Related-Key Cryptanalysis of FUTURE

Amit Jana[1], Smita Das[1], Ayantika Chatterjee[1], Debdeep Mukhopadhyay[1], and Yu Sasaki[2]

[1] Indian Institute of Technology, Kharagpur
{janaamit001,smita1995star,cayantika,debdeep.mukhopadhyay}@gmail.com
[2] NTT Social Informatics Laboratories and NIST Associate
{yusasaki0930}@gmail.com

Keywords: Related-Key Cryptanalysis, Boomerang Attack, FUTURE.

Abstract. At Africacrypt 2022, Gupta et al. introduced FUTURE, a 64-bit lightweight block cipher based on an MDS matrix and designed in an SPN structure, with a focus on achieving single-cycle encryption and low implementation cost, especially in unrolled architectures. While the designers evaluated its security under various attack models, they did not consider related-key cryptanalysis. In this work, we address this gap by analyzing the security of FUTURE in the related-key setting using techniques based on Mixed Integer Linear Programming (MILP). We first propose a simplified and generalizable approach for applying MILP to model any MDS or near-MDS-based cipher that follows the substitution-permutation paradigm. Using our MILP framework, we construct an 8-round related-key distinguisher on FUTURE, requiring 2^{58} plaintexts, 2^{58} XOR operations, and negligible memory. We further identify a full-round (i.e., 10 rounds) boomerang distinguisher with a probability of $2^{-45.8}$, enabling a distinguishing attack with 2^{48} data and time complexity. In addition, we develop a full-round key recovery attack on FUTURE with data, time, and memory complexities of 2^{18}, 2^{70}, and 2^{64}, respectively. Although all known single-key attacks remain impractical (with time complexities of at least 2^{112}), our results demonstrate a full-round cryptanalysis of FUTURE in the related-key setting, thereby challenging its claimed security guarantees.

1 Introduction

In recent years, the demand for cryptographic solutions optimized for resource-constrained environments–such as RFID tags, sensor networks, and contactless smart cards–has led to the development of lightweight cryptographic primitives. Unlike traditional cryptographic methods like AES [21] and SHA-256 [33], which are designed for systems with substantial processing power and memory, lightweight cryptography prioritizes efficiency across various metrics including hardware cost, power utilization, and latency. Block ciphers, which can be thought of a pseudo-random permutations to transform plaintext into ciphertext blocks of fixed lengths, are mainly categorized into Feistel structures

and substitution-permutation networks (SPNs). Feistel structures, used in ciphers like TWINE [46] and Piccolo [39], are cost-effective but require more rounds to ensure security, while SPNs offer robust security but can be more resource-intensive. The field of lightweight cryptography has expanded significantly, with ciphers such as PRESENT [16], KATAN [19], SIMON & SPECK [6], PRINCE [17], MANTIS [7], LED [24], MIDORI [4], and GIFT [5] being optimized for parameters like code size, latency, and energy consumption. Moreover, tweakable block ciphers like SKINNY [7], CRAFT [8], and QARMA [3], enhance encryption and authentication modes. Additionally, CRAFT addresses challenges such as resistance to Differential Fault Analysis (DFA) attacks.

Several lightweight block ciphers, including LED, MIDORI, and SKINNY, build on the fundamental structure of the AES round function, modifying its components to enhance performance. AES employs MDS (Maximum Distance Separable) matrices in its round function to achieve strong diffusion, which is essential for robust security against various cryptographic threats. However, incorporating MDS matrices into lightweight block ciphers poses a challenge due to their high implementation cost. This often necessitates additional rounds in these ciphers to maintain security against attacks such as differential and linear attacks. As a result, many lightweight block ciphers opt for lighter components, such as near-MDS matrices and bit-permutations, to avoid the high costs associated with MDS matrices. This approach helps manage implementation costs while still aiming to achieve effective diffusion, even though MDS matrices offer superior diffusion benefits.

Mixed Integer Linear Programming (MILP) is a well-established optimization technique used to find the optimal solution for a linear objective function subject to a set of linear constraints. In 2011, Mouha *et al.* introduced an automated differential path search method utilizing MILP [31], which helps generate lower bounds for the number of active S-boxes. At that time, the method could not account for the differential properties of the S-box, limiting its application to bit-oriented ciphers like PRESENT and LS-designs [30]. This limitation was later addressed by Sun *et al.* [45,44], who developed two distinct methods to model the differential propagation of S-boxes using systems of inequalities. The first approach uses logical conditions to represent differential properties through linear inequalities. The second approach employs a geometric method to capture all possible input-output difference transitions through an S-box, computing the H-representation (convex hull) of this set using the SageMath inequality generator function and simplifying constraints with a greedy approach. Sasaki and Todo [36] further advanced this technique by incorporating a MILP-based optimization phase to achieve a more compact representation of S-boxes with fewer constraints. Additionally, Boura and Coggia [18] enhanced the method by reducing the number of constraints needed to capture the differential properties of an S-box by adding related constraints from the set of constraints generated by the SageMath inequality generator function.

In 2022, Gupta *et al.* introduced a new 64-bit lightweight block cipher known as FUTURE [25], which stands out for its exceptionally low implementation cost

compared to other block ciphers, particularly when implemented in an unrolled fashion. Notably, FUTURE is one of the few lightweight ciphers where all the round components are new, and it employs an MDS matrix for its diffusion layer. The internal functions of the cipher are designed for high hardware efficiency, with the MDS matrix and S-box being specifically optimized to minimize hardware costs. The S-box used is reported to match the cryptographic quality of those in SKINNY and PICCOLO. Hardware benchmarks on FPGA and ASIC platforms demonstrated that FUTURE outperforms several well-known lightweight ciphers in terms of size, critical path, and throughput, achieving superior results across multiple metrics.

Researchers have explored various attack methods on the FUTURE cipher in single-key settings. In [26], a bit-based Mixed integer linear programming (MILP) approach was used to identify both differential and linear distinguishers, revealing distinguishers up to five rounds with probabilities of 2^{-58} and 2^{-62}, respectively. In [37], a full-round key recovery attack was presented by combining the meet-in-the-middle (MITM) technique with MILP, resulting in data, time, and memory complexities of 2^{64}, 2^{126}, and 2^{34}, respectively. Similarly, Lin *et al.* [29] applied a MILP-assisted MITM strategy to the full-round cipher, achieving complexities of 2^{64} for data, 2^{124} for time, and 2^{48} for memory. Roy *et al.* [35] proposed an attack leveraging biclique structures, achieving data, time, and memory complexities of 2^{48}, $2^{125.54}$, and 2^{32}, respectively. In another work, Mondal *et al.* [30] utilized the Yoyo technique in the secret-key setting, successfully distinguishing up to five and six rounds with data complexities of $2^{9.83}$ and $2^{58.83}$, respectively. More recently, Xu *et al.* [49] introduced a full-round key recovery attack using integral cryptanalysis, offering improved complexities compared to an exhaustive search over the full codebook.

In the context of related-key cryptanalysis, it is important to note that block ciphers are often employed within various modes of operation, where the security proofs may assume that block ciphers are ideal ciphers. Hence block ciphers are required to resist related-key attacks in such modes. Also, related-key attacks can be effectively applied to hashing modes such as the Davies-Meyer mode. In these constructions, encryption keys may act as message blocks, while plaintexts serve as chaining values. This configuration makes hash functions susceptible to related-key attacks. By selecting message pairs with specific differences, an attacker can effectively generate related keys and observe how these differences propagate through the cipher to influence the hash output. This means that using block ciphers in hash functions can open up unexpected weaknesses. So, it is important to carefully check their security when the inputs (like keys) are related in some way.

Although theoretical attacks on FUTURE have been explored in the single-key setting, no related-key cryptanalysis has been conducted so far, and the original design specification did not address this model. This paper aims to fill that gap by presenting a thorough analysis of related-key cryptanalysis on FUTURE cipher. We construct a bit-oriented MILP framework to identify enhanced differential characteristics. While [26] introduces a bit-oriented MILP model for

analyzing FUTURE in the single-key context, it does not provide sufficient details for related-key scenarios. In this work, we offer a complete description of how to formulate a bit-oriented MILP model tailored to the related-key setting of the FUTURE cipher. This framework also offers a simplified approach for formulating constraints for the individual components of each round, making it adaptable to analyze other MDS or near-MDS matrix-based SPN ciphers. Leveraging our proposed MILP framework, we present a full cryptanalysis of the FUTURE cipher in the related-key setting.

Our Contributions. Our contributions are three-fold, as follows:

- We propose an extensive bit-based related-key MILP model for the FUTURE cipher, which can be helpful for building MILP models for any MDS matrix-based SPN ciphers. We revisit Boura and Coggia's [18] work due to insuffcient information in their proposed algorithm to generate an optimal number of constraints to capture the behavior of DDT. From our understanding, we provide a revised algorithm that produces similar results by Boura and Coggia [18, Algorithm 1], but with a larger set of final constraints. Additionally, we provide a detailed explanation of how to construct a primitive representation of MDS (or near-MDS) matrices using a companion matrix approach, which is compatible with the cipher structure.
- Utilizing this technique, we demonstrate an 8-round related-key differential characteristic for FUTURE with a probability of $2^{-56.4}$, which leads to a distinguisher with 2^{58} data, 2^{58} time, and negligible memory complexities.
- In addition, we construct a full-round related-key boomerang distinguisher with practical complexity, followed by a key recovery attack that also operates within practical bounds, demonstrating a full-round break of the cipher. A comprehensive comparison of existing attack techniques and their respective complexities is presented in Table 1.

Attack	Types	#Rounds	Settings	Prob.	Complexity			Reference
					Data	Time	Memory	
Differential	–	5	Single-key	2^{-58}	–	–	–	[26]
MITM	Key Recovery	10	Single-key	–	2^{64}	2^{126}	2^{34}	[37]
MITM	Key Recovery	10	Single-key	–	2^{64}	2^{124}	2^{48}	[29]
Biclique	Key Recovery	10	Single-key	–	$\leq 2^{48}$	$2^{125.53}$	2^{32}	[35]
Yoyo	Distinguisher	6	Single-key	–	$2^{58.83}$	–	–	[30]
		8	Known-key	–	2^{15}	–	–	
Integral	Key Recovery	10	Single-key	–	2^{63}	$2^{123.7}$ Enc.	2^{16}	[49]
				–	2^{64}	2^{112} Enc.	$2^{16.1}$	
Differential	Distinguisher	8	Related-key	$2^{-56.4}$	2^{58}	2^{58} XOR	Negligible	This Work
Boomerang	Distinguisher	10	Related-key	$2^{-45.8}$	2^{48}	2^{48} XOR	Negligible	This Work
Boomerang	Key Recovery	10	Related-key	–	2^{48}	2^{112} Enc./Dec.	Negligible	This Work
				–	2^{18}	2^{70} Enc./Dec.	2^{64}	This Work

Table 1: A Comparison of Different Attacks on FUTURE

Outline of the Paper. The paper is structured as follows: Section 2 provides an overview of the FUTURE cipher. In Section 4, we present a brief introduction to related-key differential and boomerang cryptanalysis. Section 4 explains the bit-oriented MILP model used for the analysis. In Section 5, we apply this model to construct related-key differential, boomerang distinguishes, and then the key recovery for the FUTURE cipher. Finally, Section 6 concludes the paper with remarks and suggestions for future work.

2 Description of FUTURE

FUTURE is an SPN-based 64-bit lightweight block cipher designed to have applications on low hardware cost and latency. It has a key size of 128-bit.

The Round Function. The round structure of the FUTURE cipher consists of four operations: SubCell, MixColumn, ShiftRow, and AddRoundKey, as illustrated in Figure 2. Notably, the MixColumn operation is omitted in the final round. The cipher processes a 64-bit input state S arranged as a 4×4 matrix, where each cell is a nibble (i.e., $s_i \in \{0,1\}^4$ for $0 \leq i \leq 15$), as shown in the Figure 1a. Furthermore, the round structure is depicted in the following Figure 11.

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

(a) The FUTURE State

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$S(x)$	1	3	0	2	7	E	4	D	9	A	C	6	F	5	8	B

(b) The S-box Table

Fig. 1: The State Representation and S-box Table of FUTURE Cipher

SubCell. The nonlinear transformation in the round function is defined by the SubCell operation, which applies a 4-bit S-box to each cell of the state matrix. This transformation is depicted in Figure 1b.

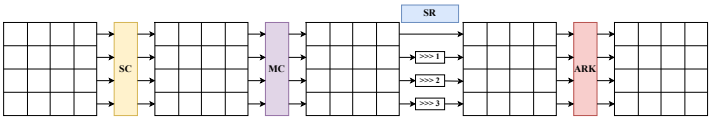


Fig. 2: Round Function

MixColumn. The linear operation is represented by the finite field matrix multiplication involving the MDS (maximum distance separable) matrix (μ) and the state matrix, where the matrix elements are in $GF(2^4)$. The MDS matrix is illustrated in Figure 3a. Matrix and vector multiplications are performed in the field $\mathbb{F}_{2^4}$, defined by the primitive polynomial $x^4 + x + 1$.

$$\mu = \begin{bmatrix} 8 & 9 & 1 & 8 \\ 3 & 2 & 9 & 9 \\ 2 & 3 & 8 & 9 \\ 9 & 9 & 8 & 1 \end{bmatrix} \qquad \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{bmatrix} \rightarrow \begin{bmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_{13} & s_1 & s_5 & s_9 \\ s_{10} & s_{14} & s_2 & s_6 \\ s_7 & s_{11} & s_{15} & s_3 \end{bmatrix}$$

(a) The MixColumn Matrix (b) The ShiftRow Operation

Fig. 3: The MixColumn Matrix and ShiftRow Operation of FUTURE Cipher

ShiftRow. Each row $(row(i), i = 0, 1, 2, 3)$ of the state matrix is rotated to the right by i positions following the MixColumn operation. This process is illustrated in Figure 3b.

AddRoundKey. The 64-bit round keys (sub-keys) $SK_i, i = 0, 1, \ldots, 10$ are XORed to the state S in each round. Additionally, the final round sub-key SK_{10} is XORed with the state before producing the ciphertext.

Key Schedule. In FUTURE encryption, a 128-bit secret key K is divided into two halves k_0 and k_1 for generating round and whitening keys. k_0 acts as the whitening key and generates each round sub-keys $\mathrm{SK}_i, i = 0, 1, \ldots, 10$ depends on whether i is even or odd. If i is even, k_0 is left-rotated by $5 \cdot \frac{i}{2}$ bits; if i is odd, k_1 undergoes the same left rotation. Left rotation involves circularly shifting bits. Additionally, except for the 5^{th} and 10^{th} rounds, a single '1' bit is XORed into specific positions within 4-bit cells during each encryption round, with these operations defined by round constants.

3 Related-Key Cryptanalysis

A related-key attack [9] involves analyzing a cipher using multiple keys with known mathematical relationships between them. The attacker has access to encryption or decryption functions with these keys and aims to determine the actual secret keys. The simplest form uses a constant (Δ) XOR relation between keys, such as $K_2 = K_1 \oplus \Delta$. Related-key attacks offer more freedom compared to other attacks but can be more challenging to implement. Resistance to such attacks is crucial, as exemplified by the design goals of AES cipher. This paper employs differential attacks and boomerang attacks in a related-key context.

3.1 Related-Key Differential Cryptanalysis

Differential cryptanalysis is an effective method for analyzing and attacking symmetric-key ciphers by examining the differences between pairs of plaintexts and ciphertexts, known as "differential". Introduced by Biham and Shamir [13] in 1990, this technique seeks to identify specific patterns in these differences, termed "differential characteristics", that are unique to the encryption algorithm. By studying these patterns, cryptanalysts can infer the internal state of

the cipher and, with sufficient data, uncover the secret key. This approach can lead to distinguishing attacks and key-recovery attacks.

In related-key attack settings, an attacker can establish or enforce a relationship between multiple keys and has access to the corresponding encryption and decryption functions for all these keys. Consider a tuple $(\Delta_{\text{in}}, \Delta_K, \Delta_{\text{out}})$ as an n-round related-key differential for a keyed round function f_K, where f_K^i represents the output after the i-th round for $i = 0, 1, \ldots, n-1$. This differential is valid if, for some plaintext P and key K, the equation $f_K^{n-1}(P) \oplus f_{K\oplus\Delta_K}^{n-1}(P \oplus \Delta_{\text{in}}) = \Delta_{\text{out}}$ holds. Let $S_{P,K}^i$ denote the internal state of the round function at round i with inputs P and K. The tuple $(\Delta_{\text{in}}, \Delta_K, \Delta S_0, \ldots, \Delta S_{n-1} = \Delta_{\text{out}})$ is an n-round related-key differential characteristic if $(\Delta_{\text{in}}, \Delta_{\text{out}}, \Delta_K)$ is an n-round related-key differential and for all i, $S_{P,K}^i \oplus S_{P\oplus\Delta_{\text{in}},K\oplus\Delta_K}^i = \Delta S_i$.

Let $p = \Pr[(\Delta_{\text{in}}, \Delta_K) \to \Delta_{\text{out}}]$ represent the probability that the related-key differential $(\Delta_{\text{in}}, \Delta_K, \Delta_{\text{out}})$ holds. This implies that if $\frac{1}{p}$ number of plaintexts P and keys K are selected uniformly at random, the equation $f_K(P) \oplus f_{K\oplus\Delta_K}(P \oplus \Delta_{\text{in}}) = \Delta_{\text{out}}$ will be satisfied redonce in average with probability more than 60%.

3.2 Related-Key Boomerang Attack.

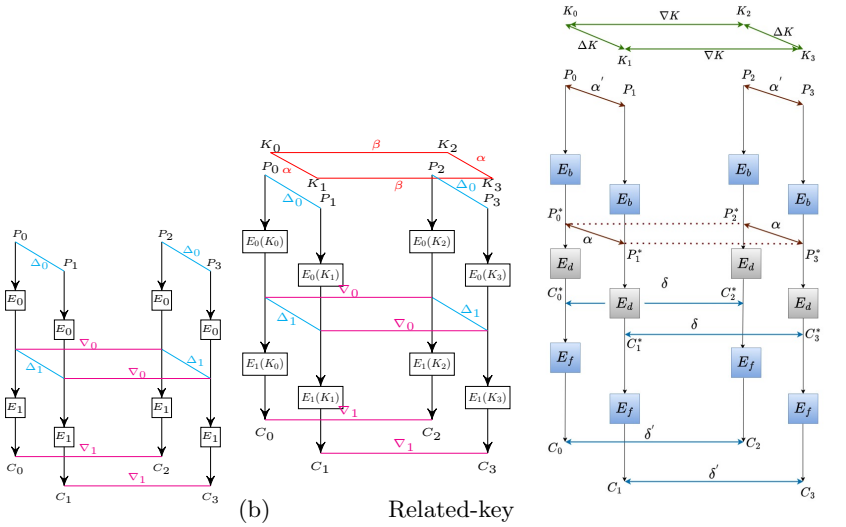
The boomerang attack, introduced by Wagner in [48], is a differential cryptanalysis method that combines two high-probability differentials to enhance the chances of breaking a cipher. This is described in Figure 4a. For a block cipher $E = E_1 \circ E_0$, with differentials $\Delta_0 \xrightarrow{p} \Delta_1$ for E_0 and $\nabla_0 \xrightarrow{q} \nabla_1$ for E_1, the attack checks if differential relationships hold, with an expected probability of success given by:

$$\Pr(E^{-1}(E(x) \oplus \nabla_1) \oplus E^{-1}(E(x \oplus \Delta_0) \oplus \nabla_1) = \Delta_0] = p^2 \cdot q^2.$$

The procedure for mounting the distinguisher in adaptive settings is as follows:

1. Request the ciphertexts $C_0 = E(P_0)$ and $C_1 = E(P_1)$, where $P_1 = P_0 \oplus \Delta_0$.
2. Request the plaintexts $P_2 = E^{-1}(C_2)$ and $P_3 = E^{-1}(C_3)$, where $C_2 = C_0 \oplus \nabla_1$ and $C_3 = C_1 \oplus \nabla_1$.
3. Verify if $P_2 \oplus P_3 = \Delta_0$?

To amplify this attack, the amplified boomerang attack [27] was proposed which works in a non-adaptive (chosen-plaintext attack) scenario. In this attack, the expected probability to get a right quartet will be $p^2 \cdot q^2 \cdot 2^{-n}$. Furthermore, in [10,11], they have pointed out that any value of Δ_1 and ∇_0 can be considered as long as $\Delta_1 \neq \nabla_0$. As a result, the probability of the right quartet is increased to $2^{-n} \cdot \hat{p}^2 \cdot \hat{q}^2$, where $\hat{p} = \sqrt{\sum_i \Pr^2(\Delta_0 \to \Delta_1^i)}$ and $\hat{q} = \sqrt{\sum_j \Pr^2(\nabla_0^j \to \nabla_1)}$. Note that this amplification can be done in the adaptive setting to increase the probability to $\hat{p}^2 \cdot \hat{q}^2$ from $p^2 \cdot q^2$. The sandwich attack [22] further refines this approach by decomposing the cipher into three parts and using the Boomerang Connectivity Table [20] (BCT) to systematically analyze the connections between



(a) Boomerang (b) Related-key Boomerang (c) Boomerang Key Recovery

Fig. 4: The Boomerang Framework

input and output differences, improving the probability approximation of the distinguisher.

The related-key boomerang attack [12], depicted in Figure 4b, utilizes both key and plaintext differences. It assumes that the upper sub-cipher E_0 follows a differential characteristic $\Delta_0 \xrightarrow{p} \Delta_1$ under a key difference $\alpha = K_0 \oplus K_1 = K_2 \oplus K_3$, while the lower sub-cipher E_1 has a differential characteristic $\nabla_0 \xrightarrow{q} \nabla_1$ under a key difference $\beta = K_0 \oplus K_2 = K_1 \oplus K_3$. A related-key distinguisher is built using four different unknown keys: K_0, $K_1 = K_0 \oplus \alpha$, $K_2 = K_0 \oplus \beta$, and $K_3 = K_1 \oplus \beta$. The related-key boomerang distinguisher in the adaptive scenario is executed as follows:

1. Request the ciphertext pairs (C_0, C_1), where $C_0 = E_{K_0}(P_0)$ and $C_1 = E_{K_1}(P_1)$, with $P_0 \oplus P_1 = \Delta_0, K_0 \oplus K_1 = \alpha$.
2. Request the plaintexts pairs (P_2, P_3), where $P_2 = E^{-1}_{K_2}(C_2)$ and $P_3 = E^{-1}_{K_3}(C_3)$, with $K_0 \oplus K_2 = \beta, K_1 \oplus K_3 = \beta, C_2 = C_0 \oplus \nabla_1$ and $C_3 = C_1 \oplus \nabla_1$.
3. Verify if $P_2 \oplus P_3 = \Delta_0$?

3.3 Related-Key Boomerang Key Recovery Attack

In general, based on the boomerang distinguisher, the attacker extends the input and/or output by one or two rounds and filters these rounds to enable key recovery in an adaptive setting. In [40,50], the authors proposed a generic and unified framework for key recovery in both rectangle and boomerang attacks. Following their approach, we present the boomerang key recovery attack in this work.

For boomerang key recovery, we treat the cipher as $E = E_f \circ E_d \circ E_b$, where E_d represents a boomerang distinguisher with probability p^2, i.e.,

$$\Pr[E_d^{-1}\big(E_d(P_0^*) \oplus \delta\big) \oplus E_d^{-1}\big(E_d(P_0^* \oplus \alpha) \oplus \delta\big)] = p^2.$$

The framework for the related-key boomerang attack is depicted in Figure 4c. The goal of key recovery is to identify partial subkeys used in E_b and E_f by leveraging the boomerang distinguisher over E_d, thereby recovering the master key more efficiently than an exhaustive search. According to Figure 4c, the input difference $\alpha^{'}$ of the distinguisher propagates to the difference α through E_b, which then propagates back to $\alpha^{'}$ through E_b^{-1}. Similarly, the difference δ propagates to $\delta^{'}$ through E_f.

Let V_b (resp. V_f) denote the difference space spanned by all possible $\alpha^{'}$ (resp. $\delta^{'}$), where $r_b = \log_2 |V_b|$ (resp. $r_f = \log_2 |V_f|$). Let K_b (resp. K_f) be the subset of subkey bits used in E_b (resp. E_f) that influence the differential propagation $\alpha^{'} \rightarrow \alpha$ (resp. $\delta \rightarrow \delta^{'}$). Define $m_b = |K_b|$ (resp. $m_f = |K_f|$) as the number of subkey bits in K_b (resp. K_f). To improve complexity, a new approach for key recovery is to partially guess a subset of K_b (resp. K_f) denoted by $K_b^{'}$ (resp. $K_f^{'}$), which will be guessed first. Let $m_b^{'} = |K_b^{'}|$ (resp. $m_f^{'} = |K_f^{'}|$). Suppose that with the guessed subkey bits, an $r_b^{'}$-bit (resp. $r_f^{'}$-bit) condition on the differential propagation $\alpha^{'} \rightarrow \alpha$ (resp. $\delta \rightarrow \delta^{'}$) can be verified. Additionally, we define $r_b^* = r_b - r_b^{'}$ (resp. $r_f^* = r_f - r_f^{'}$) and $m_b^* = m_b - m_b^{'}$ (resp. $m_f^* = m_f - m_f^{'}$).

In the related-key boomerang, the keys related to the master key K_0 are determined, where $K_1 = K_0 \oplus \Delta K$, $K_2 = K_0 \oplus \nabla K$, and $K_3 = K_0 \oplus \nabla K \oplus \Delta K$. In the attack, we add some extra rounds at the end of the distinguisher (i.e., $E = E_f \circ E_d$) and perform the key recovery steps. The key recovery for the boomerang attack in the related-key settings is as follows:

1. Construct a set S with y number of structures, where each structure contains 2^{r_f} number of ciphertexts. Let $D = |S_0| = y \cdot 2^{r_f}$. For each structure, query all the ciphertexts inside it under the four related-keys K_0, K_1, K_2, K_3 and store the corresponding plaintext-ciphertext pair in the sets S_0, S_1, S_2, S_3 respectively. Thus, the query and collection of four sets of data are as follows.

$$\begin{aligned} S_0 &= \{(P_0, C_0) | P_0 = E_{K_0}^{-1}(C_0), C_0 \in S\}, \\ S_1 &= \{(P_1, C_1) | P_1 = P_0 \oplus \alpha, C_1 = E_{K_1}(P_1), P_0 \in S_0\}, \\ S_2 &= \{(P_2, C_2) | P_2 = E_{K_2}^{-1}(C_2), C_2 \in S\}, \\ S_3 &= \{(P_3, C_3) | P_3 = P_2 \oplus \alpha, C_3 = E_{K_3}(P_3), P_2 \in S_2\}. \end{aligned}$$

2. Split $K_f^{'}$ (of $m_f^{'}$ bits) into two parts $G_L || G_R$, where G_L is of size t-bits, $0 \leq t \leq m_f^{'}$.
3. Guess G_R:
   3.1. Initialize a list of key counters of $(t + m_f^*)$-bits for G_L and unguessed key bits of K_f.

3.2. Guess the t-bit of G_L:

3.2.1. For each $i = 0, 1, 2, 3$ and the cihphertexts $C_i \in S_i$, do partial decryptions under $K_f^{'}$ to get $C_i^* = Partial\_Dec_{K_f^{'}}(C_i)$. Thus, the set containing C_1^* can be thought of as a set containing $y \cdot 2^{r_f^{'}}$ substructures with $2^{r_f^*}$ ciphertexts.

3.2.2. Construct sets as

$$S_{0,1} = \{(P_0, C_0^*, P_1, C_1^*) | P_1 = P_0 \oplus \alpha, C_1^* = Partial\_Dec_{K_f^{'}}(C_1)\},$$
$$S_{2,3} = \{(P_2, C_2^*, P_3, C_3^*) | P_3 = P_2 \oplus \alpha, C_3^* = Partial\_Dec_{K_f^{'}}(C_3)\}.$$

Insert $S_{0,1}$ and $S_{2,3}$ into a hash table according to $(n - r_f^*)$ inactive bits of C_0^*, C_2^* and C_1^*, C_3^* respectively.

3.2.3. There are $y \cdot 2^{r_f^{'}}$ possible values for the $(n - r_f^*)$ bits of C_1^* and $2^{(n-r_f^*)}$ possible values for the $(n - r_f^*)$ bits of C_3^*. For each index, we chose two distinct entries (P_0, C_0^*, P_1, C_1^*) and (P_2, C_2^*, P_3, C_3^*) to generate the quartet. The number of quartets will be

$$\binom{\frac{|S_{0,1}|}{y \cdot 2^{r_f^{'}} \cdot 2^{(n-r_f^*)}}}{2} \cdot y \cdot 2^{r_f^{'}} \cdot 2^{n-r_f^*} \approx D \cdot 2^{2 \cdot r_f^* - n - 1}, \text{ where } |S_{0,1}| = y \cdot 2^{r_f}.$$

3.2.4. Determine the key candidates involved in E_f and increase the corresponding counters. Also, we denote the time complexity for processing one quartet by ϵ.

3.3. Select the top $2^{t+m_f^*-h}$ hits on the counters as possible right subkey candidates that deliver a h-bit or higher advantage.

3.4. Guess the remaining $(|K| - m_f)$ bits of the key and search it exhaustively to find the original key.

Data Complexity. For each structure in S, the number of possible ciphertext pairs will be $2^{2 \cdot r_f}$. Therefore, among such $y \cdot 2^{2 \cdot r_f}$ pairs, the expected number of pairs left to satisfy the difference δ will be $y \cdot 2^{r_f}$. Thus, if s be the expected number of right boomerang quartets, then we have $y \cdot 2^{r_f} \cdot p^2 = s \implies y = s \cdot 2^{-r_f}/p^2$ and $D = s/p^2$. Therefore, for each queries of four sets S_0, S_1, S_2, S_3, the data complexity will be $D^{'} = 2 \cdot D = 4 \cdot s/p^2$.

Time Complexity. Since the time complexity to collect the data is $T_0 = D^{'}$, the time complexity to perform partial decryptions (at Step 3.2.1) is $T_1 = 2^{m_f^{'}} \cdot T_0 = s \cdot 2^{m_f^{'}+2}/p^2$. Also, the time complexity to generate the sets $S_{0,1} \backslash S_{2,3}$ (at step 3.2.2) is $T_2 = 2^{m_f^{'}} \cdot D = s \cdot 2^{m_f^{'}+1}/p^2$. Moreover, the time complexity (at step 3.2.3) of generating and processing quartet candidates is $T_3 = 2^{m_f^{'}} \cdot D \cdot 2^{2 \cdot r_f^* - n - 1} \cdot \epsilon = s \cdot 2^{m_f^{'}+2 \cdot r_f^* - n} \cdot \epsilon/p^2$. Finally, the time complexity for the exhaustive search is $2^{m_f^{'}-t} \cdot 2^{k+t+m_f^{'}-h} = 2^{k-h}$, where $h \leq t + m_f^*$.

Memory Complexity. The overall memory complexity M of the attack requires to store the data in the sets and for the key counters. Thus, we have $M = D^{'} + D + 2^{t+m_f^*}$.

On h**.** According to [38], the success probability of differential analysis is

$$p_s = \int_{\frac{\sqrt{s \cdot S_N} - \phi^{-1}(1-2^{-h})}{\sqrt{S_N+1}}}^{\infty} \phi(x) \cdot dx,$$

where S_N is the signal-to-noise ratio and $S_N = \frac{2^{-n} \cdot p^2}{2^{-2n}}$ for the boomerang attack.

On ϵ. A straightforward approach to determine the remaining subkey bits suggested by a quartet candidate is to use a guess and check method. This involves guessing the remaining subkey bits and verifying whether the quartet is valid under the given guess. This method does not require additional memory, while ϵ represents the number of partial encryptions or decryptions needed.

4 Mixed-Integer Linear Programming models

Mixed-integer linear programming (MILP) has been successfully utilized to develop automated search algorithms for differential and linear cryptanalysis. Two primary modeling approaches exist for implementing ciphers: the word-oriented model and the bit-oriented model. In the word-oriented model, the cipher state is treated as a sequence of words, with each word represented as a binary variable. In contrast, the bit-oriented model represents each bit of the cipher state as a binary variable, ensuring the generation of the most optimal and valid differential characteristics without any inconsistencies in the trail under the independent subkey assumption. The MILP constraints introduced in Mouha *et al.*'s method are insufficient to fully capture the differential propagation behavior in linear diffusion layers built from non-MDS codes. In [43], the authors first proposed a bit-oriented model specifically for SPN ciphers that utilize bit permutation-based linear layers.

In this section, we model the FUTURE cipher components as constraints to construct a bit-based MILP model for analyzing differential characteristics. To build this model, the S-box, permutation, and matrix multiplication over a finite field are represented by linear inequalities with binary variables. In [26], the authors present a bit-based MILP model for the FUTURE cipher aimed at searching for single-key differential and linear characteristics. However, the details provided are incomplete, particularly regarding the generation of linear inequalities for the S-box and the conversion of the MDS matrix to a binary matrix using the companion matrix representation. By employing linear inequalities, one can construct a comprehensive bit-based MILP model that automatically identifies differential characteristics.

4.1 Constraints for SubCell Operation

For differential cryptanalysis using bit-based MILP, the goal is to generate a minimal number of constraints involving input and output bits of an S-box to capture the actual behavior of the differences according to the difference distribution table (DDT). Let us assume that, $(x_0, \ldots, x_{n-1})$ and $(y_0, \ldots, y_{n-1})$ represent the input and output bit differences of an $n \times n$ S-box respectively. The problem corresponds to modeling the fact that $(x_0, \ldots, x_{n-1}) \rightarrow (y_0, \ldots, y_{n-1})$ is a possible difference transition in a DDT. In this regard, two different approaches were proposed in 2014 by Sun *et al.* [45,44]. The first is a geometrical one and consists of computing the H-representation of the convex hull of the set of possible transitions. The second one is based on logical condition modeling. The first approach is to use the Sagemath inequality generator by taking all the valid difference transition points from the DDT and it generates the number of linear inequalities which satisfies all the valid difference transition points. However, the number of inequalities using Sagemath is typically quite high with many redundant inequalities. The authors of [45] applied a greedy algorithm to reduce the number of constraints. In this approach, the algorithm adds to the solution set, the best possible inequality which can remove the highest number of impossible difference transition points among those that have not been removed yet. Later, Sasaki and Todo in [36] proposed a new reduction algorithm to further reduce the number of constraints compared to the greedy approach. They proposed to model the problem of minimizing the set of inequalities that remove all the impossible difference transition points as a MILP problem itself and solve it by some solver. More precisely, their method consists of assigning a binary variable z_i to each inequality in which $z_i = 1$ denotes that inequality i is included in the system. Then for each impossible difference transition point j, add the corresponding constraints in the list $\mathcal{L}_j$ which leads to the inequality as $\sum_{i \in \mathcal{L}_j} z_i \geq 1$. Finally, the MILP solver is used for minimizing $\sum_i z_i$ giving a solution to optimize the constraints to capture the DDT of an S-box.

However, there are other works [47,28,34] that further reduce the number of constraints to capture DDT of S-box by proposing new approaches to generate additional inequalities, surpassing the Sagemath inequality generator or using Boura and Coggia's approach. In this work, we follow the method outlined by Boura and Coggia [18] to generate the constraints that capture the DDT of a FUTURE S-box. The DDT of the FUTURE S-box is provided in Table 4a (in Appendix A). The algorithm presented by Boura and Coggia [18, Algorithm 1] for deriving a set of inequalities from the DDT of an S-box lacks certain details. Specifically, in step 7, the authors state that if C_{new} removes a new set of impossible transitions, C_{new} should be added as new constraints to capture the DDT of the S-box. However, the precise meaning of C_{new} is unclear. It is not explicitly explained whether the set S of impossible transitions from C_{new} should contain elements distinct from the elements of S_i, for all i, where each S_i is a set of impossible transitions for every constraint in $\mathcal{C}_{set}$. To clarify this, we revisited their approach and outlined the complete steps necessary to generate the S-box constraints based on our understanding.

Algorithm 1 Revised Boura and Coggia's Approach to Compute a Set of Inequalities from DDT of an S-box.

procedure ComputeConstraints($\mathcal{VDP}, k(\geq 2)$)
  $\mathcal{H}_{set} \leftarrow$ `inequality_generator(`$\mathcal{VDP}$`)`
  $\mathcal{C}_{set} \leftarrow \mathcal{H}_{set}$
  $\mathcal{D}_{set} \leftarrow \{\}$
  for all $\alpha \in \mathcal{VDP}$ **do**
    $\mathcal{H}^{\alpha}_{set} = \{C \in \mathcal{H}_{set} | C(\alpha) = 0\}$
  for all $\mathcal{H}^{\alpha}_{set}, \alpha \in \mathcal{VDP}$ **do**
    if $k \geq |\mathcal{H}^{\alpha}_{set}|$ **then**
      for all $\{C_1, \cdots, C_k\} \subseteq \mathcal{H}^{\alpha}_{set}$ **do**
        $C_{new} = C_1 + \cdots + C_k$
        Add C_{new} into $\mathcal{D}_{set}$
  for all constraints $C_i \in \mathcal{H}_{set}, 1 \leq i \leq |\mathcal{H}_{set}|$ **do**
    Construct the set $S_i = \{\beta \in \mathcal{IDP} | C_i(\beta) < 0\}$
  $\mathcal{L} = [S_1, \cdots, S_{|\mathcal{H}_{set}|}]$ ▷ A list of sets
  for all constraints $C_j \in \mathcal{D}_{set}, 1 \leq j \leq |\mathcal{D}_{set}|$ **do**
    Construct the set $S = \{\beta \in \mathcal{IDP} | C_j(\beta) < 0\}$
    if $\nexists$ any S_i from $\mathcal{L}$ such that $S \subseteq S_i$ **then**
      Add S in the list $\mathcal{L}$
      Add C_j in $\mathcal{C}_{set}$
  Apply Sasaki *et al.*'s [36] approach from $\mathcal{C}_{set}$ to finally get the optimal number of constraints to capture the DDT of S-box

Suppose, a difference transition $x \rightarrow y, x, y \in \mathbb{F}_2^n$ through the S-box can be seen as a vector of $\mathbb{F}_2^{2n}$, involving $2n$ binary variables represented as $(x_0, \ldots, x_{n-1}, y_0, \ldots, y_{n-1})$ or as $(x_0, \ldots, x_{2n-1})$. Let $\mathcal{VDP}$ and $\mathcal{IDP}$ denote a set of valid and impossible difference transition points according to the DDT of FUTURE S-box. For example, $0x12 \in \mathcal{VDP}$ and $0x11 \in \mathcal{IDP}$ are the valid and invalid difference points according to the DDT in Table 4a. Given $\mathcal{VDP}$ to the `inequality_generator()` function in the `sage.geometry.polyhedron` class of Sagemath returns a list of inequalities as the H-set representation of the convex hull of all possible transitions in a DDT. For FUTURE S-box, we get returns 214 inequalities. We denote this list of inequalities as $\mathcal{H}_{set}$. This set $\mathcal{H}_{set}$ has the following properties: (1.) each $\alpha \in \mathcal{VDP}$ must satisfy all the constraints in $\mathcal{H}_{set}$ and (2.) each $\beta \in \mathcal{IDP}$ will not be satisfied by at least one of the constraints in $\mathcal{H}_{set}$. The interesting point here is that the addition of any number of constraints always maintains the above two properties. Although, adding the constraints by choosing all the subsets (of cardinality k) of constraints from $\mathcal{H}_{set}$ and then append it to $\mathcal{H}_{set}$ can increase the list $\mathcal{H}_{set}$ remarkably high. Instead, for each $\alpha \in \mathcal{VDP}$, the authors choose the constraints from $\mathcal{H}_{set}$ which satisfies by the point α and store them in another list $\mathcal{H}^{\alpha}_{set}$. Then, for each set $\mathcal{H}^{\alpha}_{set}$, choose $\binom{|\mathcal{H}^{\alpha}_{set}|}{k}$ constraints, denoted $C_1, \ldots, C_k$, and add $C_{new} = C_1 + \cdots + C_k$ to a set $\mathcal{D}_{set}$. A list of sets, $\mathcal{L}$, is constructed, where each entry contains the impossible transition points for each constraint in $\mathcal{D}_{set}$. Finally, the constraints from $\mathcal{D}_{set}$

are filtered and added to $\mathcal{L}$, ensuring that the set of impossible transition points S is not a subset of any set already in $\mathcal{L}$. After this filtration, Sasaki and Todo's method is applied to the selected constraints to generate the optimal constraints for capturing the DDT of the S-box. These steps are detailed in Algorithm 1.

Using this algorithm with $k = 2$, we initially generate 971 constraints for C_{set}, which are eventually reduced to 17 constraints by applying the method proposed by Sasaki and Todo. In comparison, [18, Algorithm 1] reports approximately 500 initial constraints for the PRESENT S-box with $k = 2$, which are claimed to be reduced to 17 using the same technique. However, upon verifying their implementation[3], we observed that the number of resulting constraints is around 22, not 17 as stated.

By applying our revised method described in Algorithm 1, we obtained 1138 initial constraints for $\mathcal{C}_{new}$ for the PRESENT S-box, which were similarly reduced to 17 using Sasaki and Todo's method. Using this revised algorithm, we successfully derived 17 constraints for both the FUTURE and PRESENT S-boxes to accurately capture their DDT, as shown in Figure 17 and Figure 18 (Appendix A). Additionally, when our revised algorithm is applied to all 4×4 S-boxes, the resulting number of constraints matches the claims made by Boura and Coggia, as summarized in Table 3 (see Appendix A). However, it is worth noting from Table 3 that while the results match for most S-boxes, the values differ for the MIDORI S0 and TWINE S-boxes when using the revised Algorithm 1.

4.2 Constraints for MixColumn Operation

For word-based MILP modeling, Mouha *et al.* [31] modeled the MixColumn matrix multiplication using its branch number, i.e., a lower bounds on the number of active S-boxes. Whereas for a bit-based model, an MDS (or near MDS) matrix μ must be converted to a binary matrix over the base field $\mathbb{F}_2$, which is called the primitive representation of M. In [41], the authors give a short description of the primitive representation of μ using a companion matrix. However, in [42], Sun *et al.* provided a method to obtain a primitive representation using linear maps with matrix representation. In this work, we thoroughly explore how to efficiently compute a primitive representation of μ using a companion matrix that is compatible with the cipher's bit format, whether it follows a least significant bit (LSB) to the most significant bit (MSB) order or an MSB to LSB order. Finally, to model matrix multiplication in MILP, we might need several binary XOR operations.

For 1-XOR operation, $c = a \oplus b, a, b, c \in \{0, 1\}$, Mouha *et al.* [31] modeled it using 4 constraints and 3 variables as $a + b + c \geq 2d_1, d_1 \geq a, b, c$, where d_1 is a dummy variable. Similarly, for $d = a \oplus b \oplus c, a, b, c, d \in \{0, 1\}$, known as a 2-XOR operation, the approach requires 8 constraints and 5 variables. Yin *et*

[3] We refer to the code available at https://github.com/dnlcog/efficient$_m ilp_m odelings$, which accompanies the paper titled "Efficient MILP Modelings for S-boxes and Linear Layers of SPN Ciphers," published in ToSC 2020.

al. [51] showed a method to model it using 8 constraints and 4 variables. However, Fu *et al.* [23] efficiently model the 1-XOR operation using only one constraint $a+b+c+d=2d_1, a, b, c, d, d_1 \in \{0,1\}$. Based on this approach, the authors in [26] extend this approach to model the n-XOR operations $a_0 \oplus \ldots \oplus a_{n-1} = b$, as follows.

$$a_0 + \ldots + a_{n-1} + b = \begin{cases} (n+2)d_1 - (nd_2 + (n-2)d_3 + \ldots + 2d_{\frac{n}{2}+1}) & \text{if } n \text{ is even} \\ (n+1)d_1 - ((n-1)d_2 + (n-3)d_3 + \ldots + 2d_{\frac{n-1}{2}+1}) & \text{if } n \text{ is odd} \end{cases}$$

In our model, we adopt this approach for n-XOR operations for MixColumn matrix multiplication. In FUTURE cipher, the multiplication by 4×4 MDS matrix μ is performed over $GF(2^4)$, defined by the primitive polynomial x^4+x+1. Let, α be a primitive element, serving as a root of the polynomial x^4+x+1. The MDS matrix μ includes field elements **1, 2, 3, 8**, and **9** from $GF(2^4)$. To model the multiplication by μ for bit-oriented MILP, we need to convert the 4×4 MDS matrix μ over $GF(2^4)$ into a primitive representation of μ, i.e., a 16×16 binary matrix over the base field $\mathbb{F}_2$. Using linear maps with matrix representation, the authors [26] express the corresponding 4×4 binary matrices of these field elements in Figure 5.

$$\mathbf{1} = \begin{bmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix}, \quad \mathbf{2} = \begin{bmatrix} 0&1&0&0 \\ 0&0&1&0 \\ 1&0&0&1 \\ 1&0&0&0 \end{bmatrix}, \quad \mathbf{3} = \begin{bmatrix} 1&1&0&0 \\ 0&1&1&0 \\ 1&0&1&1 \\ 1&0&0&1 \end{bmatrix}, \quad \mathbf{8} = \begin{bmatrix} 1&0&0&1 \\ 1&1&0&0 \\ 0&1&1&0 \\ 0&0&1&0 \end{bmatrix}, \quad \mathbf{9} = \begin{bmatrix} 1&1&0&0 \\ 0&1&1&0 \\ 1&0&1&1 \\ 1&0&0&1 \end{bmatrix}$$

Fig. 5: 4×4 binary matrix representation of the field elements in μ

Note that, the primitive representation of μ by replacing the corresponding field elements **1, 2, 3, 8**, and **9** is compatible with cipher representation from MSB to LSB. However, this primitive representation would not be compatible with the cipher representation from LSB to MSB. Here we will describe how to construct the primitive representation of μ using a companion matrix to model the cipher which would be compatible in both ways. We know that $\mathbf{2} = 0010 \in GF(2^4)$ is the root α of the primitive polynomial x^4+x+1 over $GF(2^4)$. Let us assume that, the state of the cipher represented from MSB to LSB, i.e., $S = s_{63}||s_{62}||\ldots||s_0$. In this case, the companion matrix representation of α of the monic primitive polynomial $c_0+c_1x+c_2x^2+c_3x^3+x^4, c_i \in \mathbb{F}_2$ can be written as

$$2 = \alpha = \begin{bmatrix} c_3&1&0&0 \\ c_2&0&1&0 \\ c_1&0&0&1 \\ c_0&0&0&0 \end{bmatrix} = \begin{bmatrix} 0&1&0&0 \\ 0&0&1&0 \\ 1&0&0&1 \\ 1&0&0&0 \end{bmatrix} \text{ with } 1 = \alpha^0 = \begin{bmatrix} 1&0&0&0 \\ 0&1&0&0 \\ 0&0&1&0 \\ 0&0&0&1 \end{bmatrix}.$$

The other field elements of μ can be computed as

$$3 = \alpha + 1 = \begin{bmatrix} 1\,1\,0\,0 \\ 0\,1\,1\,0 \\ 1\,0\,1\,1 \\ 1\,0\,0\,1 \end{bmatrix}, 8 = \alpha^3 = \begin{bmatrix} 1\,0\,0\,1 \\ 1\,1\,0\,0 \\ 0\,1\,1\,0 \\ 0\,0\,1\,0 \end{bmatrix}, 9 = \alpha^3 + 1 = \begin{bmatrix} 1\,1\,0\,0 \\ 0\,1\,1\,0 \\ 1\,0\,1\,1 \\ 1\,0\,0\,1 \end{bmatrix}.$$

Thus the 16×16 binary matrix M_1, representing the primitive form of μ, corresponds to the cipher's bit representation from MSB to LSB over $\mathbb{F}_2$ is given in Figure 12b (in Appendix A). On the other hand, if the cipher is represented from LSB to MSB, i.e., $S = s_0||s_1||\ldots||s_{63}$, then the companion matrix representation of α of the monic primitive polynomial $c_0 + c_1x + c_2x^2 + c_3x^3 + x^4, c_i \in \mathbb{F}_2$ can be written as

$$2 = \alpha = \begin{bmatrix} 0\,0\,0\,c_0 \\ 1\,0\,0\,c_1 \\ 0\,1\,0\,c_2 \\ 0\,0\,1\,c_3 \end{bmatrix} = \begin{bmatrix} 0\,0\,0\,1 \\ 1\,0\,0\,1 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \end{bmatrix} \text{ with } 1 = \alpha^0 = \begin{bmatrix} 1\,0\,0\,0 \\ 0\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \end{bmatrix}.$$

Similarly, the other field elements of μ can be computed as

$$3 = \alpha + 1 = \begin{bmatrix} 1\,0\,0\,1 \\ 1\,1\,0\,1 \\ 0\,1\,1\,0 \\ 0\,0\,1\,1 \end{bmatrix}, 8 = \alpha^3 = \begin{bmatrix} 0\,1\,0\,0 \\ 0\,1\,1\,0 \\ 0\,0\,1\,1 \\ 1\,0\,0\,1 \end{bmatrix}, 9 = \alpha^3 + 1 = \begin{bmatrix} 1\,1\,0\,0 \\ 0\,0\,1\,0 \\ 0\,0\,0\,1 \\ 1\,0\,0\,0 \end{bmatrix}.$$

Thus the 16×16 binary matrix M_2, serving as the primitive representation of μ, corresponds to the bit order from LSB to MSB over $\mathbb{F}_2$ is given in Figure 12b (in Appendix A). Apart from these two companion matrix representations, using any other form of companion matrix in the model either by transposing it or by reordering the rows/columns of the above two matrices would not be compatible with the cipher representation. This is because altering the companion matrix used to construct the primitive representation M would change the bit sequences. As a result, multiplying M (a 16×16 matrix) by the state (a 16×4 matrix) would not produce a correct state consistent with the cipher's structure.

Finally, the 4×4 state matrix of FUTURE cipher can be further deduced to 16×4 binary matrix. Let, the 16-bit column vectors as $y = (y_0, y_1, \cdots, y_{15})^T$ and $t = (t_0, t_1, \cdots, t_{15})^T$, where $t = M \cdot y$. The 16 constraints corresponding to one column transformation of the state after the MixColumn operation are given in Figure 14 (see Appendix A). Therefore, for all four columns of the state, a total of $16 \cdot 4 = 64$ constraints are required to represent the differential propagation through the MixColumn operation.

4.3 Constraints for ShiftRow Operation

The ShiftRow operation performs a row-wise shift at the nibble level, which can be represented as a bit-wise permutation $\pi : \{0,1\}^{64} \rightarrow \{0,1\}^{64}$. To model this

operation, the binary variables resulting from the MixColumn step are permuted by ShiftRow. After that, 64 new binary variables are introduced and assigned to these permuted values. If x_i and y_i represent the input and output binary variables respectively, the constraint $y_i = \pi(x_i)$ is added to the MILP model. To reduce the number of constraints, the output binary variables can be directly permuted according to the ShiftRow bit-wise permutation π while modeling the MixColumn operation.

4.4 Constraints for AddRoundKey Operation

The AddRoundKey operation directly XORs the state bits with the round keys. In the bit-oriented related-key model, the state difference is XORed directly with the sub-key difference. To model the XOR operation between the key and state differences ($c = a \oplus b$), we use the following constraints without introducing dummy variables: $c \geq a - b,\ c \geq b - a,\ c \leq a + b,\ c \leq 2 - a - b$.

4.5 Construction of the Objective Function

The objective function of a MILP model can be designed to minimize the number of active S-boxes. In a bit-oriented MILP model, there will be no inconsistencies in the propagation of bit differences through rounds, provided the S-box constraints accurately represent its DDT. To account for an active S-box in a bit-based model, we introduce a dummy variable along with four additional constraints for each S-box. Let the input bit differences of an S-box be represented by $(\delta x_3, \delta x_2, \delta x_1, \delta x_0)$ and define a new binary dummy variable, d_0. This dummy variable d_0 will determine whether the S-box is active or inactive based on the following constraints: $\sum_{i=0}^{n-1} \delta x_i \geq d_0, d_0 \geq \delta x_i, i = 0, 1, 2, 3$. The objective function is then to minimize the sum of the dummy variables d_i for each S-box position in the rounds. To calculate the probability of the differential trail produced by the model, the probability of each active S-box from the DDT must be checked, and the overall probability of the differential characteristic is obtained by multiplying these values. For a clearer understanding of the MILP model applied to the FUTURE cipher, we provide our MILP model implementation in [2].

5 Results

This section presents an analysis of the differential characteristics of FUTURE in the related-key attack setting. The differential characteristics are determined using the methodology in Section 4.

5.1 Related-Key Differential Distinguishers

To search for differential characteristics of FUTURE in the related-key setting, we constructed a MILP model using the Gurobi Python API [1]. The necessary

constraints for building the model across rounds are outlined in Section 4. A summary of the related-key characteristics for different rounds, along with their probabilities, is presented in Table 2. This model enables us to search for related-key differential characteristics up to 7 rounds. However, due to the large number of constraints and variables, the model struggles to complete the search for 6 and 7 rounds. For the 7-round case, we identified several differential characteristics with 22 active S-boxes and a probability of approximately 2^{-48}. The 7-round differential characteristic is shown in Figure 16 in Appendix. To confirm the individual probabilities for each S-box, the DDT, and inverse DDT are provided in Table 4a and Table 4b, respectively. Additionally, we identified three distinct clustering effects for the 7-round differential characteristic (see Table 2) from 50 different solutions generated by the model, where the characteristics share the same input and output. This clustering further increases the probability of the differential characteristic to $3 \cdot 2^{-48} \approx 2^{-46.4}$. Furthermore, for 8 rounds, the solver could not reach a near-optimal solution due to the large number of constraints and variables. Therefore, we extended the 7-round differential characteristic by adding an additional round. Using the MILP model, we verified that seven S-boxes are active in the final round, with a probability of 2^{-17}. Consequently, the overall probability for the 8-round differential characteristic (see Figure 6) becomes $2^{-46.4} \cdot 2^{-17} = 2^{-63.4}$. For the distinguishing attack, the number of required plaintext pairs should be more than $2^{63.4}$ to ensure a high enough success probability for the distinguisher. Since the attacker only has 2^{63} plaintext pairs with a fixed input difference, an alternative strategy is to consider truncated differences in the last round to increase the overall trail probability.

In Figure 6, the S-box input differences at the last round are $1, f, 3, 9, a, 2$. According to the DDT (Table 4), the input difference 1 transitions to output differences 2 or 3 with probability 2^{-1}, i.e., $\Pr[1 \xrightarrow{\text{S-box}} 001*] = 2^{-1}$. Similarly, the probabilities for other differential transitions $2 \rightarrow 00*1$, $3 \rightarrow 1*10$, $9 \rightarrow 111*$, $a \rightarrow 100*$, and $f \rightarrow 101*$ are each improved approximately by a factor of 2. Thus, by considering truncated differentials, the improved probability for the 8-round related-key differential becomes $2^{-56.4}$, which is an improvement approximately by a factor of 2^7.

This can be directly leveraged to mount an attack on the security notion of indistinguishability against FUTURE reduced to 8 rounds. The attack procedure is detailed in Algorithm 2 (see Appendix A). In this distinguisher, the attacker requires 2^{57} plaintext pairs, thus 2^{58} data complexity, effectively exhausting the entire plaintext space. The offline time complexity amounts to 2^{58} XOR operations. The attack does not necessitate storing intermediate values, except for one ciphertext when $C_i \oplus C_i^{'} = \Delta C$ is satisfied. Therefore, the memory complexity is minimal, or effectively negligible. Furthermore, the attacker can reduce the data complexity by restricting the input space along with its associated probability. Additionally, to further minimize the data complexity for the boomerang attack, an effective strategy is to utilize structured input sets. However, in this case, the memory complexity will increase.

Experimental Verification. As previously mentioned, the bit-oriented MILP model guarantees no inconsistencies in the solutions it returns. redDuring our experiments, we evaluated differential characteristics using automated tools. We were able to successfully verify several characteristics whose probabilities are higher than 2^{-32}, confirming their practical relevance and effectiveness. The implementation used to verify these characteristics is available in [2].

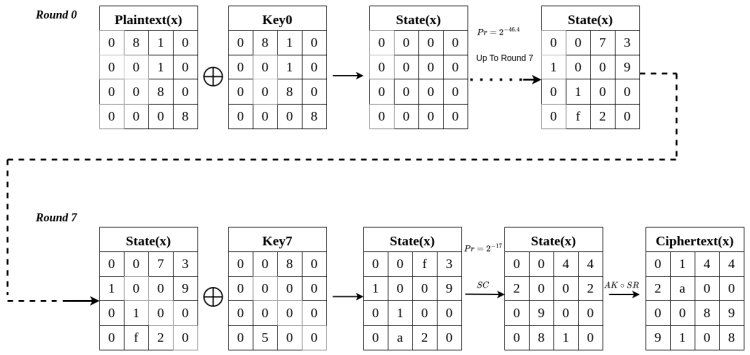


Fig. 6: Eight Round Related-Key Differential Characteristic of FUTURE Cipher

#Rounds	#Active S-box	Differential		Probability
		Input Differences	Output Difference	
4	2	ΔP = 0x2300 0010 0001 0000 ΔK_0 = 0x2300 0010 0001 0000 ΔK_1 = 0x0004 0000 0000 0000	ΔC = 0x0000 4000 0440 008c	2^{-5}
5	6	ΔP = 0x1201 01c0 0000 0000 ΔK_0 = 0x0000 01c0 0000 0000 ΔK_1 = 0x0000 0200 0020 0002	ΔC = 0x0008 0000 8000 0802	2^{-14}
6	11	ΔP = 0xc840 0000 0000 0005 ΔK_0 = 0xc840 0000 0000 0005 ΔK_1 = 0x0000 0000 0000 2480	ΔC = 0x0200 0010 0001 d420	2^{-27}
7	22	ΔP = 0x0000 8000 1180 0008 ΔK_0 = 0x0000 8000 1180 0008 ΔK_1 = 0x0000 0000 000b 0000	ΔC = 0x0000 0007 1002 0000	2^{-48}
8	29	ΔP = 0x0000 8000 1180 0008 ΔK_0 = 0x0000 8000 1180 0008 ΔK_1 = 0x0000 0000 000b 0000	ΔC = 0x0144 2a00 0089 9108	$2^{-63.4}$

Table 2: Related-Key Differentials for Different Rounds of FUTURE using Bit-Oriented MILP Model

5.2 Related-Key Boomerang Distinguisher

In this section, we construct boomerang distinguishers for FUTURE over different rounds. Using our automated search model, we identify two distinct related-key differential characteristics for five rounds each, corresponding to the upper and lower halves of the boomerang. These characteristics have probabilities of 2^{-14} and 2^{-16}, respectively. For clarity, let $\Delta_0 \xrightarrow{Upper\ Trail} \Delta_1$ and

$\nabla_0 \xrightarrow{Lower\ Trail} \nabla_1$ denote the differential characteristics for the upper and lower five rounds of the full boomerang, respectively. Additionally, let α and β represent the differences in the round keys of the upper and lower trails. The full round boomerang structure is illustrated in Figure 7. Thus, the distinguishing probability for this boomerang is given by $(2^{-14})^2 \cdot (2^{-16})^2 = 2^{-60}$, which can be utilized to perform a distinguishing attack on the full-round FUTURE cipher under adaptively chosen plaintext and ciphertext (ACPC) settings. The detailed attack procedure is presented in Algorithm 3 (see Appendix A). In this distinguisher, the attacker needs 2^{60} plaintext-ciphertext quartets, which corresponds to 2^{62} data in total. The offline time complexity is $4 \cdot 2^{60} = 2^{62}$ XOR operations. The attack does not require storing intermediate values, except for one plaintext P_i when $P_2^i \oplus P_3^i = \Delta_0$ is satisfied. As a result, the memory complexity is negligible.

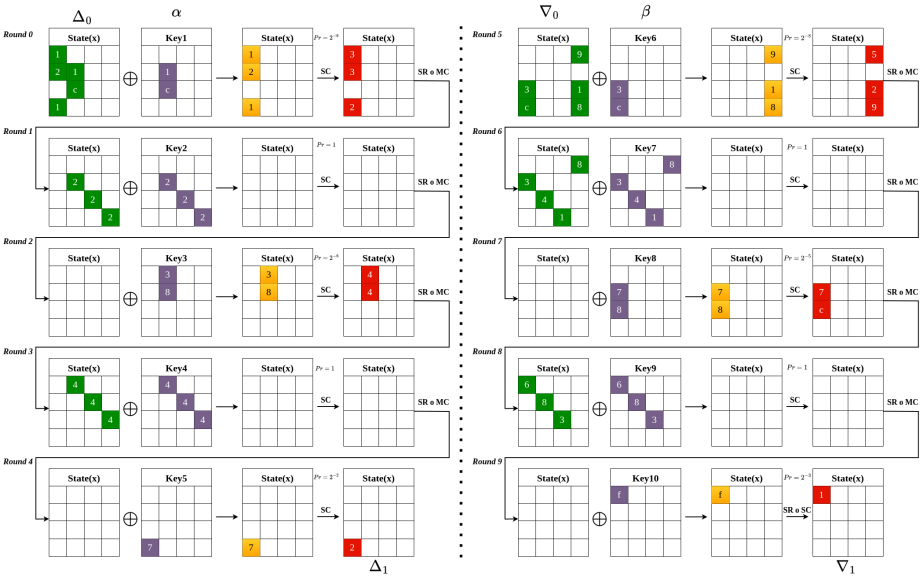


Fig. 7: Full Round Boomerang Distinguisher

Checking Incompatibilities in the Boomerang. In boomerang-style attacks, selecting compatible differential characteristics for E_0 and E_1 is crucial, as independent choices can lead to incompatibility and reduce the probability of generating a right quartet to zero. Murphy [32] highlighted that dependencies between characteristics can benefit attackers. Biryukov *et al.* introduced the middle-round S-box trick [14], and later Biryukov and Khovratovich [15] proposed techniques like the ladder and S-box switch to improve probabilities.

These ideas were formalized by Dunkelman et al. as the sandwich attack [22], which divides the cipher into three parts, enhancing the overall probability. Further, to evaluate the middle part efficiently and systematically, the authors [20] introduced a boomerang connectivity table (BCT) for a single round.

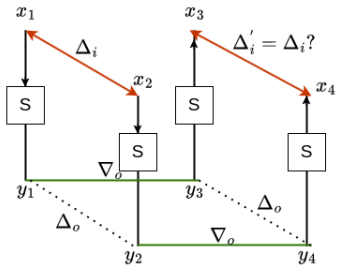


Fig. 8: Single S-box Layer in the Middle of the Boomerang

Suppose that the middle layer at the fourth round of the given boomerang (Figure 7) is composed of 16 S-box layers independently. For more clarity, we only chose one S-box layer which is depicted in Figure 8. According to Figure 8, the BCT [20] is defined in the following way.

$$\mathrm{BCT}(\Delta_i, \nabla_o) = \{x \in \{0,1\}^4 : S^{-1}(S(x) \oplus \nabla_o) \oplus S^{-1}(S(x \oplus \Delta_0) \oplus \nabla_o) == \Delta_0\}.$$

This BCT provides a unified representation of existing observations on checking the inconsistency as well as the dependencies to further increase the probability of the boomerang for a single round.

Incompatibility. Incompatibility occurs when, as shown in Figure 8, the boomerang connection table (BCT) entry $\mathrm{BCT}(\Delta_i, \nabla_o) = 0$, meaning the boomerang cannot be formed. If $\mathrm{BCT}(\Delta_i, \nabla_o) \neq 0$, the differential characteristics are compatible to form the quartet for the boomerang.
Ladder Switch. The ladder switch, introduced in [15], occurs when $\Delta_i \neq 0$ and $\nabla_o = 0$, resulting in $\mathrm{BCT}(\Delta_i, \nabla_o) = 2^4$, i.e., $\Pr[\Delta_i^{'} = \Delta_i] = 1$. Geometrically, when $\nabla_o = 0$, the upper planes coincide, and the input pairs (x_3, x_4) on the opposite plane are directly replaced by (x_1, x_2). In a similar fashion, if $\Delta_i = 0$ and $\nabla_o \neq 0$, the lower planes coincide, and the input pairs (y_1, y_3) on the opposite plane are directly replaced by (y_2, y_4).
S-box Switch. The S-box switch, introduced in [15], occurs when $\mathrm{DDT}(\Delta_i, \Delta_o) \neq 0$ and $\Delta_o = \nabla_o$, resulting in $\mathrm{BCT}(\Delta_i, \nabla_o) = \mathrm{DDT}(\Delta_i, \Delta_o)$, i.e., $\Pr[\Delta_i^{'} = \Delta_i] = \frac{\mathrm{DDT}(\Delta_i, \Delta_o)}{2^4}$. Geometrically, when $\Delta_o = \nabla_o$, the upper planes interchange their input pairs, i.e., the input pairs (x_3, x_4) on the opposite plane are directly replaced by (x_2, x_1), consistent with $\mathrm{DDT}(\Delta_i, \Delta_o)$.

Based on the switch techniques and the BCT, we verified the compatibility of the full round boomerang distinguisher shown in Figure 7. In this distinguisher,

the S-box layer in the fifth round (Round 4) is chosen as the middle layer. We examine the state difference at the Round 4 S-box layer for the upper differential trail and the state difference at the Round 5 S-box layer for the lower trail. This setup is illustrated in Figure 15 (see Appendix A). As shown, only the third S-box is active in the upper trail, while all S-box nibbles are active in the lower trail. Consequently, all nibbles except the third in the middle layer fall under the ladder switch category, resulting in a probability of 1. For the third nibble position, $\Delta_i = 0x07$ and $\nabla_o = 0x0c$, where we confirmed that $\mathrm{BCT}(0x07, 0x0c) = 2$, validating the compatibility of our differential characteristics to form the full-round boomerang distinguisher.

Refinements to the Boomerang Distinguisher. In the previous paragraph, we demonstrated the compatibility of the two differential characteristics necessary to form a full-round boomerang using middle-round switch effects. Now, we will delve into a more detailed analysis of how these switching effects can be leveraged to significantly enhance the boomerang probability. As shown in Figure 15, there are three active S-boxes at positions 12, 14, and 15 in the lower trail during round 5. Tracing this lower trail backward, the first column ($(0, 0, 0, 8)^T$) contains a single active nibble, $0x08$, at the third position following the inverse ShiftRow operation. This nibble difference, $0x08$, arises from the difference $0x09$ after the inverse S-box operation in round 5. Notably, the other two active S-boxes in round 5 do not impact the first column after the inverse ShiftRow and MixColumn operations, as depicted in Figure 9.

According to Figure 9, if we chose all possible differences δ from $0x9$ through S-box inverse, i.e., $0x9 \xrightarrow{inverse\ \mathrm{DDT}} \{0x1, 0x7, 0x8, 0xa, 0xc, 0xe\}$ (see Table 4b), we get different $\delta_3 (= \eta_3) \in \{0xc, 0x8, 0xd, 0xf, 0xa, 0x9\}$ through inverse ShiftRow and MixColumn operations. Finally, we checked that if $\delta_3 (= \eta_3) \in \{0xc, 0xa, 0x9\}$, then $\mathrm{BCT}(\zeta (= 0x7), \eta) \neq 0$. This demonstrates that the two differential characteristics are compatible for forming the quartet in the boomerang if the output differences are $\{0x8, 0xc, 0xe\}$ from the input difference $0x09$ through the inverse S-box operation at round 5 (at position 15 in the lower half). This increases the probability from 2^{-3} ($= \Pr[0x9 \xrightarrow{S^{-1}} 0x8]$) to 3/8. Furthermore, any possible output differences from the input differences $0x05$ and $0x02$ (i.e., $\Pr[0x5 \xrightarrow{S^{-1}} *] = 1, \Pr[0x2 \xrightarrow{S^{-1}} *] = 1$) do not affect the first column after the ShiftRow and MixColumn inverse operations, increasing the probability from 2^{-5} to 1. Similarly, the output difference corresponding to the active S-box at round 4 for the upper half can be arbitrary, i.e., $0x7 \xrightarrow{\mathrm{DDT}} *$. This also increases the probability from 2^{-2} to 1. As a result, for the one lower half, the probability improves by a factor of $2^5 \cdot 3$. Thus, for the two parallel lower halves, the probability improves by a factor of $(2^{6.6})^2 = 2^{13.2}$. For upper halves, the probability improves by a factor of $(2^2)^2 = 2^4$. Additionally, we account for the probability that $\mathrm{BCT}(7, \delta_3) \neq 0$ for the middle-round switch at the round 4 S-box operation. Since $\delta_3 \in \{0xc, 0xa, 0x9\}$, the probability of $\mathrm{BCT}(7, \delta_3) \neq 0$ is lower bounded

by the minimum of their respective probabilities, i.e.,

$$\Pr[\mathrm{BCT}(7,\delta_3) \neq 0] \geq \frac{\min\{\mathrm{BCT}(7,0xc), \mathrm{BCT}(7,0xa), \mathrm{BCT}(7,0x9)\}}{2^4}$$
$$= \frac{\min\{2,4,4\}}{2^4} = 2^{-3},$$

where $\mathrm{BCT}(7,0xc) = 2, \mathrm{BCT}(7,0xa) = 4$, and $\mathrm{BCT}(7,0xc) = 4$. Finally, the refined probability for the boomerang becomes $2^4 \cdot 2^{13.2} \cdot 2^{-60} \cdot 2^{-3} = 2^{-45.8}$. This scenario can be further mapped to a Sandwich attack ($E = E_1 \circ E_m \circ E_0$) with probability $\bar{p}^2 \cdot r \cdot \bar{q}^2$, where $\bar{p} = 2^{-12.4}$, $r = 2^{-3}$, and $\bar{q} = 2^{-9}$. As a result, the data, time, and memory complexities of the distinguishing attack are reduced to $2^{48}(= 4 \cdot 2^{46})$ plaintexts, 2^{48} XOR operations, and negligible memory, respectively.

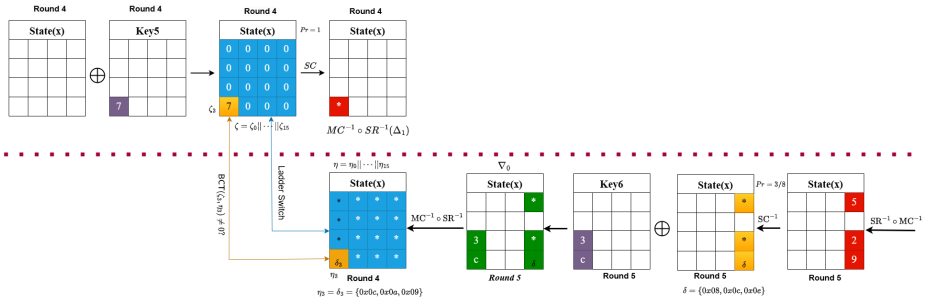


Fig. 9: Middle Round Switching Effects using Truncated Differences

Experimental Verification. For this boomerang distinguisher, we have experimentally verified both the upper and lower differential characteristics along with their corresponding probabilities. The implementation used for verification is provided in [2].

5.3 Key Recovery Attack

In this section, we present the full-round boomerang key recovery attack. First, we demonstrate how four key nibbles can be recovered using the full-round boomerang distinguisher. Next, we select an 8-round boomerang distinguisher and extend the lower part of its differential trail by two additional rounds to facilitate key recovery.

Key Recovery from the Full Round Distinguisher. Recalling the full-round boomerang distinguisher (see Figure 7) described in the previous section, the data, time, and memory complexities to satisfy the boomerang trail are 2^{48}, 2^{48}, and a negligible amount of memory, respectively. The idea is that

once an attacker obtains the satisfying quartets of plaintexts and ciphertexts as (P_0, P_1, P_2, P_3) and (C_0, C_1, C_2, C_3), they can immediately filter the active S-box differentials from the first and last rounds of the upper and lower parts of the differential trails. According to Figure 7, three active S-box differentials (at positions 0, 4, and 12) in the first round of both the upper differential trails can be directly filtered, and the respective key nibbles $K_0[0]$, $K_0[4]$, and $K_0[8]$ are uniquely retrieved from the plaintext nibble pairs $(P_0[0], P_2[0])$, $(P_0[4], P_2[4])$, and $(P_0[8], P_2[8])$. Similarly, from the last round of the lower differential trail, an attacker can filter the 0^{th} S-box differential and retrieve the key nibble $K_{10}[0]$ from $C_0[0]$ and $C_2[0]$.

The remaining task is a straightforward exhaustive search over the remaining 28 nibbles, requiring 2^{112} computations. Thus, the data, time, and memory complexities for the full-round key recovery are 2^{48}, 2^{112}, and a negligible amount of memory, respectively.

Key Recovery from an 8-Round Distinguisher. In this attack, we choose an 8-round boomerang distinguisher with four round upper and lower differential trails with probabilities 2^{-4} each. Thus, the probability of the boomerang distinguisher would be 2^{-16}. The 8-round boomerang distinguisher is depicted in Figure 13 (see Appendix A).

For this full round related-key boomerang key recovery attack, as described in Section 3, we treat the cipher as $E = E_f \circ E_d$, where E_d is an 8-round distinguisher of probability $p^2 = 2^{-16}$, and the number of rounds in E_f is 2. This is depicted in Figure 7. In this case, the other parameters are $n = 128, k = 128, m_f = 24 \times 4 = 96, r_f = 15 \times 4 = 60$.

The best guessing parameters are $m_f^{'} = 8 \times 4 = 32, m_f^* = 16 = 64, r_f^{'} = 4 \times 4 = 16$, and $r_f^* = 11 \times 4 = 44$. Therefore, using the unified and generic boomerang key recovery attack (described in Section 3.3), the complexities will be as follows.

- The data complexity is $D^{'} = \frac{4 \cdot s}{p^2} = s \cdot 2^{18}$.
- The memory complexity is $M = D^{'} + D + 2^{t+m_f^*} = s \cdot 2^{18} + s \cdot 2^{16} + 2^{64+t}$.
- For time complexity:
  - $T_1 = 2^{m_f^{'}} \cdot D^{'} = s \cdot 2^{18} \cdot 2^{32} = s \cdot 2^{50}$.
  - $T_2 = 2^{m_f^{'}} \cdot D = s \cdot 2^{48}$.
  - $T_3 = 2^{m_f^{'}} \cdot D \cdot 2^{2 \cdot r_f^* - n - 1} \cdot \epsilon = 2^{32} \cdot s \cdot 2^{16} \cdot 2^{88-64-1} \cdot \epsilon = s \cdot 2^{71} \cdot \epsilon$.
  - $T_4 = 2^{128-h}$, where $h \leq t + m_f^*$.

Note that ϵ represents the number of partial decryption, which is around $2/10 = 2^{-2.25}$ encryptions. Thus, if we set $s = a, h = 60, t = 0$, then the data, time, and memory complexities will be $2^{18}, 2^{70}$, and 2^{64} respectively.

6 Conclusion and Future Works

In this work, we present a comprehensive implementation of bit-oriented MILP models for the FUTURE lightweight block cipher in related-key settings. This

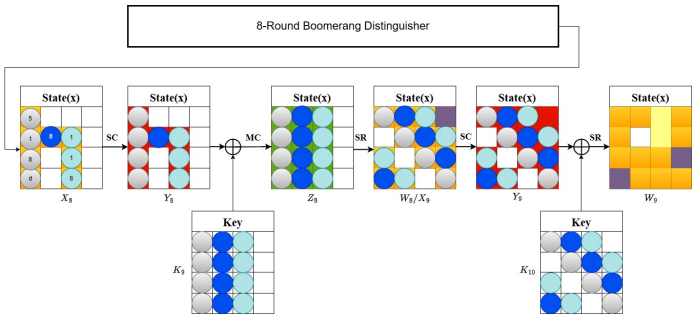


Fig. 10: The Full Round Boomerang Attack of FUTURE

approach can be extended to model MDS (or near-MDS) based SPN ciphers in the future. Utilizing this model, we explored related-key differential characteristics across different rounds, identifying a seven-round differential characteristic with a probability of $2^{-46.4}$. We further extended this characteristic by adding an extra round, providing a distinguisher with data complexity of 2^{58}, time complexity of 2^{58} XOR operations, and negligible memory requirements. Additionally, we developed a full-round boomerang distinguisher with a probability of 2^{-60} based on the round-reduced differential characteristics. By applying a one-round middle switch effect, we refined the boomerang probability from 2^{-60} to $2^{-45.8}$. Consequently, the complexities of the attack are improved to 2^{48} plaintexts, 2^{48} XOR operations, and negligible memory.Furthermore, we present a key recovery attack with data, time, and memory complexities of 2^{18}, 2^{70}, and 2^{64}, respectively.

In future work, it would be valuable to explore optimizing the probability of the distinguisher, rather than focusing solely on the number of active S-boxes. This could potentially enhance the overall probability of the distinguisher. Additionally, recent advancements in automated tools for cryptanalysis present an opportunity to develop a tool for conducting truncated differential and sandwich attacks, capturing more dependencies in the middle rounds, and further improving the probabilities of differential and boomerang distinguishers. Lastly, another interesting direction for future research would be to propose an efficient key recovery attack based on the distinguishers presented in this work.

References

1. Linear Programming Formulation With Gurobi Python API. `https://www.gurobi.com/resources/ch4-linear-programming-with-python`
2. Unoptimized MILP Codes and Their Verifications using C. `https://drive.google.com/drive/folders/1e5wZ6wy5xd8AZUV1v7E2PtBUzAg5c9_1`
3. Avanzi, R.: The QARMA block cipher family. almost MDS matrices over rings with zero divisors, nearly symmetric even-mansour constructions with non-involutory central rounds, and search heuristics for low-latency s-boxes. IACR Trans. Sym-

metric Cryptol. **2017**(1), 4–44 (2017). https://doi.org/10.13154/TOSC.V2017.I1.4-44

4. Banik, S., Bogdanov, A., Isobe, T., Shibutani, K., Hiwatari, H., Akishita, T., Regazzoni, F.: Midori: A block cipher for low energy. In: Iwata, T., Cheon, J.H. (eds.) Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9453, pp. 411–436. Springer (2015). https://doi.org/10.1007/978-3-662-48800-3_17, https://doi.org/10.1007/978-3-662-48800-3_17
5. Banik, S., Pandey, S.K., Peyrin, T., Sasaki, Y., Sim, S.M., Todo, Y.: GIFT: A small present - towards reaching the limit of lightweight encryption. In: Fischer, W., Homma, N. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2017 - 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings. Lecture Notes in Computer Science, vol. 10529, pp. 321–345. Springer (2017). https://doi.org/10.1007/978-3-319-66787-4_16, https://doi.org/10.1007/978-3-319-66787-4_16
6. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The SIMON and SPECK lightweight block ciphers. In: Proceedings of the 52nd Annual Design Automation Conference, San Francisco, CA, USA, June 7-11, 2015. pp. 175:1–175:6. ACM (2015). https://doi.org/10.1145/2744769.2747946, https://doi.org/10.1145/2744769.2747946
7. Beierle, C., Jean, J., Kölbl, S., Leander, G., Moradi, A., Peyrin, T., Sasaki, Y., Sasdrich, P., Sim, S.M.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) Advances in Cryptology - CRYPTO 2016 - 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part II. Lecture Notes in Computer Science, vol. 9815, pp. 123–153. Springer (2016). https://doi.org/10.1007/978-3-662-53008-5_5, https://doi.org/10.1007/978-3-662-53008-5_5
8. Beierle, C., Leander, G., Moradi, A., Rasoolzadeh, S.: CRAFT: lightweight tweakable block cipher with efficient protection against DFA attacks. IACR Trans. Symmetric Cryptol. **2019**(1), 5–45 (2019). https://doi.org/10.13154/TOSC.V2019.I1.5-45
9. Biham, E.: New types of cryptanalytic attacks using related keys. J. Cryptol. **7**(4), 229–246 (1994). https://doi.org/10.1007/BF00203965, https://doi.org/10.1007/BF00203965
10. Biham, E., Dunkelman, O., Keller, N.: The rectangle attack - rectangling the serpent. In: Pfitzmann, B. (ed.) Advances in Cryptology - EUROCRYPT 2001, International Conference on the Theory and Application of Cryptographic Techniques, Innsbruck, Austria, May 6-10, 2001, Proceeding. Lecture Notes in Computer Science, vol. 2045, pp. 340–357. Springer (2001). https://doi.org/10.1007/3-540-44987-6_21, https://doi.org/10.1007/3-540-44987-6_21
11. Biham, E., Dunkelman, O., Keller, N.: New results on boomerang and rectangle attacks. In: Daemen, J., Rijmen, V. (eds.) Fast Software Encryption, 9th International Workshop, FSE 2002, Leuven, Belgium, February 4-6, 2002, Revised Papers. Lecture Notes in Computer Science, vol. 2365, pp. 1–16. Springer (2002). https://doi.org/10.1007/3-540-45661-9_1, https://doi.org/10.1007/3-540-45661-9_1
12. Biham, E., Dunkelman, O., Keller, N.: Related-key boomerang and rectangle attacks. In: Cramer, R. (ed.) Advances in Cryptology - EUROCRYPT 2005, 24th

Annual International Conference on the Theory and Applications of Cryptographic Techniques, Aarhus, Denmark, May 22-26, 2005, Proceedings. Lecture Notes in Computer Science, vol. 3494, pp. 507–525. Springer (2005). https://doi.org/10.1007/11426639_30, https://doi.org/10.1007/11426639_30
13. Biham, E., Shamir, A.: Differential cryptanalysis of des-like cryptosystems. In: Menezes, A., Vanstone, S.A. (eds.) Advances in Cryptology - CRYPTO '90, 10th Annual International Cryptology Conference, Santa Barbara, California, USA, August 11-15, 1990, Proceedings. Lecture Notes in Computer Science, vol. 537, pp. 2–21. Springer (1990). https://doi.org/10.1007/3-540-38424-3_1, https://doi.org/10.1007/3-540-38424-3_1
14. Biryukov, A., Cannière, C.D., Dellkrantz, G.: Cryptanalysis of SAFER++. In: Boneh, D. (ed.) Advances in Cryptology - CRYPTO 2003, 23rd Annual International Cryptology Conference, Santa Barbara, California, USA, August 17-21, 2003, Proceedings. Lecture Notes in Computer Science, vol. 2729, pp. 195–211. Springer (2003). https://doi.org/10.1007/978-3-540-45146-4_12, https://doi.org/10.1007/978-3-540-45146-4_12
15. Biryukov, A., Khovratovich, D.: Related-key cryptanalysis of the full AES-192 and AES-256. In: Matsui, M. (ed.) Advances in Cryptology - ASIACRYPT 2009, 15th International Conference on the Theory and Application of Cryptology and Information Security, Tokyo, Japan, December 6-10, 2009. Proceedings. Lecture Notes in Computer Science, vol. 5912, pp. 1–18. Springer (2009). https://doi.org/10.1007/978-3-642-10366-7_1, https://doi.org/10.1007/978-3-642-10366-7_1
16. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J.B., Seurin, Y., Vikkelsoe, C.: PRESENT: an ultra-lightweight block cipher. In: Paillier, P., Verbauwhede, I. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2007, 9th International Workshop, Vienna, Austria, September 10-13, 2007, Proceedings. Lecture Notes in Computer Science, vol. 4727, pp. 450–466. Springer (2007). https://doi.org/10.1007/978-3-540-74735-2_31, https://doi.org/10.1007/978-3-540-74735-2_31
17. Borghoff, J., Canteaut, A., Güneysu, T., Kavun, E.B., Knezevic, M., Knudsen, L.R., Leander, G., Nikov, V., Paar, C., Rechberger, C., Rombouts, P., Thomsen, S.S., Yalçin, T.: PRINCE - A low-latency block cipher for pervasive computing applications - extended abstract. In: Wang, X., Sako, K. (eds.) Advances in Cryptology - ASIACRYPT 2012 - 18th International Conference on the Theory and Application of Cryptology and Information Security, Beijing, China, December 2-6, 2012. Proceedings. Lecture Notes in Computer Science, vol. 7658, pp. 208–225. Springer (2012). https://doi.org/10.1007/978-3-642-34961-4_14, https://doi.org/10.1007/978-3-642-34961-4_14
18. Boura, C., Coggia, D.: Efficient milp modelings for sboxes and linear layers of spn ciphers. IACR Transactions on Symmetric Cryptology **2020**(3), 327–361 (2020)
19. Cannière, C.D., Dunkelman, O., Knezevic, M.: KATAN and KTANTAN - A family of small and efficient hardware-oriented block ciphers. In: Clavier, C., Gaj, K. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2009, 11th International Workshop, Lausanne, Switzerland, September 6-9, 2009, Proceedings. Lecture Notes in Computer Science, vol. 5747, pp. 272–288. Springer (2009). https://doi.org/10.1007/978-3-642-04138-9_20, https://doi.org/10.1007/978-3-642-04138-9_20
20. Cid, C., Huang, T., Peyrin, T., Sasaki, Y., Song, L.: Boomerang connectivity table: A new cryptanalysis tool. In: Nielsen, J.B., Rijmen, V. (eds.) Advances in Cryptology - EUROCRYPT 2018 - 37th Annual International Conference on the Theory

and Applications of Cryptographic Techniques, Tel Aviv, Israel, April 29 - May 3, 2018 Proceedings, Part II. Lecture Notes in Computer Science, vol. 10821, pp. 683–714. Springer (2018). https://doi.org/10.1007/978-3-319-78375-8_22, https://doi.org/10.1007/978-3-319-78375-8_22

21. Daemen, J., Rijmen, V.: The Design of Rijndael: AES - The Advanced Encryption Standard. Information Security and Cryptography, Springer (2002). https://doi.org/10.1007/978-3-662-04722-4, https://doi.org/10.1007/978-3-662-04722-4
22. Dunkelman, O., Keller, N., Shamir, A.: A practical-time related-key attack on the KASUMI cryptosystem used in GSM and 3g telephony. J. Cryptol. **27**(4), 824–849 (2014). https://doi.org/10.1007/S00145-013-9154-9
23. Fu, K., Wang, M., Guo, Y., Sun, S., Hu, L.: Milp-based automatic search algorithms for differential and linear trails for speck. In: Fast Software Encryption: 23rd International Conference, FSE 2016, Bochum, Germany, March 20-23, 2016, Revised Selected Papers 23. pp. 268–288. Springer (2016)
24. Guo, J., Peyrin, T., Poschmann, A., Robshaw, M.J.B.: The LED block cipher. In: Preneel, B., Takagi, T. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6917, pp. 326–341. Springer (2011). https://doi.org/10.1007/978-3-642-23951-9_22, https://doi.org/10.1007/978-3-642-23951-9_22
25. Gupta, K.C., Pandey, S.K., Samanta, S.: Future: a lightweight block cipher using an optimal diffusion matrix. In: International Conference on Cryptology in Africa. pp. 28–52. Springer (2022)
26. Ilter, M.B., Selçuk, A.A.: Milp-aided cryptanalysis of the future block cipher. In: International Conference on Information Technology and Communications Security. pp. 153–167. Springer (2022)
27. Kelsey, J., Kohno, T., Schneier, B.: Amplified boomerang attacks against reduced-round MARS and serpent. In: Schneier, B. (ed.) Fast Software Encryption, 7th International Workshop, FSE 2000, New York, NY, USA, April 10-12, 2000, Proceedings. Lecture Notes in Computer Science, vol. 1978, pp. 75–93. Springer (2000). https://doi.org/10.1007/3-540-44706-7_6, https://doi.org/10.1007/3-540-44706-7_6
28. Li, T., Sun, Y.: Superball: A new approach for MILP modelings of boolean functions. IACR Trans. Symmetric Cryptol. **2022**(3), 341–367 (2022). https://doi.org/10.46586/TOSC.V2022.I3.341-367
29. Lin, H., Zou, J., Li, J.: The differential meet-in-the-middle attack on FUTURE and CRAFT. In: Proceedings of the 2023 13th International Conference on Communication and Network Security, ICCNS 2023, Fuzhou, China, December 6-8, 2023. pp. 151–158. ACM (2023). https://doi.org/10.1145/3638782.3638805, https://doi.org/10.1145/3638782.3638805
30. Mondal, S.K., Rahman, M., Sarkar, S., Adhikari, A.: Yoyo cryptanalysis on future. Int. J. Appl. Cryptogr. **4**(3/4), 238–249 (2024). https://doi.org/10.1504/IJACT.2024.138453, https://doi.org/10.1504/IJACT.2024.138453
31. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Information Security and Cryptology: 7th International Conference, Inscrypt 2011, Beijing, China, November 30–December 3, 2011. Revised Selected Papers 7. pp. 57–76. Springer (2012)
32. Murphy, S.: The return of the cryptographic boomerang. IEEE Trans. Inf. Theory **57**(4), 2517–2521 (2011). https://doi.org/10.1109/TIT.2011.2111091, https://doi.org/10.1109/TIT.2011.2111091

33. National Institute of Standards and Technology: Secure Hash Standard (SHS). Federal Information Processing Standards (FIPS) Publication 180-4 (August 2015). https://doi.org/https://doi.org/10.6028/NIST.FIPS.180-4
34. Pal, D., Chandratreya, V.P., Chowdhury, D.R.: New techniques for modeling sboxes: An MILP approach. In: Deng, J., Kolesnikov, V., Schwarzmann, A.A. (eds.) Cryptology and Network Security - 22nd International Conference, CANS 2023, Augusta, GA, USA, October 31 - November 2, 2023, Proceedings. Lecture Notes in Computer Science, vol. 14342, pp. 318–340. Springer (2023). https://doi.org/10.1007/978-981-99-7563-1_15, https://doi.org/10.1007/978-981-99-7563-1_15
35. Roy, H.S., Dey, P., Mondal, S.K., Adhikari, A.: Cryptanalysis of full round FUTURE with multiple biclique structures. Peer Peer Netw. Appl. **17**(1), 397–409 (2024). https://doi.org/10.1007/S12083-023-01600-Y
36. Sasaki, Y., Todo, Y.: New algorithm for modeling s-box in milp based differential and division trail search. In: Innovative Security Solutions for Information Technology and Communications: 10th International Conference, SecITC 2017, Bucharest, Romania, June 8–9, 2017, Revised Selected Papers 10. pp. 150–165. Springer (2017)
37. Schrottenloher, A., Stevens, M.: Simplified modeling of MITM attacks for block ciphers: New (quantum) attacks. IACR Trans. Symmetric Cryptol. **2023**(3), 146–183 (2023). https://doi.org/10.46586/TOSC.V2023.I3.146-183
38. Selçuk, A.A.: On probability of success in linear and differential cryptanalysis. J. Cryptol. **21**(1), 131–147 (2008). https://doi.org/10.1007/S00145-007-9013-7
39. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: Piccolo: An ultra-lightweight blockcipher. In: Preneel, B., Takagi, T. (eds.) Cryptographic Hardware and Embedded Systems - CHES 2011 - 13th International Workshop, Nara, Japan, September 28 - October 1, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6917, pp. 342–357. Springer (2011). https://doi.org/10.1007/978-3-642-23951-9_23, https://doi.org/10.1007/978-3-642-23951-9_23
40. Song, L., Zhang, N., Yang, Q., Shi, D., Zhao, J., Hu, L., Weng, J.: Optimizing rectangle attacks: A unified and generic framework for key recovery. In: Agrawal, S., Lin, D. (eds.) Advances in Cryptology - ASIACRYPT 2022 - 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5-9, 2022, Proceedings, Part I. Lecture Notes in Computer Science, vol. 13791, pp. 410–440. Springer (2022). https://doi.org/10.1007/978-3-031-22963-3_14, https://doi.org/10.1007/978-3-031-22963-3_14
41. Sun, B., Liu, Z., Rijmen, V., Li, R., Cheng, L., Wang, Q., Alkhzaimi, H., Li, C.: Links among impossible differential, integral and zero correlation linear cryptanalysis. In: Gennaro, R., Robshaw, M. (eds.) Advances in Cryptology - CRYPTO 2015 - 35th Annual Cryptology Conference, Santa Barbara, CA, USA, August 16-20, 2015, Proceedings, Part I. Lecture Notes in Computer Science, vol. 9215, pp. 95–115. Springer (2015). https://doi.org/10.1007/978-3-662-47989-6_5, https://doi.org/10.1007/978-3-662-47989-6_5
42. Sun, L., Wang, W., Wang, M.: Milp-aided bit-based division property for primitives with non-bit-permutation linear layers. IET Inf. Secur. **14**(1), 12–20 (2020). https://doi.org/10.1049/IET-IFS.2018.5283
43. Sun, S., Hu, L., Song, L., Xie, Y., Wang, P.: Automatic security evaluation of block ciphers with s-bp structures against related-key differential attacks. In: Lin, D., Xu, S., Yung, M. (eds.) Information Security and Cryptology - 9th International Conference, Inscrypt 2013, Guangzhou, China, November 27-30, 2013, Revised

Selected Papers. Lecture Notes in Computer Science, vol. 8567, pp. 39–51. Springer (2013). https://doi.org/10.1007/978-3-319-12087-4_3, https://doi.org/10.1007/978-3-319-12087-4_3

44. Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L., Fu, K.: Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties. Cryptology ePrint Archive, Paper 2014/747 (2014), https://eprint.iacr.org/2014/747

45. Sun, S., Hu, L., Wang, P., Qiao, K., Ma, X., Song, L.: Automatic security evaluation and (related-key) differential characteristic search: application to simon, present, lblock, des (l) and other bit-oriented block ciphers. In: Advances in Cryptology–ASIACRYPT 2014: 20th International Conference on the Theory and Application of Cryptology and Information Security, Kaoshiung, Taiwan, ROC, December 7-11, 2014. Proceedings, Part I 20. pp. 158–178. Springer (2014)

46. Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: $\textnormal{\textsc{TWINE}}$: A lightweight block cipher for multiple platforms. In: Knudsen, L.R., Wu, H. (eds.) Selected Areas in Cryptography, 19th International Conference, SAC 2012, Windsor, ON, Canada, August 15-16, 2012, Revised Selected Papers. Lecture Notes in Computer Science, vol. 7707, pp. 339–354. Springer (2012). https://doi.org/10.1007/978-3-642-35999-6_22, https://doi.org/10.1007/978-3-642-35999-6_22

47. Udovenko, A.: MILP modeling of boolean functions by minimum number of inequalities. IACR Cryptol. ePrint Arch. p. 1099 (2021), https://eprint.iacr.org/2021/1099

48. Wagner, D.A.: The boomerang attack. In: Knudsen, L.R. (ed.) Fast Software Encryption, 6th International Workshop, FSE '99, Rome, Italy, March 24-26, 1999, Proceedings. Lecture Notes in Computer Science, vol. 1636, pp. 156–170. Springer (1999). https://doi.org/10.1007/3-540-48519-8_12, https://doi.org/10.1007/3-540-48519-8_12

49. Xu, Z., Cui, J., Hu, K., Wang, M.: Integral attack on the full FUTURE block cipher. IACR Cryptol. ePrint Arch. p. 549 (2024), https://eprint.iacr.org/2024/549

50. Yang, Q., Song, L., Zhang, N., Shi, D., Wang, L., Zhao, J., Hu, L., Weng, J.: Optimizing rectangle and boomerang attacks: A unified and generic framework for key recovery. J. Cryptol. **37**(2), 19 (2024). https://doi.org/10.1007/S00145-024-09499-1

51. Yin, J., Ma, C., Lyu, L., Song, J., Zeng, G., Ma, C., Wei, F.: Improved cryptanalysis of an ISO standard lightweight block cipher with refined MILP modelling. In: Chen, X., Lin, D., Yung, M. (eds.) Information Security and Cryptology - 13th International Conference, Inscrypt 2017, Xi'an, China, November 3-5, 2017, Revised Selected Papers. Lecture Notes in Computer Science, vol. 10726, pp. 404–426. Springer (2017). https://doi.org/10.1007/978-3-319-75160-3_24, https://doi.org/10.1007/978-3-319-75160-3_24

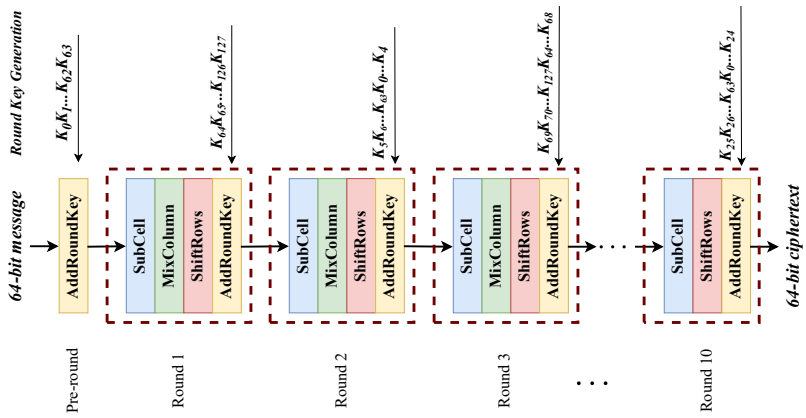


Fig. 11: FUTURE Encryption Scheme

S-box	# Inequalities		
	[18, Algorithm 1]		Revised Algorithm 1
	Claimed	Actual[3]	
PRESENT	17	22	17
PRINCE	19	24	19
MIDORI S0	16	19	17
MIDORI S1	20	23	20
KLEIN	19	22	19
PICCOLO	16	20	16
LBlock s0	17	21	17
LBlock s1	17	22	17
LBlock s2	17	22	17
LBlock s3	17	22	17
LBlock s4	17	21	17
LBlock s5	17	22	17
LBlock s6	17	22	17
LBlock s7	17	22	17
LBlock s8	17	21	17
LBlock s9	17	21	17
TWINE	19	26	20

Table 3: Number of inequality Comparison between Boura and Coggia's Algorithm and Our Revised Algorithm

Algorithm 2 Distinguishing Attack against FUTURE Reduced to 8 Rounds

1: **procedure** DISTINGUISHER((ΔP = $0x0000800011800008$,ΔSK_0 = $0x0000800011800008$, ΔSK_1 = $0x0000020000200002$) $\xrightarrow{\text{8 round differential}}$ Truncated Difference from the Last Round S-box Operation.)
2: Randomly choose a key $K = SK_0||SK_1 \xleftarrow{\$} \{0,1\}^{128}$.
3: Form another key $K^{'} = K \oplus \Delta SK_0||\Delta SK_1$. ▷ Steps 2 and 3 are chosen by the oracle.
4: Choose 2^{57} distinct plaintexts P_i, $i = 1, 2, \ldots, 2^{57}$.
5: **for** $i = 1$ to 2^{57} **do**
6: Query P_i to the encryption oracle under the key K and obtain the corresponding ciphertext $C_i = E_K(P_i)$.
7: Query $P_i^{'} = P_i \oplus \Delta P$ to the encryption oracle under the key $K^{'}$ and obtain the corresponding ciphertext $C_i^{'} = E_{K'}(P_i^{'})$.
8: **if** $C_i \oplus C_i^{'} == \Delta C$ **then**
9: Return 1 ▷ The oracle is the FUTURE reduced to 8 rounds.
10: Return 0 ▷ The oracle is a random permutation.

$$M_1 = \begin{bmatrix}
1&0&0&1&0&0&0&1&1&0&0&0&1&0&0&1\\
1&1&0&0&1&0&0&0&0&1&0&0&1&1&0&0\\
0&1&1&0&0&1&0&0&0&0&1&0&0&1&1&0\\
0&0&1&0&0&0&1&1&0&0&0&1&0&0&1&0\\
1&1&0&0&0&1&0&0&0&0&0&1&0&0&0&1\\
0&1&1&0&0&0&1&0&1&0&0&0&1&0&0&0\\
1&0&1&1&1&0&0&1&0&1&0&0&0&1&0&0\\
1&0&0&1&1&0&0&0&0&0&1&1&0&0&1&1\\
0&1&0&0&1&1&0&0&1&0&0&1&0&0&0&1\\
0&0&1&0&0&1&1&0&1&1&0&0&1&0&0&0\\
1&0&0&1&1&0&1&1&0&1&1&0&0&1&0&0\\
1&0&0&0&1&0&0&1&0&0&1&0&0&0&1&1\\
0&0&0&1&0&0&0&1&1&0&0&1&1&0&0&0\\
1&0&0&0&1&0&0&0&1&1&0&0&0&1&0&0\\
0&1&0&0&0&1&0&0&0&1&1&0&0&0&1&0\\
0&0&1&1&0&0&1&1&0&0&1&0&0&0&0&1
\end{bmatrix}$$

(a) For MSB to LSB

$$M_2 = \begin{bmatrix}
0&1&0&0&1&1&0&0&1&0&0&0&0&1&0&0\\
0&1&1&0&0&0&1&0&0&1&0&0&0&1&1&0\\
0&0&1&1&0&0&0&1&0&0&1&0&0&0&1&1\\
1&0&0&1&1&0&0&0&0&0&0&1&1&0&0&1\\
1&0&0&1&0&0&0&1&1&1&0&0&1&1&0&0\\
1&1&0&1&1&0&0&1&0&0&1&0&0&0&1&0\\
0&1&1&0&0&1&0&0&0&0&0&1&0&0&0&1\\
0&0&1&1&0&0&1&0&1&0&0&0&1&0&0&0\\
0&0&0&1&1&0&0&1&0&1&0&0&1&1&0&0\\
1&0&0&1&1&1&0&1&0&1&1&0&0&0&1&0\\
0&1&0&0&0&1&1&0&0&0&1&1&0&0&0&1\\
0&0&1&0&0&0&1&1&1&0&0&1&1&0&0&0\\
1&1&0&0&1&1&0&0&0&1&0&0&1&0&0&0\\
0&0&1&0&0&0&1&0&0&1&1&0&0&1&0&0\\
0&0&0&1&0&0&0&1&0&0&1&1&0&0&1&0\\
1&0&0&0&1&0&0&0&1&0&0&1&0&0&0&1
\end{bmatrix}$$

(b) For LSB to MSB

Fig. 12: The Primitive Representation of μ When the State is Represented from LSB to MSB or from MSB to LSB

A Additional Details

I\O	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	4	0	0	0	0	0	4	4	0	0	0	0	0
2	0	4	0	4	0	2	0	2	0	0	0	0	2	0	2	0
3	0	0	0	4	2	0	2	0	0	0	4	0	0	2	0	2
4	0	0	0	0	4	0	4	0	0	0	0	0	0	4	0	4
5	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
6	0	4	0	4	0	2	0	2	0	0	0	0	2	0	2	0
7	0	0	4	0	0	2	0	2	0	4	0	0	2	0	2	0
8	0	0	0	0	2	0	2	0	4	2	0	2	4	0	0	0
9	0	2	2	0	0	2	2	0	0	0	2	2	0	0	2	2
a	0	0	0	0	0	4	0	0	4	2	0	2	0	2	0	2
b	0	2	2	0	0	0	2	2	0	0	2	2	2	0	0	2
c	0	0	0	0	2	0	2	0	4	2	0	2	0	0	4	0
d	0	2	2	0	2	0	0	2	0	0	2	2	2	2	0	0
e	0	0	0	0	0	0	0	4	4	2	0	2	0	2	0	2
f	0	2	2	0	2	2	0	0	0	0	2	2	0	2	2	0

(a) DDT of S-box

I\O	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	0	0	0	4	0	0	2	0	2	0	2	0	2
2	0	4	0	0	0	0	0	4	0	2	0	2	0	2	0	2
3	0	4	4	4	0	0	4	0	0	0	0	0	0	0	0	0
4	0	0	0	2	4	2	0	0	2	0	0	0	2	2	0	2
5	0	0	2	0	0	2	2	2	0	2	4	0	0	0	0	2
6	0	0	0	2	4	2	0	0	2	2	0	2	2	0	0	0
7	0	0	2	0	0	2	2	2	0	0	0	2	0	2	4	0
8	0	0	0	0	0	0	0	0	4	0	4	0	4	0	4	0
9	0	4	0	0	0	0	0	4	2	0	2	0	2	0	2	0
a	0	4	0	4	0	0	0	0	0	2	0	2	0	2	0	2
b	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
c	0	0	2	0	0	2	2	2	4	0	0	2	0	2	0	0
d	0	0	0	2	4	2	0	0	0	0	2	0	0	2	2	2
e	0	0	2	0	0	2	2	2	0	2	0	0	4	0	0	2
f	0	0	0	2	4	2	0	0	0	2	2	2	0	0	2	0

(b) Inverse DDT of S-box

Table 4: Differential Distribution Tables of the S-box and Its inverse with Input/Output (I/O) Differences Represented in Hexadecimal Format.

Algorithm 3 Boomerang Distinguishing Attack against the FUTURE Cipher

procedure DISTINGUISHER($\Delta_0, \Delta_1, \nabla_0, \nabla_1, \alpha, \delta$)
  Randomly choose a key $K_0 \xleftarrow{\$} \{0,1\}^{128}$. ▷ Steps 2 and 3 are chosen by the oracle.
  Form other keys $K_1 = K_0 \oplus \alpha, K_2 = K_0 \oplus \beta$, and $K_3 = K_0 \oplus \alpha \oplus \beta$.
  Choose 2^{60} distinct plaintext pairs as $(P_0^i, P_1^i = P_0^i \oplus \Delta_0)$, $i = 1, 2, \ldots, 2^{60}$.
  for $i = 1$ to 2^{60} **do**
    Query P_0^i to the encryption oracle under the key K_0 and obtain the corresponding ciphertext $C_0^i = E_{K_0}(P_0^i)$.
    Query $P_1^i = P_0^i \oplus \Delta_0$ to the encryption oracle under the key K_1 and obtain the corresponding ciphertext $C_1^i = E_{K_1}(P_1^i)$.
    Compute $C_2^i = C_0^i \oplus \nabla_1$ and $C_3^i = C_1^i \oplus \nabla_1$.
    Query C_2^i to the decryption oracle under the key K_2 and obtain the corresponding plaintext $P_2^i = E_{K_2}(C_2^i)$.
    Query C_3^i to the decryption oracle under the key K_3 and obtain the corresponding plaintext $P_3^i = E_{K_3}(C_3^i)$.
    if $P_2^i \oplus P_3^i == \Delta_0$ **then**
      Return 1 ▷ The oracle is the FUTURE cipher.
  Return 0 ▷ The oracle is a random permutation.

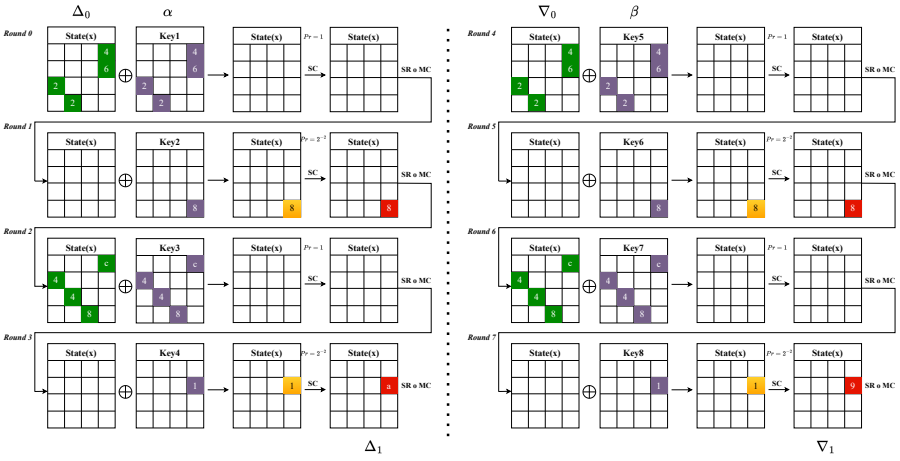


Fig. 13: An 8-Round Boomerang Distinguisher

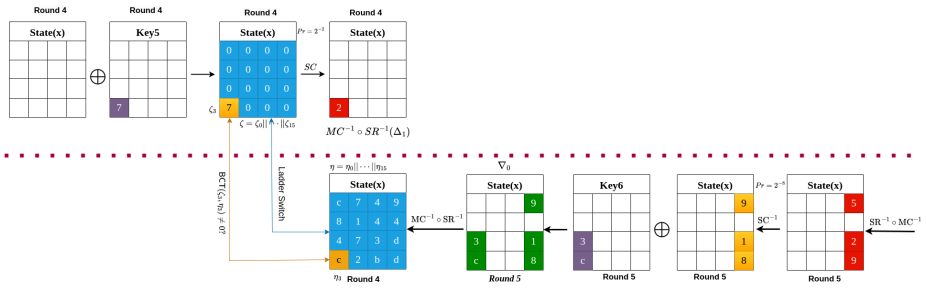


Fig. 15: Middle Round Switching Effects

$$
\begin{aligned}
y_0 + y_3 + y_7 + y_8 + y_{12} + y_{15} + t_0 - 6d_0 + 4d_1 + 2d_2 &= 0 \\
y_0 + y_1 + y_4 + y_9 + y_{12} + y_{13} + t_1 - 6d_3 + 4d_4 + 2d_5 &= 0 \\
y_1 + y_2 + y_5 + y_{10} + y_{13} + y_{14} + t_2 - 6d_6 + 4d_7 + 2d_8 &= 0 \\
y_2 + y_6 + y_7 + y_{11} + y_{14} + t_3 - 6d_9 + 4d_{10} + 2d_{11} &= 0 \\
y_0 + y_1 + y_5 + y_{11} + y_{15} + t_4 - 6d_{12} + 4d_{13} + 2d_{14} &= 0 \\
y_1 + y_2 + y_6 + y_8 + y_{12} + t_5 - 6d_{15} + 4d_{16} + 2d_{17} &= 0 \\
y_0 + y_2 + y_3 + y_4 + y_7 + y_9 + y_{13} + t_6 - 8d_{18} + 6d_{19} + 4d_{20} + 2d_{21} &= 0 \\
y_0 + y_3 + y_4 + y_{10} + y_{11} + y_{14} + y_{15} + t_7 - 8d_{22} + 6d_{23} + 4d_{24} + 2d_{25} &= 0 \\
y_1 + y_4 + y_5 + y_8 + y_{11} + y_{15} + t_8 - 6d_{26} + 4d_{27} + 2d_{28} &= 0 \\
y_2 + y_5 + y_6 + y_8 + y_9 + y_{12} + t_9 - 6d_{29} + 4d_{30} + 2d_{31} &= 0 \\
y_0 + y_3 + y_4 + y_6 + y_7 + y_9 + y_{10} + y_{13} + t_{10} - 8d_{32} + 6d_{33} + 4d_{34} + 2d_{35} &= 0 \\
y_0 + y_4 + y_7 + y_{10} + y_{14} + y_{15} + t_{11} - 6d_{36} + 4d_{37} + 2d_{38} &= 0 \\
y_3 + y_7 + y_8 + y_{11} + y_{12} + t_{12} - 6d_{39} + 4d_{40} + 2d_{41} &= 0 \\
y_0 + y_4 + y_8 + y_9 + y_{13} + t_{13} - 6d_{42} + 4d_{43} + 2d_{44} &= 0 \\
y_1 + y_5 + y_9 + y_{10} + y_{14} + t_{14} - 6d_{45} + 4d_{46} + 2d_{47} &= 0 \\
y_2 + y_3 + y_6 + y_7 + y_{10} + y_{15} + t_{15} - 6d_{48} + 4d_{49} + 2d_{50} &= 0
\end{aligned}
$$

Fig. 14: 16 Constraints Correspond to One Column Transformation After the MixColumn Operation

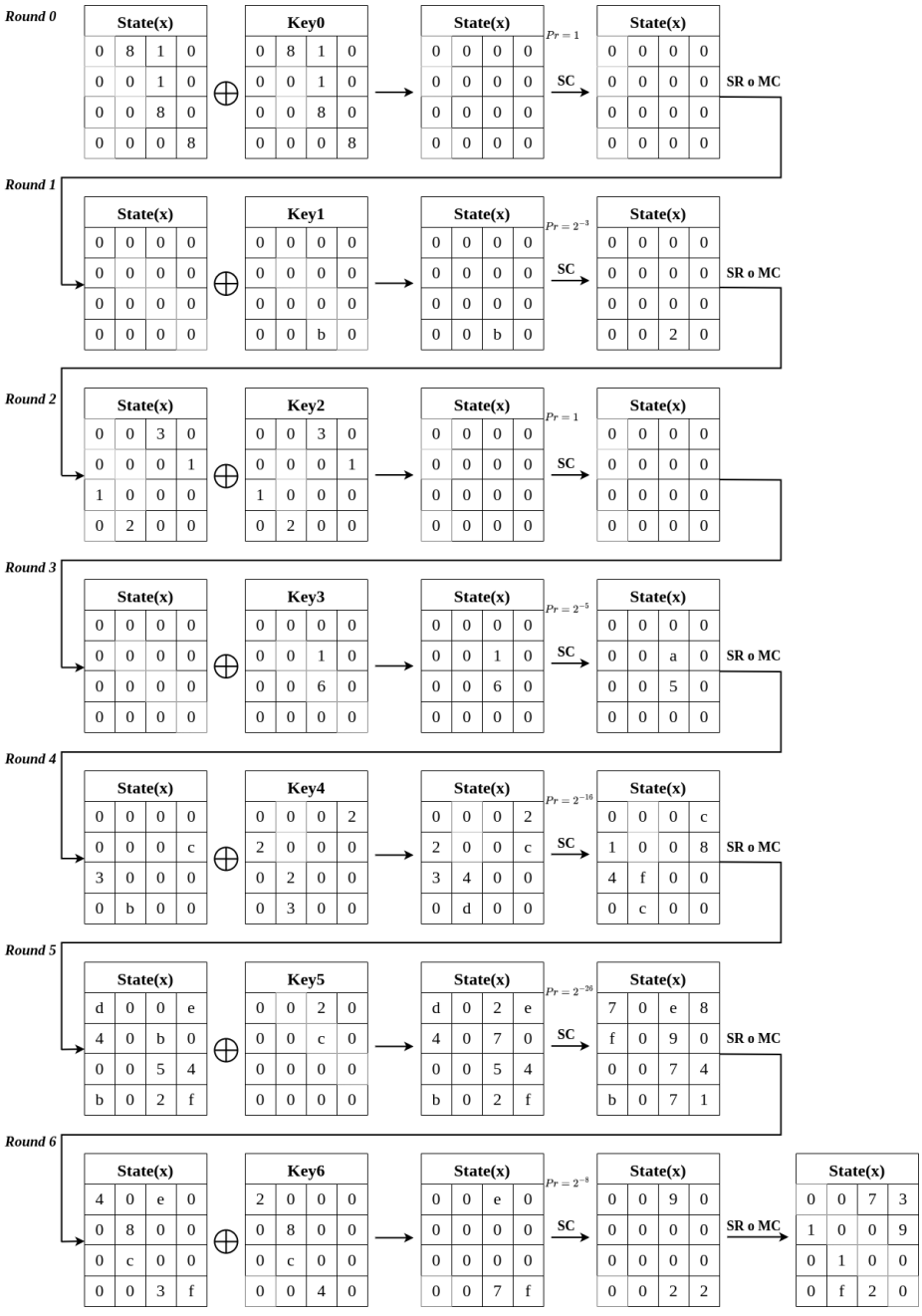


Fig. 16: Seven Round Related-Key Differential Characteristic of FUTURE Cipher

$$
\begin{aligned}
3x_0 + x_1 - 5x_2 - 3x_3 + 2y_0 + 8y_1 + 7y_2 + 5y_3 + 0 &\geq 0\\
-2x_0 - x_1 + 2x_2 + 2x_3 + y_0 - 2y_1 + y_2 - 3y_3 + 5 &\geq 0\\
-3x_0 - x_1 - 2x_2 + 2x_3 - y_0 - 3y_1 + y_2 + 3y_3 + 7 &\geq 0\\
4x_0 - 3x_1 + 2x_2 + x_3 - y_0 + 3y_1 + 0y_2 + y_3 + 0 &\geq 0\\
-2x_0 + x_1 - 3x_2 + 2x_3 + 5y_0 + 2y_1 + 3y_2 + 3y_3 + 0 &\geq 0\\
2x_0 - 3x_1 - 3x_2 - x_3 - 3y_0 - 2y_1 - y_2 - 3y_3 + 13 &\geq 0\\
4x_0 + x_1 - x_2 + 3x_3 - 4y_0 + 2y_1 + 0y_2 - 2y_3 + 3 &\geq 0\\
-x_0 + 0x_1 - 2x_2 + 4x_3 + y_0 + 2y_1 - 3y_2 + 3y_3 + 2 &\geq 0\\
-2x_0 - x_1 - x_2 - x_3 + 3y_0 + 2y_1 - 3y_2 - 3y_3 + 8 &\geq 0\\
-2x_0 + x_1 + 0x_2 - 2x_3 - 2y_0 + y_1 + 2y_2 - y_3 + 5 &\geq 0\\
-2x_0 - 2x_1 + 2x_2 - 2x_3 - y_0 - 2y_1 - 2y_2 + y_3 + 9 &\geq 0\\
4x_0 + 6x_1 + 5x_2 + 2x_3 - y_0 - 3y_1 - y_2 - y_3 + 0 &\geq 0\\
0x_0 + x_1 + x_2 - 2x_3 + 0y_0 + y_1 + 2y_2 + 2y_3 + 0 &\geq 0\\
-x_0 + x_1 + 2x_2 + 2x_3 + 3y_0 + 0y_1 - y_2 - 3y_3 + 2 &\geq 0\\
x_0 - 2x_1 - 2x_2 + 0x_3 + 2y_0 - 2y_1 - y_2 + 2y_3 + 5 &\geq 0\\
2x_0 + 3x_1 - x_2 - 3x_3 + 3y_0 - y_1 + 2y_2 - 3y_3 + 5 &\geq 0\\
x_0 + 2x_1 - x_2 - 2x_3 - 2y_0 - 2y_1 - y_2 + 2y_3 + 6 &\geq 0
\end{aligned}
$$

Fig. 17: 17 Number of Constraints to Capture DDT of FUTURE S-box

$$
\begin{aligned}
3x_0 + 4x_1 + 4x_2 + x_3 - 2y_0 - 0y_1 - 2y_2 + y_3 + 0 &\geq 0\\
2x_0 - 2x_1 + 3x_2 - 4x_3 - y_0 - 4y_1 - 4y_2 + y_3 + 11 &\geq 0\\
-3x_0 + 2x_1 + 2x_2 + 6x_3 + 2y_0 - 3y_1 + 2y_2 + 4y_3 + 0 &\geq 0\\
-x_0 - 5x_1 + 4x_2 + 2x_3 - 5y_0 - 3y_1 - 2y_2 - 5y_3 + 16 &\geq 0\\
2x_0 - 4x_1 - 4x_2 - x_3 + 3y_0 - 5y_1 + 2y_2 - 3y_3 + 12 &\geq 0\\
x_0 + 4x_1 - 3x_2 - 3x_3 + 5y_0 + 6y_1 + 2y_2 + 6y_3 + 0 &\geq 0\\
2x_0 + 3x_1 - 2x_2 - 3x_3 - 3y_0 - y_1 + 2y_2 + y_3 + 6 &\geq 0\\
x_0 - 3x_1 + x_2 + 0x_3 + 2y_0 + 3y_1 + 2y_2 + 3y_3 + 0 &\geq 0\\
x_0 - 2x_1 - 2x_2 + x_3 - 2y_0 + 0y_1 - 2y_2 + 2y_3 + 6 &\geq 0\\
2x_0 + x_1 - 2x_2 + 4x_3 + y_0 + 4y_1 + 4y_2 - 2y_3 + 0 &\geq 0\\
-x_0 - 4x_1 - 4x_2 - 3x_3 - 2y_0 + 4y_1 - 2y_2 - y_3 + 13 &\geq 0\\
-5x_0 - 4x_1 + 3x_2 - 2x_3 - 4y_0 - 3y_1 + 2y_2 - y_3 + 14 &\geq 0\\
-5x_0 + 3x_1 - 4x_2 - 2x_3 + 2y_0 - 3y_1 - 4y_2 - y_3 + 14 &\geq 0\\
-x_0 + 4x_1 - 5x_2 + 2x_3 - 2y_0 - 3y_1 - 5y_2 - 5y_3 + 16 &\geq 0\\
3x_0 - 2x_1 + 3x_2 + 4x_3 + 4y_0 + 5y_1 - y_2 - 2y_3 + 0 &\geq 0\\
-x_0 + 2x_1 + 2x_2 - 2x_3 - y_0 + y_1 - y_2 - y_3 + 4 &\geq 0\\
x_0 + 5x_1 + 5x_2 - 2x_3 + 5y_0 + y_1 + 5y_2 - 2y_3 + 0 &\geq 0
\end{aligned}
$$

Fig. 18: 17 Number of Constraints to Capture DDT of PRESENT S-box

$$
\begin{aligned}
-x_0 + 2x_1 - 2x_2 - x_3 + 0y_0 + 0y_1 - 2y_2 - y_3 + 5 &\geq 0\\
4x_0 - 2x_1 - x_2 + 0x_3 - y_0 + 3y_1 + 3y_2 + 2y_3 + 0 &\geq 0\\
-x_0 - 2x_1 + 2x_2 - x_3 - 2y_0 + 2y_1 - 2y_2 - 2y_3 + 8 &\geq 0\\
-2x_0 + 2x_1 + x_2 + 0x_3 - y_0 + y_1 + 2y_2 - 2y_3 + 3 &\geq 0\\
x_0 - 2x_1 + x_2 + 2x_3 - 3y_0 - y_1 - 2y_2 + 2y_3 + 5 &\geq 0\\
2x_0 + 3x_1 + 2x_2 + 2x_3 + 0y_0 - y_1 + y_2 - 2y_3 + 0 &\geq 0\\
-x_0 + x_1 + 0x_2 + 0x_3 + 2y_0 - y_1 + 2y_2 + y_3 + 0 &\geq 0\\
3x_0 + 4x_1 + 5x_2 - 2x_3 + 0y_0 - y_1 - 2y_2 + 5y_3 + 0 &\geq 0\\
2x_0 + 2x_1 - 3x_2 - 3x_3 + 0y_0 - y_1 - y_2 - 3y_3 + 8 &\geq 0\\
x_0 - 3x_1 + x_2 - x_3 + 3y_0 + 4y_1 + 2y_2 + y_3 + 0 &\geq 0\\
x_0 - 3x_1 - x_2 - 2x_3 + 3y_0 - y_1 - 3y_2 + 2y_3 + 7 &\geq 0\\
-x_0 - 2x_1 - 2x_2 + x_3 + 2y_0 + 2y_1 + 0y_2 - 2y_3 + 5 &\geq 0\\
-x_0 - 2x_1 + 2x_2 + x_3 + 2y_0 - 2y_1 - 2y_2 - 2y_3 + 7 &\geq 0\\
-2x_0 - 2x_1 - x_2 + 0x_3 - 2y_0 - 2y_1 + y_2 - 2y_3 + 9 &\geq 0\\
2x_0 + x_1 + 0x_2 + x_3 - 3y_0 + 0y_1 + 2y_2 + 3y_3 + 0 &\geq 0\\
2x_0 + 4x_1 - 3x_2 + 2x_3 + 0y_0 - y_1 + 2y_2 + 3y_3 + 0 &\geq 0\\
x_0 - 2x_1 - x_2 + 2x_3 - 3y_0 + y_1 - 2y_2 + 2y_3 + 5 &\geq 0
\end{aligned}
$$

Fig. 19: 17 Number of Constraints to Capture DDT of LBlock s0 S-box

$$
\begin{aligned}
-2x_0 - x_1 + x_2 - 2x_3 - y_0 + 2y_1 - 2y_2 - 2y_3 + 8 &\geq 0\\
-3x_0 - 3x_1 + 6x_2 - 4x_3 + 4y_0 - y_1 + 2y_2 + 5y_3 + 5 &\geq 0\\
-5x_0 + 4x_1 - x_2 + 5x_3 - 2y_0 - 2y_1 - 3y_2 + 5y_3 + 8 &\geq 0\\
4x_0 + 3x_1 - x_2 + 3x_3 + 2y_0 - 3y_1 + 2y_2 + 2y_3 + 0 &\geq 0\\
2x_0 - 3x_1 - 4x_2 - 4x_3 + 3y_0 + 2y_1 + y_2 - 4y_3 + 10 &\geq 0\\
-x_0 + x_1 + 2x_2 + 2x_3 + 2y_0 + y_1 - y_2 - 2y_3 + 2 &\geq 0\\
3x_0 - 2x_1 + 3x_2 - x_3 - y_0 - 2y_1 - 3y_2 + 2y_3 + 6 &\geq 0\\
7x_0 + 5x_1 + 4x_2 + 7x_3 - 3y_0 + y_1 - 2y_2 - 2y_3 + 0 &\geq 0\\
-2x_0 - 2x_1 + 2x_2 - x_3 + 2y_0 + 3y_1 + 3y_2 + 5y_3 + 0 &\geq 0\\
-x_0 + 2x_1 - 2x_2 + 3x_3 - 2y_0 + 3y_1 - 4y_2 + 2y_3 + 5 &\geq 0\\
-x_0 + 2x_1 + 2x_2 - 3x_3 + 3y_0 - 2y_1 + y_2 - 2y_3 + 5 &\geq 0\\
3x_0 - x_1 - 4x_2 + x_3 + 2y_0 - y_1 + 4y_2 + 4y_3 + 2 &\geq 0\\
x_0 - 2x_1 + 2x_2 + 3x_3 - 3y_0 - y_1 + y_2 - 2y_3 + 5 &\geq 0\\
-2x_0 + 2x_1 - 3x_2 - x_3 - 2y_0 - y_1 + 3y_2 - 3y_3 + 9 &\geq 0\\
-3x_0 + 5x_1 - 5x_2 - x_3 + 2y_0 + 3y_1 - 2y_2 + 4y_3 + 6 &\geq 0\\
2x_0 + 2x_1 + 0x_2 - 2x_3 + 2y_0 + y_1 - 2y_2 - y_3 + 3 &\geq 0\\
-3x_0 - 2x_1 - 4x_2 + 2x_3 + y_0 + 3y_1 + 3y_2 - y_3 + 6 &\geq 0\\
2x_0 + 2x_1 + 2x_2 - x_3 - 3y_0 - 2y_1 - y_2 + y_3 + 4 &\geq 0\\
2x_0 - x_1 + 2x_2 + 3x_3 - y_0 - 3y_1 - 3y_2 - 2y_3 + 6 &\geq 0
\end{aligned}
$$

Fig. 20: 19 Number of Constraints to Capture DDT of Prince S-box

$$
\begin{aligned}
-2x_0 + 5x_1 - 4x_2 - x_3 - 3y_0 - 4y_1 - 2y_2 + 5y_3 + 11 &\geq 0\\
-2x_0 + 3x_1 - 2x_2 + 3x_3 + y_0 + 2y_1 + 3y_2 + 2y_3 + 0 &\geq 0\\
0x_0 - x_1 + 2x_2 - x_3 - y_0 - 2y_1 + y_2 - 2y_3 + 5 &\geq 0\\
-x_0 - 3x_1 - x_2 - 3x_3 + 0y_0 + y_1 + 3y_2 + y_3 + 5 &\geq 0\\
-3x_0 + 3x_1 + 2x_2 - 3x_3 + y_0 - 3y_1 - 2y_2 + y_3 + 8 &\geq 0\\
5x_0 + 2x_1 + 3x_2 - 3x_3 - y_0 + 2y_1 - 4y_2 - 4y_3 + 7 &\geq 0\\
x_0 - 2x_1 - 2x_2 - 2x_3 + y_0 - 2y_1 - 2y_2 - 2y_3 + 10 &\geq 0\\
5x_0 - 3x_1 + 3x_2 + 2x_3 - y_0 - 4y_1 - 4y_2 + 2y_3 + 7 &\geq 0\\
4x_0 + 5x_1 + 2x_2 + 5x_3 + y_0 - 3y_1 - 2y_2 - 3y_3 + 1 &\geq 0\\
5x_0 + 4x_1 + 5x_2 + 4x_3 - 3y_0 - 2y_1 + y_2 - 2y_3 + 0 &\geq 0\\
-2x_0 + x_1 - 3x_2 - 4x_3 + 5y_0 + 2y_1 + 4y_2 - 3y_3 + 7 &\geq 0\\
-2x_0 - x_1 - 4x_2 + 5x_3 - 3y_0 + 5y_1 - 2y_2 - 4y_3 + 11 &\geq 0\\
4x_0 + x_1 - 2x_2 + x_3 - y_0 + 2y_1 - 3y_2 + 2y_3 + 2 &\geq 0\\
-2x_0 - 4x_1 - 3x_2 + x_3 + 5y_0 - 3y_1 + 4y_2 + 2y_3 + 7 &\geq 0\\
x_0 - 3x_1 - 2x_2 - 3x_3 + 4y_0 + 5y_1 + 2y_2 + 5y_3 + 1 &\geq 0\\
x_0 + 2x_1 + 3x_2 + 2x_3 - 2y_0 + 3y_1 - 2y_2 + 3y_3 + 0 &\geq 0\\
-3x_0 - 3x_1 + 2x_2 + 3x_3 + y_0 + y_1 - 2y_2 - 3y_3 + 8 &\geq 0
\end{aligned}
$$

Fig. 21: 17 Number of Constraints to Capture DDT of Midori s0 S-box

$$
\begin{aligned}
-x_0 - 2x_1 + x_2 - 3x_3 + y_0 - 3y_1 - 3y_2 + 2y_3 + 9 &\geq 0\\
2x_0 + 3x_1 + 0x_2 - x_3 + 2y_0 - 3y_1 - y_2 - 2y_3 + 5 &\geq 0\\
2x_0 - 2x_1 - x_2 + 2x_3 + y_0 - 2y_1 + 0y_2 - 2y_3 + 5 &\geq 0\\
2x_0 + 3x_1 - 2x_2 + 3x_3 - y_0 + 4y_1 + 4y_2 - y_3 + 0 &\geq 0\\
3x_0 - 4x_1 - 3x_2 - 2x_3 + 2y_0 + 4y_1 - 2y_2 - y_3 + 8 &\geq 0\\
2x_0 + 3x_1 + 4x_2 + 4x_3 - 2y_0 - 3y_1 + y_2 + 2y_3 + 0 &\geq 0\\
2x_0 + 3x_1 - 2x_2 - x_3 - 3y_0 - y_1 + 3y_2 + 3y_3 + 4 &\geq 0\\
-2x_0 - 4x_1 - x_2 + 2x_3 + 3y_0 - 4y_1 + 4y_2 - 2y_3 + 9 &\geq 0\\
3x_0 - 2x_1 + 2x_2 - 3x_3 - 3y_0 - 2y_1 + y_2 - y_3 + 8 &\geq 0\\
x_0 + 4x_1 - 5x_2 + 3x_3 + y_0 + 3y_1 + 4y_2 + 2y_3 + 0 &\geq 0\\
-4x_0 + 3x_1 - 5x_2 - x_3 - 2y_0 - 2y_1 - 3y_2 - 4y_3 + 16 &\geq 0\\
3x_0 + 3x_1 - x_2 + 2x_3 + 3y_0 + 0y_1 - 2y_2 + y_3 + 0 &\geq 0\\
x_0 - 2x_1 + 0x_2 - 2x_3 + 2y_0 - 2y_1 - y_2 + 2y_3 + 5 &\geq 0\\
x_0 + 3x_1 + 4x_2 + 2x_3 + y_0 + 4y_1 - 5y_2 + 3y_3 + 0 &\geq 0\\
-x_0 + 2x_1 + 5x_2 - 5x_3 + 3y_0 + 2y_1 + 4y_2 - 3y_3 + 4 &\geq 0\\
-3x_0 - 2x_1 + 3x_2 + 3x_3 + 2y_0 + 3y_1 - y_2 - y_3 + 4 &\geq 0\\
-2x_0 - 4x_1 - 3x_2 + 2x_3 - 3y_0 - 4y_1 - 2y_2 + y_3 + 14 &\geq 0\\
-2x_0 - 2x_1 - 3x_2 - 4x_3 - 4y_0 + 3y_1 - 5y_2 - y_3 + 16 &\geq 0\\
-x_0 - 4x_1 + 2x_2 + 2x_3 + 2y_0 + 4y_1 + 3y_2 + 4y_3 + 0 &\geq 0
\end{aligned}
$$

Fig. 22: 19 Number of Constraints to Capture DDT of Klein S-box

$$
\begin{aligned}
-2x_0 - 2x_1 - 2x_2 + x_3 - 2y_0 + y_1 + y_2 - y_3 + 7 &\geq 0 \\
-x_0 - x_1 + x_2 - x_3 - y_0 + y_1 - y_2 + y_3 + 4 &\geq 0 \\
-x_0 - 3x_1 + 3x_2 - 3x_3 - y_0 - 2y_1 + y_2 - 3y_3 + 10 &\geq 0 \\
4x_0 + 3x_1 + 5x_2 + 4x_3 + 2y_0 - 2y_1 - y_2 - 2y_3 + 0 &\geq 0 \\
3x_0 - 4x_1 + 2x_2 + x_3 + y_0 + y_1 + 3y_2 + 3y_3 + 0 &\geq 0 \\
-4x_0 - 2x_1 - 4x_2 + x_3 + 4y_0 + 3y_1 - 5y_2 - 2y_3 + 12 &\geq 0 \\
x_0 + 3x_1 + 3x_2 + 5x_3 - 4y_0 - y_1 + 2y_2 + 4y_3 + 0 &\geq 0 \\
2x_0 - 2x_1 - x_2 + 2x_3 + y_0 + 2y_1 + 0y_2 + 3y_3 + 0 &\geq 0 \\
x_0 + 5x_1 + 3x_2 + 2x_3 + 2y_0 - 4y_1 - 4y_2 + 2y_3 + 3 &\geq 0 \\
-2x_0 - x_1 - 3x_2 - 3x_3 - 3y_0 - 3y_1 - 2y_2 + y_3 + 14 &\geq 0 \\
-4x_0 - 4x_1 + 2x_2 + 4x_3 + 3y_0 - y_1 - 2y_2 - 4y_3 + 10 &\geq 0 \\
4x_0 - 5x_1 - 2x_2 - 3x_3 - 4y_0 - y_1 - 4y_2 - 2y_3 + 16 &\geq 0 \\
-2x_0 + 2x_1 - x_2 - x_3 + 2y_0 - y_1 + 2y_2 - 2y_3 + 5 &\geq 0 \\
2x_0 + x_1 + 2x_2 - 5x_3 + 3y_0 + 2y_1 + 5y_2 + 4y_3 + 0 &\geq 0 \\
-2x_0 - 2x_1 - x_2 - 2x_3 + 7y_0 + 5y_1 + 6y_2 + 7y_3 + 0 &\geq 0 \\
-2x_0 + 3x_1 - 3x_2 - x_3 - 3y_0 + y_1 - 2y_2 + 3y_3 + 8 &\geq 0 \\
x_0 + 5x_1 - 3x_2 - 5x_3 + 3y_0 + 6y_1 + 2y_2 - 4y_3 + 6 &\geq 0 \\
-2x_0 + x_1 + 3x_2 - x_3 + 2y_0 - 2y_1 - 2y_2 + 3y_3 + 4 &\geq 0 \\
-3x_0 + 2x_1 + 2x_2 - x_3 - 2y_0 + 3y_1 + y_2 - 3y_3 + 6 &\geq 0 \\
3x_0 + 5x_1 - 2x_2 + 2x_3 - 5y_0 - 2y_1 + 4y_2 - y_3 + 5 &\geq 0
\end{aligned}
$$

Fig. 23: 20 Number of Constraints to Capture DDT of TWINE S-box