Greedy Algorithm for Representative Sets: Applications to IVLBC and GIFT-64 in Impossible Differential Attack

Manjeet Kaur^{®*1}, Tarun Yadav^{®^{†2}}, Manoj Kumar^{®^{‡3}}, and Dhananjoy Dey^{®^{§4}}

^{1,4}Indian Institute of Information Technology, Lucknow, U.P., INDIA-226 002 ^{2,3}Scientific Analysis Group, DRDO, Delhi, INDIA-110 054

Abstract

The impossible differential (ID) attack is crucial for analyzing the strength of block ciphers. The critical aspect of this technique is to identify IDs, and the researchers introduced several methods to detect them. Recently, the researchers extended the mixed-integer linear programming (MILP) approach by partitioning the input and output differences to identify IDs. The researchers proposed techniques to determine the representative set and partition table of a set over any nonlinear function. In this paper, we introduce a deterministic algorithm using the greedy approach [1] for finding the representative set and partition table of a set over any nonlinear function. This algorithm iteratively selects the set that covers the maximum remaining elements of the target set. This performs better than the existing algorithms in terms of time complexity. We use this algorithm to compute the representative sets and partition tables of the two block ciphers, IVLBC and GIFT-64, and identify 6-round IDs for them. Our research contributes a deterministic algorithm to compute the representative set and partition table of a set over any nonlinear function with at most 16-bit input size.

Keywords: GIFT-64, Impossible Differentials, IVLBC, MILP, Representative Set

1 Introduction

Currently, an impossible differential (ID) attack is one of the most significant techniques in the security analysis of block ciphers [2–4]. Identifying IDs is an essential step in this technique. For a cipher E, a difference pair (x, y) is called an ID if $x \not\xrightarrow{E} y$. Due to the huge search space, it is not practical to manually search for IDs. Therefore, the focus has shifted to automatic tools for finding IDs. The researchers have introduced several methods, including the \mathcal{U} -method [5], the UID-method [6], and some extensions by Wu and Wang [7]. In [8,9], the researchers suggested the MILP method to identify IDs by imposing additional limits on the input and output differences within the MILP models of differential trails [10–14]. This approach, however, is inefficient to analyze all difference pairs due to the huge search space, which is 2^{2n} , where n denotes the block size. After that, Hu *et al.* [15] presented a new technique for reducing the search space in SPN ciphers using MILP. This technique partitions the complete collection of difference

^{*}Email: rmm21101@iiitl.ac.in

[†]Email: tarunyadav.sag@gov.in

[‡]Email: manojkumar.sag@gov.in

[§]Email: ddey@iiitl.ac.in

pairs into smaller and non-overlapping sets, each of which contains a representative pair. If the representative pair is feasible through MILP, then all pairs in the corresponding sets are also feasible; otherwise, the sets may consist of IDs. This method minimizes the search space to a reasonable size by eliminating sets that do not contain any IDs. Subsequently, we identify IDs through the analysis of the remaining pairs. Moreover, the researchers [15] proposed an algorithm to calculate the representative set and partition table; however, this method does not produce the same results for the fixed input. Then, Wang *et al.* [16] presented the MILP-based methods to determine the minimal representative sets and partition tables over S-boxes and super boxes. In our paper, we introduce a deterministic algorithm for computing the representative set and hash table of a set over any nonlinear function, which has better time complexity. We then applies it on the two lightweight block ciphers, IVLBC [17] and GIFT-64 [18]. Our detailed contributions are as follows:

- 1. We introduce a deterministic algorithm based on greedy approach [1] to compute the representatives and partition table of a set over any 16-bit nonlinear function. This algorithm performs better than existing ones in terms of time complexity, as shown in Table 1.
- We compute the representative set and the hash table of IVLBC and then utilize these to identify 6-round IVLBC IDs using the MILP method, as shown in Algorithm 3. Additionally, our chosen search space does not contain any 7- or 8-round IDs.
- 3. Similarly, we calculate the representatives and partition table for GIFT-64 and detect the IDs for 6-round GIFT-64. Moreover, we do not obtain any IDs for 7- or 8-round GIFT-64.

We organize the remaining paper as follows: Section 2 describes the details of the MILP method for identifying IDs, including relevant definitions used in this paper. We propose a greedy-based algorithm in Section 3 that identifies the representative set and partition table of a set over a nonlinear function. Additionally, we identify IVLBC and GIFT-64 IDs. At last, Section 4 summarizes the paper.

2 Notations and Preliminaries

This paper uses lowercase letters (x) and uppercase letters (X) to represent differences and sets of differences, respectively. For a cipher E, $x \stackrel{E}{\mapsto} x'$ and $x \stackrel{E}{\not\mapsto} x'$ represent possible and impossible propagations, respectively. If $x \stackrel{E}{\mapsto} x'$ satisfies for all $(x, x') \in X \otimes X'$, then we write it as $X \stackrel{E}{\to} X'$. The Cartesian product $X \otimes X'$ is defined by $\{(x, x') \mid x \in X \text{ and } x' \in X'\}$. We use the notation $\sum_i X_i$ instead of $\bigcup_i X_i$ when $\bigcap_i X_i = \emptyset$. The expression $X - X' = \{x : x \in X \text{ and } x \notin X'\}$ denotes the difference between two sets. The notation $f \circ g$ (right to left) represents the composition of the functions f and g.

For identifying the IDs of an *n*-bit block cipher E, we are required to solve 2^{2n} MILP models in [8,9]. These MILP models, with extra restrictions on input and output differences, are the same as those used for identifying differential trails. To address the extensive search space, we employ the MILP method, which involves partitioning the entire search space into smaller, disjoint sets [15]. For this, we write E as $E = E_2 \circ E_1 \circ E_0$ and let $X_0, X_3 \subseteq \mathbb{F}_2^n$ be the sets of differences. The process has three steps: (i) partitioning the set, (ii) pairwise examination, and (iii) identification of IDs. Therefore, we partition the set $X_0 \otimes X_3$ into smaller, disjoint sets, as explained below:

$$X_0 \otimes X_3 = \sum_{i=0}^{\delta} \sum_{j=0}^{\delta'} H_1[x_1^i] \otimes H_2[x_2^j],$$

where $X_0 = \sum_{i=0}^{\delta} H_1[x_1^i]$, and $X_3 = \sum_{j=0}^{\delta'} H_2[x_2^j]$.

To achieve this, we require two representative sets, X_1 and X_2 , such that $H_1[x_1^i] \xrightarrow{E_0} x_1^i$ and $x_2^j \xrightarrow{E_2} H_2[x_2^j]$, where $x_1^i \in X_1$ and $x_2^j \in X_2$. The partition of $X_0 \otimes X_3$ over E is illustrated in Figure 1. For each pair (x_1^i, x_2^j) , we evaluate whether $x_1^i \xrightarrow{E_1} x_2^j$ using MILP models [8,9]. If the condition is satisfied, the set $H_1[x_1^i] \otimes H_2[x_2^j]$ does not have any IDs. Otherwise, the set may contain IDs, as depicted in Figure 2. To identify all IDs, analyze the remaining sets that may include IDs.



Figure 1: The partition of sets X_0 and X_3 [15]



Figure 2: If $x_1 \xrightarrow{E_1} x_2$, then $X_0 \otimes X_3$ may contain some IDs. Otherwise, $X_0 \xrightarrow{E} X_3$

Algorithm 1 outlines the entire process for identifying all IDs within the set $X_0 \otimes X_3$. Further, we discuss some definitions utilized in this paper.

Definition 1 (Representative Set and Partition Table). If f is any nonlinear function and X is the set of input differences of f. We have,

for all
$$x \in X$$
, $\exists s \in S$ such that $x \stackrel{J}{\mapsto} s$.

Algorithm 1: Find IDs over E

Input: $E = E_2 \circ E_1 \circ E_0$, where $E_i = E_{i,m-1} ||E_{i,m-2}|| \dots ||E_{i,0}$ for i = 0, 2**Output:** A set *I* consists of IDs ¹ for $i \in \{0, 2\}$ do for $j \in \{0, 1, \dots, m-1\}$ do 2 To compute representative set $X_{i,j}$ and partition table $H_{i,j}$, we use Algorithm 3 to $E_{i,j}$; з 4 $I' \leftarrow \phi;$ 5 for $(x_{1,m-1}, x_{1,m-2}, \dots, x_{2,0}) \in X_{1,m-1} \otimes X_{1,m-2} \otimes \dots \otimes X_{2,0}$ do Create a MILP model with the input and output differences $(x_{1,m-1}, x_{1,m-2}, \ldots, x_{1,0})$ and 6 $(x_{2,m-1}, x_{2,m-2}, \ldots, x_{2,0})$, respectively over E_1 ; if model is not feasible then 7 $| I' \leftarrow I' \bigcup \{ (x_{1,m-1}, x_{1,m-2}, \dots, x_{2,0}) \};$ 8 9 $I \leftarrow \phi;$ 10 for $(x_{1,m-1}, x_{1,m-2}, \dots, x_{2,0}) \in I'$ do Construct MILP models to check each pattern in $H_{1,m-1} \otimes H_{1,m-2} \otimes \ldots \otimes H_{2,0}$, and add the impossible ones into I; ¹² return I;

Then, we call S the representative set of X over f and

$$\bigcup_{s \in S} \{ x \mid x \stackrel{f}{\mapsto} s \} = X.$$

By eliminating the repeated elements from $\{x \mid x \stackrel{f}{\mapsto} s\}$ for all $s \in S$, we have a partition table of X with respect to f by storing the elements of S as keys and sets after partitioning as values in a hash table.

Hu *et al.* [15] introduced an algorithm to compute a representative set and partition table for a set X over a nonlinear function f. However, it does not produce the same output with any fixed input. Wang *et al.* [16] then introduced the optimal MILP-based algorithm to determine the representative set and partition table. In our paper, we develop a deterministic algorithm that employs greedy method from [1] to identify the representatives and partition table of X, which performs better than existing ones in terms of time complexity.

3 Proposed Algorithm and Applications

This section introduces an algorithm to compute the representatives and the partition table of a set over a nonlinear function. Then, we discuss its applications to two lightweight block ciphers, IVLBC and GIFT-64.

3.1 Proposed Algorithm

The proposed algorithm employs Cormen et al.'s [1] greedy approach to find the representative set and the partition table of a set over any nonlinear function, as shown in Algorithm 3. The procedure begins by constructing a hash table, H, with the output differences over f as keys and the sets of associated input differences as values (Lines 11–17). Then, we compute the representative set S by iteratively selecting the keys from H such that the associated value covers the maximum uncovered elements of the set until X is empty (Lines 18–25). Finally, we form the partition table of X by removing the overlapping elements from the sets corresponding to S (Lines 5–10). Our algorithm consistently yields the same result for any fixed input. It has a time complexity of $\mathcal{O}(2^{2|X|})$ and a space complexity of $\mathcal{O}(2^{2|X|})$, Algorithm 2: Compute representatives and partition table of X over a nonlinear function f [15]

Input: $X \subseteq \mathbb{F}_2^k$ and $f : \mathbb{F}_2^k \to \mathbb{F}_2^k$ Output: S, H $S \leftarrow \phi;$ ² $H \leftarrow \phi;$ while $X \neq \phi$ do 3 $x \xleftarrow{\mathcal{R}} X;$ compute X' such that $x \xrightarrow{f} X'$; $x' \xleftarrow{\mathcal{R}} X';$ 6 $S \leftarrow S \cup \{x'\};$ compute X'' such that $X'' \xrightarrow{f} x'$; $H[x'] \leftarrow X'';$ 9 $X \leftarrow X - X''$: 10 $H' \leftarrow \phi;$ 11 for $h \in S$ do 12 $H'[h] \leftarrow H[h];$ 13 for $h' \in S$ do 14 if $h' \neq h$ then 15 $H[h'] \leftarrow H[h'] - H'[h];$ 16 17 return S, H

respectively. Table 1 illustrates the comparative analysis of algorithms and shows that our algorithm has better time complexity compared to the existing ones.

In the end, we employ our algorithm on two lightweight block ciphers, IVLBC and GIFT-64, and calculate the representatives and partition tables within a maximum of 186 minutes. Following this, we identify the IDs for both the ciphers.

3.2 Application to IVLBC

The literature has not yet identified any IDs on IVLBC. This section provides a detailed description of IVLBC and then identify the IDs using the MILP method, as discussed in Section 2 as there is no impossible differential attack on it till now. IVLBC was proposed by Huang *et al.* in 2022 [17], with two variants, IVLBC-80 and IVLBC-128. It has 29 rounds, and each round consists of four operations: AddRoundKey (ARK), SubCells (SC), PermuteNibbles (PN), and MixColumns (MC), as shown in Figure 3.



Figure 3: Round function of IVLBC

Algorithm 3: Compute representatives and partition table of X over f using greedy approach

Input: $X \subseteq \mathbb{F}_2^k$ and $f : \mathbb{F}_2^k \to \mathbb{F}_2^k$ Output: S, H1 size $\leftarrow 2^n$; ² $H \leftarrow calculate_H(f, size);$ $S \leftarrow find_representative_set(H, X);$ 4 $H' \leftarrow \phi$; 5 for $h \in S$ do $H'[h] \leftarrow H[h];$ 6 for $h' \in S$ do 7 if $h' \neq h$ then 8 $H[h'] \leftarrow H[h'] - H'[h];$ 9 10 return S, H;11 Function calculate_H(f, size): $H \leftarrow \phi;$ 12 for $x \in \{0, 1, \dots, size - 1\}$ do 13 for $z \in \{0, 1, \dots, size - 1\}$ do 14 $y \leftarrow f(z) \oplus f(z \oplus x);$ 15 $H[y] \leftarrow H[y] \cup \{x\};$ 16 return H; 17 ¹⁸ Function find_representative_set(H, X): $S \leftarrow \phi;$ 19 while $X \neq \phi$ do 20 $key \leftarrow$ select the key from H such that the corresponding value 21 covers maximum elements of X; 22 $S \leftarrow S \cup \{key\};$ 23 $X \leftarrow X - H[key];$ 24 return S; 25

Table 1: Comparative Analysis

Algorithm	Туре	Space Complexity	Time Complexity	Input Size
[15]	Probabilistic	$\mathcal{O}(2^{2n})$	$\mathcal{O}(2^{3n})$	16-bit
[16]	Optimal	$\mathcal{O}(f(n) \cdot 2^{2n})$	$\mathcal{O}(f(n) \cdot 2^{2^n})$	16-bit
Ours	Deterministic	$\mathcal{O}(2^{2n})$	$\mathcal{O}(2^{2n})$	16-bit

3.2.1 IDs of IVLBC

This paper considers only a single-key setting to detect IDs for IVLBC; consequently, we can omit the add round key operation. As a result, we represent the round function of IVLBC as follows.

$$\mathbf{F} = MC \circ PN \circ SC.$$

An r-round cipher, E, can be written as follows:

$$\mathbf{F}^{r} = MC \circ PN \circ SC \circ MC \circ PN \circ SC \circ \mathbf{F}^{r-4} \circ MC \circ PN \circ SC \circ MC \circ PN \circ SC$$
$$= MC \circ PN \circ SC \circ MC \circ SC \circ PN \circ \mathbf{F}^{r-4} \circ MC \circ PN \circ SC \circ MC \circ SC \circ PN$$
$$= \underbrace{SC \circ MC \circ SC}_{E_{2}} \circ \underbrace{PN \circ \mathbf{F}^{r-4} \circ MC \circ PN}_{E_{1}} \circ \underbrace{SC \circ MC \circ SC}_{E_{0}}$$

We rearrange PN and SC at the ends of E and neglect the two linear operations, MC and PN, from our analysis, as they do not affect the identification of IDs. Both E_0 and E_2 comprise four parallel super boxes [19], as indicated in Equation 1 and Figure 4. Since all the super boxes are the same, therefore, it is sufficient to calculate the representatives

Representative Set	Values
	0x0, 0x26, 0x113, 0x11a, 0x12a, 0x12e, 0x133, 0x23a, 0x2af, 0x444, 0x1013, 0x101b, 0x102a,
	0x102e, 0x1033, 0x1103, 0x110b, 0x1130, 0x11b0, 0x120a, 0x120e, 0x12a0, 0x12e0, 0x1303,
V where $h \in \{1, 2\}$ and $0 \leq i \leq 2$	0x1330, 0x168e, 0x19ec, 0x1fe8, 0x202a, 0x20af, 0x220a, 0x22a0, 0x25af, 0x28ad, 0x28af,
$A_{k,j}$, where $k \in \{1,2\}$ and $0 \leq j \leq 5$	0x2a0f, 0x2af0, 0x2b1e, 0x4044, 0x4060, 0x4278, 0x4404, 0x4440, 0x4444, 0x461d, 0x52ce,
	0x679f, 0x6891, 0x6a1d, 0x7b8e, 0x7ccb, 0x7fdd, 0x8aee, 0x8da7, 0x8df9, 0x9392, 0x9887,
	0x9969, 0x9c9e, 0x9fed, 0xb8ac, 0xba1c, 0xbae9, 0xbcf9, 0xc1b9, 0xc39e, 0xd2fa, 0xd9aa

and partition table for a single super box.

$$E_i = E_{i,3} ||E_{i,2}||E_{i,1}||E_{i,0}, \text{ where } i = 0,2$$
 (1)



Figure 4: Super boxes representation of IVLBC $(i \in \{0, 2\})$

The representative sets for super boxes are computed using Algorithm 3, with results detailed in Table 2. The representative sets consist of 68 elements for $E_{i,j}$, where i = 0, 2 and $0 \le j \le 3$. Therefore, the number of MILP models over E_1 is equal to $(68^4 - 1) \times (68^4 - 1) \approx 2^{48.70}$. It is not feasible to test all corresponding MILP models with our available resources. We limit our selection to 16 elements and focus on pairs that have exactly two active blocks for the 6-round, 7-round, and 8-round IVLBC models. We investigate input-output difference pairs, which may contain IDs and then conduct an exhaustive search of all possible sets to identify IDs. In the end, we obtain 1,44,360 IDs for 6-round IVLBC, whereas no IDs are detected for 7-round and 8-round IVLBC.

3.3 Application to GIFT-64

Banik *et al.* introduced GIFT at CHES 2017 [18] with its two variants, GIFT-64-128 and GIFT-128-128, depending on SPN structure. This paper focuses exclusively on GIFT-64-128, referred to as GIFT-64, which comprises 28 rounds. Each round function comprises three operations: the substitution layer (S), bit-permutation (P), and the addition of round key/constants, as depicted in Figure 5.



Figure 5: Round Function of GIFT-64

The single-key setting on GIFT-64 is the only focus of our analysis. Therefore, we can ignore the add-round keys or constants in each round. Consequently, we define the round function as follows:

$$\mathbf{F} = P \circ S$$

Moreover, the two consecutive rounds in GIFT-64 can be expressed as follows:

$$S \circ P \circ S = P_2 \circ S \circ P_1 \circ S$$

where $P_1 = [0, 5, 10, 15, 12, 1, 6, 11, 8, 13, 2, 7, 4, 9, 14, 3]$ and $P_2 = [0, 4, 8, 12, 1, 5, 9, 13, 2, 6, 10, 14, 3, 7, 11, 15]$. We can represent the r-round cipher as follows.

$$\mathbf{F}^{r} = P \circ S \circ P \circ S \circ \mathbf{F}^{r-4} \circ P \circ S \circ P \circ S$$
$$= P \circ P_{2} \circ S \circ P_{1} \circ S \circ \mathbf{F}^{r-4} \circ P \circ P_{2} \circ S \circ P_{1} \circ S$$
$$= \underbrace{S \circ P_{1} \circ S}_{E_{2}} \circ \underbrace{\mathbf{F}^{r-4} \circ P \circ P_{2}}_{E_{1}} \circ \underbrace{S \circ P_{1} \circ S}_{E_{0}}$$

The researchers in [15,16] presented representative sets of E_0 and E_2 in GIFT-64 and claimed that 8-round GIFT-64 is free of IDs. However, the researchers' representative sets do not cover the entire set. Therefore, it is necessary to recalculate the representative sets for this specific cipher. Consequently, we use our algorithm to compute the representatives of a super box over E_0 and E_2 in GIFT-64. As indicated in Table 3, the representative sets over $E_{0,j}$ and $E_{2,j}$, where $0 \le j \le 3$, contain 26 and 25 elements, respectively. As a result, the total number of MILP models that have the potential to contain IDs is $(26^4 - 1) \times (25^4 - 1) \approx 2^{37.38}$, which is not practical to solve. We therefore select 14 elements from each representative set and evaluate the pairs that contain exactly two 16-bit active blocks. Finally, we identify 4,661 IDs for 6-round GIFT-64, but there are no 7-round or 8-round IDs within the chosen search space.



Figure 6: Super boxes representation of GIFT-64 $(i \in \{0, 2\})$

4 Conclusion

We introduced a greedy-based deterministic algorithm to identify representative sets and partition tables needed for ID identification through MILP. We conducted a comparison of existing algorithms, demonstrating better performance

Representative Set	Values
	0x0, 0x7d6, 0x839, 0xb05, 0x1025, 0x300b, 0x3dff, 0x63d0, 0x777e, 0x79df, 0x7c7c, 0x7da0,
$X_{1,j}$, where $0 \le j \le 3$	0x86df, 0x90b3, 0x9f32, 0xa7d7, 0xaaaa, 0xadad, 0xb799, 0xca77, 0xcaca, 0xcccc, 0xdd77,
	0xdfb7, 0xdfd9, 0xeeee
	0x0, 0xd, 0xa1, 0x7f7, 0x9b0, 0x1102, 0x1edd, 0x39b9, 0x76db, 0x90f9, 0x99bf, 0x9c59,
$X_{2,j}$, where $0 \le j \le 3$	0x9e0d, 0x9f9d, 0xb111, 0xb799, 0xbb9b, 0xcf97, 0xd6dd, 0xd9d9, 0xdbec, 0xdf90, 0xdfdb,
	0xf3cf, 0xf991

than others for 16-bit nonlinear functions. Through this algorithm, we computed the representatives and the partition tables for IVLBC and GIFT-64, then identified 6-round IDs for both ciphers. In this paper, we only explored the limited search space because of limited resources. Therefore, in the future, we will explore the entire search space to identify longer IDs as well as consider more ciphers for the identification of IDs. Furthermore, we will extend our algorithm over nonlinear functions with larger input sizes.

References

- [1] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to algorithms. MIT press, 2022.
- [2] E. Biham, A. Biryukov, and A. Shamir, "Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials," in Advances in Cryptology—EUROCRYPT'99: International Conference on the Theory and Application of Cryptographic Techniques Prague, Czech Republic, May 2–6, 1999 Proceedings 18, pp. 12–23, Springer, 1999.
- [3] L. Knudsen, "Deal-a 128-bit block cipher," complexity, vol. 258, no. 2, p. 216, 1998.
- [4] C. Boura and M. NAYA-PLASENCIA, "Impossible differential cryptanalysis," Symmetric Cryptography, Volume 2: Cryptanalysis and Future Directions, p. 47, 2024.
- [5] J. Kim, S. Hong, J. Sung, S. Lee, J. Lim, and S. Sung, "Impossible differential cryptanalysis for block cipher structures," in Progress in Cryptology-INDOCRYPT 2003: 4th International Conference on Cryptology in India, New Delhi, India, December 8-10, 2003. Proceedings 4, pp. 82–96, Springer, 2003.
- [6] Y. Luo, X. Lai, Z. Wu, and G. Gong, "A unified method for finding impossible differentials of block cipher structures," *Information Sciences*, vol. 263, pp. 211–220, 2014.
- [7] S. Wu and M. Wang, "Automatic search of truncated impossible differentials for word-oriented block ciphers," in International Conference on Cryptology in India, pp. 283–302, Springer, 2012.
- [8] T. Cui, S. Chen, K. Jia, K. Fu, and M. Wang, "New automatic search tool for impossible differentials and zero-correlation linear approximations," *Cryptology ePrint Archive*, 2016.
- [9] Y. Sasaki and Y. Todo, "New impossible differential search tool from design and cryptanalysis aspects: Revealing structural properties of several ciphers," in Advances in Cryptology-EUROCRYPT 2017: 36th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Paris, France, April 30-May 4, 2017, Proceedings, Part III 36, pp. 185–215, Springer, 2017.

- [10] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "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.
- [11] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "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.
- [12] E. Biham and A. Shamir, "Differential cryptanalysis of des-like cryptosystems," Journal of CRYPTOLOGY, vol. 4, pp. 3–72, 1991.
- [13] L. R. Knudsen and M. Robshaw, The block cipher companion. Springer Science & Business Media, 2011.
- [14] J. Daemen and V. Rijmen, The design of Rijndael, vol. 2. Springer, 2002.
- [15] K. Hu, T. Peyrin, and M. Wang, "Finding all impossible differentials when considering the ddt," in International Conference on Selected Areas in Cryptography, pp. 285–305, Springer, 2022.
- [16] S. Wang, D. Feng, T. Shi, B. Hu, J. Guan, K. Zhang, and T. Cui, "New methods for bounding the length of impossible differentials of spn block ciphers," *IEEE Transactions on Information Theory*, 2024.
- [17] X. Huang, L. Li, and J. Yang, "Ivlbc: An involutive lightweight block cipher for internet of things," *IEEE Systems Journal*, vol. 17, no. 2, pp. 3192–3203, 2022.
- [18] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "Gift: A small present: Towards reaching the limit of lightweight encryption," in *Cryptographic Hardware and Embedded Systems-CHES 2017:* 19th International Conference, Taipei, Taiwan, September 25-28, 2017, Proceedings, pp. 321–345, Springer, 2017.
- [19] J. Daemen and V. Rijmen, "New criteria for linear maps in aes-like ciphers," Cryptography and Communications, vol. 1, pp. 47–69, 2009.