New Impossible Differential Search Tool from Design and Cryptanalysis Aspects

Yu Sasaki and Yosuke Todo

NTT Secure Platform Laboratories
3-9-11 Midori-cho, Musashino-shi, Tokyo 180-8585 Japan
{sasaki.yu,todo.yosuke}@lab.ntt.co.jp

Abstract. In this paper, a new tool searching for impossible differentials against symmetric-key primitives is presented. Compared to the previous tools, our tool can detect any contradiction between input and output differences, and it can take into account the property inside the S-box when its size is small e.g. 4 bits. In addition, several techniques are proposed to evaluate 8-bit S-box. With this tool, the number of rounds of impossible differentials are improved from the previous best results by 1 round for Midori128, LILLIPUT, and Minalpher. The tool also finds new impossible differentials of ARIA and MIBS. We manually verify the impossibility of the searched results, which reveals new structural properties of those designs.
Our tool can be implemented only by slightly modifying the previous differential search tool using Mixed Integer Linear Programming (MILP), while the previous tools need to be implemented independently of the differential search tools. This motivates us to discuss the usage of our tool particular for the design process. With this tool, the maximum number of rounds of impossible differentials can be proven under reasonable assumptions and the tool is applied to various concrete designs.

Key words: symmetric-key, impossible differential, mixed integer linear programming, Midori, LILLIPUT, Minalpher, ARIA, MIBS

1 Introduction

Designing symmetric-key primitives becomes more and more complicated to simultaneously satisfy various goals such as security against many notions, efficiency in high-end software, low-implementation cost in hardware, and so on.

A popular design approaches is *substitution-permutation network (SPN)*, in which a state is composed of small words, and is updated by iteratively applying a round function consisting of a non-linear layer and a linear layer. In the non-linear layer, the state is updated by looking up a word-wise precomputed table called S-box. In the linear layer, the state is mixed with some linear operations.

A lot of designs were proposed in the last decade. It is now necessary for the community to carefully but quickly evaluate their security. Automated evaluation tools are useful to evaluate various designs in short term. Regarding the

differential cryptanalysis and linear cryptanalysis, automated tools have been well-developed. In particular, evaluating the lower bound of the number of active S-boxes with mixed-integer-linear programming (MILP) is becoming popular in the design of SPN primitives [1]. Meanwhile, automated tools for other cryptanalytic approaches are not as sophisticated as differential and linear cryptanalysis.

Impossible differential cryptanalysis [2, 3] is one of the most major and effective cryptanalytic approaches. In short, for a target function F, it reveals a pair of input and output differences (Δ_i, Δ_o) that cannot be connected. Namely, two input values x, x' satisfying $x \oplus x' = \Delta_i$ never satisfy $F(x) \oplus F(x') = \Delta_o$. Because an ideally designed function should not have this property, the detection of (Δ_i, Δ_o) immediately leads to a distinguishing attack. The most important part of the attack is finding such a distinguisher. In particular, the attacker aims to maximize the number of rounds.

Such (Δ_i, Δ_o) are detected by the *miss-in-the-middle* approach [4] and Kim et al. [5] presented the first automated tool called $\mathcal{U}$-method. Suppose that one wants to examine if (Δ_i, Δ_o) is impossible. First it propagates Δ_i in forwards (with F) by r_f rounds, and checks if the difference of each word is known active, active, inactive, or unknown. Then, it propagates Δ_o in backwards (with F^{-1}) by r_b rounds and checks the same information. Finally, it finds contradiction in the middle, detecting that (Δ_i, Δ_o) is impossible for $r_f + r_b$ rounds.

Several researches extended the $\mathcal{U}$-method, e.g. UID-method by Luo et al. [6, 7] or some extension by Wu and Wang [8]. Those detect more complicated contradiction than the $\mathcal{U}$-method. Although some advancement was made, usability of the previous tools are limited as explained below.

- To be as generic as possible, the recent tools consider complicated differential impact through the linear layer, which requires more sensitive implementation. Even with this effort, only particular contradictions can be analyzed.
- Most of the previous tools cannot take into account differential property inside the S-box. Several analysis against a particular S-box in a particular primitive may analyze its differential property [9, 10], however such an analysis cannot be extended to a generic tool.
- Most of the previous tools for impossible differential cryptanalysis cannot be used to evaluate other cryptanalytic approaches, e.g. differential and linear cryptanalysis. Derbez and Fouque proposed a tool for the meet-in-the-middle attack that can also be used for impossible differential cryptanalysis [11]. However, it cannot find better impossible pairs compared to [5, 6, 8].

Our Contributions. In this paper, we propose a new automated tool to find impossible differentials. Our tool is based on the previous tool for (standard) differential cryptanalysis with MILP.

In the differential search with MILP, the attacker describes possible differential propagation patterns in a round function by using linear inequalities. Then, the attacker runs a solver for MILP, which returns the minimum number of active S-boxes under the given propagation patterns. In this research, to examine the impossibility of (Δ_i, Δ_o), we simply add constraints to fix the input and

output differences to (Δ_i, Δ_o). Due to the added constraints, the lower bound of the number of active S-boxes usually increases. In some case, (Δ_i, Δ_o) cannot be satisfied, thus the MILP solver returns an error code implying that no solution exists. In other words, Δ_i and Δ_o are impossible pairs. We can examine various pairs of (Δ_i, Δ_o) to enlarge the search space.

Our tool leads to stronger cryptanalytic results than the previous tools owing to the following advantages.

Analyzing inside S-boxes: The previous differential-bound search using MILP [12] can model the possible differential propagation patterns in the differential distribution table (DDT) of the S-box. Our tool inherits this advantage. Thus impossible differentials taking into account DDT can be found.

Arbitrary Contradiction: The MILP solver automatically judges whether or not the solution exists. Thus, the attacker does not have to predict the mechanism of contradiction in advance, which significantly increases the versatility of the tool.

Multi-Purpose Tool: We convert the previous MILP-based differential search into impossible differential search by just adding constraints to fix input and output differences. Thus only with a single tool, security against differential cryptanalysis and impossible differential cryptanalysis can be evaluated. This feature is especially useful for future primitive designers who need to evaluate both cryptanalysis.

Arbitrary S-box Mode: MILP requires too many inequalities to represent differential propagations in DDT of 8-bit S-boxes. Thus, the tool is infeasible for 8-bit S-boxes in a straightforward manner. Here, we introduce an *arbitrary S-box* where impossible differentials for the arbitrary S-box is always valid for arbitrary S-box choice. The arbitrary S-box can be described efficiently, which enables us to evaluate 8-bit S-boxes.

Quick Search for Truncated Impossible Differential: A single pair of input and output differences can be impossible for more rounds than truncated differentials. Meanwhile, evaluating all the pairs is infeasible and the search range is often limited to single-active word. Here we present a technique to make the search fast only by aiming truncated impossible differentials.

We apply the proposed tool to various designs. The results improving the existing impossible differentials are summarized in Table 1. Although one of the advantages of the tool is that the attacker can detect impossible differentials without analyzing contradicting reasons, we manually analyze why the detected (Δ_i, Δ_o) is impossible. The manual verification not only demonstrates the correctness of the tool, but also reveals the structural properties of the target designs that have not been known before. We believe that the contradicting reasons analyzed in this paper for Midori128, LILLIPUT, and Minalpher lead to new understanding about their designs.

Our automated tool is useful to test many design choices during the design process of new primitives. Thus, we also discuss the usage of the tool for the design. For example, when the tool finds several impossible pairs of (Δ_i, Δ_o), the designers may want to patch the design to avoid such (Δ_i, Δ_o). By using

Table 1. Application Results. 'KR' denotes 'key recovery.'

Target	Ref.	#Rounds Prev.	#Rounds Ours	Search Mode	Goal	Remarks
Midori128	[13]	6	7	specific S-box	characteristic	
LILLIPUT	[14]	8	9	specific S-box	characteristic	
Minalpher	[15]	6.5	7.5	arbitrary S-box	truncated	large state
ARIA	[16]	4	4	arbitrary S-box	truncated	8-bit S-box, improve KR
MIBS	[17]	8	8	specific S-box	characteristic	new impossible differentials

We also list all 8-round impossible differentials of RECTANGLE [10] in Appendix F.

the arbitrary S-box mode, we can easily check whether (Δ_i, Δ_o) is dependent on the S-box. If it is dependent on the S-box, it may be prevented by replacing the S-box. If it is independent, it needs to modify the linear layer to prevent it.

Moreover, because it catches any contradiction, the tool provides a certain level of provable security about the existence of impossible differentials with reasonable assumptions and reasonable search range. In details, provable security can be discussed when a single word is active in the input and output differences, and we can set two-level of the assumption; 1) S-box is public and each subkey is chosen independently and uniformly at random and 2) keyed S-box is used and for each key the S-box is chosen uniformly at random. We apply the tool to various designs to prove the maximum number of rounds of impossible differentials. Finally, we propose an *optimal pick technique* which dramatically reduces the execution time only when the tool is used for obtaining the proof.

Paper Outline. Notations and related work are introduced in Sect. 2. Framework of our tool is introduced in Sect. 3. Application on various designs improving previous impossible differentials are shown in Sect. 4. A technique to reduce the search complexity is explained in Sect. 5. Advantages of our tool in the design process are explained in Sect. 6. Our research is partially overlapped with [18]. The relationship between [18] and this paper is explained in Appendix A.

2 Related Work

2.1 Terminologies in Impossible Differential Cryptanalysis

- We call a pair of input and output differences (Δ_i, Δ_o) that cannot be connected an *impossible differential characteristic* or *impossible characteristic*.
- We call a pair of a closed set of input differences and a closed set of output differences in which any pair cannot be connected as a *truncated impossible differential*.
- When we do not distinguish the above two, we call it *impossible differential*.

2.2 Differential Search with Mixed Integer Linear Programming

Here we explain an automated tool for differential cryptanalysis, not impossible differential cryptanalysis, which will be a base of our tool.

Mouha et al. [1] showed that the problem to search for the minimum number of active S-boxes can be modeled with mixed integer linear programming (MILP). The approach is now very popular for designing a new primitive. For example, resistance against differential and linear cryptanalysis of Skinny [19] recently proposed at CRYPTO 2016 was evaluated by MILP.

The approach by Mouha et al. [1] is effective for evaluating word-oriented ciphers, while several ciphers are not word-oriented. For example, PRESENT [20] applies 4-bit S-box, then the bit-permutation moves four bits from a single S-box to four different S-boxes. In order to apply MILP to such a structure, Sun et al. [12] developed a method to model all possible differential propagations bit by bit even for the S-box.

Modeling Differential Propagations with MILP. We explain how to model valid differential propagations of PRESENT in bitwise. Note that one round of PRESENT consists of subkey addition, S-box applications, and bit-permutation.

At first, binary variables to represent active or inactive of each bit are defined for all rounds; $x_0, x_1, \ldots, x_{63}$ are for 64 bits in the plaintext, $x_{64}, x_{65}, \ldots, x_{127}$ are for 64 bits after round 1, $x_{128}, x_{129}, \ldots, x_{191}$ are for 64 bits after round 2, and so on. Each variable takes '1' if the bit has the difference, and takes '0' otherwise. Then, the constraint to ensure at least 1 active bit is added, which can be written as '$x_0 + x_1 + \cdots + x_{63} \geq 1$.' Finally, constraints to be valid differential propagations are added. Here, the bit-permutation only changes the order of variables and subkey addition can be ignored because it does not change the difference. The following denotes the variables involved in the first round, in which a 64-bit plaintext difference $x_0, \ldots, x_{63}$ are updated to $x_{64}, \ldots, x_{127}$.

$$\begin{bmatrix} x_0, x_1, x_2, x_3 \\ x_4, x_5, x_6, x_7 \\ x_8, x_9, x_{10}, x_{11} \\ x_{12}, x_{13}, x_{14}, x_{15} \\ \cdots \\ x_{60}, x_{61}, x_{62}, x_{63} \end{bmatrix} \xrightarrow{\text{S-box}} \begin{bmatrix} x_{64}, x_{68}, x_{72}, x_{76} \\ x_{80}, x_{84}, x_{88}, x_{92} \\ x_{96}, x_{100}, x_{104}, x_{108} \\ x_{112}, x_{116}, x_{120}, x_{124} \\ \cdots \\ x_{115}, x_{119}, x_{123}, x_{127} \end{bmatrix} \xrightarrow{\text{BitPerm}} \begin{bmatrix} x_{64}, x_{65}, x_{66}, x_{67} \\ x_{68}, x_{69}, x_{70}, x_{71} \\ x_{72}, x_{73}, x_{74}, x_{75} \\ x_{76}, x_{77}, x_{78}, x_{79} \\ \cdots \\ x_{124}, x_{125}, x_{126}, x_{127} \end{bmatrix}$$

The most difficult part is describing all possible propagation patterns for 16 S-boxes, e.g. $x_0, x_1, x_2, x_3 \longrightarrow x_{64}, x_{68}, x_{72}, x_{76}$, with a system of linear inequalities. Sun et al. [12] showed two approaches to solve the problem.

Fact 1 *Linear inequalities to constrain input and output variables of the S-box only to valid patterns can be generated by using either the computation tool called SageMath or several logical operations.*

How to use SageMath is well explained in [12] and more details of logical computations can be seen in [21]. We rely on Fact 1 about the description of S-box,

and the choice of SageMath and logical operations does not impact to our tool. Meanwhile, the following limitation of those approaches should be noted.

Fact 2 *Both of SageMath and the logical operations can be used only when the S-box size is small.*

In our computational environment, both methods can be feasible only up to 5-bit S-box. No method is known to model bigger S-box, e.g. 8-bit S-box.

MILP returns a solution of the system optimizing a given objective function. In differential cryptanalysis, the attacker's goal is minimizing the number of active S-boxes, which can be defined as "Minimize $\sum_i (x_{4i} \vee x_{4i+1} \vee x_{4i+2} \vee x_{4i+3})$."

The system can be solved by the MILP solver to find the optimal solution. We use Gurobi Optimizer [22] as the MILP solver.

2.3 Impossible Differential Search with MILP

Cui et al. have recently posted their work to Cryptology ePrint Archive presenting that impossible differentials (and zero-correlation approximations) can be searched with MILP [18]. Although we have independently reached the same idea and used it to evaluate a lot of designs, the work by Cui et al. became the first article to report the impossible differential search tool based on MILP. Comparison between [18] and this work will be explained in Appendix A. We explain [18] in details because we use the same idea in this work.

The impossible differential search with MILP heavily relies on the previous differential bound search with MILP. As explained in Sect. 2.2, in the differential bound search, users describe all possible differential propagations and then run the MILP solver to find the best solution. This can be converted to check if a specific pair of input and output differences, (Δ_i, Δ_o), is impossible or not by adding constraints to fix input and output differences to (Δ_i, Δ_o). Due to those additional constraints, the MILP solver may not be able to find the solution, thus returns some error code indicating that *the system is infeasible*, which tells that (Δ_i, Δ_o) is an impossible differential characteristic.

Example 1 *Let $p_0, p_1, \ldots, p_{b-1}$ and $c_0, c_1, \ldots, c_{b-1}$ be variables that represent active/inactive of plaintext bits and ciphertext bits, respectively, where b is the block size. To test if $(\Delta_i, \Delta_o) = (\texttt{0x1}, \texttt{0x1})$ is impossible, the MILP solver should run with the following constraints added.*

$$p_0 = 1, p_1 = 0, c_2 = 0, \ldots, p_{b-1} = 0,$$
$$c_0 = 1, c_1 = 0, c_2 = 0, \ldots, c_{b-1} = 0.$$

To find truncated impossible differentials, Cui et al. iterated the search for one pair many times by changing input and output differences. Because testing all possible pairs is computationally infeasible, Cui et al. limit the differences to be some subset, e.g. single-active word.

As summarized in Fact 2, the MILP-based search only can work for small S-box. Cui et al. left application to ciphers using 8-bit S-box open.

3 Composite Framework for Differential and Impossible Differential Searches

Our basic idea is the same as Cui et al. [18], which runs the MILP solver with fully specified input and output differences and checks if the system is infeasible or not. This enables the following improvements compared to the previous tools.

Analyzing inside S-boxes: This property is also well advertised in [18]. Regarding small S-box, up to 5-bit S-box in our computational environment, all possible differential propagations through S-box can be modeled with MILP. Hence impossible differential characteristics considering the differential property of S-box can be identified.

Arbitrary Contradiction: Whether the system is infeasible or not is automatically judged by the MILP solver. Hence, the attackers do not need to predict contradicting reasons. The tool can catch any contradiction, which significantly reduces the attackers' task. Note that this feature is not focused in [18], and we present several extending techniques based on this feature.

Both properties have huge potential to find new impossible characteristics, thus the number of attacked rounds may be extended. Actually, in Sect. 4 we show several applications detecting new characteristics that seem impossible to find with by-hand analysis or previous search tools.

Note that running time of our tool for a single pair of input and output differences is significantly shorter than the differential search. This can be explained that the solver can stop only by detecting one characteristic. In the previous differential bound search, the bottleneck of the tool is increasing the lower bound. Finding some upper bound (some solution of the system) is usually fast.

Limitations. While it is very effective against primitives using 4- or 5-bit S-box, it is important to discuss limitations of this approach.

1. The idea cannot be applied to larger S-boxes, thus Cui et al. left application to 8-bit S-boxes open. Moreover, though it is not mentioned in [18], applying this idea to primitives with a large block size, say 256 bits or 512 bits, is computationally hard even if the S-box size is small.
2. The MILP solver checks if a given input and output differences are impossible to solve. We iterate this test for various input and output differences. To keep the computation time practical, as suggested by [18], the choices of input and output differences must be limited to a reasonable subset, i.e. only one active word. Actually, searching for impossible characteristics with two-active words is computationally infeasible.

New Techniques. In this work, we present several additional techniques that partially overcome both of the above-mentioned limitations of the basic idea.

Algorithm 1 Generating System of Inequalities in Previous Differential Search

Require: number of rounds r, system of inequalities for S-boxes and linear layer
Ensure: system of inequalities

1: Write an objective function.$_A$
2: Write constraints ensuring at least 1 active bit in input.$_B$
3: **for** round = 1 to r **do**
4: Write constraints for the S-boxes.$_C$
5: Write constraints for the linear layer.
6: **end for**

1. For large S-boxes and large block sizes, we stop modeling the differential property precisely, instead analyzing an imaginary S-box, called *arbitrary S-box*, in which differential propagations of any S-box become a subset of the ones in the arbitrary S-box. The arbitrary S-box can be modeled very efficiently, which allows to analyze large primitives including 8-bit S-box.
2. The tool for a single pair of (Δ_i, Δ_o) can be extended to truncated differentials by simply running it for multiple pairs. However, this limits the search range only to 1 active word. We show a technique to make the tool more efficient only by aiming truncated impossible differentials, which can be implemented only by changing the constraints of input and output differences.

3.1 Composite Framework

Another remarkable advantage of our tool is that users can switch differential-bound search and impossible-differential search very easily. This helps primitive designers, generally required to evaluate the resistance against both of differential and impossible differential cryptanalysis. Here we introduce our framework to generate system of inequalities depending on the target to evaluate.

Most of the symmetric-key primitives can be described as an iteration of the round function consisting of the non-linear and linear layers. We explain our tool by following this structure. Our tool focuses on the primitive whose non-linear layer is the parallel application of S-boxes. The tool relies on the previous MILP-based differential search that models differential propagations in bitwise [12, 23, 21]. Here, we recall how a system of inequalities is generated in previous work.

First, the number of rounds, r, is fixed. Then, an objective function, e.g. minimizing the number of active S-boxes, is defined. It also constrains the system so that at least one S-box is activated. The remaining is writing constraints for the valid differential propagations through the S-boxes and linear layer for r rounds, which can be done with [12, 23, 21]. The procedure is summarized in Algorithm 1. Underlines in Algorithm 1 will be later referred by Algorithm 2.

We slightly modify Algorithm 1 so that impossible differentials can be evaluated with several techniques. The goal of the tool can be either the differential bound (DB) or the impossibility of the given input and output differences (ID), which can be specified in the parameter "GOAL". For converting DB to ID, the

Algorithm 2 Generating System of Inequalities in Composite Framework

Require: number of rounds r, system of inequalities for S-boxes and linear layer, GOAL $\in$ {DB, ID}, MODE $\in$ {SPECFIC, ARBITRARY}, and OBJECT $\in$ {TRUNCATED,CHARACTERISTIC}
Ensure: system of inequalities

/* Lines 1–5 correspond to $\_\__A$ in Alg. 1. */
1: **if** GOAL = DB **then**
2: Write an objective function.
3: **else if** GOAL = ID **then**
4: Leave an objective function empty.
5: **end if**

/* Lines 6–14 correspond to $\_\__B$ in Alg. 1. */
6: **if** GOAL = DB **then**
7: Write constraints ensuring at least 1 active bit in input.
8: **else if** GOAL = ID **then**
9: **if** OBJECT = CHARACTERISTIC **then**
10: Fully specify active or inactive for each input and output bit.
11: **else if** OBJECT = TRUNCATED **then**
12: Specify input and output difference in a truncated level.
13: **end if**
14: **end if**

15: **for** round = 1 to r **do**

/* Lines 16–20 correspond to $\_\__C$ in Alg. 1. */
16: **if** TARGET = ID and MODE = ARBITRARY **then**
17: Write constraints for the differentially ideal S-box.
18: **else**
19: Write constraints for the S-boxes as in specification.
20: **end if**

21: Write constraints for the linear layer.
22: **end for**

users need to modify only two parts; make the objective function empty and specify input and output differences.

For impossible differentials, the users can further choose several search modes specified in the parameter "MODE". To be more precise, the S-boxes can be fixed to particular ones (SPECIFIC) or can be treated as more general ones (ARBITRARY).

The users can also choose which of truncated differential (TRUNCATED) or a single impossible differential characteristic (CHARACTERISTIC) is searched as a parameter "OBJECT".

The updated framework to generate the system of inequalities for each setting is given in Algorithm 2. Note that the basic idea in [18] corresponds to "GOAL = ID", "MODE = SPECIFIC", and "OBJECT = CHARACTERISTIC." In the following sections, we will discuss the purpose of each search mode.

Hereafter, we explain details of impossible differential search ("GOAL=ID"). We first explain how to search impossible differential characteristics ("OBJECT =CHARACTERISTIC") with the specific S-box mode and the arbitrary S-box mode in Sect. 3.2 and 3.3, respectively. We then explain the case of truncated impossible differential ("OBJECT=TRUNCATED") in Sect. 3.4.

3.2 Specific S-box Mode for Impossible Characteristic

In the specific S-box mode, the users derive the differential distribution table (DDT) from the actual S-boxes, and construct the MILP model to describe all valid differential propagations by using the existing method [12, 23, 21]. Then differences in all input and output bits are constrained to the target pair. The analysis is iterated for various input and output differences chosen from a reasonable subset, i.e. only one word is active.

The specific S-box mode can maximize the number of rounds of impossible differentials. Thus the attackers may prefer to choose this mode.

Impact of Key Schedule. The tool does not take into account the key schedule, thus we need a careful discussion about the impact of its omission.

The search by MILP describes a system of inequalities for the entire rounds by iterating a system of one-round differential propagation. Thus all valid propagations for one round are also valid in the evaluation of multiple rounds independently of the propagation in neighboring rounds and subkey values. This is true only if all subkeys are independent and chosen uniformly at random, which is not true in practical designs with a particular key schedule.

In summary, what the MILP simulates is the worst-case scenario (for the attackers). Namely, even if some differential propagations cannot occur for multiple rounds, the tool regards it possible, which leads to the following observation.

Observation 1 *Impossible differential characteristics found in the specific S-box mode are always impossible independently of the choice of key schedule.*

3.3 Arbitrary S-box Mode for Impossible Characteristic

In the arbitrary S-box mode, we assume an imaginary S-box in which any non-zero input difference can be propagated to any non-zero output difference. Then, a set of valid differential propagations of any bijective S-box can be a subset of the one in the arbitrary S-box. DDT for the 4-bit arbitrary S-box is shown in Table 13 in Appendix H.

Valid differential propagations of the n-bit arbitrary S-box can be described only by $2n$ inequalities. Let $i_0, i_1, \ldots, i_{n-1}$ and $o_0, o_1, \ldots, o_{n-1}$ be binary variables to represent active or inactive of input and output bits respectively. We write the constraints such that if input (resp. output) is 0, each output bit

(resp. input bit) is 0, namely

$$\begin{aligned} i_0 + i_1 + \cdots + i_{n-1} - o_0 \geq 0, \quad & o_0 + o_1 + \cdots + o_{n-1} - i_0 \geq 0, \\ i_0 + i_1 + \cdots + i_{n-1} - o_1 \geq 0, \quad & o_0 + o_1 + \cdots + o_{n-1} - i_1 \geq 0, \\ \cdots \quad & \cdots \\ i_0 + i_1 + \cdots + i_{n-1} - o_{n-1} \geq 0, \quad & o_0 + o_1 + \cdots + o_{n-1} - i_{n-1} \geq 0. \end{aligned}$$

The advantage of the arbitrary S-box compared to the specific S-box is efficiency owing to a small number of constrains to describe differential propagations. The arbitrary S-box mode is useful in the following two cases.

8-bit S-boxes: There is no known method to describe differential propagations of 8-bit S-boxes in MILP. Here by using the arbitrary S-box, the tool can be applied to 8-bit S-boxes.

Large Block Size: Even if the S-box size is small, say 4 bits, it is computationally hard to evaluate a large block size, say 256 bits. Again the arbitrary S-box enables analysis.

Note that, differently from the specific S-box mode, the analysis can no longer exploit properties inside the S-box. However, the analysis can still exploit another advantage that the tool catches any contradiction, and this advantage is often big enough to find new impossible differential characteristics. Actually, we found new characteristics of ARIA (8-bit S-boxes) [16] and of Minalpher (4-bit S-box, 256-bit block) [15], which will be explained in Sect. 4.

Similarly to Sect. 3.2, MILP simulates the worst-case scenario. Namely, even if some differential propagations cannot occur for some specific S-box, the tool regards it possible.

Observation 2 *Impossible differential characteristics found in the arbitrary S-box mode are always impossible independently of the choice of S-box and key schedule.*

3.4 Searching for Truncated Impossible Differential

The tool for a single characteristic can be extended to truncated differentials by simply running the tool for multiple pairs of input and output differences. However, this approach easily becomes computational infeasible when the number of active words is more than 1. Actually, searching for two active words is already too heavy. Let n and c be the number of S-boxes per round and the size of each S-box, respectively. Then, the number of pairs of input and output differences with 1-active word is $\left(n \cdot (2^c - 1)\right)^2$, which is $O(n^2 \cdot 2^{2c})$, while one with two active words is $\left(\binom{n}{2} \cdot (2^{2c} - 1)\right)^2$, which is $O(n^4 \cdot 2^{4c})$. Generally for d input active words and d' output active words, the number of pairs to test is given by

$$O(n^{d+d'} \cdot 2^{(d+d')c}). \tag{1}$$

With $n = 16$ and $c = 4$, which is a popular choice for lightweight ciphers, we need to evaluate 2^{16} pairs for single-active word ($d = d' = 1$) while 2^{32} pairs for 2-active words ($d = d' = 2$).

Here, we show a technique to make the tool more efficient only by aiming truncated impossible differentials in both of the specific S-box and the arbitrary S-box modes. Let $i_0, i_1, \ldots, i_{n-1}$ and $o_0, o_1, \ldots, o_{n-1}$ be variables to represent active or inactive of n input and n output bits to truncate. Then, we write the following constraint (along with constraints fixing the other bits to 0):

$$i_0 + i_1 + \cdots + i_{n-1} \geq 1, \qquad o_0 + o_1 + \cdots + o_{n-1} \geq 1.$$

Note that if there exists at least one solution satisfying the constraints, the tool returns that the system is feasible. Hence, the truncated impossible differential search is less accurate than the impossible characteristic search, while execution time is significantly reduced. Compared to Eq. (1), 1 inequality is enough for each active word position. Thus the number of pairs to test is given by

$$O(n^{d+d'}), \tag{2}$$

which enables to evaluate multiple active words differences. Actually, we searched for truncated impossible differentials on ARIA [16] with this technique. Then, we found new truncated impossible differentials with $d = 2$ active input words and $d' = 5$ output active words, which will be explained in Sect. 4.3.

4 Applications from Cryptanalysis Aspect

4.1 Midori128

Midori is a low energy block cipher designed by Banik et al. in 2015 [13]. Midori provides two different block lengths; Midori64 and Midori128 have 64-bit and 128-bit block lengths, respectively. Both ciphers accept 128-bit secret key.

Specification. Midori128 uses the SPN structure with AES-like state. The state is arranged in a 4×4 matrix as

$$S = \begin{pmatrix} s_0 & s_4 & s_8 & s_{12} \\ s_1 & s_5 & s_9 & s_{13} \\ s_2 & s_6 & s_{10} & s_{14} \\ s_3 & s_7 & s_{11} & s_{15} \end{pmatrix}. \tag{3}$$

The bit length of every cell s_i is 8 bits.

The round function consists of `SubCell`, `ShuffleCell`, `MixColumn`, and `KeyAdd`. In `SubCell`, 8-bit S-boxes $\mathtt{SSb}_0$, $\mathtt{SSb}_1$, $\mathtt{SSb}_2$, and $\mathtt{SSb}_3$ are used and $s_i \leftarrow \mathtt{SSb}_{i \bmod 4}(s_i)$ where $0 \leq i \leq 15$. Four 8-bit S-boxes $\mathtt{SSb}_i$ are constructed by 4-bit S-box $\mathtt{Sb}_1$, where $\mathtt{Sb}_1$ is defined as follows.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$\mathtt{Sb}_1(x)$	1	0	5	3	E	2	F	7	D	A	9	B	C	8	4	6

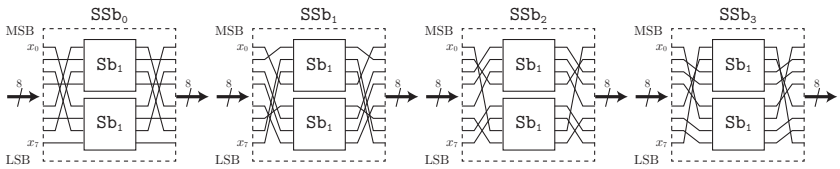

Fig. 1. $\mathtt{SSb}_0$, $\mathtt{SSb}_1$, $\mathtt{SSb}_2$, and $\mathtt{SSb}_3$

Then, $\mathtt{SSb}_i$ are constructed as $\mathtt{SSb}_i = p_i^{-1} \circ (\mathtt{Sb}_1 \| \mathtt{Sb}_1) \circ p_i$, where two $\mathtt{Sb}_1$ are applied to top and bottom halves in $(\mathtt{Sb}_1 \| \mathtt{Sb}_1)$. Note that $\mathtt{SSb}_i$ is involution, and we later show that impossible differentials are improved by exploiting this property. Figure 1 shows the specification of $\mathtt{SSb}_i$. In `ShuffleCell`, each cell is permuted as $(s_0, s_1, \ldots, s_{15}) \leftarrow (s_0, s_{10}, s_5, s_{15}, s_{14}, s_4, s_{11}, s_1, s_9, s_3, s_{12}, s_6, s_7, s_{13}, s_2, s_8)$. In `MixColumns`, the following multiplication

$$\begin{pmatrix} s_i \\ s_{i+1} \\ s_{i+2} \\ s_{i+3} \end{pmatrix} = \begin{pmatrix} 0\,1\,1\,1 \\ 1\,0\,1\,1 \\ 1\,1\,0\,1 \\ 1\,1\,1\,0 \end{pmatrix} \begin{pmatrix} s_i \\ s_{i+1} \\ s_{i+2} \\ s_{i+3} \end{pmatrix}$$

is applied for $i = 0, 4, 8, 12$. In `KeyAdd`, the i-th n-bit round key is XORed with a state. The number of rounds of Midori128 is 20. Moreover, only `SubCell` is applied in the final round function.

Previous Cryptanalysis. Several third-party cryptanalyses have been proposed, and the full-round Midori64 was broken by the invariant subspace attack [24] and nonlinear invariant attack [25] under the weak-key setting. On the other hand, there are no cryptanalysis against full-round Midori128. Regarding the impossible differential attack on Midori128, the designers found 6-round impossible differentials such that only one cell is active in the input and output [13]. Then, Zhen et al. found 6-round impossible differentials that are advantageous for the key recovery but the number of rounds is not increased [26].

Configurations for the Tool. The block size of Midori128 is 128 bits and the S-boxes size is 8 bits. However, since the 8-bit S-boxes are represented as concatenation of two 4-bit S-boxes, we can regard that there are thirty-two 4-bit S-boxes in each round. The search space for impossible differential characteristics is large, hence we run our tool in the arbitrary S-box mode.

When the arbitrary S-box mode is chosen for Midori, it is sufficient to evaluate truncated impossible differentials rather than impossible differential characteristics. This is because, for any choice of the differential value of the active nibble in the plaintext, the set of possible output differences of the active S-box in the first round is identical. In other words, when (Δ_i, Δ_o) is an impossible differential characteristic, for any other 1-nibble difference Δ_i' in the same active nibble position, (Δ_i', Δ_o) becomes impossible.

Table 2. 7-Round Truncated Impossible Differentials against Midori128

ID	ΔP	ΔC	Remarks
001T	$(0\alpha_1 00, 0000, 0000, 0000)$	$(0\beta_1 00, 0000, 0000, 0000)$	manually verified
002T	$(0\beta_1 00, 0000, 0000, 0000)$	$(0\alpha_1 00, 0000, 0000, 0000)$	manually verified
003T	$(0000, \alpha_0 000, 0000, 0000)$	$(0000, \beta_0 000, 0000, 0000)$	
004T	$(0000, \beta_0 000, 0000, 0000)$	$(0000, \alpha_0 000, 0000, 0000)$	
005T	$(0000, 0\alpha_1 00, 0000, 0000)$	$(0000, 0\beta_1 00, 0000, 0000)$	
006T	$(0000, 0\beta_1 00, 0000, 0000)$	$(0000, 0\alpha_1 00, 0000, 0000)$	
007T	$(0000, 0000, \alpha_0 000, 0000)$	$(0000, 0000, \beta_0 000, 0000)$	
008T	$(0000, 0000, \beta_0 000, 0000)$	$(0000, 0000, \alpha_0 000, 0000)$	
009T	$(0000, 0000, 0\alpha_1 00, 0000)$	$(0000, 0000, 0\beta_1 00, 0000)$	
010T	$(0000, 0000, 0\beta_1 00, 0000)$	$(0000, 0000, 0\alpha_1 00, 0000)$	
011T	$(0000, 0000, 0000, \alpha_0 000)$	$(0000, 0000, 0000, \beta_0 000)$	
012T	$(0000, 0000, 0000, \beta_0 000)$	$(0000, 0000, 0000, \alpha_0 000)$	

We limit the input and output differences to 1 active nibble. The number of such input differences is 32, and we have the same number of output differences. In the end, we run MILP for $32 * 32 = 1024$ pairs of input and output differences.

List of 7-Round Truncated Impossible Differentials. We ran our tool with the above configuration. The tool required about 0.03 seconds par pair and it took about 0.5 minutes to test 1024 pairs.

As a result, our tool found 12 truncated impossible differentials for 7 rounds, which improves the previous best result by 1 round. We list 12 truncated impossible differentials in Table 2. Note that α_i is active in 4 bits where the active bits go to top four bits after p_i is applied, while β_i is active in 4 bits where the active bits go to bottom four bits after p_i is applied. Every truncated impossible differential consists of $15^2 = 225$ impossible differential characteristics.

Manual Verification of ID001T and ID002T. Although one of the major advantages of the tool is that the attacker does not have to analyze the reason of contradiction, we would like to verify the reason. The analysis reveals a new structural property of Midori128 exploiting the involution of $\mathtt{SSb}_i$, which seems to be useful for future analysis. We first prove ID001T.

Theorem 1. *The input difference* $(0\alpha_1 00, 0000, 0000, 0000)$ *cannot propagate to the output difference* $(0\beta_1 00, 0000, 0000, 0000)$ *after 7 rounds of Midori128, where only top four bits of* $p_1(\alpha_1)$ *and bottom four bits of* $p_1(\beta_1)$ *are active.*

Proof. In Fig. 2, the input difference is propagated in forwards by 3.5 rounds, and the output difference is propagated in backwards by 3 rounds.

Let us focus on the forward propagation. From the definition, the differential form of α_1 is $(*, *, 0, 0, 0, 0, *, *)$ thus $p_1(\alpha_1) = (*, *, *, *, 0, 0, 0, 0)$, where $*$ and 0

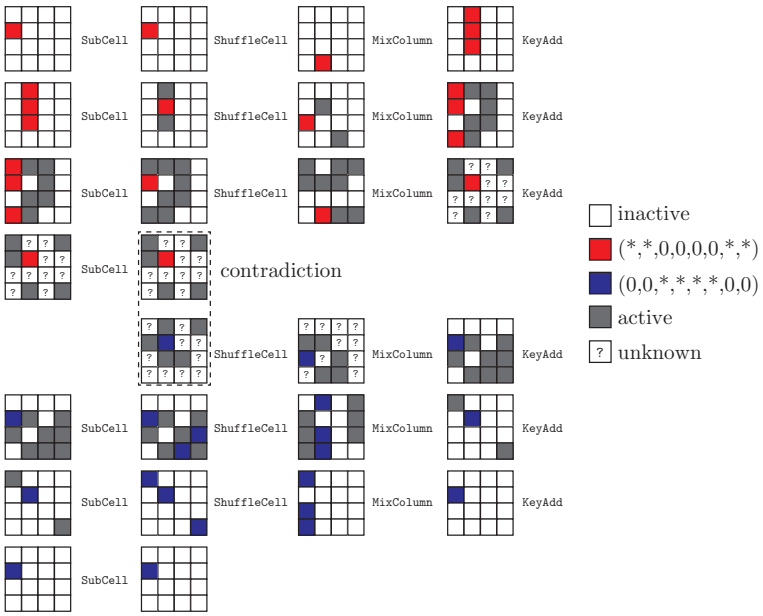

Fig. 2. 7-Round Truncated Impossible Differential of Midori128; ID001T

are active and inactive, respectively. In `SubCell` in the first round, $\mathtt{SSb}_1(\alpha_1) = p_1^{-1} \circ (\mathtt{Sb}_1 \| \mathtt{Sb}_1) \circ p_1(\alpha_1)$ is computed. $(\mathtt{Sb}_1 \| \mathtt{Sb}_1)$ preserves that only top 4 bits are active, and active bit positions go back to α_i after the application of p_1^{-1}. The position of the active byte moves from s_1 to s_7 by `ShuffleCell`, then is diffused to s_4, s_5, and s_6 by `MixColumns`. S-boxes are applied in the second round again, but $\mathtt{SSb}_0$ and $\mathtt{SSb}_2$ do not preserve the form of α_1 due to the different bit permutations p_0 and p_2. Therefore, only s_5 preserves the differential form of α_1. Similar analysis is continued during the 3.5-round forward propagation.

The differential form of β_1 is $(0, 0, *, *, *, *, 0, 0)$. With the same reason as α_1, the differential form of β_1 is preserved after the computation of $\mathtt{SSb}_1^{-1}(\beta_1)$, and 1 byte preserves the difference β_1 after 3 round decryption.

On one hand, from the forward 3.5-round propagation, only top half of $p_1(s_5)$ is active and bottom half is inactive. On the other hand, from the 3-round backward propagation, only bottom half of $p_1(s_5)$ is active and top half is inactive. This is a contradiction, therefore ID001T is manually verified. □

ID002T can be proved by exchanging the position of α_1 and β_1 of ID001T. Note that all impossible differentials found by our tool have the similar structure. Therefore, we expect that ID003T–ID012T can be verified similarly.

Table 3. S-box in LILLIPUT (Hex)

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

Table 4. Nibble Permutation (Decimal)

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

4.2 LILLIPUT

LILLIPUT is a lightweight block cipher designed by Berger et al. in 2015 [14] in which the block size and the key size are 64 bits and 80 bits, respectively. LILLIPUT adopts an extended generalized Feistel network (EGFN) [27].

Specification. A 64-bit plaintext is loaded to a 64-bit state X^0, which is divided into sixteen 4-bit nibbles, $X^0_{15}\|X^0_{14}\|\cdots\|X^0_0$. The round function, RF, takes as input a previous state X^j and a 32-bit subkey $SK^j \triangleq SK^j_7\|SK^j_6\|\cdots SK^j_0$ and updates the state to X^{j+1} with three operations $\mathcal{F}$, $\mathcal{L}$, and $\mathcal{P}$.

Non-linear layer $\mathcal{F}$: Copy the right half of the state, XOR the subkey, apply an S-box to each nibble, finally XOR the results to the left half of the state. Namely, $X^j_{8+i} \leftarrow X^j_{8+i} \oplus S(X^j_{7-i} \oplus SK^j_i)$, $i = 0, 1, \ldots, 7$, where $S(\cdot)$ is a 4-bit S-box defined in Table 3.

Linear layer $\mathcal{L}$: Update the left half of the state with several XORs.

$$X^j_{15} \leftarrow X^j_{15} \oplus X^j_7 \oplus X^j_6 \oplus X^j_5 \oplus X^j_4 \oplus X^j_3 \oplus X^j_2 \oplus X^j_1,$$
$$X^j_{15-i} \leftarrow X^j_{15-i} \oplus X^j_7 \text{ for } i = 1, 2, \ldots, 6.$$

Permutation layer $\mathcal{P}$: Permute nibble positions with π defined in Table 4.

The round function is iterated 30 times in which the permutation π is omitted in the last round. Because we are discussing distinguishers in which several rounds will be added for the key recovery, we do not omit the last permutation. The illustration of the round function can be seen in Fig. 5.

Previous Impossible Differential. The designers searched for truncated impossible differentials with $\mathcal{U}$-method [5] and found two 8-round truncated impossible differentials, e.g. the input difference $(0, 0, 0, 0, 0, 0, 0, \alpha, 0, 0, 0, 0, 0, 0, 0, 0)$ is incompatible with the output difference $(0, 0, 0, \beta, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$. We stress that the designers searched for them independently of the S-box choice.

Configurations for the Tool. Because both of the block size and the S-box size are small in LILLIPUT, we run our tool in the specific S-box mode to maximize the number of rounds of the distinguisher. In our experiment, we limited the input and output differences to only 1 active nibble.

Considering the Feistel network, having an active nibble in the left half of the input and in the right half of the output can maximize the number of rounds.

Table 5. 9-Round Impossible Differential Characteristics against LILLIPUT

ID	$(\Delta L^0, \Delta R^0)$	$(\Delta L^9, \Delta R^9)$	Remarks
001 - 015	$(0000000\alpha, 00000000)$	$(00000000, 00000\alpha00)$	manually verified
016 - 030	$(000000\alpha0, 00000000)$	$(00000000, 00\alpha00000)$	
031 - 045	$(000000\alpha0, 00000000)$	$(00000000, 0000000\alpha)$	
...	...	...	...
181 - 195	$(000\alpha0000, 00000000)$	$(00000000, 0000000\alpha)$	
196	$(00000020, 00000000)$	$(00000000, 00000200)$	manually verified
197	$(00000030, 00000000)$	$(00000000, 00000300)$	manually verified
198	$(00000080, 00000000)$	$(00000000, 00000800)$	manually verified
199	$(00000090, 00000000)$	$(00000000, 00000900)$	manually verified
200	$(000000e0, 00000000)$	$(00000000, 00000e00)$	manually verified
201	$(000000f0, 00000000)$	$(00000000, 00000f00)$	manually verified
202	$(00007000, 00000000)$	$(00000000, 00000700)$	
203	$(0000e000, 00000000)$	$(00000000, 00000e00)$	
204 - 216	$(000\beta0000, 00000000)$	$(00000000, 000000\beta0)$	manually verified
217	$(00010000, 00000000)$	$(00000000, 00000050)$	

The number of such input differences is $8 * 15 = 120$, where 8 is for the active nibble position and 15 is for non-zero difference in the active nibble. The number of output differences is the same. In the end, we run MILP for $120 * 120 = 14400$ pairs of input and output differences.

List of 9-Round Impossible Differential Characteristics. We ran our tool with the above configuration. The tool required about 0.2 seconds par pair and it took about 1 hour to test 14400 pairs.

As a result, we found 217 impossible differential characteristics for 9 rounds, which improves the previous best result by 1 round. We list a part of 217 impossible characteristics in Table 5. The complete list is available in Table B in Appendix B. Note that α in the impossible characteristics with ID 001 to 195 can be any non-zero value but must be the same between input and output. β in ID 204 to 216 can be 1,2,3,4,5,6,7,8,9,10,11,14, or 15.

Manual Verification of ID196 to ID201. Because some of detected impossible characteristics exploit the property of DDT, the analysis is completely different from the previous truncated impossible differentials. Verifying ID001–ID015 is relatively simple (but cannot be detected by the previous tools), which actually does not use the property inside the S-box.[1] The proof of ID001–ID015 is given in Appendix C, and we expect ID016–ID195 can be proven similarly.

[1] We realized this fact only after we finished manual verification. The tool outputs a list of 217 pairs, and at that time we had no clue about the contradicting reason.

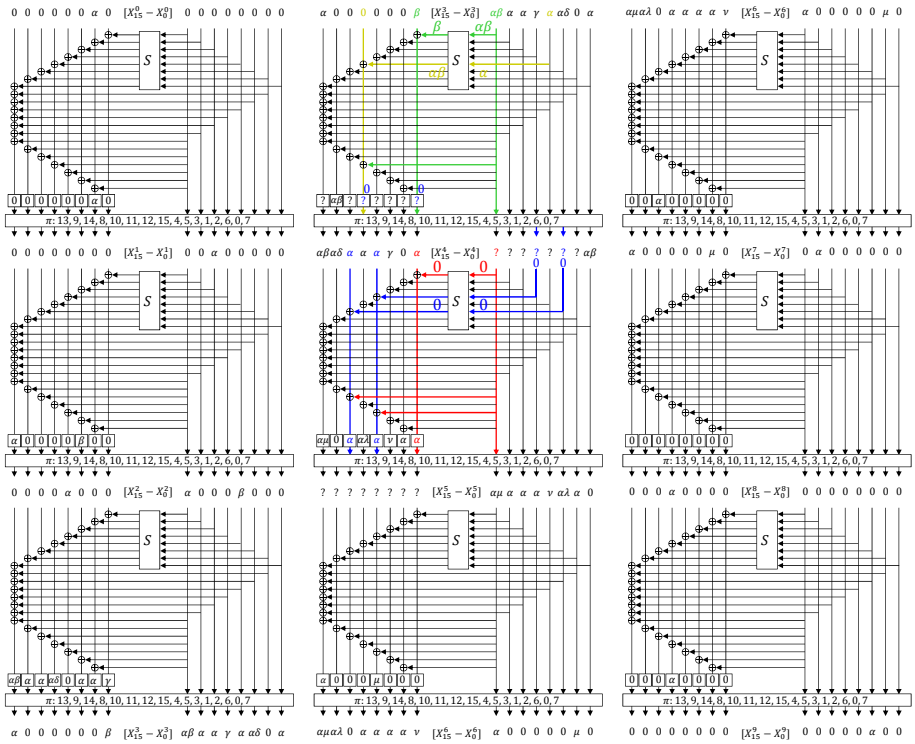

Fig. 3. 9-Round Impossible Differential Characteristic of LILLIPUT; ID196–201

ID196–ID201 essentially exploit the differential property of the S-box. Here, we explain the details of the contradicting reasons of ID196–ID201.

Theorem 2. *The input difference* $(000000\alpha0, 00000000)$ *cannot propagate to the output difference* $(00000000, 00000\alpha00)$ *after 9 rounds of* LILLIPUT*, where* $\alpha \in \{2, 3, 8, 9, e, f\}$.

Proof. In Fig. 3, the input (resp. output) difference is propagated in forwards (resp. backwards) by 4 rounds. We first focus on the forward propagation.

- In the second round, we denote by β the output difference of the active S-box. Note that β may or may not be equal to α.
- In the third round, we further introduce γ and δ for the output difference from the S-boxes. In Fig. 3, we denote by $\alpha\beta$ and $\alpha\delta$ abbreviations of $\alpha \oplus \beta$ and $\alpha \oplus \delta$ respectively. Note that $\alpha \oplus \beta$ and $\alpha \oplus \delta$ may or may not be non-zero.
- In the forth round, difference is unknown in many nibbles, denoted by '?'.

We do the same for the last 4 rounds and detect the contradiction in the middle.

1. We focus on $X_8^4 \oplus S(X_7^4) = X_4^5$ in the fifth round, in which $\Delta X_8^4 = \Delta X_4^5 = \alpha$, which eventually leads to $\Delta X_7^4 = 0$ (red lines in Fig. 3).
2. We then focus on $X_{11}^4 \oplus S(X_4^4) \oplus X_7^4 = X_1^5$, in which $\Delta X_{11}^4 = \Delta X_1^5 = \alpha$ and $\Delta X_7^4 = 0$. Hence, $\Delta X_4^4 = 0$. Similarly, $\Delta X_2^4 = 0$ (blue in Fig. 3).
3. We focus on $X_8^3 \oplus S(X_7^3) = X_4^4$ in the fourth round, in which $\Delta X_8^3 = \beta$ and $\Delta X_4^4 = 0$. Hence $\Delta S(X_7^3)$ must be β while $\Delta X_7^3 = \alpha \oplus \beta$ (green in Fig. 3). Considering that β is originally defined as an output difference of the S-box whose input difference is α, we have the following necessary condition for this 9-round characteristic to be possible.

$$\exists \beta, x, y : \begin{cases} S(x) \oplus S(x \oplus \alpha) = \beta \\ S(y) \oplus S(y \oplus \alpha \oplus \beta) = \beta \end{cases} \tag{4}$$

   Whether this condition is satisfied or not depends on the S-box, especially on its DDT, which is shown in Table 14 in Appendix for LILLIPUT.
   When $\alpha = 9$, β can be $3, 7, 8, 9, c, e, f$ for the first equation in (4). Then, $(\alpha \oplus \beta, \beta)$ can be computed as $(a, 3), (e, 7), (1, 8), (0, 9), (5, c), (7, e), (6, f)$. The second equation in (4) constrains that one of them must be a valid propagation. From DDT in Table 14, all of them cannot occur, which proves that the 9-round characteristic in Fig. 3 is impossible when $\alpha = 9$. Note that the condition (4) can be satisfied when $\alpha \neq 0, 9$.
4. We then further focus on $X_{12}^3 \oplus S(X_3^3) \oplus X_7^3 = X_2^4$ in the fourth round. $\Delta X_{12}^3 = \Delta X_2^4 = 0$ and $\Delta X_7^3 = \alpha \oplus \beta$, which derives $\Delta S(X_3^3) = \alpha \oplus \beta$. Meanwhile, $\Delta X_3^3 = \alpha$ (yellow in Fig. 3). Thus besides (4), we obtain the following necessary condition.

$$\exists z : S(z) \oplus S(z \oplus \alpha) = \alpha \oplus \beta \tag{5}$$

   To avoid redundancy, we omit listing all candidates, but from DDT conditions (4) and (5) cannot be satisfied simultaneously when $\alpha \in \{2, 3, e, f\}$.
5. To prove the case $\alpha = 8$, we further proceed the analysis. Because it requires too much details, we write the full proof in Appendix D.

With the above argument, Theorem 2 is proven. □

Remarks. We would like to emphasize once again that the advantage of our tool is that we can obtain a list of all impossible differential characteristics without considering the contradicting reason. We also manually verified ID204 to ID216, while we could not catch the contradicting reason for ID202, ID203, and ID217 by hand. In particular, ID217 is the only pair that the difference of active nibbles in the input and output are different. We leave their verification open.

4.3 ARIA

ARIA is a 128-bit block cipher and provides three secret-key lengths: 128, 192, and 256 bits [16]. ARIA is standardized by Korean Agency for Technology and

Standards (KATS) and is described by RFC5794 and RFC6209. ARIA uses Substitution-Permutation Network (SPN) structure, and the state is represented by 16 bytes. The round function consists of Substitution layer SL and Diffusion layer DL. Appendix H.3 shows the specification of ARIA.

Previous Cryptanalysis. Wu et al. proposed a truncated impossible differential on 4.5-round ARIA ($(DL \circ SL)^4 \circ DL$) as

$$(0,0,0,a,a,0,a,0,a,a,0,0,0,a,a,0) \xrightarrow{4.5R} (0,h,0,0,0,0,0,0,h,h,h,0,0,0,h,0),$$

where a and h denote any non-zero difference. Based on it, they attacked 6-round ARIA ($SL \circ (DL \circ SL)^5$) [28]. Then, Li et al. showed new truncated impossible differentials on 4.5-round ARIA and the data-time tradeoff for the attack on 6-round ARIA [29]. One of Li's truncated impossible differentials improved Wu's by reducing the number of active output bytes to 4 from 5, implying that the number of involved subkeys is less, and the time complexity is improved. However, the data complexity is greater than the time complexity. The total complexity is not very improved. Another Li's truncated impossible differential is

$$\begin{aligned}(0,b,0,a\oplus b,a\oplus b,0,a,0,a,a\oplus b,b,0,0,a,a\oplus b,b)\\ \xrightarrow{4.5R} (0,h,0,0,0,h,0,0,0,0,0,0,h,0,h,0),\end{aligned}$$

where a, b, and h denote any non-zero difference. This contributes to reducing the data complexity because the number of independent non-zero differences increases. Unfortunately, the number of involved subkeys increases to 14, and the time complexity is greater than the data complexity. In the end, the total complexity is not very improved.

Configurations for the Tool. Since the S-boxes size of ARIA is 8 bits, we run our tool in the arbitrary S-box mode. Similar to Midori128, we only execute truncated impossible differential search. Our goal is to improve Li's truncated impossible differentials. Namely, we search for 4.5-round truncated impossible differentials, where input and output differences take 3 independent differences and the number of involved subkey is reduced from 14. To search such truncated impossible differentials efficiently, our tool searches for truncated impossible differentials for 3.5 rounds ($SL \circ (DL \circ SL)^3$), where every active byte can take any difference. Then, found truncated differentials are trivially extended to 4.5 rounds by applying DL to the begging and end. Finally, we evaluate the number of input and output differences.

4.5-Round Truncated Impossible Differentials. We ran our tool with the above configuration. As a result, we found a truncated impossible differential as

$$\begin{aligned}(a,0,0,0,0,0,0,a,0,a,0,a,a,0,a,0) &\xrightarrow{DL} (0,a,0,0,a,0,0,0,0,0,0,0,0,0,0,0)\\ &\xrightarrow{3.5R} (h,g,0,0,0,0,0,h\oplus g,0,h\oplus g,0,g,0,0,0,0)\\ &\xrightarrow{DL} (h\oplus g,0,0,0,h\oplus g,h,h,0,0,h,0,0,0,g,0,g)\end{aligned}$$

where a, h, and g are non-zero differences. The number of involved subkeys is 13, and it decreases by one byte from that of Li's truncated impossible differentials. It implies that we can improve the time complexity of their key recovery attack.

4.4 Minalpher

Minalpher is an authenticated encryption scheme designed by Sasaki et al. in 2015 [15]. Minalpher uses 256-bit core permutation called Minalpher-P, which is based on Substitution-Permutation Network (SPN) structure using 4-bit S-boxes. Appendix H.4 shows the specification of Minalpher-P.

Previous Cryptanalysis. The designers found 6.5-round truncated impossible differentials by using the $\mathcal{U}$-method by Kim et al. These are the longest impossible differentials discovered by the $\mathcal{U}$-method.

Configurations for the Tool. While the S-boxes size is 4 bits, the block size, i.e., 256 bits, is very large. Therefore, we run our tool in the arbitrary S-box mode aiming truncated impossible differentials with 1 active nibble in the input and output differences. The number of such differences is 64 for both of input and output. In the end, we run MILP for $64 * 64 = 4096$ pairs.

List of 7.5-Round Truncated Impossible Differentials. The tool required about a few seconds per pair. As a result, our experiment found 1152 truncated impossible differentials for 7.5 rounds, which improves the previous best truncated impossible differentials by 1 round. We list 1152 truncated impossible differentials in supporting material, and Table 6 shows several examples. Column ΔP shows the position of the active nibble in plaintext, and column ΔC shows the position of the active nibble in ciphertext. Every truncated impossible differential consists of $15^2 = 225$ impossible differential characteristics.

4.5 MIBS

MIBS is a lightweight block cipher designed by Izadi et al. in 2009 [17]. The block length is 64, and it provides two key lengths: 64- and 80-bit secret key. Appendix H.5 shows the specification of MIBS.

Table 6. 7.5-Round Truncated Impossible Differentials of Minalpher-P

ID	ΔP	ΔC	Remarks
0001T	$A[0][0]$	$A[0][2]$	manually verified
0002T	$A[0][0]$	$A[0][3]$	
0003T	$A[0][0]$	$A[0][4]$	
0004T	$A[0][0]$	$A[0][5]$	
⋮	⋮	⋮	
1152T	$B[3][7]$	$B[3][7]$	

Previous Cryptanalysis. Bay et al. found two 8-round truncated impossible differentials [30]. Then, Wu and Wang found four additional 8-round truncated impossible differentials [8].

Configurations for the Tool. The block size of MIBS is 64 bits and the S-boxes size is 4 bits. Therefore, we run our tool in the specific S-box mode to maximize the number of rounds of the distinguisher. In our experiment, we limited the input and output differences to only 1 active nibble.

Considering the Feistel network, the number of differences we need to test is exactly the same as the case of LILLIPUT in Sect. 4.2. Thus we run MILP for $120 \times 120 = 14400$ pairs of input and output differences.

List of 8-Round Impossible Differential Characteristics. The tool required about 7.7 seconds par pair using single core and it took about 30 hours to test 14400 pairs.

Our tool found six 8-round truncated impossible differentials, which are the same as results by Wu's method. However, our method additionally found 2×120 impossible differential characteristics, which are not nibble-oriented truncated impossible differentials. We list all impossible differentials in Table 7, where α and β are any non-zero value. ID001–ID240 are impossible differential characteristics that our tool newly found. If the differences (γ, ϵ) takes differences that are shown by `x` in Table 8, the pairs of input and output differences is impossible differential characteristics.

5 Differential Possibility Equivalence Technique

In Sect. 4.1, we searched for all truncated impossible differentials with one active nibble. However, since `ShuffleCell` and `MixColumn` in Midori128 are byte-wise operations, we should search for all impossible characteristics with one active byte if possible. Moreover, the search in Sect 4.1 never exploited the property of $\mathtt{Sb}_1$ because the tool was run in the arbitrary S-box mode. This section explains how to run the tool in the specific S-box mode in a feasible time.

Table 7. 8-Round Impossible Differential Characteristics against MIBS

ID	ΔP	ΔC	Remarks
001T	$(00000000, 000000\alpha 0)$	$(0000\beta 000, 00000000)$	Bay
002T	$(00000000, 0000\alpha 000)$	$(000000\beta 0, 00000000)$	Wu
003T	$(00000000, 00\alpha 00000)$	$(0000000\beta, 00000000)$	Bay
004T	$(00000000, 0000000\alpha)$	$(00\beta 00000, 00000000)$	Wu
005T	$(00000000, 00\alpha 00000)$	$(0000\beta 000, 00000000)$	Wu
006T	$(00000000, 0000\alpha 000)$	$(00\beta 00000, 00000000)$	Wu
001-120	$(00000000, 000\gamma 0000)$	$(00000\epsilon 00, 00000000)$	
121-240	$(00000000, 00000\epsilon 00)$	$(000\gamma 0000, 00000000)$	

Table 8. Pairs of Impossible Differences Found by Our Tool for MIBS

		ϵ													
		1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
	1	x	x	x	0	x	x	0	0	0	x	0	0	x	0	x
	2	0	x	0	x	x	x	0	x	x	0	0	0	x	x	0
	3	x	0	x	x	0	0	0	x	x	x	0	0	0	x	x
	4	x	x	0	x	0	0	0	0	0	x	x	x	x	x	0
	5	x	0	0	0	x	x	0	x	x	x	x	x	0	0	0
	6	x	0	x	x	0	x	x	0	x	0	x	0	x	0	0
γ	7	0	0	0	0	0	x	x	x	0	x	x	0	x	x	x
	8	x	x	x	0	x	0	x	x	0	0	x	0	0	x	0
	9	0	x	x	0	0	x	x	0	x	x	0	x	0	x	0
	a	0	x	0	x	x	0	x	0	x	x	x	0	0	0	x
	b	x	0	0	0	x	0	x	0	x	0	0	x	x	x	x
	c	0	0	x	x	x	0	x	x	0	x	0	x	x	0	0
	d	0	0	x	x	x	x	0	0	0	0	x	x	0	x	x
	e	0	x	x	0	0	0	0	x	x	0	x	x	x	0	x
	f	x	x	0	x	0	x	x	x	0	0	0	x	0	0	x

As described in Sect. 3.4, the number of all pairs with d input active words and d' output active words is $O(n^{d+d'} 2^{(d+d')c})$, where n and c are the number of S-boxes per round and the size of each S-box, respectively. If we want to evaluate all impossible differential characteristics on Midori128 with 1 byte active, our tool has to execute $16^2 \times 255^2 \approx 2^{24}$ MILP instances. Then, it takes about 200 days to complete all instances even if an MILP instance is solved within one second. Therefore, we need an efficient method to evaluate all instances.

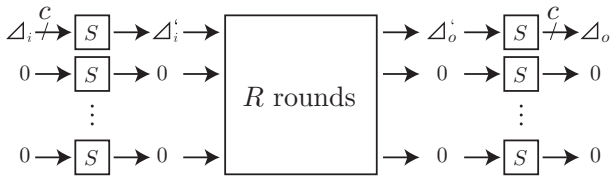

Fig. 4. Differential Possibility Equivalence Technique

5.1 Procedure of Differential Possibility Equivalence Technique

The *differential possibility equivalence technique* is useful technique to reduce the number of MILP instances that our tool has to solve.[2] Figure 4 shows the outline of the technique. Assuming that we search for impossible differential characteristics in which the first words of plaintexts and ciphertexts are active, we want to evaluate $(2^c - 1)^2$ pairs of input and output differences. First, we solve one MILP instance and obtain that $(\Delta_i \to \Delta_i' \to \Delta_o' \to \Delta_o)$ is possible differential characteristic for one tuple of $(\Delta_i, \Delta_i', \Delta_o', \Delta_o)$. Next, we evaluate a set $\mathcal{I}$ whose elements are all Δ such that $\Delta \to \Delta_i'$ is possible. Similarly, we evaluate a set $\mathcal{O}$ whose elements are all Δ such that $\Delta_o' \to \Delta$ is possible. Then, pairs in $(\mathcal{I} \times \mathcal{O})$ are always possible differential characteristics via (Δ_i', Δ_o'). Therefore, we do not need to evaluate all pairs in $(\mathcal{I} \times \mathcal{O})$ using MILP. Since the numbers of elements in $\mathcal{I}$ and $\mathcal{O}$ are $2^{c/2}$ on average, we can efficiently reduce the number of MILP instances that our tool has to solve.

We estimate the effectiveness of differential possibility equivalence technique.

Theorem 3. *Let n and c be the number of S-boxes per round and the size of each S-box, respectively. Our tool aims to find impossible differential with d input active words and d' output active words. Then, the number of trials that we have to solve MILP instances is $2^{d+d'}\left((d+d')\log_e(2^c-1)+O(1)\right)$ on average.*

The proof of theorem 3 is shown in Appendix G. Accurately, we can more efficiently collect N input and output differences than the estimation by Theorem 3 because every trial can always choose a pair without duplication. On the other hand, this error is not serious because N' differences are evaluated in the same time in one trial.

We searched for impossible differential characteristics with one active byte on Midori128 by using the differential possibility equivalence technique. In our experiment, this technique reduces the number of MILP instances that our tool has to solve from 2^{24} to $546865 \approx 2^{19}$. As a result, we found two new impossible differential characteristics, which are shown in Table 9. The total time that our

[2] The motivation of the differential possibility equivalence technique is quite different from truncated impossible differential. The truncated impossible search overlooks impossible characteristics only with one possible characteristic in the truncated set. When the number of impossible characteristics is small, truncated impossible differential search is not useful.

Table 9. 7-Round Impossible Differential Characteristics against Midori128

ID	ΔP		ΔC	
	Position	Value	Position	Value
001	s_1	0x04	s_8	0x43
002	s_1	0x0C	s_8	0x43

tool evaluates all impossible differential characteristics with one active byte is about 24 days in single core.

6 Applications from Design Aspect

6.1 Design Tool using Specific S-box Mode

Let us discuss using the tool for the design process of new primitives. Attack tools can always be used to evaluate how many rounds are attacked after the design is completed. Here we want to discuss a more interactive process. In many SPN-based designs, the designers evaluate many candidates with MILP and pick up the best choice. For example, the designers of Midori chose an almost-MDS matrix for `MixColumn`, and tested all parameters for `ShuffleCell`. Similarly, the designers of Skinny tested all light non-MDS matrices for `MixColumns` and the designers of Minalpher tested all parameters of a `ShiftRows`-like operation.

To run our tool in the specific S-box mode, S-boxes must be fixed in advance. This situation occurs when the choice of S-boxes has a high priority in the design. For example, Midori[13] chose the S-box with the lowest depth, and FIDES [31] and PICARO [32] chose the S-box that can be masked easily.

In our tool, all the components but for key schedule are simulated. Therefore, when we assume that subkeys are XORed to all words of the state before S-boxes, the tool can provide a certain level of proof, which is detailed below.

Observation 3 *Suppose that the tool does not find any impossible differential characteristic for r rounds after testing all paired input and output differences in a certain subset in the specific S-box mode. Then, the number of rounds of the longest impossible differential satisfying those input and output differences is at most $r-1$ by assuming that all subkeys are independent and chosen uniformly at random.*

Proof. Suppose that the tool can find specific differential propagations for given input and output differences. We now assume subkeys are XORed to all words of the state. Therefore, the output difference of any S-box are computed as

$$\Delta_o = S(x \oplus sk) \oplus S(x \oplus sk \oplus \Delta_i),$$

where Δ_i and Δ_o denote the input and output difference, respectively. The tool does not evaluate the value of x, but we now assume that all subkeys are

independent and chosen uniformly at random. Since Δ_o can take all possible output differences in DDT, the differential propagations that the tool finds are always valid in this assumption. □

If we can verify that all input and output differences with one active word are possible in the specific S-box mode, we say that the cipher is secure against impossible differential with one active word under the *subkey uniform assumption.*

Remarks about Proof in [18]. Cui et al. claimed that the tool can be used to prove the longest impossible differentials under the condition that input and output differences belong to the tested subset. After evaluating several ciphers, they claimed that *"we proof that the longest impossible differentials for LBlock, TWINE and Piccolo ciphers are really 14, 14 and 7 rounds respectively."* Unfortunately, Cui et al. are misinterpreting what the tool does.

In the evaluation with MILP, all valid propagations for one round are also valid in the evaluation of multiple rounds irrespectively of the propagation in neighboring rounds and subkey values. This is true only if all subkeys are independent and chosen uniformly at random. Therefore, even if no impossible differential is found for r rounds by MILP, it cannot ensure the non-existence for r rounds for real ciphers with particular key schedule.

6.2 Design Tool using Arbitrary S-box Mode

The arbitrary S-box mode is also useful for the design tool. When we run our tool in the specific S-box mode for the design tool, S-boxes must be fixed in advance. Meanwhile, if the choice of the linear layer has a higher priority, we would like to recommend the arbitrary S-box mode. The arbitrary S-box mode have two advantages: it can be executed before S-boxes are not specified and is generally more efficient than the specific S-box mode. In addition, the arbitrary S-box mode leads to several benefit to the designers.

Evaluating Linear Layer: The designers often test many choices of the S-boxes and of the linear layer. Because exhaustively testing all combinations is infeasible, the designers need to evaluate them independently. The arbitrary S-box mode finds impossible differential characteristics that are independent from the choice of the S-box, which makes possible to evaluate the security of the linear layer. In addition, the arbitrary S-box mode enables the designers to proceed the design of S-boxes and the design linear layer in parallel, which can shorten the design period.

Distinguishing Contradicting Reasoning: When impossible differentials are found for some rounds, the designers may prefer to patch the design or choose other design candidates. Then it is convenient for the designer to know whether the detected differentials can be prevented by changing S-boxes or not. In the arbitrary S-box mode, the contradiction is clearly caused by the linear layer.

Actually, impossible differential characteristics ID001–ID195 of LILLIPUT can be found by both the specific and arbitrary S-box modes, but the others ID196–ID217 can be found only by the specific S-box mode. Thus, we can immediately know ID001–ID195 are impossible differential characteristics independent of the choice of the S-box and cannot be prevented by replacing the S-box.

Similarly to Sect. 6.1, the fact that no impossible differential is found gives a certain level of security proof as follows.

Observation 4 *Suppose that the tool does not find any impossible differential characteristic for r rounds after testing all paired input and output differences in a certain subset in the arbitrary S-box mode. Then, the number of rounds of the longest impossible differential satisfying those input and output differences is at most $r1$ by assuming that all S-boxes are keyed bijective S-boxes that are independent and chosen uniformly at random.*

If we can verify that all pairs of input and output differences with one active word are possible in the arbitrary S-box mode, we say that the cipher is secure against impossible differential with one active word under the *keyed (uniform) bijective S-boxes assumption.*

6.3 Optimal Pick Technique; Application to MIBS

When ciphers have heavy diffusion layer, MILP solver requires too much time to verify whether or not a given pair of input and output differences is possible. For example, suppose that we evaluate resistance of MIBS against 9-round impossible differential. As discussed in Sect. 4.5, we need to test 14400 pairs of input and output differences. However, the tool could not finish the evaluation of 1 pair even after a couple of hours. Proving the security of 9-round MIBS with the direct application of our tool is infeasible.

Optimal Pick Technique. We propose an *optimal pick technique*, which dramatically reduces the computation time to prove the resistance against impossible differentials, i.e. to prove the existence of differential characteristic. Suppose that we are given a pair of input and output differences. The optimal pick technique well works when there are many differential characteristics satisfying a pair of given input and output differences. The intuition of this technique is as follows. We partially constrain the difference of the state in a middle round as well as the input and output differences. Suppose that our aim is to prove the resistance against r-rounds impossible differentials, and we expect that the proof is possible. Let X_{i-1} be a difference of the input of the i-th round. Our tool constrains a pair of input and output differences (X_0, X_r), and additional b bits of $X_{\lceil r/2 \rceil}$, where b is heuristically chosen. In our experiments, these additional constraints often reduce the execution time of the MILP solver. To prove the resistance against impossible differential, it is sufficient to find only one characteristic satisfying the constraint. Therefore, if the solver takes too long for a

Table 10. Provable Security against Impossible Differentials

Target	#Rounds	Assumption	Remarks
	8	subkey uniform	1 active byte
Midori128	8	keyed bijective 4-bit S-boxes	1 active byte
	7	keyed bijective 8-bit S-boxes	1 active byte
LILLIPUT	10	subkey uniform	1 active nibble
Minalpher	9.5	keyed bijective S-boxes	1 active nibble
ARIA	5	keyed bijective S-boxes	1 active byte
MIBS	9	subkey uniform	1 active nibble
SIMON	12	subkey uniform	1 active bit
TWINE	15	subkey uniform	1 active nibble
LBlock	15	subkey uniform	1 active nibble
Piccolo	8	subkey uniform	1 active nibble
RECTANGLE	9	subkey uniform	1 active nibble
Skinny-64	12	subkey uniform	1 active nibble
Midori64	7	subkey uniform	1 active nibble
CLEFIA	10	keyed bijective 8-bit S-boxes	1 active byte

choice of constrained b bits, we give up searching for the b bits, and test another b bits by expecting that the new b bits are easy to compute.

In application to 9-round MIBS, for pairs of input and output differences (X_0, X_9) we used the optimal pick technique with the following strategy.

- Four nibbles in X_4 are additionally constrained ($b = 16$).
- For all 2^{16} choices of additional constraints, we evaluate whether or not it is possible to satisfy (X_0, X_4, X_9). If the execution time reaches 10 seconds, we stop the evaluation and proceed the next additional constraints.
- Once we find an additional constraint X_4 satisfying the input and output differences (X_0, X_9), we return that the pair (X_0, X_9) is possible.

The second strategy is the essence of the optimal pick technique. The execution time of the MILP solver becomes too long for some choice of X_4, and the second strategy allows us to escape from the unlucky choice. As a result, we successfully proved that there is no 9-round impossible differential characteristics with one active nibble under the subkey uniform assumption. Note that the optimal pick technique only can be used for the proving approach, i.e. it cannot be used to find impossible differential characteristics because we terminate the MILP search when the execution time reaches 10 seconds.

6.4 List of Evaluated Designs

We proved the maximal number of rounds of impossible differential characteristics for many designs. Besides the already discussed five designs, we evaluated SIMON [33], TWINE [34], LBlock [35], Piccolo [36], RECTANGLE [10], Skinny [19], Midori64 [13], and CLEFIA [37] as shown in Table 10. We confirmed that

there are no impossible differential characteristics within the parameters of input and output differences in Remark column. For example, if we regard $\mathtt{SSb}_i$ in Midori128 as keyed 8-bit bijective S-boxes, we proved that there are no 7-round impossible differential characteristics with 1 active byte. However, if we exploit the structure of $\mathtt{SSb}_i$ and regard $\mathtt{Sb}_1$ as keyed 4-bit bijective S-boxes, 7-round impossible characteristics can be found as explained in Sect. 4.1. In such an assumption, we proved that there are no 8-round impossible differential characteristics with 1 active byte. Moreover, 8 rounds are also secure in the subkey uniform assumption.

References

1. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In Wu, C., Yung, M., Lin, D., eds.: Inscrypt 2011. Volume 7537 of LNCS., Springer (2011) 57–76
2. Knudsen, L.: DEAL - a 128-bit block cipher. Technical report no. 151. Department of Informatics, University of Bergen, Norway (1998)
3. Biham, E., Biryukov, A., Shamir, A.: Cryptanalysis of skipjack reduced to 31 rounds using impossible differentials. In Stern, J., ed.: EUROCRYPT '99. Volume 1592 of LNCS., Springer (1999) 12–23
4. Biryukov, A.: Miss-in-the-middle attack. In van Tilborg, H.C.A., ed.: Encyclopedia of Cryptography and Security. Springer (2005)
5. Kim, J., Hong, S., Sung, J., Lee, C., Lee, S.: Impossible differential cryptanalysis for block cipher structures. In Johansson, T., Maitra, S., eds.: INDOCRYPT 2003. Volume 2904 of LNCS., Springer (2003) 82–96
6. Luo, Y., Wu, Z., Lai, X., Gong, G.: A unified method for finding impossible differentials of block cipher structures. Cryptology ePrint Archive, Report 2009/627 (2009)
7. Luo, Y., Lai, X., Wu, Z., Gong, G.: A unified method for finding impossible differentials of block cipher structures. Inf. Sci. **263** (2014) 211–220
8. Wu, S., Wang, M.: Automatic search of truncated impossible differentials for word-oriented block ciphers. In Galbraith, S.D., Nandi, M., eds.: INDOCRYPT 2012. Volume 7668 of LNCS., Springer (2012) 283–302
9. Tezcan, C.: Improbable differential attacks on present using undisturbed bits. J. Computational Applied Mathematics **259** (2014) 503–511
10. Zhang, W., Bao, Z., Lin, D., Rijmen, V., Yang, B., Verbauwhede, I.: Rectangle: A bit-slice lightweight block cipher suitable for multiple platforms. Cryptology ePrint Archive, Report 2014/084 (2014) `http://eprint.iacr.org/2014/084`.
11. Derbez, P., Fouque, P.: Automatic search of meet-in-the-middle and impossible differential attacks. In Robshaw, M., Katz, J., eds.: CRYPTO 2016, Part II. Volume 9815 of LNCS., Springer (2016) 157–184
12. 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 Sarkar, P., Iwata, T., eds.: ASIACRYPT 2014 Part I. Volume 8873 of LNCS., Springer (2014) 158–178
13. 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.: ASIACRYPT 2015, Part II. Volume 9453 of LNCS., Springer (2015) 411–436

14. Berger, T.P., Francq, J., Minier, M., Thomas, G.: Extended generalized feistel networks using matrix representation to propose a new lightweight block cipher: LILLIPUT. IEEE Transactions on Computers **65** (2015) 2074–2089
15. Sasaki, Y., Todo, Y., Aoki, K., Naito, Y., Sugawara, T., Murakami, Y., Matsui, M.: Minalpher v1.1. Submitted to CAESAR (2015)
16. Kwon, D., Kim, J., Park, S., Sung, S.H., Sohn, Y., Song, J.H., Yeom, Y., Yoon, E., Lee, S., Lee, J., Chee, S., Han, D., Hong, J.: New block cipher: ARIA. In Lim, J.I., Lee, D.H., eds.: ICISC 2003. Volume 2971 of LNCS., Springer (2003) 432–445
17. Izadi, M., Sadeghiyan, B., Sadeghian, S.S., Khanooki, H.A.: MIBS: A new lightweight block cipher. In Garay, J.A., Miyaji, A., Otsuka, A., eds.: CANS 2009. Volume 5888 of LNCS., Springer (2009) 334–348
18. Cui, T., Jia, K., Fu, K., Chen, S., Wang, M.: New automatic search tool for impossible differentials and zero-correlation linear approximations. Cryptology ePrint Archive, Report 2016/689 (2016) `http://eprint.iacr.org/2016/689`.
19. 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.: CRYPTO 2016, Part II. Volume 9815 of LNCS., Springer (2016) 123–153
20. 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.: CHES 2007. Volume 4727 of LNCS., Springer (2007) 450–466
21. Sasaki, Y., Todo, Y.: New differential bounds and division property of LILLIPUT: Block cipher with extended generalized feistel network. In Avanzi, R., Heys, H., eds.: SAC 2016. LNCS, Springer (2016)
22. Inc., G.O.: Gurobi optimizer 6.5. Official webpage, `http://www.gurobi.com/` (2015)
23. 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. IACR Cryptology ePrint Archive **2014** (2014) 747
24. Guo, J., Jean, J., Nikolić, I., Qiao, K., Sasaki, Y., Sim, S.M.: Invariant subspace attack against full midori64. Cryptology ePrint Archive, Report 2015/1189 (2015) `http://eprint.iacr.org/2015/1189`.
25. Todo, Y., Leander, G., Sasaki, Y.: Nonlinear invariant attack –practical attack on full scream, iscream, and midori64. Cryptology ePrint Archive, Report 2016/732 (2016) `http://eprint.iacr.org/2016/732`.
26. Zhan, C., Xiaoyun, W.: Impossible differential cryptanalysis of midori. Cryptology ePrint Archive, Report 2016/535 (2016) `http://eprint.iacr.org/2016/535`.
27. Berger, T.P., Minier, M., Thomas, G.: Extended generalized feistel networks using matrix representation. In Lange, T., Lauter, K.E., Lisonek, P., eds.: SAC 2013. Volume 8282 of LNCS., Springer (2013) 289–305
28. Wu, W., Zhang, W., Feng, D.: Impossible differential cryptanalysis of reduced-round ARIA and camellia. J. Comput. Sci. Technol. **22**(3) (2007) 449–456
29. Ruilin Li, Bing Sun, P.Z., Li, C.: New impossible differential cryptanalysis of aria. Cryptology ePrint Archive, Report 2008/227 (2008) `http://eprint.iacr.org/2008/227`.
30. Bay, A., Jr., J.N., Vaudenay, S.: Cryptanalysis of reduced-round MIBS block cipher. In Heng, S., Wright, R.N., Goi, B., eds.: CANS 2010. Volume 6467 of LNCS., Springer (2010) 1–19

31. Bilgin, B., Bogdanov, A., Knezevic, M., Mendel, F., Wang, Q.: Fides: Lightweight authenticated cipher with side-channel resistance for constrained hardware. In Bertoni, G., Coron, J., eds.: CHES 2013. Volume 8086 of LNCS., Springer (2013) 142–158
32. Piret, G., Roche, T., Carlet, C.: PICARO - A block cipher allowing efficient higher-order side-channel resistance. In Bao, F., Samarati, P., Zhou, J., eds.: ACNS 2012. Volume 7341 of LNCS., Springer (2012) 311–328
33. Beaulieu, R., Shors, D., Smith, J., Treatman-Clark, S., Weeks, B., Wingers, L.: The simon and speck families of lightweight block ciphers. Cryptology ePrint Archive, Report 2013/404 (2013) `http://eprint.iacr.org/2013/404`.
34. Suzaki, T., Minematsu, K., Morioka, S., Kobayashi, E.: TWINE: A lightweight block cipher for multiple platforms. In Knudsen, L.R., Wu, H., eds.: SAC 2012. Volume 7707 of LCNS., Springer (2012) 339–354
35. Wu, W., Zhang, L.: Lblock: A lightweight block cipher. In Lopez, J., Tsudik, G., eds.: ACNS 2011. Volume 6715 of LNCS., Springer (2011) 327–344
36. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita, T., Shirai, T.: Piccolo: An ultra-lightweight blockcipher. In Preneel, B., Takagi, T., eds.: CHES 2011. Volume 6917 of LNCS., Springer (2011) 342–357
37. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In Biryukov, A., ed.: FSE 2007. Volume 4593 of LNCS., Springer (2007) 181–195
38. Fu, K., Wang, M., Guo, Y., Sun, S., Hu, L.: Milp-based automatic search algorithms for differential and linear trails for speck. In Peyrin, T., ed.: FSE 2016. Volume 9783 of LNCS., Springer (2016) 268–288

A Relationship between [18] and This Paper

Cui et al. [18] have recently posted their work to Cryptology ePrint Archive (received by ePrint Archive at 11 July 2016) presenting that impossible differentials can be searched with MILP.

Actually, though the basic idea of the tool is the same, two papers extend the basic idea to quite different directions. The main focus of [18] seems to be the extension to the ARX structure and zero-correlation cryptanalysis, which is not covered by our work. Meanwhile, we are focusing on the impossible differential cryptanalysis much deeper, and trying to extend the structure that can be evaluated by the tool. Therefore, we obtained new results even for 8-bit S-boxes, in which [18] left application to 8-bit S-box open.

Another difference is enthusiasm for the application to practical designs. Considering the number of applications, [18] seems to focus on the theoretical aspects, while we are trying to evaluate more and more targets and the usage of the tool for designing new primitives is another main focus. Moreover, all applications of [18] are limited to lightweight block ciphers, i.e., the block length is at most 64 bits, while we are trying a wide variety of block lengths: ARIA and Midori128 have 128-bit block length and Minalpher-P has 256-bit block length.

In the following, we compare the contents of two papers which are exclusive each other.

Advantages of [18] Over Our Work.

- By converting differential evaluation to linear evaluation, the tool is extended to zero-correlation approximations.
- By borrowing the idea by Fu et al. about MILP on the ARX structure [38], the tool is extended to the impossible differentials for the ARX structure.
- By applying the basic idea to PRESENT, new impossible differentials are recovered while the number of attacked rounds is not improved.
- By applying the extended tool to ARX, new impossible differentials and new zero-correlation approximations are discovered against HIGHT, which improves the previous best results by 1 round.

Advantages of Our Work Over [18].

- The arbitrary S-box mode to apply the tool to 8-bit S-box.
- Focusing on the property of the tool that it can catch any contradiction, which leads to find improvement of impossible differential using 8-bit S-box.
- More applications are examined and we improved the previous best results in several applications.
- Analyzing the contradicting reasons for the detected pairs and revealed the new structural properties that may be used in future analysis.
- More precise arguments for provable security.
- The differential possibility equivalence technique for the efficient search.
- The optimal pick technique for the efficient proof.

Time Stamp We finished implementation of the first application (on Simon) on 27 May 2016. The summary of our preliminary results was presented at the rump session of CRYPTO 2016. The complete paper was first submitted to Eurocrypt 2017.

B Complete List of 9-Round Impossible Differential Characteristics of LILLIPUT

Table 11. 9-Round Impossible Differential Characteristics against LILLIPUT

ID	$(\Delta L^0, \Delta R^0)$	$(\Delta L^9, \Delta R^9)$	Remarks
001 - 015	$(0000000\alpha, 00000000)$	$(00000000, 00000\alpha00)$	manually verified
016 - 030	$(000000\alpha0, 00000000)$	$(00000000, 00\alpha00000)$	
031 - 045	$(000000\alpha0, 00000000)$	$(00000000, 0000000\alpha)$	
046 - 060	$(000000\alpha0, 00000000)$	$(00000000, 000000\alpha0)$	
061 - 075	$(00000\alpha00, 00000000)$	$(00000000, 00\alpha00000)$	
076 - 090	$(00000\alpha00, 00000000)$	$(00000000, 0000000\alpha)$	
091 - 105	$(00000\alpha00, 00000000)$	$(00000000, 000000\alpha0)$	
106 - 120	$(00000\alpha00, 00000000)$	$(00000000, 00000\alpha00)$	
121 - 135	$(0000\alpha000, 00000000)$	$(00000000, 00\alpha00000)$	
136 - 150	$(0000\alpha000, 00000000)$	$(00000000, 0000000\alpha)$	
151 - 165	$(0000\alpha000, 00000000)$	$(00000000, 000000\alpha0)$	
166 - 180	$(000\alpha0000, 00000000)$	$(00000000, 00\alpha00000)$	
181 - 195	$(000\alpha0000, 00000000)$	$(00000000, 0000000\alpha)$	
196	$(00000020, 00000000)$	$(00000000, 00000200)$	manually verified
197	$(00000030, 00000000)$	$(00000000, 00000300)$	manually verified
198	$(00000080, 00000000)$	$(00000000, 00000800)$	manually verified
199	$(00000090, 00000000)$	$(00000000, 00000900)$	manually verified
200	$(000000e0, 00000000)$	$(00000000, 00000e00)$	manually verified
201	$(000000f0, 00000000)$	$(00000000, 00000f00)$	manually verified
202	$(00007000, 00000000)$	$(00000000, 00000700)$	
203	$(0000e000, 00000000)$	$(00000000, 00000e00)$	
204	$(00010000, 00000000)$	$(00000000, 00000010)$	manually verified
205	$(00020000, 00000000)$	$(00000000, 00000020)$	manually verified
206	$(00030000, 00000000)$	$(00000000, 00000030)$	manually verified
207	$(00040000, 00000000)$	$(00000000, 00000040)$	manually verified
208	$(00050000, 00000000)$	$(00000000, 00000050)$	manually verified
209	$(00060000, 00000000)$	$(00000000, 00000060)$	manually verified
210	$(00070000, 00000000)$	$(00000000, 00000070)$	manually verified
211	$(00080000, 00000000)$	$(00000000, 00000080)$	manually verified
212	$(00090000, 00000000)$	$(00000000, 00000090)$	manually verified
213	$(000a0000, 00000000)$	$(00000000, 000000a0)$	manually verified
214	$(000b0000, 00000000)$	$(00000000, 000000b0)$	manually verified
215	$(000e0000, 00000000)$	$(00000000, 000000e0)$	manually verified
216	$(000f0000, 00000000)$	$(00000000, 000000f0)$	manually verified
217	$(00010000, 00000000)$	$(00000000, 00000050)$	

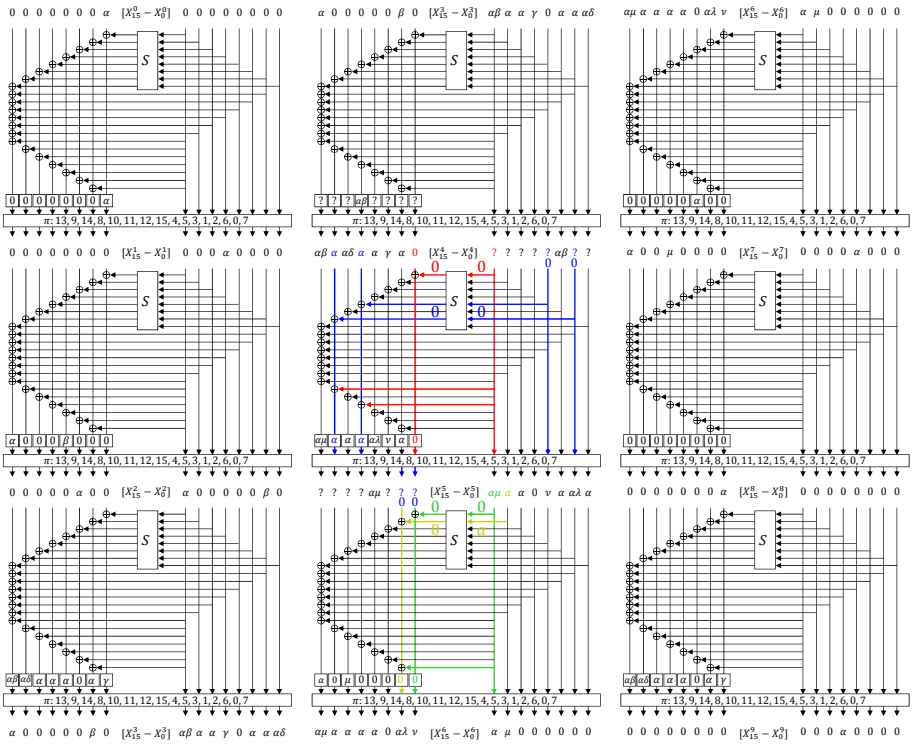

Fig. 5. 9-Round Impossible Differential Characteristic of LILLIPUT; ID001–015

C Proof of Contradicting Mechanism of ID001–ID015

Towards the better understandings about the EGFN, we analyze the mechanism of the construction. We begin with proving ID001 to 015, which seems the simplest of 217 cases.

Theorem 4. *The input difference* $(0000000\alpha, 00000000)$ *cannot propagate to the output difference* $(00000000, 00000\alpha00)$ *after 9 rounds of* LILLIPUT*, where* α *is any non-zero 4-bit value.*

Proof. In Fig. 5, the input (resp. output) difference is propagated in forwards (resp. backwards) by 4 rounds.

- The propagation in the first round is straightforward.
- In the second round, we denote by β the output difference of the active S-box. Note that β may or may not be equal to α.
- In the third round, we further introduce γ and δ for the output difference from the S-boxes. In Fig. 5, we denote by $\alpha\beta$ and $\alpha\delta$ abbreviations of $\alpha \oplus \beta$ and $\alpha \oplus \gamma$ respectively. Note that it is unknown whether or not $\alpha \oplus \beta$ and $\alpha \oplus \gamma$ are non-zero.

- In the forth round, difference is unknown in many nibbles, denoted by "?".

We do the same for the last 4 rounds and detect the contradiction in the middle as follows, which is explained in 4 steps.

1. We focus on $X_8^4 \oplus S(X_7^4) = X_4^5$ in the fifth round, in which $\Delta X_8^4 = \Delta X_4^5 = 0$, thus $\Delta S(X_7^4) = 0$. Because the S-box in LILLIPUT is bijective, we can deduce that $\Delta X_7^4 = 0$. The network related to this step is colored by red in Fig. 5.
2. We then focus on $X_{12}^4 \oplus S(X_3^4) \oplus X_7^4 = X_2^5$ in the fifth round, in which $\Delta X_{12}^4 = \Delta X_2^5 = \alpha$ and we detected that $\Delta X_7^4 = 0$. Hence, $\Delta X_3^4 = 0$. With the same analysis, $\Delta X_1^4 = 0$. Those will move to X_8^5 and X_9^5 in the next round. The network related to this step is colored by blue.
3. We focus on $X_8^5 \oplus S(X_7^5) = X_4^6$ in the sixth round, in which $\Delta X_4^6 = 0$ and we detected that $\Delta X_8^5 = 0$. Hence $\Delta X_7^5 = 0$, which is colored by green.
4. Finally, we focus on $X_9^5 \oplus S(X_6^5) \oplus X_7^5 = X_5^6$ in the sixth round (colored by yellow). On one hand $\Delta X_9^5 = \Delta X_7^5 = \Delta X_5^6 = 0$ leads to $\Delta S(X_6^5) = 0$, on the other hand $\Delta X_6^5 = \alpha$, which is non-zero. This is contradiction which completes the proof of Theorem 4. □

D Proof of Contradicting Mechanism of ID198

In this section, we continue the proof of Theorem 2 when $\alpha = 8$. Please read the proof in Sect. 4.2 before read this section.

The proof in Sect. 4.2 revealed two necessary conditions (4) and (5). Although both conditions can be satisfied with $\alpha = 8$, the difference β can be uniquely fixed, namely $\beta = e$ and thus $\alpha \oplus \beta = 6$. (Remember that we are discussing the case of $\alpha = 8$.) This information will be utilized later.

In this proof, we take over three nibble differences determined in the proof in Sect. 4.2; $\Delta X_7^4 = 0$, $\Delta X_4^4 = 0$ and $\Delta X_0^4 = \alpha \oplus \beta = 6$. Those will move to ΔX_{15}^5, ΔX_{10}^5, and ΔX_{13}^5 respectively, which are illustrated in Fig. 6. We no longer analyze the first 4 rounds, thus Fig. 6 contains only the last 5 rounds.

Since the proof in Sect. 4.2 has 5 steps, we resume the proof from Step 6.

6. We focus on $X_{10}^5 \oplus S(X_5^5) \oplus X_7^5 = X_3^6$ in the sixth round, in which $\Delta X_{10}^5 = \Delta X_3^6 = 0$ and $\Delta X_7^5 = \alpha \oplus \mu$. Hence, $\Delta S(X_5^5) = \alpha \oplus \mu$ (green in Fig. 6). Considering that μ is originally defined as an output difference of the S-box whose input difference is α, we have the following necessary condition;

$$\exists \mu, x, y : \begin{cases} S(x) \oplus S(x \oplus \alpha) = \mu \\ S(y) \oplus S(y \oplus \alpha) = \alpha \oplus \mu, \end{cases} \tag{6}$$

   where α is now fixed to 8. Looking at DDT, μ can take only two choices;

$$(\alpha, \mu, \alpha \oplus \mu) = (8, 6, e), (8, e, 6) \tag{7}$$

7. We then prove the corollary that $\alpha \oplus \lambda \neq 0$. To achieve it, we recall that λ was originally defined as the output difference of the S-box whose input difference is μ (yellow in the seventh round). The proof is simple. Now μ is either 6 or e. From DDT, whichever the input difference is 6 or e, the output difference cannot be 8. Hence, $\lambda \neq 8$ which leads to $\alpha \oplus \mu = 8 \oplus \mu \neq 0$.

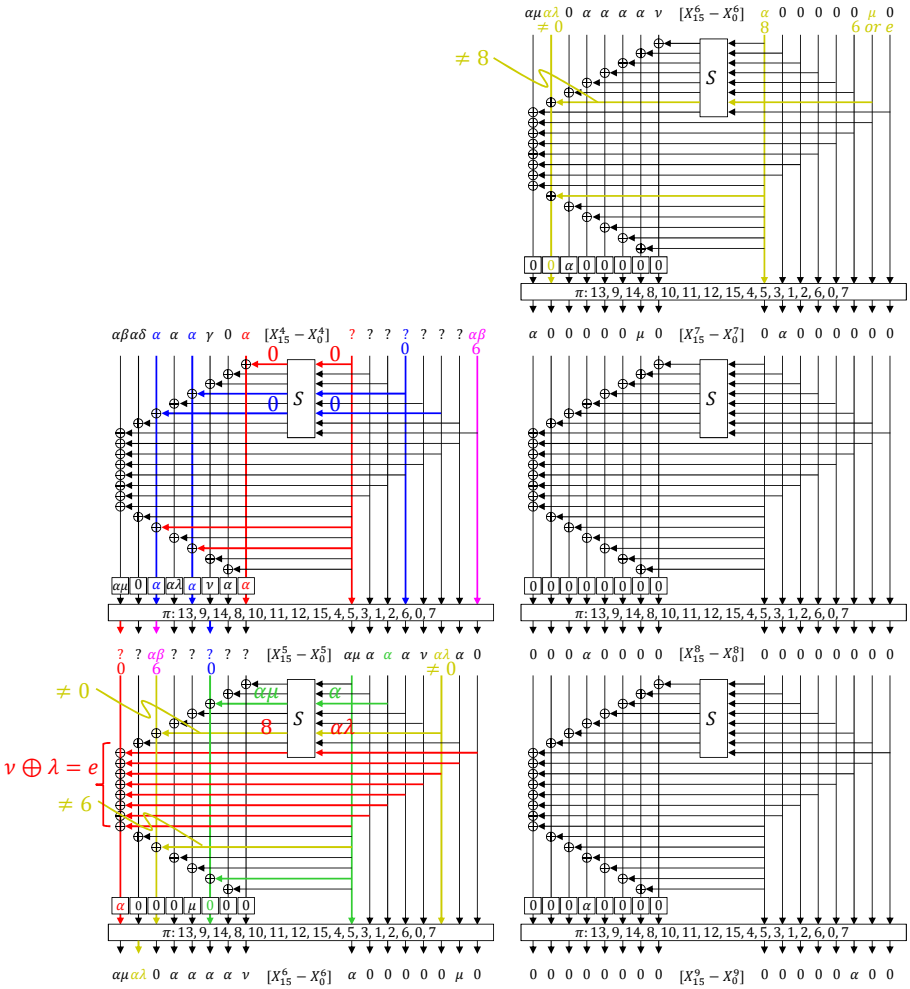

Fig. 6. The Last 5 Rounds in ID198

8. We focus on $X_{13}^5 \oplus S(X_2^5) \oplus X_7^5 = X_6^6$, in which $\Delta X_{13}^5 = 6$, $\Delta X_7^5 = \alpha \oplus \mu$, $\Delta X_6^6 = 0$. Hence, we deduce that $\Delta S(X_2^5) = 6 \oplus \alpha \oplus \mu$. Meanwhile, ΔX_2^5 is $\alpha \oplus \lambda$ which is known to be non-zero from the corollary above. Because the S-box is bijective, $6 \oplus \alpha \oplus \mu$ must be non-zero, which leads to $\alpha \oplus \mu \neq 6$. Combining with the two choices in (7), $(\alpha, \mu, \alpha \oplus \mu)$ is now uniquely determined to $(8, 6, e)$. The analysis in this step is colored by yellow in the sixth round.
9. Next, we focus on the left-most nibble in the sixth round (red), namely

$$X_{15}^5 \oplus S(X_0^5) \oplus X_1^5 \oplus X_2^5 \oplus X_3^5 \oplus X_4^5 \oplus X_5^5 \oplus X_6^5 \oplus X_7^5 = X_7^6.$$

By assigning the known differences, the equation becomes

$$0 \oplus 0 \oplus \alpha \oplus (\alpha \oplus \lambda) \oplus \nu \oplus \alpha \oplus \alpha \oplus \alpha \oplus e = \alpha,$$

which is converted into $\nu \oplus \lambda = e$. Considering that ν and λ are the output difference S-box whose output difference is $\alpha(= 8)$ and $\mu(= 6)$, from DDT, the possible candidates of $(\nu, \lambda) = (4, a), (2, c,), (3, d)$.

10. Finally, we focus on the 5th S-box in the sixth round (red) where the input difference is $\alpha \oplus \lambda(= 8 \oplus \lambda)$ and the output difference is $(\alpha \oplus \beta) \oplus (\alpha \oplus \mu) = 8$. For any of three choices of $\lambda = a, c, d$, from DDT, $8 \oplus \lambda(= 2, 4, 5)$ cannot produce the output difference 8. This completes the proof of the impossibility of ID198, and thus Theorem 2. □

E Proof of Contradicting Mechanism of ID0001T and ID0017T of Minalpher

In this section, we manually verify the contradicting mechanism of ID0001T and ID0017T of Minalpher.

We first show the contradicting mechanism of ID0001T because the reason is more simple than ID0017T. ID0001T shows that 7.5-round Minalpher has truncated impossible differential when $A[0][0]$ of plaintexts and $A[0][2]$ of ciphertexts are active and the others are inactive. Figure 7 shows the differential propagation. The focus is that the 3rd column in the composite function of `XM` and `MC` in Round 4. We get the input as $(\alpha, 0, \beta, \gamma, \delta, \epsilon, 0, 0)$ because of the propagation from plaintexts, where $\alpha \neq 0$, $\delta \neq 0$, and $\epsilon \neq 0$. Apply `XM` and `MC`, and the output is computed as

$$\begin{pmatrix} \alpha \oplus \gamma \\ \alpha \oplus \beta \\ \beta \oplus \gamma \\ \alpha \oplus \beta \oplus \gamma \\ \alpha \oplus \gamma \oplus \delta \oplus \epsilon \\ \alpha \oplus \beta \oplus \delta \oplus \epsilon \\ \beta \oplus \gamma \oplus \epsilon \\ \alpha \oplus \beta \oplus \gamma \oplus \delta \end{pmatrix}.$$

From $\alpha \oplus \beta \oplus \gamma = 0$, $\alpha \oplus \beta \oplus \gamma \oplus \delta = \delta \neq 0$. However, the propagation from ciphertexts derives $\delta = 0$, and it is contradictory.

Next, we show the contradicting mechanism of ID0017T. In this case, we need to use the case analysis. ID0017T shows that 7.5-round Minalpher has truncated impossible differential when $A[0][0]$ of plaintexts and $B[0][1]$ of ciphertexts are active and the others are inactive. Figure 8 shows the differential propagation. In the input difference of `XM` in Round 6, two nibbles labeled yellow is dependently active because of the propagation from ciphertexts.

First, assume that nibble labeled yellow is non active. The focus is that the 1st column in `MC` in Round 4. We get the input as $(\alpha, \beta, 0, \gamma)$ because of the

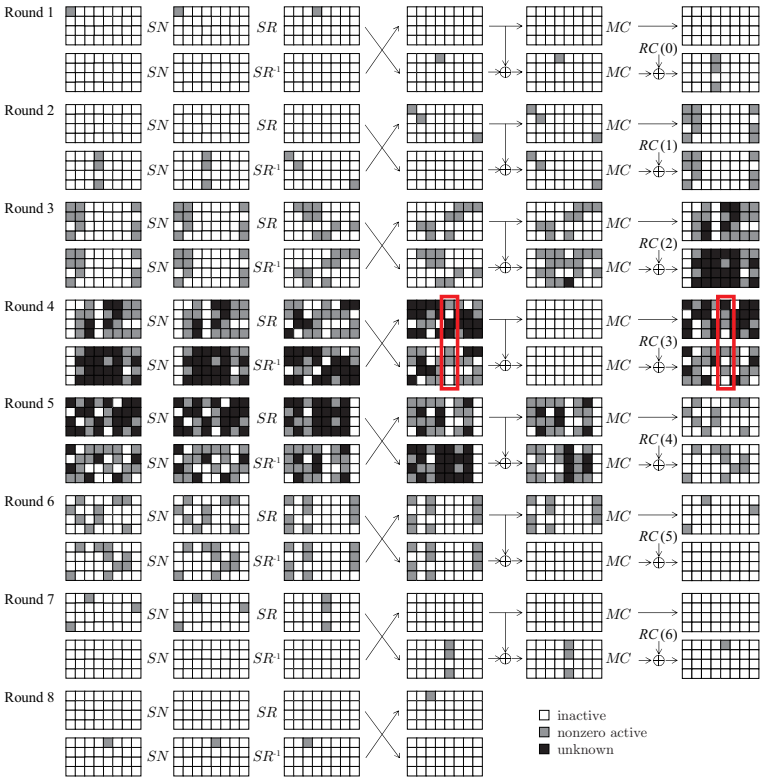

Fig. 7. Differential Propagation in ID0001T

propagation from plaintexts. Apply `MC`, and the output is computed as

$$\begin{pmatrix} \alpha \oplus \beta \oplus \gamma \\ \alpha \oplus \beta \\ \beta \oplus \gamma \\ \alpha \oplus \gamma \end{pmatrix}.$$

From $\alpha \oplus \beta = 0$, $\beta \oplus \gamma = \alpha \oplus \gamma$. However, the propagation from ciphertexts derives $\beta \oplus \gamma \neq 0$ and $\alpha \oplus \gamma = 0$, and it is contradictory.

Next, assume that nibble labeled yellow is active. The focus is that the 3rd column in the composite function of `XM` and `MC` in Round 4. We get the input as $(\alpha, \beta, \gamma, 0, 0, 0, \delta, \epsilon)$ because of the propagation from plaintexts, where $\beta \neq 0$,

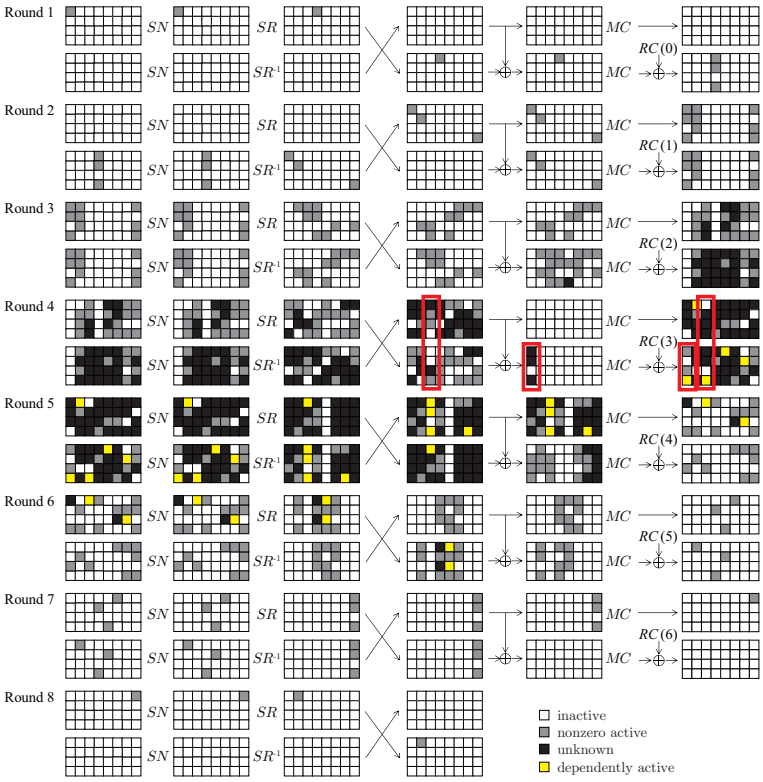

Fig. 8. Differential Propagation in ID0017T

$\gamma \neq 0$, and $\epsilon \neq 0$. Apply `XM` and `MC`, and the output is computed as

$$\begin{pmatrix} \alpha \oplus \beta \\ \alpha \oplus \beta \oplus \gamma \\ \beta \oplus \gamma \\ \alpha \oplus \gamma \\ \alpha \oplus \beta \oplus \epsilon \\ \alpha \oplus \beta \oplus \gamma \oplus \delta \\ \beta \oplus \gamma \oplus \delta \oplus \epsilon \\ \alpha \oplus \gamma \oplus \delta \oplus \epsilon \end{pmatrix}.$$

From $\alpha \oplus \beta = 0$, $\beta \oplus \gamma \oplus \delta \oplus \epsilon = \alpha \oplus \gamma \oplus \delta \oplus \epsilon$. However, the propagation from ciphertexts derives $\beta \oplus \gamma \oplus \delta \oplus \epsilon = 0$ and $\alpha \oplus \gamma \oplus \delta \oplus \epsilon \neq 0$, and it is contradictory.

F Complete List of 8-Round Impossible Differential Characteristics of RECTANGLE

RECTANGLE is a 64-bit PRESENT-like cipher designed by Zhang et al. [10]. The state has 64 bits represented by 4 rows and 16 columns. In each round, a 4-bit S-box is applied to each of 16 columns, and then bit position of row i is rotated to left by r_i bits, where $r_i = 0, 1, 12, 13$ for $i = 0, 1, 2, 3$ respectively.

In [10], the designers claim "We found some 8-round impossible differential distinguishers for RECTANGLE" but presented only 1 impossible characteristic. Hence, we give a complete list of impossible differential characteristics starting and ending with 1 active nibble.

By testing 28800 pairs in the specific S-box mode, we found 320 impossible differential characteristics against 8-round RECTANGLE. Due to the rotation property of the state, 320 patterns are rotated version of the following 20 impossible characteristics starting from the left most active column.

Table 12. 8-Round Impossible Differential Characteristics against RECTANGLE

ID	Δ_i	Δ_o	Remarks
01	(0000000000000003)	(0000000000000004)	
02	(0000000000000003)	(0000000000200000)	
03	(0000000000000005)	(0000000000002000)	
04	(0000000000000005)	(0000000000004000)	
05	(0000000000000005)	(0000000000020000)	
06	(0000000000000005)	(0000000000080000)	
07	(0000000000000005)	(0000000200000000)	
08	(0000000000000005)	(4000000000000000)	
09	(000000000000000b)	(0000000000000004)	
10	(000000000000000b)	(0000000000200000)	
11	(000000000000000c)	(0000000000000002)	
12	(000000000000000c)	(0000000000000004)	
13	(000000000000000c)	(0000000000000020)	
14	(000000000000000c)	(0000000000000080)	
15	(000000000000000c)	(0000000000200000)	
16	(000000000000000c)	(0004000000000000)	verified in [10]
17	(000000000000000e)	(0000000000004000)	
18	(000000000000000e)	(0000000200000000)	
19	(000000000000000f)	(0000000000004000)	
20	(000000000000000f)	(0000000200000000)	

G Proof of Theorem 3

Suppose that let n and c be the number of S-boxes per round and the size of each S-box, respectively. To search for impossible differential characteristics with d input active words and d' output active words, we first choose the positions of active words in both plaintexts and ciphertexts. Then, the number of possible input and output differences is $N = (2^c - 1)^{d+d'}$. From the coupon collector's problem, we can collect N pairs by trying NH_N input and output differences, where H_N is the N-th harmonic number. Moreover, since $N' = (2^{c-1})^{d+d'}$ input and output differences are evaluated from one trial in average, the number of trials that we have to solve MILP instances is represented as NH_N/N'. From $H_N = \log_e(N) + O(1)$,

$$\begin{aligned}
\frac{HH_N}{N'} &= \frac{N(\log_e(N) + O(1))}{N'} \\
&= \frac{(2^c - 1)^{d+d'}((d+d')\log_e((2^c - 1)) + O(1))}{(2^{c-1})^{d+d'}} \\
&\approx 2^{d+d'}((d+d')\log_e((2^c - 1)) + O(1))
\end{aligned}$$

H Auxiliary Data

H.1 Differential Distribution Table for Arbitrary S-box

In Table 13, '$*$' denotes the possible differential propagation (some non-zero value) and '0' denotes the impossible differential propagation. The n-bit arbitrary S-box can be modeled in MILP only by $2n$ inequalities.

Table 13. DDT for 4-Bit Arbitrary S-box

		Δ_o														
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	0	$*$	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	2	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	3	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	4	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	5	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	6	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
Δ_i	7	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	8	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	9	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	A	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	B	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	C	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	D	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	E	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$
	F	0	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$	$*$

H.2 Differential Distribution Table for S-box in Lilliput

Table 14. DDT for S-box in Lilliput

		Δ_o															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
Δ_i	0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
	1	0	2	0	0	0	0	2	0	0	2	2	2	4	0	0	2
	2	0	0	0	2	2	0	2	2	0	4	0	2	0	2	0	0
	3	0	2	0	0	0	2	2	2	2	0	0	0	0	2	0	4
	4	0	0	0	2	0	2	0	0	0	0	2	4	0	2	2	2
	5	0	4	2	2	0	2	0	2	0	2	2	0	0	0	0	0
	6	0	0	2	0	0	0	4	2	0	0	2	0	2	2	2	0
	7	0	0	0	2	2	2	2	0	2	0	4	0	2	0	0	0
	8	0	2	2	4	2	0	2	0	0	0	0	0	0	0	2	2
	9	0	0	0	2	0	0	0	2	4	2	0	0	2	0	2	2
	a	0	0	2	0	2	0	0	4	2	0	2	2	0	0	0	2
	b	0	2	0	0	2	2	0	2	0	0	0	2	2	0	4	0
	c	0	2	0	0	2	0	0	0	2	2	2	0	0	4	2	0
	d	0	2	4	2	0	0	0	0	2	0	0	2	2	2	0	0
	e	0	0	2	0	4	2	0	0	0	2	0	0	2	2	0	2
	f	0	0	2	0	0	4	2	0	2	2	0	2	0	0	2	0

H.3 Specification of ARIA

ARIA uses Substitution-Permutation Network (SPN) structure, and the state is represented by 16 bytes. The round function consists of Substitution layer SL and Diffusion layer DL. Let x and y be the input and output, respectively, and $y = DL \circ SL(x)$. Substitution layer SL applies 16 S-boxes in parallel, where two S-boxes S_1 and S_2 and the inverse S_1^{-1} and S_2^{-1} are used in this layer. Then,

DL diffuses 16 bytes using the following 16×16 binary matrix as

$$\begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_5 \\ y_6 \\ y_7 \\ y_8 \\ y_9 \\ y_{10} \\ y_{11} \\ y_{12} \\ y_{13} \\ y_{14} \\ y_{15} \end{pmatrix} = \begin{pmatrix} 0\,0\,0\,1\,1\,0\,1\,0\,1\,1\,0\,0\,0\,1\,1\,0 \\ 0\,0\,1\,0\,0\,1\,0\,1\,1\,1\,0\,0\,1\,0\,0\,1 \\ 0\,1\,0\,0\,1\,0\,1\,0\,0\,0\,1\,1\,1\,0\,0\,1 \\ 1\,0\,0\,0\,0\,1\,0\,1\,0\,0\,1\,1\,0\,1\,1\,0 \\ 1\,0\,1\,0\,0\,1\,0\,0\,1\,0\,0\,1\,0\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,0\,0\,0\,1\,1\,0\,0\,0\,1\,1 \\ 1\,0\,1\,0\,0\,0\,0\,1\,0\,1\,1\,0\,1\,1\,0\,0 \\ 0\,1\,0\,1\,0\,0\,1\,0\,1\,0\,0\,1\,1\,1\,0\,0 \\ 1\,1\,0\,0\,1\,0\,0\,1\,0\,0\,1\,0\,0\,1\,0\,1 \\ 1\,1\,0\,0\,0\,1\,1\,0\,0\,0\,0\,1\,1\,0\,1\,0 \\ 0\,0\,1\,1\,0\,1\,1\,0\,1\,0\,0\,0\,0\,1\,0\,1 \\ 0\,0\,1\,1\,1\,0\,0\,1\,0\,1\,0\,0\,1\,0\,1\,0 \\ 0\,1\,1\,0\,0\,0\,1\,1\,0\,1\,0\,1\,1\,0\,0\,0 \\ 1\,0\,0\,1\,0\,0\,1\,1\,1\,0\,1\,0\,0\,1\,0\,0 \\ 1\,0\,0\,1\,1\,1\,0\,0\,0\,1\,0\,1\,0\,0\,1\,0 \\ 0\,1\,1\,0\,1\,1\,0\,0\,1\,0\,1\,0\,0\,0\,0\,1 \end{pmatrix} \begin{pmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \\ x_{11} \\ x_{12} \\ x_{13} \\ x_{14} \\ x_{15} \end{pmatrix},$$

where let $(x_0, x_1, \ldots, x_{15})$ and $(y_0, y_1, \ldots, y_{15})$ be the input and output of DL, respectively.

H.4 Specification of Minalpher-P

Minalpher uses Substitution-Permutation Network (SPN) structure using 4-bit S-boxes. The state is arranged in two 4×8 matrices as $A \in (\mathbb{F}_2^4)^{4\times 8}$ and $B \in (\mathbb{F}_2^4)^{4\times 8}$. Let $A[i][j]$ and $B[i][j]$ be 4-bit values in row i and column j of A and B, respectively. The round function consists of `SN`, `SR`, `SM`, `XM`, `MC`, and XOR of round constant.

In `SN`, 4-bit involution S-boxes are applied to all cells in A and B. In `SR`, each cell of the state is permuted as

$$\begin{aligned}
&A[0][j] \leftarrow A[0][SR_1(j)], && B[0][j] \leftarrow B[0][SR_1^{-1}(j)], && 0 \le j < 8, \\
&A[1][j] \leftarrow A[1][SR_2(j)], && B[1][j] \leftarrow B[1][SR_2^{-1}(j)], && 0 \le j < 8, \\
&A[2][j] \leftarrow A[2][SR_1^{-1}(j)], && B[2][j] \leftarrow B[2][SR_1(j)], && 0 \le j < 8, \\
&A[3][j] \leftarrow A[3][SR_2^{-1}(j)], && B[3][j] \leftarrow B[3][SR_2(j)], && 0 \le j < 8,
\end{aligned}$$

where $SR_1 = \{6, 7, 1, 0, 2, 3, 4, 5\}$ and $SR_2 = \{4, 5, 0, 1, 7, 6, 2, 3\}$. In `SM`, the matrix A is swapped for the matrix B. Then, in `XM`, the matrix B is XORed with the matrix A. In `MC`, the following matrix

$$\begin{pmatrix} 1\,1\,0\,1 \\ 1\,1\,1\,0 \\ 0\,1\,1\,1 \\ 1\,0\,1\,1 \end{pmatrix}$$

is multiplied by A and B. Finally, the round constant is XORed with a state. The number of rounds of Minlapher-P is 18, and only `SN`, `SR`, and `SM` are applied in the final round function.

H.5 Specification of MIBS

MIBS uses Feistel Network structure with a KSP-type round function. The state is arranged in 16 nibbles, and they split into left and right halves. The left half also becomes an input of the KSP-type round function, and the output is XORed with the right half. Finally, left and right halves are swapped.

The KSP-type round function is used in MIBS. The round key is first XORed with the input, and the $\mathcal{S}$ layer then substitutes eight nibbles using a 4-bit S-box defined as

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

The $\mathcal{P}$ layer finally diffuses eight nibbles as follows:

$$
\begin{aligned}
y'_1 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_7 \oplus y_8, & y'_5 &= y_1 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_8,\\
y'_2 &= y_2 \oplus y_3 \oplus y_4 \oplus y_5 \oplus y_6 \oplus y_7, & y'_6 &= y_1 \oplus y_2 \oplus y_4 \oplus y_5 \oplus y_6,\\
y'_3 &= y_1 \oplus y_2 \oplus y_3 \oplus y_5 \oplus y_6 \oplus y_8, & y'_7 &= y_1 \oplus y_1 \oplus y_3 \oplus y_6 \oplus y_7,\\
y'_4 &= y_2 \oplus y_3 \oplus y_4 \oplus y_7 \oplus y_8, & y'_8 &= y_1 \oplus y_3 \oplus y_4 \oplus y_6 \oplus y_7 \oplus y_8,
\end{aligned}
$$

where let $(y_1, y_2, \ldots, y_8)$ and $(y'_1, y'_2, \ldots, y'_8)$ be the input and output of the $\mathcal{P}$ layer, respectively.