

New Impossible Differential Search Tool from Design and Cryptanalysis Aspects Revealing Structural Properties of Several Ciphers

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 is presented. Our tool can detect any contradiction between input and output differences. It can also take into account the property inside the S-box when its size is small e.g. 4 bits. This is natural for ciphers with bit-wise diffusion like PRESENT, while finding such impossible differentials for ciphers with word-wise diffusion is novel. In addition, several techniques are proposed to evaluate 8-bit S-box. The tool improves the number of rounds of impossible differentials from the previous best results for Midori128, LILLIPUT, and Minalpher. The tool also finds new impossible differentials for ARIA and MIBS. We manually verify the impossibility of the searched results, which reveals new structural properties of those designs. The tool can be implemented by slightly modifying the previous differential search tool using Mixed Integer Linear Programming (MILP). 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.

Keywords: 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 approach 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 keyed cipher E_K , it exploits a pair of input and output differences (Δ_i, Δ_o) that cannot be connected for any K . Namely, two input values x, x' satisfying $x \oplus x' = \Delta_i$ never satisfy $E_K(x) \oplus E_K(x') = \Delta_o$.

Such (Δ_i, Δ_o) are detected by the *miss-in-the-middle* approach [4]. The first automated search attempt was done in [3] with a technique called *shrink*. It shrinks the word size to 3 bits and finds impossible differentials of the global structure of the cipher by exhaustively testing all possible differences and values. The shrink technique is useful when the cipher consists of small number of words with big word size, e.g. 4 words of 32 bits in Skipjack, while the recent design trend is using many words with small word size, e.g. 16 words of 8 bits in AES.

Kim et al. [5] presented the 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 is 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 MILP-based tools for (standard) differential cryptanalysis, which models S-boxes in bitwise [12–14].

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 then iterate this test to examine multiple pairs of (Δ_i, Δ_o) e.g. all pairs with 1 active word both in input and output. We note that, for all existing ciphers, the longest impossible differentials have only 1 active word in both input and output. Thus, it is reasonable to conjecture that if such impossible differentials do not exist, any impossible differentials do not.

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 cryptanalyses.

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 are always valid for arbitrary S-box choice. The arbitrary S-box can be described efficiently, which enables us to evaluate 8-bit S-boxes. We note that previous work on MILP based tool aimed to model DDT precisely. One can see the catchphrase “MILP whose feasible region is exactly the set of all valid differential” in [13, 15], while modeling 8-bit S-box precisely is infeasible. Our approach is opposite of previous work, which describes DDT only roughly but can be executed in practice.

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 tool more efficient only by aiming truncated impossible differentials, which can be implemented only by changing the constraints of input and output differences.

Table 1. Application results. ‘KR’ denotes ‘key recovery.’

Target	Ref	#Rounds		Search mode	Goal	Remarks
		Prev.	Ours			
Midori128	[16]	6	7	Specific S-box	Characteristic	
LILLIPUT	[17]	8	9	Specific S-box	Characteristic	
Minalpher	[18]	6.5	7.5	Arbitrary S-box	Truncated	Large state
ARIA	[19]	4	4	Arbitrary S-box	Truncated	8-bit S-box, improve KR
MIBS	[20]	8	8	Specific S-box	Characteristic	New impossible differentials

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.

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 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 [21]. The relationship between [21] 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 [22] 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 [23] 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 whether the bits are active or inactive are defined for all rounds; x_0, x_1, \dots, x_{63} are for 64 bits in the plaintext, $x_{64}, x_{65}, \dots, x_{127}$ are for 64 bits after round 1, $x_{128}, x_{129}, \dots, x_{191}$ 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 ‘ $x_0 + x_1 + \dots + x_{63} \geq 1$.’ 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, \dots, x_{63} are updated to x_{64}, \dots, x_{127} .

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

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 [14]. 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 are feasible for S-boxes of size five bits or less. 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 [24] as the MILP solver.

3 Composite Framework for Differential and Impossible Differential Searches

We begin with explaining the basic concept of our impossible differential search tool, which has been independently discovered by Cui et al. and their paper was posted on Cryptology ePrint Archive prior to our paper [21]. Comparison between [21] and this work will be explained in Appendix A.

The tool adds several constraints to the previous differential bound search for fixing an input and output difference to a specific pair (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.

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**
-

Example 1. Let p_0, p_1, \dots, p_{b-1} and c_0, c_1, \dots, 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 $(\Delta_i, \Delta_o) = (0x1, 0x1)$ is impossible, the MILP solver should run with the following constraints added.

$$p_0 = 1, p_1 = 0, c_2 = 0, \dots, p_{b-1} = 0,$$

$$c_0 = 1, c_1 = 0, c_2 = 0, \dots, c_{b-1} = 0.$$

We then iterate this test to examine multiple pairs of (Δ_i, Δ_o) e.g. all pairs with 1 active word both in input and output.

3.1 Composite Framework

A 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 cryptanalyses. 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 through S-box in bitwise [12–14]. Here, we recall how a system of inequalities is generated.

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 [12–14]. 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 users need to modify only two parts; make the objective function empty and specify input and output differences.

Algorithm 2. Generating System of Inequalities in Composite Framework

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

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

```

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 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 [21] 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 Sects. 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 [12–14]. 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.

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

$$\begin{array}{ll}
 i_0 + i_1 + \dots + i_{n-1} - o_0 \geq 0, & o_0 + o_1 + \dots + o_{n-1} - i_0 \geq 0, \\
 i_0 + i_1 + \dots + i_{n-1} - o_1 \geq 0, & o_0 + o_1 + \dots + o_{n-1} - i_1 \geq 0, \\
 \dots & \dots \\
 i_0 + i_1 + \dots + i_{n-1} - o_{n-1} \geq 0, & o_0 + o_1 + \dots + o_{n-1} - i_{n-1} \geq 0.
 \end{array}$$

The advantage of the arbitrary S-box compared to the specific S-box is efficiency owing to a small number of constraints 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) [19] and of Minalpher (4-bit S-box, 256-bit block) [18], 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 $(n \cdot (2^c - 1))^2$, which is $O(n^2 \cdot 2^{2c})$, while one with two active words is $\binom{n}{2} \cdot (2^{2c} - 1)^2$, 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

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

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, \dots, i_{n-1} and o_0, o_1, \dots, o_{n-1} be variables to represent whether n input and n output bits in the truncated position are active or inactive.

Then, we write the following constraint (along with constraints fixing the other bits to 0):

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

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 [19] 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 [16]. 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 $SSb_0, SSb_1, SSb_2,$ and SSb_3 are used and $s_i \leftarrow SSb_{i \bmod 4}(s_i)$ where $0 \leq i \leq 15$. Four 8-bit S-boxes SSb_i are constructed by 4-bit S-box Sb_1 , where Sb_1 is defined as follows.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
$Sb_1(x)$	1	0	5	3	E	2	F	7	D	A	9	B	C	8	4	6

Then, SSb_i are constructed as $SSb_i = p_i^{-1} \circ (Sb_1 \parallel Sb_1) \circ p_i$, where two Sb_1 are applied to top and bottom halves in $(Sb_1 \parallel Sb_1)$. Note that SSb_i is involution, and we later show that impossible differentials are improved by exploiting this property. Figure 1 shows the specification

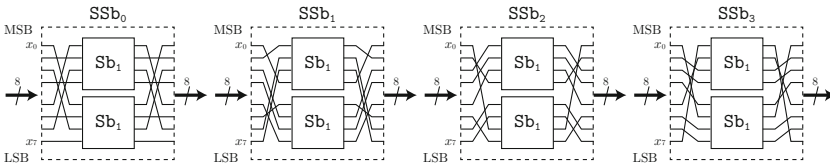


Fig. 1. SSb₀, SSb₁, SSb₂, and SSb₃

of SSb_{*i*}. In ShuffleCell, each cell is permuted as $(s_0, s_1, \dots, s_{15}) \leftarrow (s_0, s_{10}, s_5, s_{15}, s_{14}, s_4, s_{11}, s_1, s_9, s_3, s_{12}, s_6, s_7, s_{13}, s_2, s_8)$. 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 [25] and nonlinear invariant attack [26] 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 [16]. Then, Zhen et al. found 6-round impossible differentials that are advantageous for the key recovery but the number of rounds is not increased [27].

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.

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.

Table 2. 7-round truncated impossible differentials against Midori128

ID	ΔP	ΔC	Remarks
001T	($0\alpha_100, 0000, 0000, 0000$)	($0\beta_100, 0000, 0000, 0000$)	Manually verified
002T	($0\beta_100, 0000, 0000, 0000$)	($0\alpha_100, 0000, 0000, 0000$)	Manually verified
003T	($0000, \alpha_0000, 0000, 0000$)	($0000, \beta_0000, 0000, 0000$)	
004T	($0000, \beta_0000, 0000, 0000$)	($0000, \alpha_0000, 0000, 0000$)	
005T	($0000, 0\alpha_100, 0000, 0000$)	($0000, 0\beta_100, 0000, 0000$)	
006T	($0000, 0\beta_100, 0000, 0000$)	($0000, 0\alpha_100, 0000, 0000$)	
007T	($0000, 0000, \alpha_0000, 0000$)	($0000, 0000, \beta_0000, 0000$)	
008T	($0000, 0000, \beta_0000, 0000$)	($0000, 0000, \alpha_0000, 0000$)	
009T	($0000, 0000, 0\alpha_100, 0000$)	($0000, 0000, 0\beta_100, 0000$)	
010T	($0000, 0000, 0\beta_100, 0000$)	($0000, 0000, 0\alpha_100, 0000$)	
011T	($0000, 0000, 0000, \alpha_0000$)	($0000, 0000, 0000, \beta_0000$)	
012T	($0000, 0000, 0000, \beta_0000$)	($0000, 0000, 0000, \alpha_0000$)	

List of 7-Round Truncated Impossible Differentials. We ran our tool with the above configuration. The tool required about 0.03 seconds per pair and it took about 0.5 min 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 SSb_i , which seems to be useful for future analysis. We first prove ID001T.

Theorem 1. *The input difference ($0\alpha_100, 0000, 0000, 0000$) cannot propagate to the output difference ($0\beta_100, 0000, 0000, 0000$) 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 are active and inactive, respectively. In `SubCell` in the first round, $\text{SSb}_1(\alpha_1) = p_1^{-1} \circ (\text{Sb}_1 \parallel \text{Sb}_1) \circ p_1(\alpha_1)$ is computed. $(\text{Sb}_1 \parallel \text{Sb}_1)$ 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

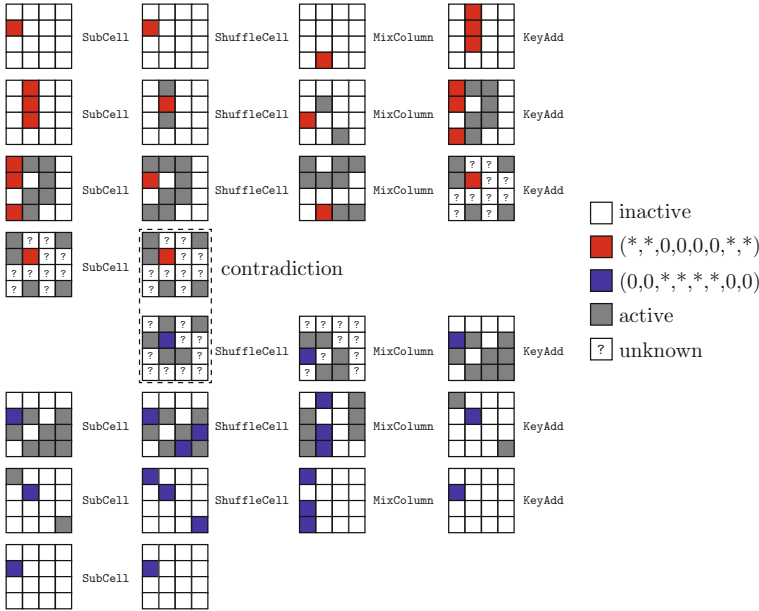


Fig. 2. 7-round truncated impossible differential of Midori128; ID001T

diffused to $s_4, s_5,$ and s_6 by **MixColumns**. S-boxes are applied in the second round again, but SSb_0 and 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 $SSb_1^{-1}(\beta_1)$, 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. \square

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.

4.2 LILLIPUT

LILLIPUT is a lightweight block cipher designed by Berger et al. in 2015 [17] in which the block size and the key size are 64 bits and 80 bits, respectively. LILLIPUT adopts an extended generalized Feistel network (EGFN) [28].

Specification. A 64-bit plaintext is loaded to a 64-bit state X^0 , which is divided into sixteen 4-bit nibbles, $X_{15}^0 || X_{14}^0 || \dots || X_0^0$. The round function, RF , takes as

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

input a previous state X^j and a 32-bit subkey $SK^j \triangleq SK_7^j || SK_6^j || \dots || SK_0^j$ 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, $X_{8+i}^j \leftarrow X_{8+i}^j \oplus S(X_{7-i}^j \oplus SK_i^j)$, $i = 0, 1, \dots, 7$, 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.

$$X_{15}^j \leftarrow X_{15}^j \oplus X_7^j \oplus X_6^j \oplus X_5^j \oplus X_4^j \oplus X_3^j \oplus X_2^j \oplus X_1^j,$$

$$X_{15-i}^j \leftarrow X_{15-i}^j \oplus X_7^j \text{ for } i = 1, 2, \dots, 6.$$

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. 3.

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. 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 0.2seconds per pair and it took about 1 h to test 14400 pairs.

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 α , 00000000)	(00000000, 00000 α 00)	Manually verified
016–030	(000000 α 0, 00000000)	(00000000, 00 α 00000)	
031–045	(000000 α 0, 00000000)	(00000000, 0000000 α)	
...
181–195	(000 α 0000, 00000000)	(00000000, 0000000 α)	
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	(000000 e 0, 00000000)	(00000000, 00000 e 00)	Manually verified
201	(000000 f 0, 00000000)	(00000000, 00000 f 00)	Manually verified
202	(00007000, 00000000)	(00000000, 00000700)	
203	(0000 e 000, 00000000)	(00000000, 00000 e 00)	
204–216	(000 β 0000, 00000000)	(00000000, 000000 β 0)	Manually verified
217	(00010000, 00000000)	(00000000, 00000050)	

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. 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 Due to the page limitation, we omit the proof of ID001–ID015. We expect ID016–ID195 can be proven similarly.

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 α 0, 00000000) cannot propagate to the output difference (00000000, 00000 α 00) 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.

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.

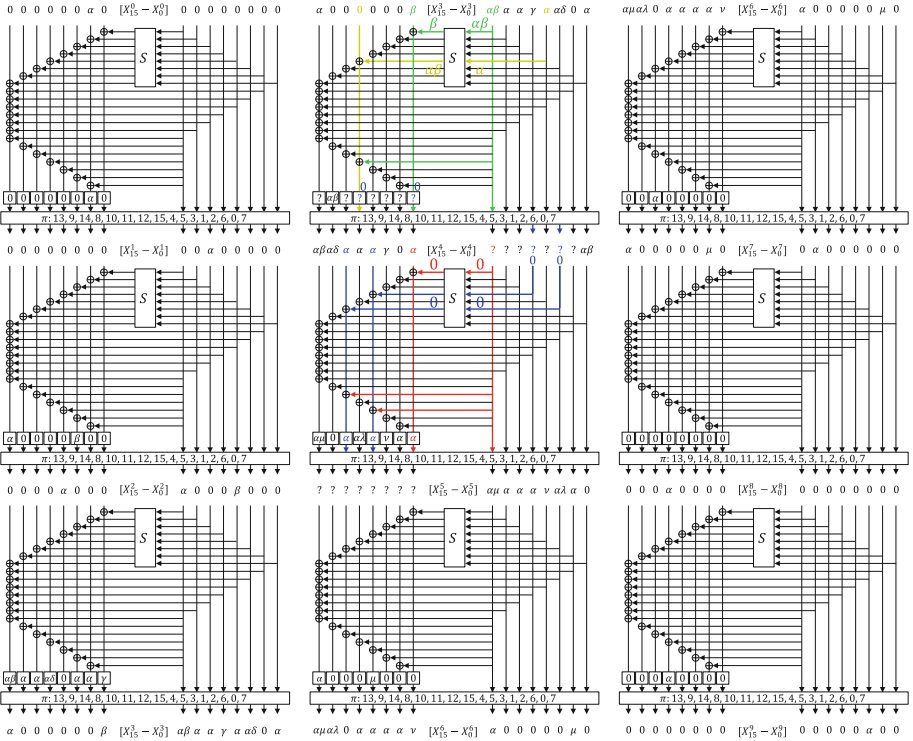


Fig. 3. 9-round impossible differential characteristic of LILLIPUT; ID196–201 (Color figure online)

- 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 fourth 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 $X_8^4 \oplus S(X_7^4) = X_8^5$ 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.

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 5, 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 omit the proof in this paper.

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 [19]. 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*. We refer to [19] for its detailed specification.

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 ($SL \circ (DL \circ SL)^5$) [29]. 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 [30]. 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

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

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 beginning 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

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

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 [18]. Minalpher uses 256-bit core permutation called Minalpher-P, which

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]	
\vdots	\vdots	\vdots	
1152T	B[3][7]	B[3][7]	

is based on Substitution-Permutation Network (SPN) structure using 4-bit S-boxes. We refer to [18] for its detailed specification.

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. 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 [20]. The block length is 64, and it provides two key lengths: 64- and 80-bit secret key. We refer to [20] for its detailed specification.

Previous Cryptanalysis. Bay et al. found two 8-round truncated impossible differentials [31]. Then, Wu and Wang found four additional 8-round truncated impossible differentials [8].

Table 7. 8-round impossible differential characteristics against MIBS

ID	ΔP	ΔC	Remarks
001T	(00000000, 000000 α 0)	(0000 β 000, 00000000)	Bay
002T	(00000000, 0000 α 000)	(000000 β 0, 00000000)	Wu
003T	(00000000, 00 α 00000)	(0000000 β , 00000000)	Bay
004T	(00000000, 0000000 α)	(00 β 00000, 00000000)	Wu
005T	(00000000, 00 α 00000)	(0000 β 000, 00000000)	Wu
006T	(00000000, 0000 α 000)	(00 β 00000, 00000000)	Wu
001–120	(00000000, 000 γ 0000)	(00000 ϵ 00, 00000000)	
121–240	(00000000, 00000 ϵ 00)	(000 γ 0000, 00000000)	

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 per pair using single core and it took about 30 h 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 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.

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

Table 8. Pairs of impossible differences found by our tool for MIBS

γ	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	
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	

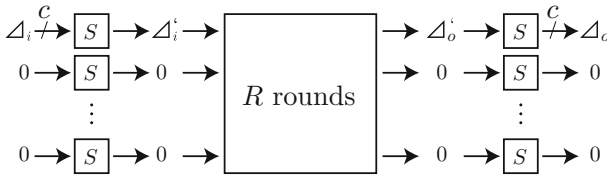


Fig. 4. Differential possibility equivalence technique

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.

5.1 Procedure of Differential Possibility Equivalence Technique

The *differential possibility equivalence technique* reduces the number of MILP instances that our tool has to solve.² Figure 4 shows the outline of the technique.

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

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 $(\Delta_i \rightarrow \Delta'_i \rightarrow \Delta'_o \rightarrow \Delta_o)$ 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 $\Delta \rightarrow \Delta'_i$ is possible. Similarly, we evaluate a set \mathcal{O} whose elements are all Δ such that $\Delta'_o \rightarrow \Delta$ is possible. Then, pairs in $(\mathcal{I} \times \mathcal{O})$ are possible characteristics via (Δ'_i, Δ'_o) , we thus do not need to evaluate them using MILP. We note that some MILP solvers have API for programming languages, e.g. Gurobi Optimizer supports API for C-language. Thus, adding such auxiliary codes is easily done. 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 $2^{d+d'}((d+d')\log_e(2^c - 1) + O(1))$ on average.*

Due to the page limitation, we omit the proof of Theorem 3. 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 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` [16] chose the S-box with the lowest depth, and `FIDES` [32] and `PICARO` [33] 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. \square

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 [21]. 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 $r - 1$ 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 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 10s, 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 10s.

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 [34], TWINE [35], LBlock [36], Piccolo [37], RECTANGLE [10],

Table 10. Provable security against impossible differentials

Target	#Rounds	Assumption	Remarks
Midori128	8	Subkey uniform	1 active byte
	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

Skinny [22], Midori64 [16], and CLEFIA [38] 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 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 SSb_i and regard 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.

A Relationship Between [21] and This Paper

Cui et al. [21] 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. 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.

Though the basic idea of the tool is the same, two papers extend the basic idea to quite different directions. The main focus of [21] 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 [21] left application to 8-bit S-box open.

Another difference is enthusiasm for the application to practical designs. Considering the number of applications, [21] 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.

Advantages of [21] 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 [39], 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 [21].

- 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.

References

1. Mouha, N., Wang, Q., Gu, D., Preneel, B.: Differential and linear cryptanalysis using mixed-integer linear programming. In: Wu, C.-K., Yung, M., Lin, D. (eds.) *Inscrypt 2011*. LNCS, vol. 7537, pp. 57–76. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-34704-7_5](https://doi.org/10.1007/978-3-642-34704-7_5)
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 1999*. LNCS, vol. 1592, pp. 12–23. Springer, Heidelberg (1999). doi:[10.1007/3-540-48910-X_2](https://doi.org/10.1007/3-540-48910-X_2)
4. Biryukov, A.: Miss-in-the-middle attack. In: van Tilborg, H.C.A. (ed.) *Encyclopedia of Cryptography and Security*. Springer, Heidelberg (2005)

5. Kim, J., Hong, S., Sung, J., Lee, S., Lim, J., Sung, S.: Impossible differential cryptanalysis for block cipher structures. In: Johansson, T., Maitra, S. (eds.) *INDOCRYPT 2003*. LNCS, vol. 2904, pp. 82–96. Springer, Heidelberg (2003). doi:[10.1007/978-3-540-24582-7_6](https://doi.org/10.1007/978-3-540-24582-7_6)
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**, 211–220 (2014)
8. Wu, S., Wang, M.: Automatic search of truncated impossible differentials for word-oriented block ciphers. In: Galbraith, S., Nandi, M. (eds.) *INDOCRYPT 2012*. LNCS, vol. 7668, pp. 283–302. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-34931-7_17](https://doi.org/10.1007/978-3-642-34931-7_17)
9. Tezcan, C.: Improbable differential attacks on present using undisturbed bits. *J. Comput. Appl. Math.* **259**, 503–511 (2014)
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.-A.: Automatic search of meet-in-the-middle and impossible differential attacks. In: Robshaw, M., Katz, J. (eds.) *CRYPTO 2016*. LNCS, vol. 9815, pp. 157–184. Springer, Heidelberg (2016). doi:[10.1007/978-3-662-53008-5_6](https://doi.org/10.1007/978-3-662-53008-5_6)
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*. LNCS, vol. 8873, pp. 158–178. Springer, Heidelberg (2014). doi:[10.1007/978-3-662-45611-8_9](https://doi.org/10.1007/978-3-662-45611-8_9)
13. 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* 2014/747 (2014)
14. 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, Cham (2016)
15. Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L., Fu, K.: Constructing mixed-integer programming models whose feasible region is exactly the set of all valid differential characteristics of SIMON. *Cryptology ePrint Archive*, report 2015/122 (2015). <http://eprint.iacr.org/2015/122>
16. 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*. LNCS, vol. 9453, pp. 411–436. Springer, Heidelberg (2015). doi:[10.1007/978-3-662-48800-3_17](https://doi.org/10.1007/978-3-662-48800-3_17)
17. 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 Trans. Comput. Comput.* **65**, 2074–2089 (2015)
18. Sasaki, Y., Todo, Y., Aoki, K., Naito, Y., Sugawara, T., Murakami, Y., Matsui, M.: Minalpher v1.1. Submitted to CAESAR (2015)
19. Kwon, D., et al.: New block cipher: ARIA. In: Lim, J.-I., Lee, D.-H. (eds.) *ICISC 2003*. LNCS, vol. 2971, pp. 432–445. Springer, Heidelberg (2004). doi:[10.1007/978-3-540-24691-6_32](https://doi.org/10.1007/978-3-540-24691-6_32)

20. Izadi, M., Sadeghiyan, B., Sadeghian, S.S., Khanooki, H.A.: MIBS: a new light-weight block cipher. In: Garay, J.A., Miyaji, A., Otsuka, A. (eds.) CANS 2009. LNCS, vol. 5888, pp. 334–348. Springer, Heidelberg (2009). doi:[10.1007/978-3-642-10433-6_22](https://doi.org/10.1007/978-3-642-10433-6_22)
21. 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>
22. Beierle, C., et al.: The SKINNY family of block ciphers and its low-latency variant MANTIS. In: Robshaw, M., Katz, J. (eds.) CRYPTO 2016. LNCS, vol. 9815, pp. 123–153. Springer, Heidelberg (2016). doi:[10.1007/978-3-662-53008-5_5](https://doi.org/10.1007/978-3-662-53008-5_5)
23. 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. LNCS, vol. 4727, pp. 450–466. Springer, Heidelberg (2007). doi:[10.1007/978-3-540-74735-2_31](https://doi.org/10.1007/978-3-540-74735-2_31)
24. Gurobi Optimization, Inc.: Gurobi optimizer 6.5 (2015). <http://www.gurobi.com/>
25. 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>
26. 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>
27. Zhan, C., Xiaoyun, W.: Impossible differential cryptanalysis of Midori. Cryptology ePrint Archive, report 2016/535 (2016). <http://eprint.iacr.org/2016/535>
28. Berger, T.P., Minier, M., Thomas, G.: Extended generalized feistel networks using matrix representation. In: Lange, T., Lauter, K., Lisoněk, P. (eds.) SAC 2013. LNCS, vol. 8282, pp. 289–305. Springer, Heidelberg (2014). doi:[10.1007/978-3-662-43414-7_15](https://doi.org/10.1007/978-3-662-43414-7_15)
29. Wu, W., Zhang, W., Feng, D.: Impossible differential cryptanalysis of reduced-round ARIA and Camellia. J. Comput. Sci. Technol. **22**(3), 449–456 (2007)
30. Li, R., Sun, B., Zhang, P., Li, C.: New impossible differential cryptanalysis of ARIA. Cryptology ePrint Archive, report 2008/227 (2008). <http://eprint.iacr.org/2008/227>
31. Bay, A., Nakahara Jr., J., Vaudenay, S.: Cryptanalysis of reduced-round MIBS block cipher. In: Heng, S.-H., Wright, R.N., Goi, B.-M. (eds.) CANS 2010. LNCS, vol. 6467, pp. 1–19. Springer, Heidelberg (2010). doi:[10.1007/978-3-642-17619-7_1](https://doi.org/10.1007/978-3-642-17619-7_1)
32. Bilgin, B., Bogdanov, A., Knežević, M., Mendel, F., Wang, Q.: FIDES: light-weight authenticated cipher with side-channel resistance for constrained hardware. In: Bertoni, G., Coron, J.-S. (eds.) CHES 2013. LNCS, vol. 8086, pp. 142–158. Springer, Heidelberg (2013). doi:[10.1007/978-3-642-40349-1_9](https://doi.org/10.1007/978-3-642-40349-1_9)
33. 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. LNCS, vol. 7341, pp. 311–328. Springer, Heidelberg (2012). doi:[10.1007/978-3-642-31284-7_19](https://doi.org/10.1007/978-3-642-31284-7_19)
34. 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>
35. 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. LNCS, vol. 7707, pp. 339–354. Springer, Heidelberg (2013). doi:[10.1007/978-3-642-35999-6_22](https://doi.org/10.1007/978-3-642-35999-6_22)

36. Wu, W., Zhang, L.: LBlock: a lightweight block cipher. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 327–344. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-21554-4_19](https://doi.org/10.1007/978-3-642-21554-4_19)
37. 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. LNCS, vol. 6917, pp. 342–357. Springer, Heidelberg (2011). doi:[10.1007/978-3-642-23951-9_23](https://doi.org/10.1007/978-3-642-23951-9_23)
38. Shirai, T., Shibutani, K., Akishita, T., Moriai, S., Iwata, T.: The 128-bit blockcipher CLEFIA (extended abstract). In: Biryukov, A. (ed.) FSE 2007. LNCS, vol. 4593, pp. 181–195. Springer, Heidelberg (2007). doi:[10.1007/978-3-540-74619-5_12](https://doi.org/10.1007/978-3-540-74619-5_12)
39. 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. LNCS, vol. 9783, pp. 268–288. Springer, Heidelberg (2016). doi:[10.1007/978-3-662-52993-5_14](https://doi.org/10.1007/978-3-662-52993-5_14)