus m_0, m_2 \oplus m_0, \ldots, m_d \oplus m_0)$. We refer to the first text m_0 of the $(d+1)$-tuple of messages as the anchor of the d-difference.*

It is straightforward to observe that a $(d+1)$-tuple is uniquely determined by its d-difference and its anchor. Analogous to the propagation of differences, the transition of d-differences is formally defined as follows.

Definition 11 (d-difference transition [42]). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n, \boldsymbol{\alpha} = (\alpha_0, \alpha_1, \ldots, \alpha_{d-1})$ and $\boldsymbol{\beta} = (\beta_0, \beta_1, \ldots, \beta_{d-1})$ be two d-differences over $\mathbb{F}_2^n$. We use the notation $\boldsymbol{\alpha} \xrightarrow[x]{F} \beta$ to denote the event that F maps the $(d+1)$-tuple of messages corresponding to the d-difference α with anchor x to a $(d+1)$-tuple of messages corresponding to d-difference β with anchor $F(x)$. More precisely, the notation $\alpha \xrightarrow[x]{F} \beta$ implies that $F(x \oplus \alpha_i) \oplus F(x) = \beta_i (0 \le i \le d-1)$.*

Definition 12 (Quasi-d-differential transition matrix [27]). *Let $F : \mathbb{F}_2^n \to \mathbb{F}_2^n$. The quasi-$d$-differential transition matrix D^F is of size $2^{n(d+1)} \times 2^{n(d+1)}$. The entry located at the $(v, \boldsymbol{\beta})$-th row and $(u, \boldsymbol{\alpha})$-th column is defined as*

$$D^F_{(v,\boldsymbol{\beta}),(u,\boldsymbol{\alpha})} = \frac{1}{2^n} \sum_{x \in \mathbb{F}_2^n,\, \boldsymbol{\alpha} \xrightarrow[F]{x} \boldsymbol{\beta}} (-1)^{u^\top x + v^\top F(x)},$$

where $(u, \boldsymbol{\alpha}) = (u, \alpha_1, \ldots, \alpha_d) \in \mathbb{F}_2^{n(d+1)}$ and $(v, \boldsymbol{\beta}) = (v, \beta_1, \ldots, \beta_d) \in \mathbb{F}_2^{n(d+1)}$.

With the quasi-d-differential transition matrix serving as a fundamental tool, we can rigorously derive the exact probability of 3-differentials.

Definition 13 (Quasi-3-differential trail [27]). *A quasi-3-differential trail for a cipher $E = E_{r-1} \circ \cdots \circ E_1 \circ E_0$ is a sequence $\varpi_0, \varpi_1, \ldots, \varpi_r$ of mask-3-differential quartets, where $\varpi_i = (u, a_i, b_i, c_i)$ $(0 \le i \le r)$. The correlation of this quasi-3-differential trail is defined as $\prod_{i=0}^{r-1} D^{E_i}_{\varpi_{i+1}, \varpi_i}$.*

Let $\varpi_0 = (0, a_0, b_0, c_0)$ and $\varpi_r = (0, a_r, b_r, c_r)$ denote the input and output mask-3-differential quartets, respectively. The exact probability of the 3-differential with input 3-difference (a_0, b_0, c_0) and output 3-difference (a_r, b_r, c_r) is

$$\Pr((a_0, b_0, c_0) \xrightarrow{E} (a_r, b_r, c_r)) = D^E_{\varpi_r, \varpi_0} = \sum_{\varpi_1, \ldots, \varpi_{r-1}} \prod_{i=0}^{r-1} D^{E_i}_{\varpi_{i+1}, \varpi_i}. \tag{7}$$

Eq. (7) provides the exact probability without any assumptions. By virtue of Eq. (7), Wang et al. [27] also presented an approach to calculate the exact probability of a `Boomerang` distinguisher $\Pr\big((\alpha, \alpha) \xrightarrow{E} (\beta, \beta)\big)$ for a cipher E. As illustrated in the left part of Fig. 15, by introducing arbitrarily valued differences η and ϵ, the propagation from the input differences (α, α) to the output differences (β, β) can be regarded as the propagation of the input 3-difference $(\alpha, \eta, \alpha \oplus \eta)$ and the output 3-difference $(\varepsilon, \beta, \beta \oplus \varepsilon)$. Thus, the exact probability of the `Boomerang` distinguisher can be derived.

D The Details of Modeling Methods

In this section, we present the SAT/SMT-based modeling methods for the propagation of quartet-differences through the `Equal`, `Copy`, `Xor`, and `Matrix-multiply`

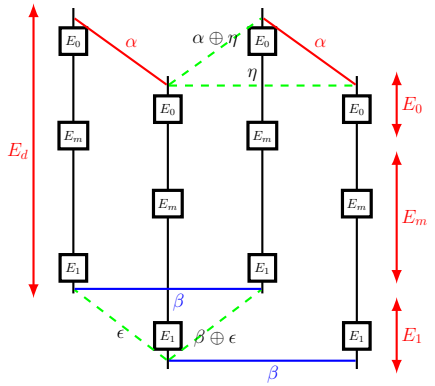


Fig. 15: The `Boomerang` distinguisher is constructed from the perspective of 3-differentials, in which the differences η and ϵ can take arbitrary values

operations, as well as the propagation of quartet-states through the `Equal`, `Copy`, `Xor`, `Matrix-multiply`, `Modular-addition`, and `Bitwise-and` operations. First, we propose the modeling method for quartet-state propagation.

Model 4. *For the* ***`Equal`*** *operation, let* $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^p)^4$ *denote the input quartet-state, and let* $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^p)^4$ *denote the output quartet-state. The method for modeling the propagation of the quartet-state through* ***`Equal`*** *is then given by*

$$y_\ell = x_0 \quad (0 \le \ell \le 3).$$

Model 5. *For the* ***`Copy`*** *operation, let* $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^p)^4$ *denote the input quartet-state, and let* $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^p)^4$ *and* $(z_0, z_1, z_2, z_3) \in (\mathbb{F}_2^p)^4$ *denote the two output quartet-states. The method for modeling the propagation of the quartet-state through* ***`Copy`*** *is then given by*

$$y_\ell = x_\ell, \quad z_\ell = x_\ell \quad (0 \le \ell \le 3).$$

Model 6. *For the* ***`Xor`*** *operation, let* $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^p)^4, (y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^p)^4$ *denote the two input quartet-states, and let* $(z_0, z_1, z_2, z_3) \in (\mathbb{F}_2^p)^4$ *denote the output quartet-state. The method for modeling the propagation of the quartet-state through* ***`Xor`*** *is then given by*

$$z_\ell = x_\ell \oplus y_\ell \quad (0 \le \ell \le 3).$$

For the `Matrix-multiply` operation, it can be expanded into expressions composed of `Equal` and `Xor` operations. Therefore, its modeling method can reuse the modeling method for `Equal` and `Xor` in an embedded manner. For the propagation rule of quartet states through `Modular-addition` and `Bitwise-and`, we have the following modeling methods.

Model 7. *For the* `Modular-addition` *operation, let* $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^p)^4, (y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^p)^4$ *denote the two input quartet-states, and let* $(z_0, z_1, z_2, z_3) \in (\mathbb{F}_2^p)^4$ *denote the output quartet-state. The method for modeling the propagation of the quartet-state through* `Modular-addition` *is then given by*

$$z_\ell = x_\ell + y_\ell \quad (0 \leq \ell \leq 3).$$

Model 8. *For the* `Bitwise-and` *operation, let* $(x_0, x_1, x_2, x_3) \in \mathbb{F}_2^4, (y_0, y_1, y_2, y_3) \in \mathbb{F}_2^4$ *denote the input quartet-state,* $\mathbb{Q}_z = (z_0, z_1, z_2, z_3) \in \mathbb{F}_2^4$ *denote its two output quartet-states. The method for modeling the propagation of quartet-state through* `Bitwise-and` *is then given by*

$$z_\ell = x_\ell \cdot y_\ell \ (0 \leq \ell \leq 3).$$

For the `Equal`, `Copy`, `Xor`, and `MatrixMultiply` operations, the propagation rules of quartet-differences are identical to those of quartet-states.

Table 4: The `Sbox` of `PRINTcipher`

x	0	1	2	3	4	5	6	7
$S(x)$	0	1	3	6	7	4	5	2

Example 1. Taking the `Sbox` of `PRINTcipher` (as shown in Table 4) as an example, let $(x_0, x_1, x_2, x_3) \in (\mathbb{F}_2^3)^4$ and $(y_0, y_1, y_2, y_3) \in (\mathbb{F}_2^3)^4$ denote the input and output quartet-states, respectively. The propagation of quartet-states through the `Sbox` can be modeled by

- $y_\ell =$(If $x_\ell = 0$ Then 0 ELSE (If $x_\ell = 1$ Then 1 Else (If $x_\ell = 2$ Then 3 Else (If $x_\ell = 3$ Then 6 Else (If $x_\ell = 4$ Then 7 Else (If $x_\ell = 5$ Then 4 Else (If $x_\ell = 6$ Then 5 Else 2))))))) $(0 \leq \ell \leq 3)$.

E Verification strategy

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; however, 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. Algorithm 1 can be regarded as modeling the forward propagation of quartet differences or quartet states from the input to the middle round and the backward propagation from the output to the middle round, with the constraint that the variables of the two middle quartets are equal.

- If the model remains unsolvable after removing the equality constraints on certain bit positions, these bit positions are irrelevant to the occurrence of contradictions. Using this method, we can eliminate positions unrelated to contradictions and further derive contradictions from the remaining ones. We adopt this method to help verify the distinguishers of `AEA-128`.
- If the model becomes solvable after removing equality constraints on certain bit positions, partial contradictions may occur at these positions. We can utilize the differential propagation properties of the Sboxes corresponding to those positions to eliminate invalid input-output differential trails of the distinguisher and manually verify the remaining feasible trails to improve efficiency. We adopt this method to help verify the distinguishers of `GIFT-64`.

- Traverse all plausible trails and disprove them. When computing resources permit, propagate the differences or states forward from the input to the middle round and backward from the output to the middle round. Then, use the properties of quartet-difference or quartet-state propagation in the middle round to disprove all these combined trails. We adopt this method to verify the distinguishers of `PRINTcipher48`.

- Automatic verification of `GEBCT-based IB` distinguishers. Since a `GEBCT-based IB` is essentially a `state-based IB` distinguisher, we perform automatic verification of `GEBCT-based IB` distinguishers by independently deriving them using the state-based method.

Besides, the correctness of the code is crucial for successfully searching for the correct distinguisher. In traditional methods, it takes time to verify the results obtained by automatic methods. Once the distinguishers are disproven, it is necessary to recheck and rerun the search program, which makes the entire distinguisher searching process extremely time-consuming. Therefore, an automatic verification technique that minimizes 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 `IB` distinguisher, the third-party solver will return a solution. The solution must correspond to trails that propagate input differences to output differences using differential propagation or state propagation. By verifying such trails, we may detect erroneous trails generated by incorrect code, thus improving the efficiency of the whole process.

F Specifications and Other Applications

In this section, we present a detailed introduction to the block ciphers SPECK, GIFT-64, PRINTcipher48 and AES-128, and elaborate on the application of our method to SKINNY-64/192, SKINNYee and CHAM.

F.1 SPECK

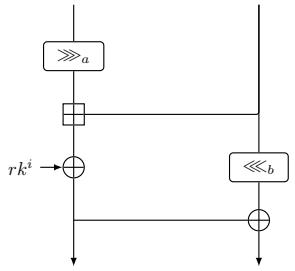


Fig. 16: One round of block cipher SPECK

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

Table 5: The parameters n, m of SPECK.

SPECK is a family of ARX ciphers proposed by researchers from the U.S. National Security Agency (NSA) [18], and has been incorporated into the RFID air interface standard ISO/IEC 29167-22.

Specification. The SPECK is usually denoted as SPECK-n/m where n and m are the block size and key size, respectively, in bits. It is referred to as SPECK-n if the key length does not need to be specified, where the parameters n and m are shown in Table 5. SPECK is an ARX cipher that uses modular addition operations. One round of SPECK is shown in Figure 16; the round function 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

$$l^{i+m-1} = (rk^i + (l^i \ggg_a)) \oplus i,$$
$$rk^{i+1} = rk^i \lll_b \oplus l^{i+m-1}.$$

The value rk^i represents the i-th round key.

Verification of Distinguisher 1. The distinguisher for SPECK-32/64 is verified as follows.

Table 6: The example of RK-IB distinguishers for SPECK

Block cipher	params	value
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	$0x000000000000000000000000000000000$
	β	$0x800000000000000080000000000000004$
	β'	$0x800000000000000080000000000000000$
SPECK-128/256	r	9
	κ	$0x000000000000008000000000000000000$
		$0x000000000000000000000000000000000$
	α	$0x000000000000000000000000000000000$
	β	$0x800000000000000080000000000000004$
	β'	$0x800000000000000080000000000000000$

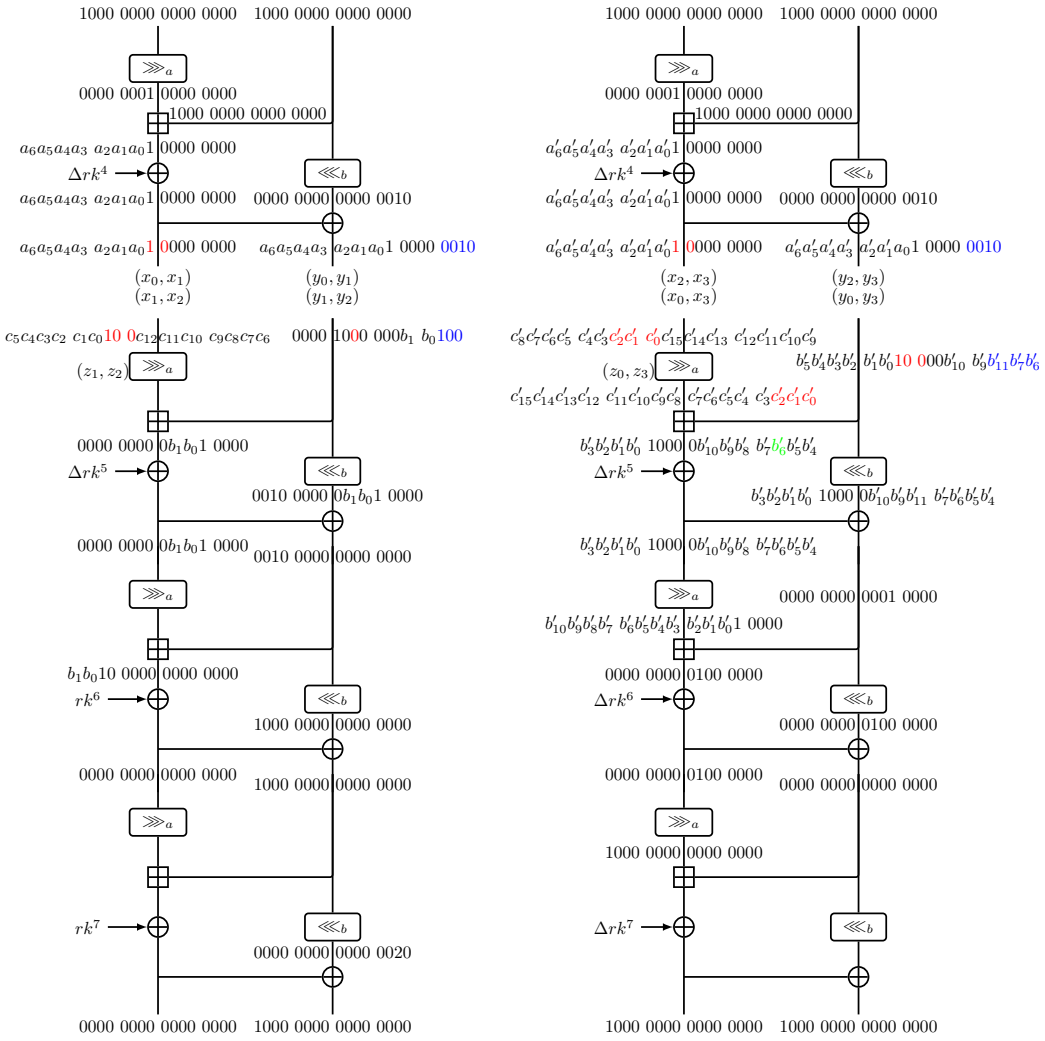


Fig. 17: Verification of the core of 8-round `RK-IB distinguisher` for `SPECK-32/64`

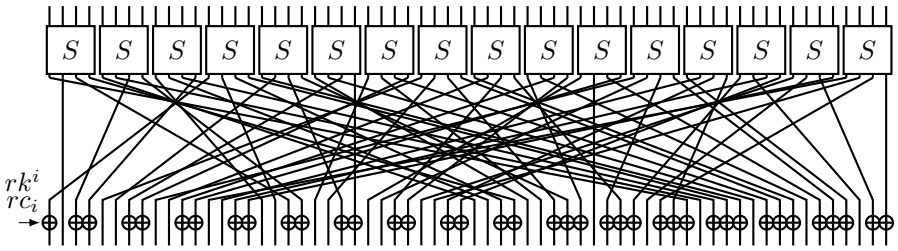


Fig. 18: One round of block cipher GIFT-64

Verification. (Proof by Contradiction) Assume that the input differences of the distinguisher can propagate to the output differences. We propagate the input differences for 5 rounds in the forward direction, and the output differences for 3 rounds in the backward direction. The differential propagation of the last 4 rounds is shown in Figure 17, where the $a_i, a'_i, b_i, b'_i, c_i, c'_i$ are either 0 or 1, and $b'_{11} = b'_8 \oplus 1$.

On 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, given $x_{0,8} \oplus x_{1,8} = 1$ and $x_{2,8} \oplus x_{3,8} = 1$, $x_{1,8} \oplus x_{2,8} = 0$, we have $x_{0,8} \oplus x_{3,8} = c'_1 = 0$. Similarly, we have $c'_0 = 0, b'_{11} = 1, b'_7 = b'_6 = 0$.

On the other hand, for the second modular addition on the right side of Fig. 17, the least three input bits of this operation are $(000, 100)$, so we have $b'_6 = 1$. This leads to a contradiction. □

F.2 GIFT-64

GIFT [38] has solidly established itself as a significant constituent in the field of lightweight cryptography. With its design focused on efficient resource usage, it stands apart from various 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 are of paramount significance. GIFT comes in two versions: GIFT-64 and GIFT-128. Both versions are SPN ciphers that utilize bit permutation operations.

Specification. GIFT-64 is a 64-bit block cipher with a 128-bit master key. One round of GIFT-64 is shown in Fig. 18; its round function involves the parallel application of a 4-bit Sbox to the state (SubCells), a bit permutation (PermBits), and an XOR operation with the round key (AddRoundKey). In particular, for the 64-bit state s_j $(0 \le j \le 63)$ in the i-th round, 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 $b_{4j+1} \leftarrow b_{4j+1} \oplus u_j$, $b_{4j} \leftarrow b_{4j} \oplus v_j$ $(0 \le j \le 15)$.

The 128-bit master key k is split into eight 16-bit words as $k = k_7 \| k_6 \| \cdots \| k_1 \| k_0$. The key scheduling procedure follows the rule below:

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

where $\ggg_i$ stands for an i-bit right rotation operation performed within each 16-bit word. Subsequently, the 32-bit round key $rk^i = u \| v$ is directly obtained by setting $u \leftarrow k_1$ and $v \leftarrow k_0$.

F.3 PRINTcipher48

`PRINTcipher` [39] 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.

Specification. `PRINTcipher48` is one variant of `PRINTcipher`. It has a block size of 48 bits, a key size of 80 bits, and an Sbox size of 3 bits. As shown in Fig. 19, it is an SPN cipher that adopts the 3-bit key-dependent bit permutation P. The combination of the 3-bit key-dependent bit permutation and the 3-bit Sbox is collectively defined as the 3-bit key-dependent Sbox.

The key schedule of `PRINTcipher48` is rather simple, which uses the same 48-bit key for all rounds. The 64-bit key segment is XORed with the internal state, while the 32-bit segment governs the key-dependent bit permutation, and each such permutation is controlled by two key bits. The keys dedicated to regulating these permutations are termed control keys.

The round function of `PRINTcipher48` includes round key XOR, bit permutation, round constant XOR, key-dependent bit permutation, and the parallel application of 3-bit Sboxes to the internal state.

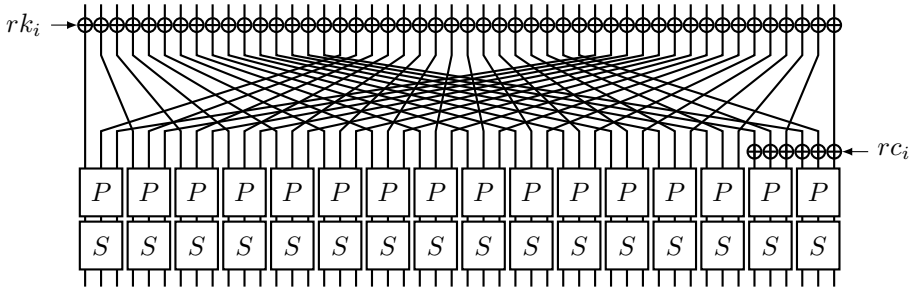


Fig. 19: One round of block cipher `PRINTcipher48`

Verification of Distinguisher 3. Let $\overline{S}[\overline{k}] = S \circ P : \mathbb{F}_2^3 \times \mathbb{F}_2^2 \to \mathbb{F}_2^3$ denote the key-dependent Sbox of `PRINTcipher48`, where P represents the 3-bit key-dependent bit permutation and S stands for the standard 3-bit Sbox. Let $(\mu, \mu', \rho, \rho') \in (\mathbb{F}_2^3)^4$ and $(\theta, \theta', \varphi, \varphi') \in (\mathbb{F}_2^3)^4$ be the input and output quartet differences, respectively. Construct the following three tables to obtain valid control keys of the key-dependent Sbox corresponding to given input and output differences:

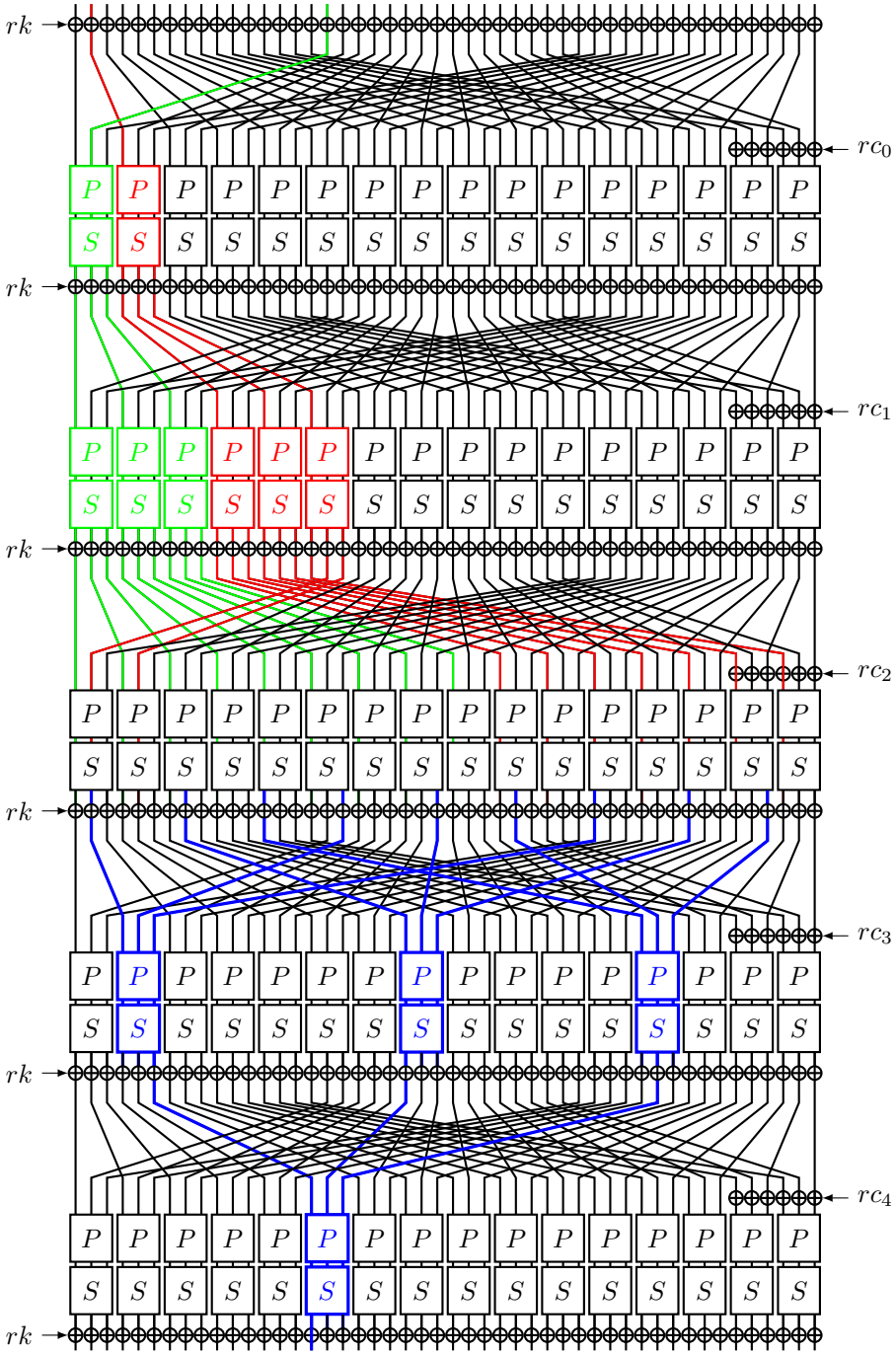


Fig. 20: All possible propagations of quartet-differential for `PRINTcipher48`

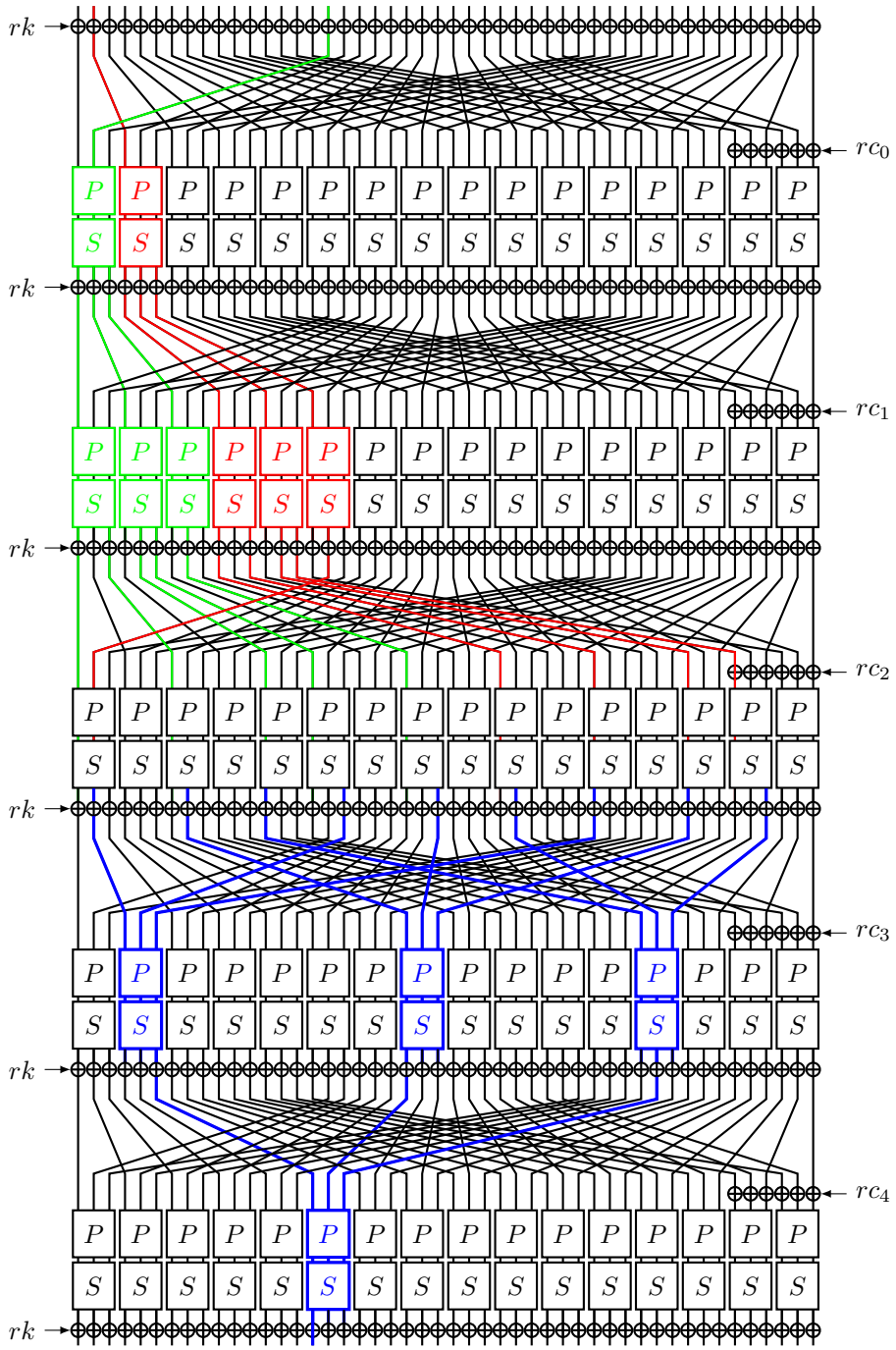


Fig. 21: Reduced possible propagations of quartet-differential for `PRINTcipher48`

$$\mathcal{T}_m(\mu,\mu',\varphi,\varphi') = \{\overline{k} | \exists x_1, x_2 \in \mathbb{F}_2^3, \text{ s.t. } \begin{cases} \overline{S}[\overline{k}](x_1) \oplus \overline{S}[\overline{k}](x_2) = \varphi, \\ \overline{S}[\overline{k}](x_1 \oplus \mu) \oplus \overline{S}[\overline{k}](x_2 \oplus \mu') = \varphi' \end{cases} \},$$

$$\mathcal{T}_u(\mu,\mu',\theta,\theta') = \{\overline{k} | \exists x_1, x_2 \in \mathbb{F}_2^3, \text{ s.t. } \begin{cases} \overline{S}[\overline{k}](x_1) \oplus \overline{S}[\overline{k}](x_1 \oplus \mu) = \theta, \\ \overline{S}[\overline{k}](x_2) \oplus \overline{S}[\overline{k}](x_2 \oplus \mu') = \theta' \end{cases} \},$$

$$\mathcal{T}_l(\rho,\rho',\varphi,\varphi') = \{\overline{k} | \exists x_0, x_1 \in \mathbb{F}_2^3, \text{ s.t. } \begin{cases} \overline{S}[\overline{k}](x_1) \oplus \overline{S}[\overline{k}](x_1 \oplus \rho) = \varphi, \\ \overline{S}[\overline{k}](x_0) \oplus \overline{S}[\overline{k}](x_0 \oplus \rho') = \varphi' \end{cases} \}.$$

Among them, $\mathcal{T}_u$ and $\mathcal{T}_l$ are essentially equivalent. We adopt distinct notations only for the sake of explicit illustration.

From the table $\mathcal{T}_m$, the following two properties hold.

Property 1.

$$\begin{aligned} \mathcal{T}_m(\mu,0,0,0) &= \emptyset \ (\mu \neq 0), \\ \mathcal{T}_m(0,\mu',0,0) &= \emptyset \ (\mu' \neq 0), \\ \mathcal{T}_m(\mu,\mu',0,0) &= \emptyset \ (\mu' \neq \mu). \end{aligned}$$

Property 2.

$$\begin{aligned} \mathcal{T}_m(\mu,0,\varphi,\varphi') &= \emptyset \ (\mu \in \{1,2,4\}, \varphi = \varphi' \in \{0,1,2,4\}), \\ \mathcal{T}_m(0,\mu',\varphi,\varphi') &= \emptyset \ (\mu' \in \{1,2,4\}, \varphi = \varphi' \in \{0,1,2,4\}), \\ \mathcal{T}_m(4,2,2,2) &= \emptyset, \mathcal{T}_m(4,2,0,0) = \emptyset. \end{aligned}$$

Using the above two properties, the verification of the distinguisher is presented as follows.

Verification. (Proof by Contradiction) Assume that the input differences of the distinguisher can propagate to the output differences. Let $(\sigma_0^r, \sigma_1^r, \sigma_2^r, \sigma_3^r)$ and $(\phi_0^r, \phi_1^r, \phi_2^r, \phi_3^r)$ denote the input and output quartet-differences of the key-dependent Sbox layer at Round r. Then ϕ_ℓ^r is determined by σ_ℓ^{r+1} $(0 \leq \ell \leq 3)$ according to the bit permutation. Let rk denote the 48-bit key XORed with the state, and ck denote the 32-bit control key. Let $\sigma_{\ell,j}^r$ and $\phi_{\ell,j}^r$ be the j-th part (partitioned by Sbox) of σ_ℓ^r and ϕ_ℓ^r, where $0 \leq j \leq 15$ and $0 \leq \ell \leq 3$. Let rk_j and ck_j $(0 \leq j \leq 15)$ be the corresponding keys.

As shown in Fig. 20, we propagate the two input differences forward for 3 rounds and the two output differences backward for 2 rounds. Since $\phi_{1,j}^2 = \phi_{3,j}^2 = 0$ $(j \in \{0,3,5,7,9,12,14\})$, according to Property 1, we obtain $\sigma_{0,j}^2 = 0$ $(j \in \{7,9,12,14\})$ and $\sigma_{2,j}^2 = 0$ $(j \in \{0,3,5,14\})$. Therefore, the propagation trails are simplified as illustrated in Fig. 21.

According to Property 2, the propagation in Table 7 holds. From Fig. 21, there are $(2^5-1)^2$ feasible pairs of (σ_0^2, σ_2^2) for the input differences. Using

Table 7: All possible values of (ϕ_1^2, ϕ_3^2) according to (σ_0^2, σ_2^2)

$(\sigma_{0,j}^2, \sigma_{2,j}^2)$	$(\phi_{1,j}^2, \phi_{3,j}^2)$	j
$\sigma_{0,j}^2 \neq 0$ or $\sigma_{2,j}^2 \neq 0$	$\{(1,0),(0,1)\}$	1, 8, 10
$\sigma_{0,j}^2 = 0$ and $\sigma_{2,j}^2 = 0$	$\{(0,0),(1,1)\}$	
$\sigma_{0,j}^2 \neq 0$ or $\sigma_{2,j}^2 \neq 0$	$\{(2,0),(0,2)\}$	6, 14
$\sigma_{0,j}^2 = 0$ and $\sigma_{2,j}^2 = 0$	$\{(0,0),(2,2)\}$	
$\sigma_{0,j}^2 \neq 0$ or $\sigma_{2,j}^2 \neq 0$	$\{(4,0),(0,4)\}$	2, 4, 11
$\sigma_{0,j}^2 = 0$ and $\sigma_{2,j}^2 = 0$	$\{(0,0),(4,4)\}$	
$\sigma_{0,j}^2 = 4$ or $\sigma_{2,j}^2 = 0$	$\{(2,0),(0,2)\}$	
$\sigma_{0,j}^2 = 0$ and $\sigma_{2,j}^2 = 2$		15
$\sigma_{0,j}^2 = 4$ and $\sigma_{2,j}^2 = 2$	$\{(2,0),(0,2),(2,2)\}$	
$\sigma_{0,j}^2 = 0$ and $\sigma_{2,j}^2 = 0$	$\{(0,0),(2,2)\}$	

Table 7, we derive all corresponding (ϕ_1^2, ϕ_3^2) values, and thus determine all propagable values from (α, α'). Since $\phi_{1,10}^5 = \phi_{3,10}^5 \neq 0$, we verify via DDT lookup and ck_j enumeration that there exists some $j \in \{1,2,4,6,8,10,11,13,15\}$ such that $\phi_{1,j}^2 = \phi_{3,j}^2 \neq 0$. Combined with the above conditions, we obtain 557568 values of $(\sigma_0^2, \sigma_2^2, \phi_1^2, \phi_3^2)$. If (α, α') propagates to (β, β'), there exists a valid pair satisfying

$$(\alpha, *, \alpha', *) \xrightarrow{\text{DDT}} (\sigma_0^2, *, \sigma_2^2, *) \xrightarrow{\text{GEBCT}} (*, \phi_1^2, *, \phi_3^2) \xrightarrow{\text{DDT}} (*, \beta, *, \beta'),$$

where $\xrightarrow{\text{DDT}}$ denotes DDT-compliant propagation and $*$ denotes an arbitrary value. Let Λ_j^r be the set of possible control keys ck_j at Round r. For each valid pair, we perform the steps below.

- For $j = 8, 10, 13, 15$, we derive Λ_j^2 from the input difference pair $(\sigma_{0,j}^2, \sigma_{2,j}^2)$ and output difference pair $(\phi_{1,j}^2, \phi_{3,j}^2)$ according to $\mathcal{T}_m$.
- Based on the table $\mathcal{T}_u$, we derive Λ_j^1 for $j = 10, 13, 14, 15$, as well as Λ_{14}^0 and Λ_{15}^0, from the difference propagation from (α, α') to (σ_0^2, σ_2^2).
- Based on the table $\mathcal{T}_l$, we derive Λ_3^3, Λ_8^3, Λ_{14}^3, and Λ_{10}^4 from the differential propagation from (ϕ_1^2, ϕ_3^2) to (β, β).
- Let $\Lambda_8 = \Lambda_8^2 \cap \Lambda_8^4$, $\Lambda_{10} = \Lambda_{10}^1 \cap \Lambda_{10}^2 \cap \Lambda_{10}^4$, $\Lambda_{13} = \Lambda_{13}^1 \cap \Lambda_{13}^2$, $\Lambda_{14} = \Lambda_{14}^0 \cap \Lambda_{14}^1 \cap \Lambda_{14}^3$, $\Lambda_{15} = \Lambda_{15}^0 \cap \Lambda_{15}^1 \cap \Lambda_{15}^2$. Check whether there exists $j \in \{8, 10, 13, 14, 15\}$ such that $\Lambda_j = \emptyset$.

By automating the above process, we observe that for all valid pairs of (σ_0^2, σ_2^2) and (ϕ_1^2, ϕ_3^2), there always exists some $j \in \{8, 10, 13, 14, 15\}$ such that $\Lambda_j = \emptyset$. Therefore, all possible trails are invalid, which proves the correctness of the distinguisher. □

F.4 AES-128

AES [40] is one of the most renowned block ciphers worldwide. Its design philosophy has had a profound impact on block ciphers. AES-128 is a 128-bit block cipher that supports a 128-bit key, and its Sbox size is 8 bits. It is an SPN cipher that employs the MDS matrix to achieve excellent diffusion. The internal state is regarded as a square array of bytes as follows, where $s_i \in \mathbb{F}_2^8$ for $0 \leq i \leq 15$.

$$S = \begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix}. \tag{8}$$

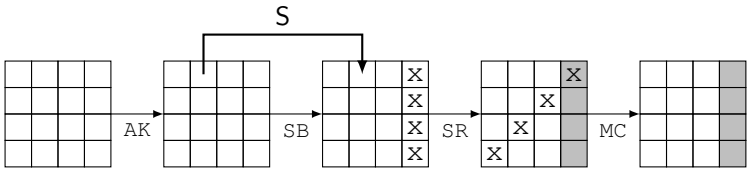


Fig. 22: One round of block cipher AES

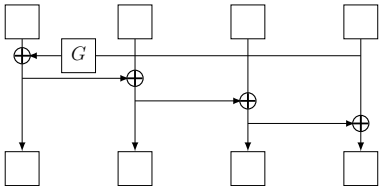


Fig. 23: The key schedule of AES-128

Specification. One encryption round of AES is depicted in Fig. 22, and it consists of the following four operations:

- AddRoundKey (AK): The round key derived from the key schedule is XORed with the internal state.
- SubBytes (SB): The 8-bit Sbox is applied to each byte of the cipher's internal state in parallel.
- ShiftRows (SR): The i-th row, where $0 \leq i \leq 3$, of the internal state is rotated right by i bytes.
- MixColumns (MC): Each column of the internal state is multiplied by the MDS matrix.

The key schedule of AES-128 is shown in Fig. 23. The function G is a 32-bit to 32-bit function that consists of:

- Rotating the input right by 1 byte.
- Processing all four bytes of this rotated input through the AES Sbox.
- Adding a fixed round constant to the output of the first Sbox.

F.5 SKINNY-64/192

SKINNY [17] is a family of tweakable block ciphers constructed under the TWEAKEY framework. The SKINNY family contains 6 different variants denoted as SKINNY-$n/(NT{\cdot}n)$. Here, $n \in \{64, 128\}$ denotes the block size, and $NT \in \{1, 2, 3\}$ stands for the number of n-bit tweakey states. The internal state and tweakey state are arranged in the form of a 4×4 matrix:

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

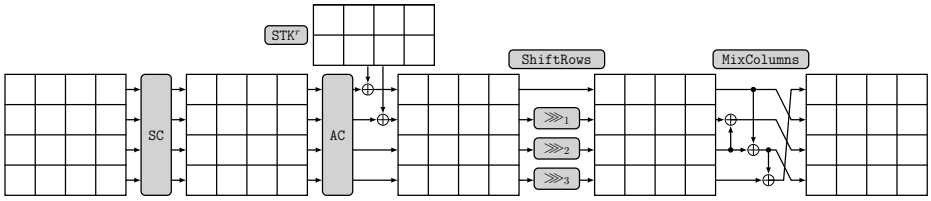


Fig. 24: One round of block cipher SKINNY

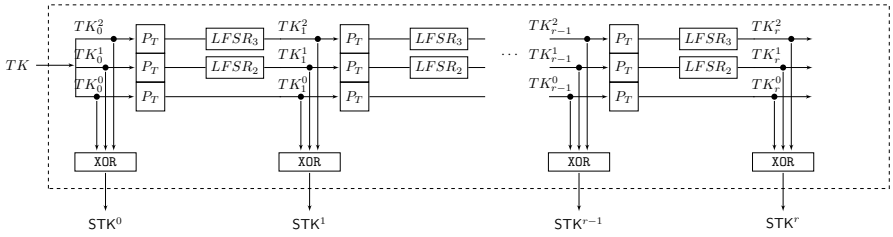


Fig. 25: Tweakey schedule of SKINNY

Specification. One round of SKINNY-64/192 is shown in Fig. 24. It uses three tweaks TK^0, TK^1 and TK^2 to derive the tweakey. One round of SKINNY consists of the following five operations:

– SubCell (SC): The 4bit Sbox is applied to each nibble of the cipher's internal state in parallel.
– AddConstant (AC): The constant is XORed with the state.
– AddRoundTweakey (ART): The first and second rows of the subtweakey are XORed with the corresponding cells in the internal state.
– ShiftRow (SR): The i-th row ($0 \leq i \leq 3$) of the internal state is rotated by i nibbles from left to right.
– MixColumn (MC): Each column of the internal state is multiplied by the matrix

$$M = \begin{pmatrix} 1\,0\,1\,1 \\ 1\,0\,0\,0 \\ 0\,1\,1\,0 \\ 1\,0\,1\,0 \end{pmatrix}.$$

For the tweakey schedule, there are three tweakey states: TK^0, TK^1, and TK^2. In each round, these three tweakey states are updated via a cell permutation PT (where $PT = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7]$). Additionally, every cell in the first and second rows of TK^1 and TK^2 is updated using LFSR_2 and LFSR_3, respectively, where these LFSRs are defined below:

$$\mathrm{LFSR}_2 : (x_3\|x_2\|x_1\|x_0) \rightarrow (x_2\|x_1\|x_0\|x_3 \oplus x_2),$$
$$\mathrm{LFSR}_3 : (x_3\|x_2\|x_1\|x_0) \rightarrow (x_0 \oplus x_3\|x_3\|x_2\|x_1),$$

The tweak schedule updated by LFSRs has the following property, which enables us to directly construct a 6-round probability-1 differential. This property is similar to the subtweakey difference cancellation of `Deoxys-BC` [15,11].

Property 3. *For `SKINNY-64/192`, if the tweakey difference ΔTK is active, there are at most 5 subtweaks (each ΔSTK^r has 8 nibbles) with inactive differences, which can be consecutive or nonconsecutive.*

Proof. Since LFSRs operate independently on each nibble, we can analyze each nibble independently. Assume the active nibble is u. Since the LFSRs are linear, the system

$$\begin{cases} a_0 \oplus a_1 \oplus a_2 = u, \\ a_0 \oplus \mathrm{LFSR}_2^1(a_1) \oplus \mathrm{LFSR}_3^1(a_2) = 0, \\ a_0 \oplus \mathrm{LFSR}_2^2(a_1) \oplus \mathrm{LFSR}_3^2(a_2) = 0 \end{cases}$$

has exactly one solution for (a_0, a_1, a_2), where LFSR_j^i denotes applying LFSR_j i times ($i > 0$, $j = 2, 3$). Furthermore, the system

$$\begin{cases} a_0 \oplus a_1 \oplus a_2 = u, \\ a_0 \oplus \mathrm{LFSR}_2^1(a_1) \oplus \mathrm{LFSR}_3^1(a_2) = 0, \\ a_0 \oplus \mathrm{LFSR}_2^2(a_1) \oplus \mathrm{LFSR}_3^2(a_2) = 0, \\ a_0 \oplus \mathrm{LFSR}_2^3(a_1) \oplus \mathrm{LFSR}_3^3(a_2) = 0 \end{cases}$$

has no solution. Therefore, there are at most 5 subtweakeys with inactive differences, and these 5 subtweakeys can be consecutive. □

Configurations and Results. Since SKINNY-64/192 adopts a linear tweakey schedule, we search for the RK-IB distinguishers by applying the GEBCT-based method in four-related-keys setting with tweakey differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\Delta TK, \Delta TK, \Delta TK', \Delta TK')$. The search proceeds in two phases:

1. Let ΔTK satisfy $\Delta STK^0 \neq 0$, $\Delta STK^j = 0$ for $1 \leq j \leq 5$, and $\Delta STK^6 \neq 0$, and let $\Delta TK'$ satisfy $\Delta STK'^r \neq 0$, $\Delta STK'^{r-j} = 0$ for $1 \leq j \leq 5$, and $\Delta STK'^{r-6} \neq 0$. Then ΔTK and $\Delta TK'$ are determined by ΔSTK^0 and $\Delta STK'^r$, respectively. Let $\widehat{x}$ denote extending a 32-bit value x to 64 bits by appending zeros to the end. Set $(\alpha, \alpha', \beta, \beta') = (\widehat{\Delta STK^0}, \widehat{\Delta STK^0}, MC \circ SR(\widehat{\Delta STK'^r}), MC \circ SR(\widehat{\Delta STK'^r}))$. This construction allows 6-round probability-1 related-key differentials at both ends.
2. Search for ΔSTK^0 and $\Delta STK'^r$ such that $(0, 0, 0, 0)$ constitutes an $(r-12)$-round core RK-IB distinguisher.

Combine them to yield an r-round RK-IB distinguisher $(\alpha, \alpha', \beta, \beta')$. It results in 3 19-round RK-IB distinguishers within 1.05 hours. Here, we restrict ΔSTK^0 and $\Delta STK'^r$ to 1-nibble active, yielding a search space of $(8 \times 15)^2 = 14400$. All results are automatically verified via the state-based method. We also confirm it is the upper bound on RK-IB distinguisher coverage within the same search space in 8.35 hours.

Comparison.

- vs. prior RK-IB distinguishers: In [11], Zhang et al. proposed 18-round RK-IB distinguishers for SKINNY-64/192; our RK-IB distinguishers cover one more round than the previous best results. This improvement arises because Zhang et al. adopted DBCT to detect contradictions derived from the dependency relations of two-round input-output differences. In contrast, our GEBCT-based method takes cross-round difference dependencies into account, which facilitates constructing contradictions with more rounds.
- vs. state-of-the-art RK-ID distinguishers: The maximum number of rounds for RK-ID distinguishers on SKINNY-64/192 is 17 [21]. Thus, the RK-IB distinguishers we identified cover two more rounds than the known RK-ID distinguishers, which is consistent with our theoretical analysis in Sec. 3.6: RK-ID distinguishers can only employ one 6-round probability-1 differential, whereas RK-IB distinguishers can utilize two.

Example. One of the 19-round RK-IB distinguishers is shown as follows. It is obtained by the GEBCT-based method and validated using the state-based method.

Distinguisher 6. *For SKINNY-64/192, $(\alpha, \alpha, \beta, \beta)$ is a 19-round RK-IB distinguisher without the initial SC operation under the tweakey differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\Delta TK, \Delta TK, \Delta TK', \Delta TK')$, where ΔTK is determined by the upper trail key differences from round 0 to round 5, denoted by $\eta = (\eta_0, \eta_1, \eta_2, \eta_3, \eta_4, \eta_5)$, and $\Delta TK'$ is determined by the lower trail key differences from round 13 to round 18, denoted by $\theta = (\theta_{13}, \theta_{14}, \theta_{15}, \theta_{16}, \theta_{17}, \theta_{18})$. The parameters are given*

as follows:

$$\begin{cases} \alpha = 0xc000000000000000, \\ \beta = 0x0000000000400000, \\ \eta_0 = 0xc0000000, \eta_1 = \eta_2 = \eta_3 = \eta_4 = \eta_5 = 0x00000000, \\ \theta_{18} = 0x00000400, \theta_{13} = \theta_{14} = \theta_{15} = \theta_{16} = \theta_{17} = 0x00000000. \end{cases}$$

F.6 SKINNYee

SKINNYee [14], proposed by Naito et al. at CRYPTO 2022, is an extension of the tweakable block cipher SKINNY [17], and is specifically designed to be compatible with their newly proposed operation mode HOMA. It has a block size of 64 bits, a key size of 128 bits and a tweak size of 256 bits. The arrangement of its internal state and tweakey state is consistent with that of SKINNY.

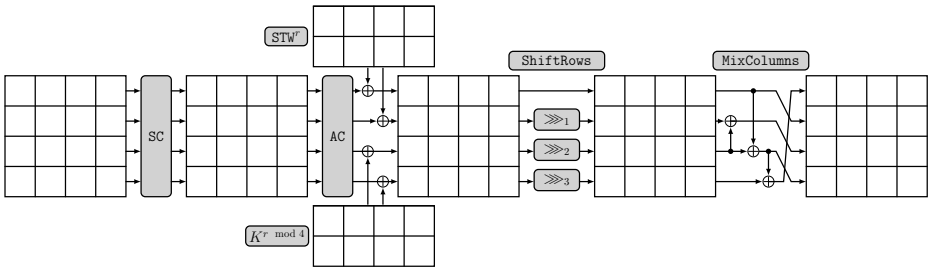


Fig. 26: One round of block cipher SKINNYee

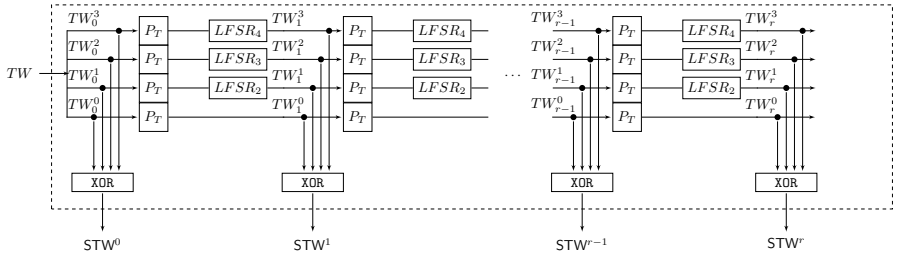


Fig. 27: Tweakey schedule of SKINNYee

Specification. One round of SKINNYee is shown in Fig. 26, and it consists of the following six operations:

– SubCell(SC): Applying the 4-bit Sbox to each nibble in parallel to the cipher's internal state.
– AddConstant(AC): The constant is XORed with the state.
– AddRoundTweak(ART): XOR the first and second rows of the subtweak with the corresponding nibbles in the internal state.
– AddRoundKey(ARK): The 128-bit master key MK is divided into four 32-bit round keys K^0, K^1, K^2 and K^3, which are XORed with the third and fourth rows of the state.
– ShiftRows(SR): The i-th row ($0 \leq i \leq 3$) of the internal state is rotated by i nibbles from left to right.
– MixColumn(MC): Each column of the internal state is multiplied by the matrix

$$M = \begin{pmatrix} 1\,0\,1\,1 \\ 1\,0\,0\,0 \\ 0\,1\,1\,0 \\ 1\,0\,1\,0 \end{pmatrix}.$$

For the tweak schedule, there are four tweak states: TW^0, TW^1, TW^2, and TW^3. In each round, these four tweak states are updated via a nibble permutation PT (where $PT = [9, 15, 8, 13, 10, 14, 12, 11, 0, 1, 2, 3, 4, 5, 6, 7]$). Additionally, all nibbles in the internal states of TW^1, TW^2, and TW^3 are updated using LFSR_2, LFSR_3, and LFSR_4, respectively, where these LFSRs are defined below:

$$\begin{aligned} &\mathrm{LFSR}_2 : (x_3\|x_2\|x_1\|x_0) \to (x_2\|x_1\|x_0\|x_3 \oplus x_2), \\ &\mathrm{LFSR}_3 : (x_3\|x_2\|x_1\|x_0) \to (x_0 \oplus x_3\|x_3\|x_2\|x_1), \\ &\mathrm{LFSR}_4 : (x_3\|x_2\|x_1\|x_0) \to (x_1\|x_0\|x_3 \oplus x_2\|x_2 \oplus x_1). \end{aligned}$$

Similar to `SKINNY-64/192`, the tweak schedule of `SKINNYee` updated by LFSRs possesses the following property, which allows us to directly construct an 8-round probability-1 differential.

Property 4. *For `SKINNYee`, if the tweak difference ΔTW is active, there will be at most 7 subtweaks (each ΔSTW_r is of 8 nibbles) with inactive differences, which can be consecutive or nonconsecutive.*

Proof. Since LFSRs operate independently on each nibble, we can analyze each nibble independently. Assume the active nibble is u. Since the LFSRs are linear, the equations

$$\begin{cases} a_0 \oplus a_1 \oplus a_2 \oplus a_3 = u, \\ a_0 \oplus \mathrm{LFSR}_2^2(a_1) \oplus \mathrm{LFSR}_3^2(a_2) \oplus \mathrm{LFSR}_3^2(a_3) = 0, \\ a_0 \oplus \mathrm{LFSR}_2^4(a_1) \oplus \mathrm{LFSR}_3^4(a_2) \oplus \mathrm{LFSR}_3^4(a_3) = 0, \\ a_0 \oplus \mathrm{LFSR}_3^6(a_1) \oplus \mathrm{LFSR}_3^6(a_2) \oplus \mathrm{LFSR}_3^6(a_3) = 0 \end{cases}$$

have exactly one solution for (a_0, a_1, a_2, a_3), where LFSR_j^i denotes the operation of applying LFSR_j i times ($i > 0$, $j = 2, 3, 4$). Furthermore, the equations

$$\begin{cases} a_0 \oplus a_1 \oplus a_2 \oplus a_3 = u, \\ a_0 \oplus \mathrm{LFSR}_2^2(a_1) \oplus \mathrm{LFSR}_3^2(a_2) \oplus \mathrm{LFSR}_3^2(a_3) = 0, \\ a_0 \oplus \mathrm{LFSR}_2^4(a_1) \oplus \mathrm{LFSR}_3^4(a_2) \oplus \mathrm{LFSR}_3^4(a_3) = 0, \\ a_0 \oplus \mathrm{LFSR}_3^6(a_1) \oplus \mathrm{LFSR}_3^6(a_2) \oplus \mathrm{LFSR}_3^6(a_3) = 0, \\ a_0 \oplus \mathrm{LFSR}_3^8(a_1) \oplus \mathrm{LFSR}_3^8(a_2) \oplus \mathrm{LFSR}_3^8(a_3) = 0 \end{cases}$$

have no solutions. Therefore, there can be at most 7 subtweakeys with inactive differences, and these 7 subtweakeys can be consecutive. □

Configurations and Results. Since `SKINNYee` adopts a linear tweak schedule, we search for the `RK-IB` distinguishers by applying the GEBCT-based method in `four-related-keys` setting with tweak differences $(\kappa_0, \kappa_1, \kappa_2, \kappa_3) = (\Delta TW, \Delta TW, \Delta TW', \Delta TW')$. The search proceeds in two phases:

1. Let ΔTW satisfy $\Delta STW^0 \neq 0$, $\Delta STW^j = 0$ for $1 \leq j \leq 7$, and $\Delta STW^8 \neq 0$, and let $\Delta TW'$ satisfy $\Delta STW'^r \neq 0$, $\Delta STW'^{r-j} = 0$ for $1 \leq j \leq 7$, and $\Delta STW'^{r-8} \neq 0$. Then ΔTW and $\Delta TW'$ are determined by ΔSTW^0 and $\Delta STW'^r$, respectively. Let $\widehat{x}$ denote extending a 32-bit value x to 64 bits by appending zeros to the end. Set $(\alpha, \alpha', \beta, \beta') = (\Delta\widehat{STW^0}, \Delta\widehat{STW^0}, MC \circ SR(\Delta\widehat{STW'^r}), MC \circ SR(\Delta\widehat{STW'^r}))$. This construction allows 8-round probability-1 related-key differentials at both ends.
2. Search for $\Delta STW^0, \Delta STW'^r$ such that $(0, 0, 0, 0)$ constitutes an $(r - 16)$-round core `RK-IB` distinguisher.

Combine them to yield an r-round `RK-IB` distinguisher $(\alpha, \alpha', \beta, \beta')$. It results in 5 23-round `RK-IB` distinguishers within 1.54 hours. Here, we restrict ΔSTW^0 and $\Delta STW'^r$ to 1-nibble active, yielding a search space of $(8 \times 15)^2 = 14400$. All results are automatically verified via the state-based method. We also confirm it is the upper bound on `RK-IB` distinguisher coverage within the same search space in 15.28 hours.

Comparison.

- vs. prior `RK-IB` distinguishers: In [10], Bonnetain et al. proposed 22-round `RK-IB` distinguishers for `SKINNYee`; our `RK-IB` distinguishers cover one more round than the previous best ones. This improvement arises because Bonnetain et al. used BCT to detect contradictions derived from one-round input-output difference dependencies. In contrast, our GEBCT-based method takes cross-round difference dependencies into account, which facilitates constructing contradictions with more rounds.
- vs. state-of-the-art `RK-ID` distinguishers: The maximum number of rounds for `RK-ID` distinguishers on `SKINNYee` is 19 [21]. Thus, the `RK-IB` distinguishers we identified cover four more rounds than the known `RK-ID` distinguishers, which is consistent with our theoretical analysis in Sec. 3.6. Specifically, `RK-ID` distinguishers can only employ one 8-round probability-1 differential, while `RK-IB` distinguishers are able to utilize two such differentials.

Example. One of the 23-round RK-IB distinguishers is shown as Distinguisher 5. It is obtained by the GEBCT-based method and validated using the state-based method. Since we conduct the IB attack based on this distinguisher, we place it across rounds 4 to 27. Notably, owing to the GEBCT-based method, the obtained distinguishers are independent of the starting round position.

F.7 CHAM

CHAM [23] is a family of block ciphers suitable for resource-constrained devices, such as Internet of Things (IoT) devices and embedded systems. Its design is optimized to impose low overhead on computation, memory and power consumption. Each member of this family is denoted as CHAM-n/m, where n stands for the block size and m for the key size. Table 8 summarizes all cipher instances and their corresponding parameters. Here, w denotes the bit length of each branch, and r denotes the total number of rounds. CHAM adopts a 4-branch generalized Feistel structure with modular addition as the core operation.

Table 8: 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

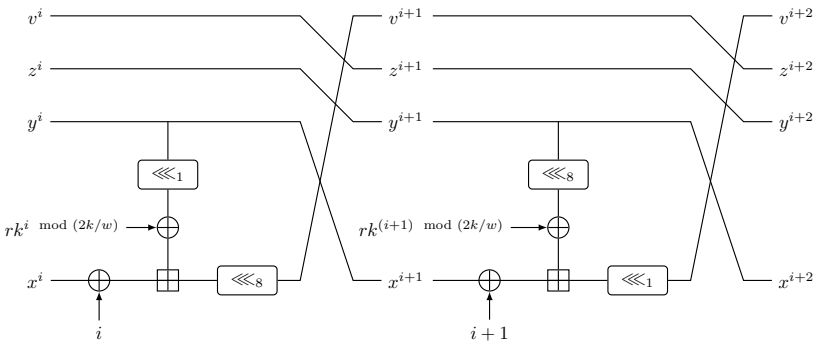


Fig. 28: Two consecutive rounds of block cipher CHAM beginning with the even i-th round

Specification. As shown in Fig. 28, CHAM-n/k encrypts four w-bit words $\left(x^0, y^0, z^0, v^0\right)$ to four w-bit words (x^r, y^r, z^r, v^r). To be more specific, in the i-th round $(0 \leq$

$i < r$):

$$\left(x^{i+1}, y^{i+1}, z^{i+1}, v^{i+1}\right) \leftarrow \left(y^i, z^i, v^i, \left(\left(x^i \oplus i\right) \boxplus \left(\left(y^i \lll a_i\right) \oplus rk^{i \bmod 2k/w}\right)\right) \lll b_i\right),$$

where $a_i = 1$ and $b_i = 8$ when $i \bmod 2 = 0$ and $a_i = 8$ and $b_i = 1$ 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:

$$\begin{aligned} rk^i &\longleftarrow K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 8), \\ rk^{(i+k/w)\oplus 1} &\longleftarrow K[i] \oplus (K[i] \lll 1) \oplus (K[i] \lll 11), \end{aligned}$$

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

Configurations and Results. CHAM's modular addition operations make it incompatible with all prior IB distinguisher search methods. As CHAM adopts a linear key schedule, we address this by treating modular addition as a large Sbox and applying our GEBCT-based method and state-based method in the four-related-keys setting setting (μ, μ, ν, ν). The search proceeds in two phases:

1. Find r_0-round related-key differentials $\alpha \xrightarrow{E^0,\mu} \gamma$ and r_1-round related-key differentials $\beta \xrightarrow{(E^1)^{-1},\nu} \delta$. This results in $r_0 = r_1 = 10$ for both CHAM-64/128 and CHAM-128/256.
2. Determine whether $(\gamma, \gamma, \delta, \delta)$ constitutes an $(r-20)$-round core RK-IB distinguisher.

Combine them to yield an r-round RK-IB distinguisher $(\alpha, \alpha', \beta, \beta')$. Finally, we get the following results.

- State-based method: For CHAM-64/128, we obtain 3 30-round RK-IB distinguishers within 0.15 hours. For CHAM-128/256, we obtain 4 28-round RK-IB distinguishers within 0.48 hours.
- GEBCT-based method: For CHAM-64/128, we obtain 8 28-round RK-IB distinguishers within 0.02 hours (50 seconds). For CHAM-128/256, we obtain 3 28-round RK-IB distinguishers within 0.02 hours (56 seconds).

This serves as evidence that the state-based method can yield distinguishers with more rounds or larger quantities compared with the GEBCT-based method.

Comparison.

- vs. prior RK-IB distinguishers: To date, no existing methods can search for RK-IB distinguishers on CHAM due to its modular addition operations.
- vs. state-of-the-art RK-ID distinguishers: The maximum number of rounds of RK-ID distinguishers for both CHAM-64/128 and CHAM-128/256 is 26 [37]. Thus, the RK-IB distinguishers we found exceed the optimized RK-ID distinguishers by four and two rounds for CHAM-64/128 and CHAM-128/256, respectively. These results are consistent with our theoretical analysis in Sec. 3.6: RK-ID distinguishers can only employ one probability-1 differential, while RK-IB distinguishers can utilize two such differentials.

Example. Representative RK-IB distinguishers for CHAM-64/128 and CHAM-128/256 are given below. The distinguisher of CHAM-64/128 is obtained by the state-based method, while that of CHAM-128/256 is derived from the GEBCT-based method and verified via the state-based method.

Distinguisher 7. *$(\alpha, \alpha, \beta, \beta)$ is an RK-IB distinguisher for 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 9.*

Table 9: The example of RK-IB distinguishers for CHAM

Block cipher	params	value
CHAM-64/128	r	30
	α	$0x0000000000000000$
	β	$0x0000000000000000$
	η	$0x00006020000000000000000000000000$
	θ	$0x6020c040000000000000000000000000$
CHAM-128/256	r	28
	α	$0x00000000000000000000000000000000$
	β	$0x00000000000000000000000000000000$
	η	$0x78081828000000000000000000000000$
		$0x00000000000000000000000000000000$
	θ	$0x0000000000000000781818 28f0103050$
		$0x00000000000000000000000000000000$

F.8 SIMON32

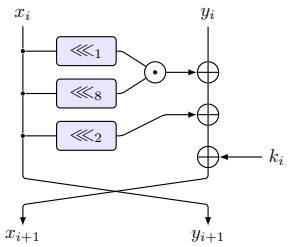


Fig. 29: Round function of SIMON

SIMON is a family of lightweight ciphers designed by the (NSA) [18] to provide high security and efficiency for use in constrained environments. SIMON32 is a

member of the SIMON family with a block size of 32 bits. One round of SIMON is illustrated in Fig. 29. The round function employs bitwise AND $\cdot$, bitwise XOR $\oplus$, and left circular shift $\lll$.

Configurations and Results. We search for 1-bit active SK-IB distinguishers using the state-based method, where only the left branch of the input differences and the right branch of the output differences are active, which enables the input difference and output difference to propagate for one round each with probability one. The search space size is $(16 \times 15)^2/4 = 14400$. We prove no 12-round SK-IB distinguishers exist within such search space in 4.39 hours; and obtain 64 11-round SK-IB distinguishers in 4.08 hours.

Comparison.

- vs. prior SK-IB distinguishers: Bonnetain et al. reported 7-round SK-IB distinguishers [18], which constitutes the first SK-IB distinguisher result on the AndRX cipher. In comparison, we find 11-round SK-IB distinguishers with our method, extending the longest known result by 4 rounds.

Example. Representative SK-IB distinguisher for SIMON32 is given below.

Distinguisher 8. $(\alpha, \alpha', \beta, \beta')$ *is an* ***SK-IB*** *distinguisher of* 11*-round* ***SIMON32****, where*

$$\begin{cases} \alpha = 0x00000001, \alpha' = 0x00000004, \\ \beta = 0x00010000, \beta' = 0x02000000. \end{cases}$$

F.9 PRESENT-80

PRESENT [44] 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 Sbox size of 4 bits. It is an SPN cipher that makes use of the operation of bit permutation.

Table 10: The Sbox 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

One round of PRESENT is depicted in Fig. 30, the round function of it involves an XOR with the round key, the application of a 4-bit Sbox (as shown in Table 10) 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

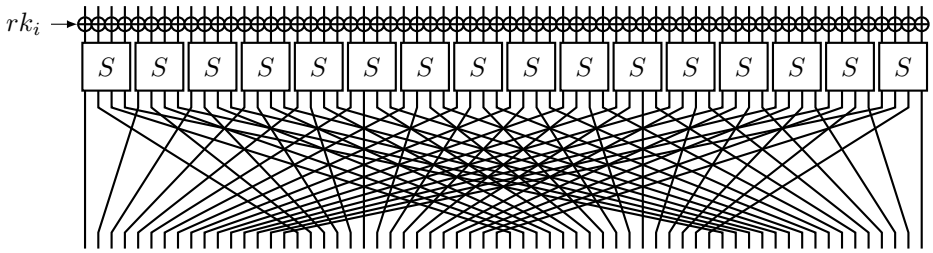


Fig. 30: One round of block cipher PRESENT

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:

$$[\mathrm{k}_{79}\mathrm{k}_{78}\ldots\mathrm{k}_1\mathrm{k}_0] = [\mathrm{k}_{18}\mathrm{k}_{17}\ldots\mathrm{k}_{20}\mathrm{k}_{19}]$$
$$[\mathrm{k}_{79}\mathrm{k}_{78}\mathrm{k}_{77}\mathrm{k}_{76}] = S\,[\mathrm{k}_{79}\mathrm{k}_{78}\mathrm{k}_{77}\mathrm{k}_{76}]$$
$$[\mathrm{k}_{19}\mathrm{k}_{18}\mathrm{k}_{17}\mathrm{k}_{16}\mathrm{k}_{15}] = [\mathrm{k}_{19}\mathrm{k}_{18}\mathrm{k}_{17}\mathrm{k}_{16}\mathrm{k}_{15}] \oplus \mathrm{round_counter}.$$

Configurations and Results. We search for 1-nibble active SK-IB distinguishers by applying the state-based method, with only the first Sbox of the input and output differences being active. The search space size is $(15\times14)^2/4 = 11025$. We prove no 7-round 1-nibble active SK-IB distinguishers exist within such search space in 0.77 hours; and obtain 58 6-round 1-nibble active SK-IB distinguishers in 0.23 hours.

Comparison.

- vs. prior SK-IB distinguishers: Due to the use of bit permutation in PRESENT, existing methods based on truncated difference search for IB distinguishers are not applicable. Our approach is the first to enable the search for IB distinguishers of block ciphers using bit permutation.
- vs. state-of-the-art SK-ID distinguishers: To our knowledge, the maximum round number of SK-ID distinguishers for PRESENT-80 is 6 [43], with no 7-round 1-nibble active SK-ID distinguisher existing [20]. Thus, the SK-IB distinguishers we found cover the same number of rounds as the optimized SK-ID distinguishers.

Example. Representative SK-IB distinguisher for PRESENT-80 is given below.

Distinguisher 9. $(\alpha, \alpha', \beta, \beta')$ *is an SK-IB distinguisher of* 6*-round PRESENT-80 without the last bit permutation, where*

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

F.10 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. 31, 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 Sboxes, these Sboxes can be either public or secret and are not necessarily permutations.

In our work, we employ public Sboxes for automatic search. Particularly, we search for RK-IB distinguishers in two of the most renowned versions, namely GOST-FB and GOST-PS. GOST-FB indicates GOST that uses eight different Sboxes 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 Sbox.

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

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 $K_7, \ldots, K_0$.

Configurations and Results. Since GOST-FB/PS uses a linear KS, we search for 1-nibble active SK-IB distinguishers by applying the state-based method. 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. 32, 33, 34 and 35, GOST-FB/PS 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 rounds 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 $((0x00000000, 0x80000000)$, $(0x80000000, 0x00000000)$, $(0x80000000, 0x00000000)$,

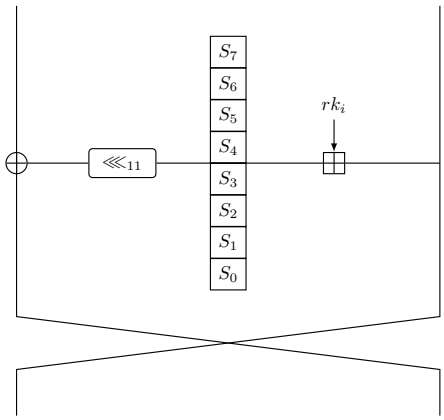


Fig. 31: One round of block cipher GOST

$(0x00000000, 0x80000000)$) is an 2-round RK-IB distinguishers. 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. We found two 2-round RK-IB distinguishers for both GOST-FB and GOST-PS within 5 minutes. Specifically, it requires $\kappa_{2,7} = 0x40000000$ or $\kappa_{2,7} = 0x20000000$. These two RK-IB distinguishers can be extended to full-round distinguishers for both GOST-FB and GOST-PS.

Comparison.

- vs. prior SK-IB distinguishers: To date, no existing methods have been able to search for RK-IB distinguishers on GOST-FB/PS due to its use of modular additions. Our results present the first full-round RK-IB distinguishers on GOST-FB/PS. As our distinguisher is full-round, we do not compare it with other distinguishers anymore.

Example. Representative SK-IB distinguisher for GOST-FB/PS is given below.

Distinguisher 10. *$(\alpha, \alpha', \beta, \beta')$ is an RK-IB distinguisher of full-round GOST-FB/PS under the key differences $\kappa_{i,j}(0 \le i \le 3, 0 \le j \le 7)$, where*

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

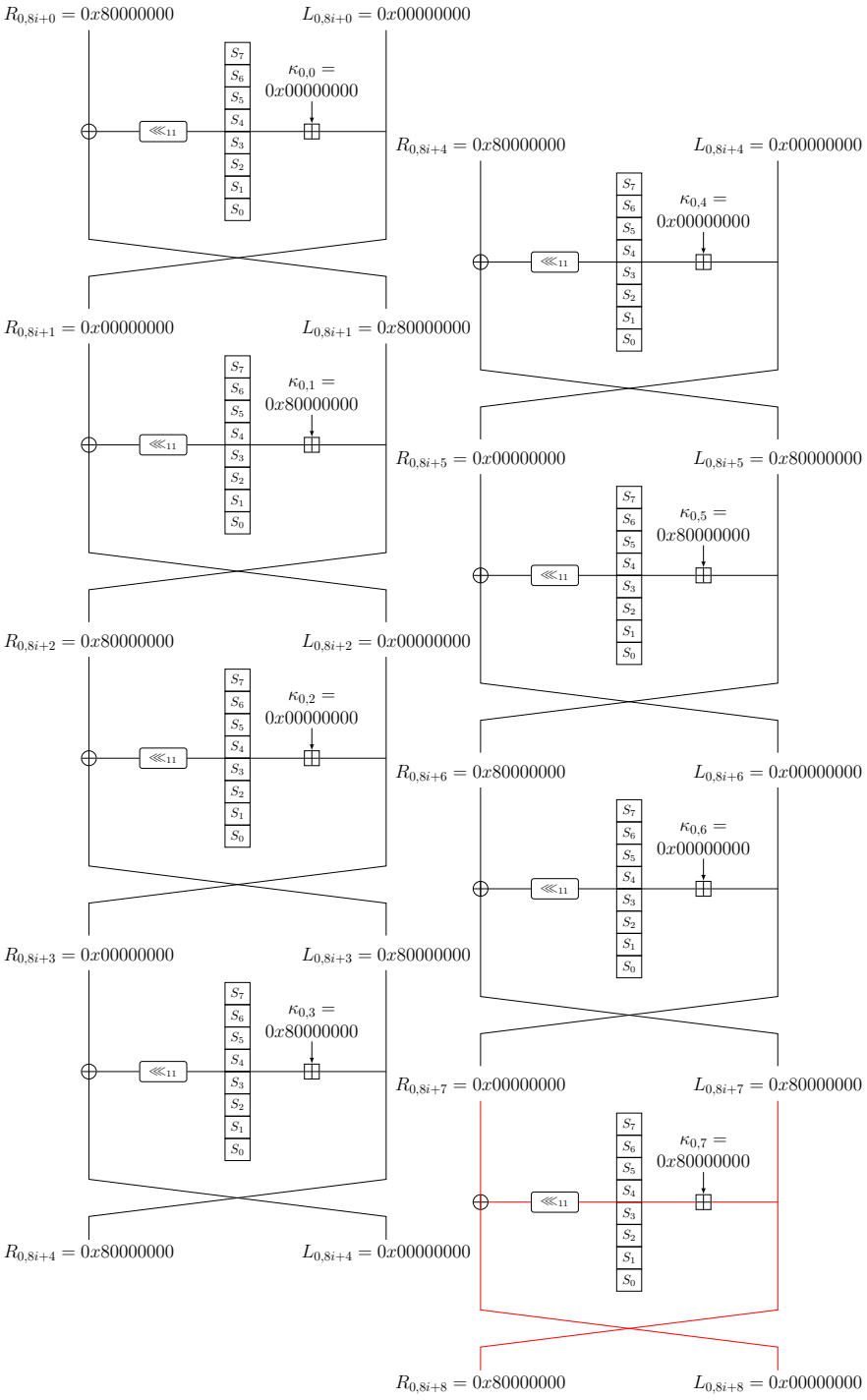


Fig. 32: The 24-round related-key differential of `GOST-FB/PS` by iterating above 3 times

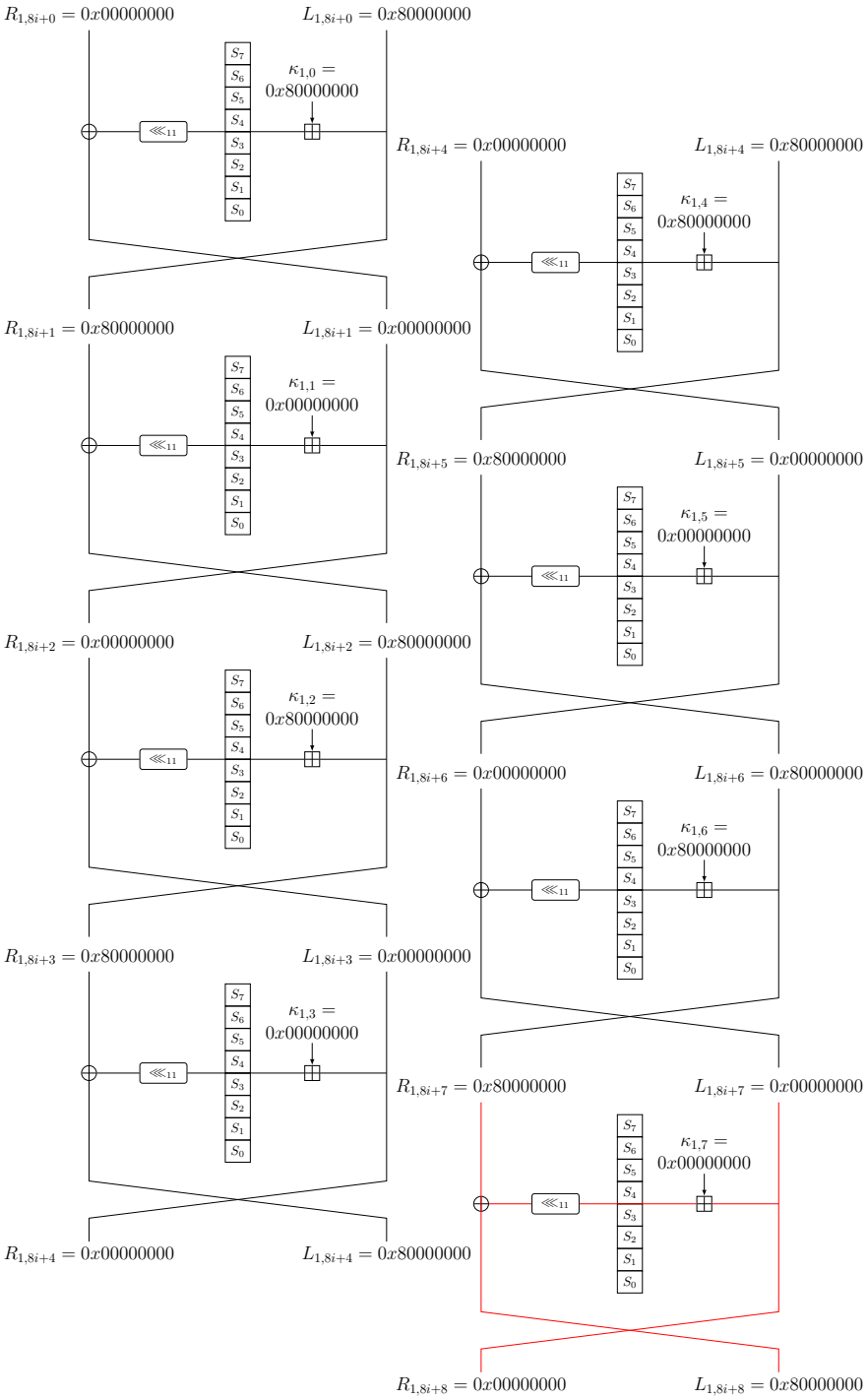


Fig. 33: The 24-round related-key differential of `GOST-FB/PS` by iterating above 3 times

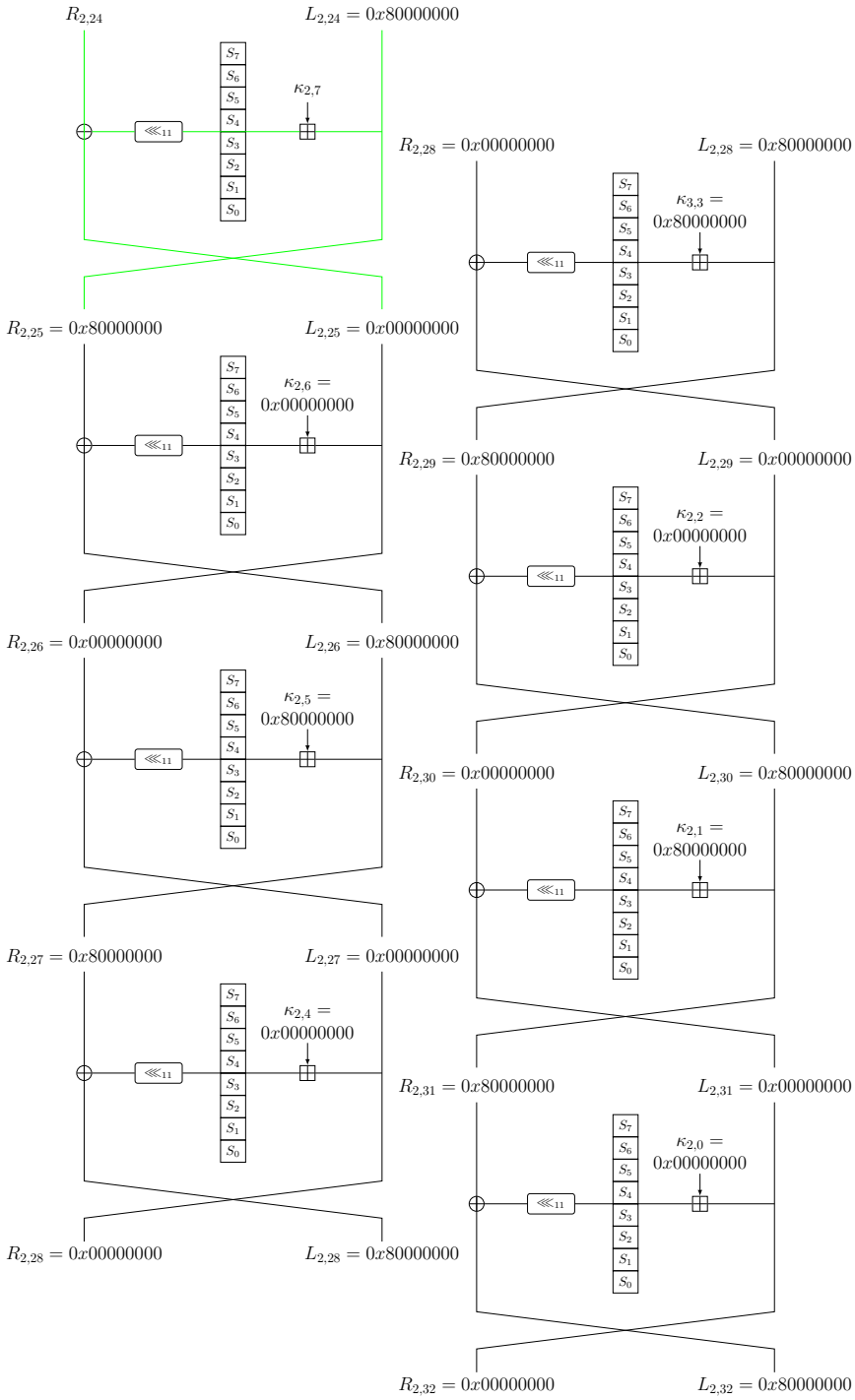


Fig. 34: The 7-round related-key differential of `GOST-FB/PS`

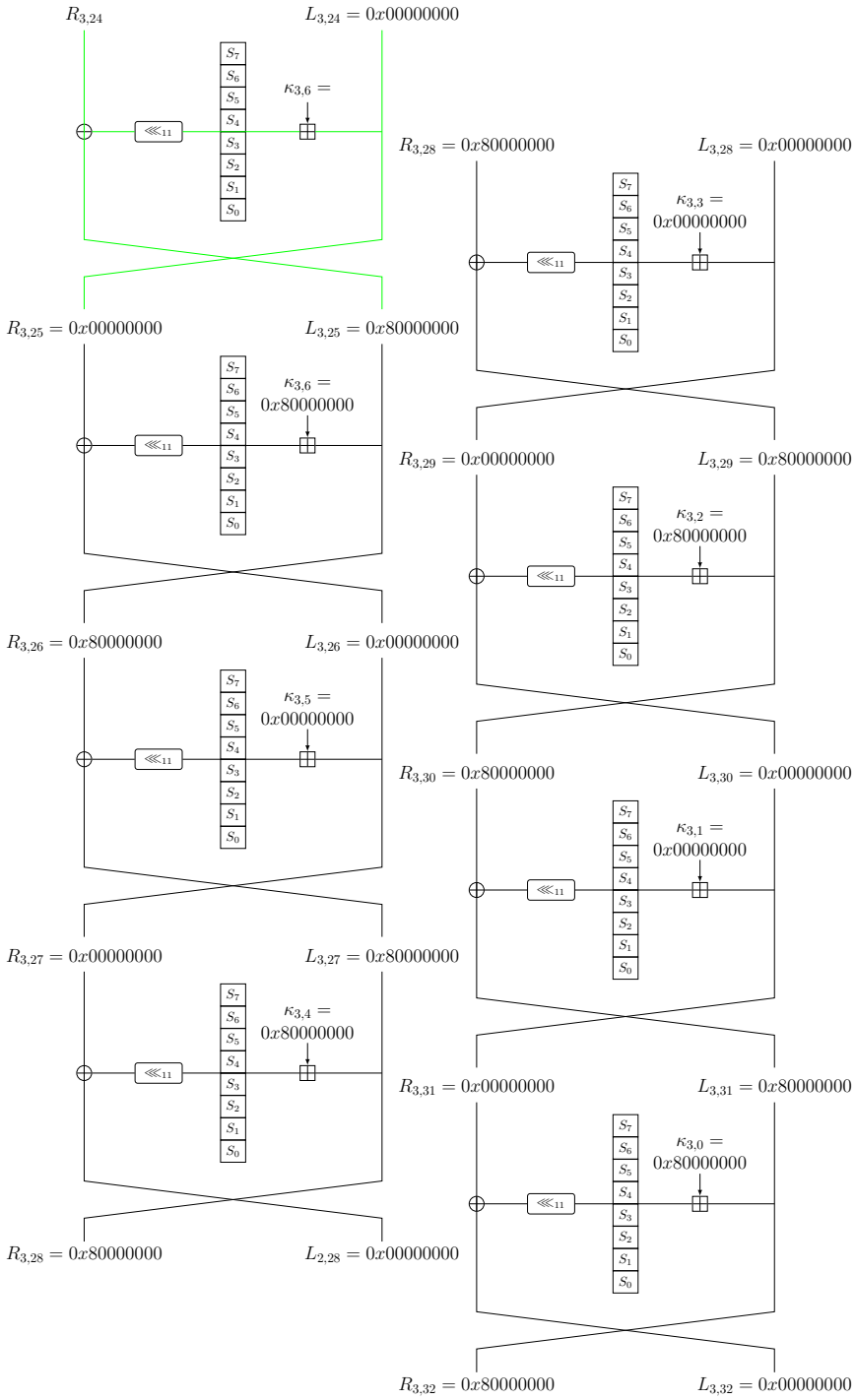


Fig. 35: The 7-round related-key differential of `GOST-FB/PS`

G Detailed Attack on SKINNYee

As SKINNYee processes a $16c$-bit ($c = 4$) plaintext and a $64c$-bit tweak (i.e., $4 \times 16c$), it is reasonable to assume that an attacker may access a data set of size $D > 2^{16c}$, provided that $D \leq 2^{16c+64c} = 2^{80c}$. An attack under this setting is known as a beyond full-codebook attack [11,45,46,47,48]. When selecting $2^{16c+\tau}$ plaintexts, the data complexity becomes $2^2 \cdot 2^{16c+\tau} = 2^{66+\tau}$. Regarding time complexity:

- Step 2(a): The time complexity is $2^2 \cdot 2^{11c} \cdot 2^{16c+\tau} \cdot 14/(16 \cdot 31) \approx 2^{104.86+\tau}$.
- Step 2(b): Processing $(2^{7c})^2$ data with 22 Sboxes yields a time complexity of $2 \cdot 2^{11c} \cdot 2^{14c} \cdot 22/(16 \cdot 31) \approx 2^{97.51}$.
- Step 2(c): Two lists of size $2^{19c+\tau}$ are constructed, resulting in a time complexity of $2 \cdot 2^{11c} \cdot 2^{19c+\tau} = 2^{121+\tau}$.
- Step 2(d): A total of $Q' = 2^{18c+2\tau}$ quartets are formed, leading to a time complexity of $2^{11c} \cdot 2^{18c+2\tau} = 2^{116+2\tau}$.
- Step 2(e): Given $Q'/Q = 2^{2\tau}$, the time complexity increases to $2^{120.1+2\tau}$. After this step, the probability that a key candidate survives is $p' = (1 - 2^{-18c})^{Q'} = e^{-2^{2\tau}}$.
- Step 3: As discussed in [10,49], the time complexity is $2^{128-2}(1-(1-e^{-2^{2\tau}})^4)$.

The maximum memory consumption occurs in Step 2(e), where 2^{26c} key candidates are stored, resulting in a memory complexity of 2^{104}.

Finally, the attack complexities are summarized as follows.

- In the full-codebook setting with $\tau = 0$, we mount a 31-round attack on SKINNYee with data complexity 2^{66}, time complexity $2^{125.75}$ and memory complexity 2^{104}.
- In the beyond full-codebook setting with $\tau = 1.2$, we mount a 31-round attack on SKINNYee with data complexity $2^{67.01}$, time complexity roughly $2^{123.53}$ (dominated by $2^{121+1.2} + 2^{120.1+2\cdot 1.2} + 2^{126}\big(1 - (1 - e^{-2^{2\cdot 1.2}})^4\big)$), and memory complexity 2^{104}.