New Differential Bounds and Division Property of LILLIPUT: Block Cipher with Extended Generalized Feistel Network

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. This paper provides security analysis of lightweight block cipher LILLIPUT, which is an instantiation of extended generalized Feistel network (EGFN) developed by Berger *et al.* at SAC 2013. Its round function updates a part of the state only linearly, which yields several security concerns. The first important discovery is that the lower bounds of the number of active S-boxes provided by the designers are incorrect. Then the new bounds are derived by using mixed integer linear programming (MILP), which shows an interesting fact that the actual bounds are better than the designers originally expected. Another contribution is the best third-party cryptanalysis. Owing to its unique computation structure, the designers expected that EGFN efficiently enhances security against integral cryptanalysis. However, the security is not enhanced as the designers expect. In fact, division property, which is a new method to find integral distinguishers, finds a 13-round distinguisher which improves the previous distinguisher by 4 rounds. The new distinguisher is further extended to a 17-round key recovery attack which improves the previous best attack by 3 rounds.

Keywords: Block-cipher · LILLIPUT · Extended generalized Feistel network · Mixed integer linear programming · Division property

1 Introduction

Lightweight cryptography is one of the most actively discussed topics in the current symmetric-key community. A huge number of designs have been proposed especially for the last decade. Here, we omit the list of all the lightweight primitives. Readers may refer to [1] for such a list. An important challenge that is common for most of those designs is achieving good security without significantly sacrificing efficiency.

One of the major approaches to design lightweight cipher is using Feistel network or generalized Feistel network (GFN), which has a property that its transformation is basically involutive thus the overhead to implement decryption circuit is minimized. Meanwhile, diffusion speed of the standard Feistel network

R. Avanzi and H. Heys (Eds.): SAC 2016, LNCS 10532, pp. 264–283, 2017.
https://doi.org/10.1007/978-3-319-69453-5_15

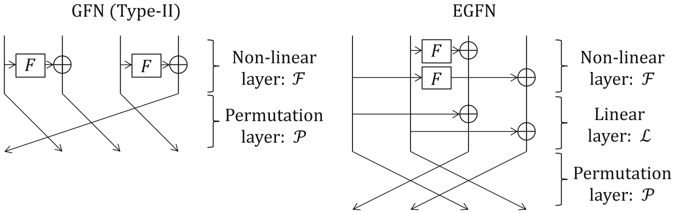
GFN (Type-II)
EGFN
F
F
Non-linear layer: $\mathcal{F}$
Permutation layer: $\mathcal{P}$
Linear layer: $\mathcal{L}$

Fig. 1. Comparison of GFN (Left) and EGFN (Right) with four branches.

is often much slower than other design approaches. To overcome this drawback, several researches have developed new ideas. Suzaki and Minematsu pointed out that security of GFN can be enhanced by replacing the way of mixing branches [2]. This is called *block-shuffle* and TWINE [3] was designed based on this idea. Zhang and Wu used modified Feistel network to design LBlock [4], which turned out to be the same network as one in TWINE [3]. The latest approach, which is a main focus in this paper, is *extended GFN (EGFN)* proposed by Berger *et al.* [5], in which an additional linear diffusion layer is inserted between the application to F-function and branch network. The comparison of GFN and EGFN is depicted in Fig. 1. In many designs, the non-linear layer is the most expensive, thus the linear layer leads to better diffusion speed with a small extra cost.

Berger *et al.* [5] specified two concrete examples of EGFN with security analysis. Unfortunately, mistakes in the security analysis were pointed out by Zhang and Wu [6] and very effective differential trails were constructed for those original choices of EGFN. To fix this drawback, Berger *et al.* combined block-shuffle [2] with EGFN, and proposed a new cipher preventing the attack by Zhang and Wu. The cipher was named LILLIPUT [7].

LILLIPUT is a lightweight block cipher, supporting 64-bit block and 80-bit key. LILLIPUT is a 16-branch EGFN with block-shuffle, in which the size of each branch is 4 bits (nibble) and the non-linear function is an application of a 4-bit S-box. Those parameter sizes are the same as TWINE and LBlock. The number of rounds is 30, which is 2 rounds less than TWINE and LBlock. This shows that the additional linear layer of LILLIPUT allows to ensure its security with a smaller number of rounds than TWINE and LBlock. The designers of LILLIPUT provided several security analysis, including minimal number of active S-boxes for every round, impossible differential attack, integral attack, differential/linear cryptanalysis, related-key attacks and chosen-key attacks. Regarding differential cryptanalysis, the minimal number of active S-boxes is listed in Table 1. Other single-key attacks are summarized in Table 2.

Our Contributions. In this paper, we show that the linear layer of EGFN and LILLIPUT yields several security concerns to be carefully discussed.

We first study differential cryptanalysis. We show that the linear layer makes the evaluation of truncated differential very complicated. The linear layer allows differences to go through the round function without going through S-box.

Table 1. Lowerbounds of number of active S-boxes for each round. NW and BW represent nibble-wise model and bit-wise model, respectively.

Approach	Rounds																Tightness
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
Branching [7]	0	1	2	3	5	9	12	14	15	17	21	24	26	28	29	31	Claimed as tight
MILP (NW, basic)	0	1	2	3	5	9	12	14	15	17	19	22	25	27	29	31	Not tight
MILP (NW, advanced)	0	1	2	3	5	9	12	14	15	17	19	23	25	28	30	32	Not tight
MILP (BW)	0	1	2	3	5	9	12	15	17	19	22	?	?	?	?	?	Tight

Table 2. Key recovery attacks in the single-key model against LILLIPUT. Related-key attack and chosen-key attacks reach 23 rounds, which are not included in this table.

Approaches	Distinguisher	Key recovery	Data	Time	Ref
Integral	9 rounds	13 rounds	2^{62}	2^{72}	[7]
Impossible differential	8 rounds	14 rounds	2^{63}	2^{77}	[7]
Division property	13 rounds	17 rounds	2^{63}	2^{77}	**Ours**

This implies that attackers need to trace the impact of linearly diffused difference over many rounds. This is quite opposite for SPN-based ciphers, say AES, in which difference in all cells is randomly updated in every round. To illustrate this fact, an example of contradicting truncated differential searched by a simple search is shown in Fig. 3. We search for the lower bounds of the number of active S-boxes with MILP. The results show that the lower bounds provided by the designers are incorrect. This is the reason why our bounds are sometimes larger and sometimes smaller than the original bounds. Then, we derive new bounds with MILP in two approaches; nibble-wise and bitwise models. The former can evaluate many rounds while the derived bounds are loose. The latter can derive tight bounds while its expensive search cost restricts the search range up to 11 rounds. The results are shown in Table 1. Interestingly, our results show that LILLIPUT is more secure than the designers have expected, e.g. the designers reported that the best characteristic could reach 16 rounds while we prove this is impossible.

We next study integral cryptanalysis. The designers evaluated the security in [5,7], where the propagation characteristic of the integral property [8] was used to search for the integral distinguisher. They showed that EGFN and LILLIPUT have higher security than GFN with block-shuffle. Actually, while TWINE and LBlock allow 15-round integral distinguisher, LILLIPUT only allows the 9-round integral distinguisher. It implies that the linear layer enhances security against the integral cryptanalysis by $6 (= 15 - 9)$ rounds. On the other hand, the linear layer does not increase the algebraic degree. Hence by constructing the integral characteristic by estimating the algebraic degree, which is often called the higher order differential cryptanalysis, the attack may be improved drastically. The division property is a new method to find integral distinguisher, which is a generalization of the integral property and can exploit low algebraic degree in the same time [9]. Thus security contribution of the linear layer can be evaluated more accurately with the the division property. As a result, we show that the division property finds a 13-round integral distinguisher, and it implies that the security is not enhanced as the designers expected. Moreover, the new distinguisher leads the attack against 17-round Lilliput (see Table 2), which is the current best attack against Lilliput.

Paper Outline. Related work and specification are introduced in Sect. 2. High-level overview of the properties we discuss on EGFN and Lilliput is given in Sect. 3. In Sect. 4 we search for new bounds of number of active S-boxes using MILP. In Sect. 5, we improve the previous best attack with division property. Finally, we conclude this paper in Sect. 6.

2 Related Work

2.1 Extended Generalized Feistel Network (EGFN)

Previous GFN has two computation layers per round; one is applying non-linear functions to some of branches and xoring the results to other branches (non-linear layer $\mathcal{F}$), and the other is permuting branches (permutation layer $\mathcal{P}$), which is often designed as a simple cyclic shift of branches. EGFN [5] adds a new diffusion layer (linear layer $\mathcal{L}$). In many designs, the non-linear layer $\mathcal{F}$ is the most expensive part, thus the linear layer $\mathcal{L}$ helps to increase the diffusion speed with a small additional cost. Berger *et al.* showed two concrete choices of $\mathcal{F}$ and $\mathcal{L}$ when the number of branches is 8 and 16 along with some security analysis. It is notable that the permutation $\mathcal{P}$ was assumed to be a simple swap of the left half and right half of the state.

Zhang and Wu [6] pointed out that the security evaluation in [5] was wrong and presented efficient differential characteristics against concrete examples in [5]. The attack relies on the choice of $\mathcal{P}$, which is a simple swap of branches.

2.2 Lilliput Specification

Lilliput [7] was designed by Berger et al. in 2015. So as to prevent the attack by Zhang and Wu [6], the designers adopted block-shuffle network [2] proposed by Suzaki *et al.* on top of EGFN so as to achieve even faster diffusion.

The block size and the key size of Lilliput are 64 bits and 80 bits, respectively. Its round function consists of 16 branches of size 4 bits. 64-bit plaintext is first loaded to sixteen 4-bit array $X_{15}, X_{14} \ldots, X_0$. Then, the round function consisting of three layers $\mathcal{F}$, $\mathcal{L}$, and $\mathcal{P}$ is iterated 30 times. The permutation layer $\mathcal{P}$ is omitted in the last round for involution reasons. An illustration of the round function is shown in Fig. 2.

The key schedule first expands the 80-bit key to 32-bit round keys for round $j, j = 0, \ldots, 29$ dented by RK^j. Because we do not analyze the key schedule, we omit its description.

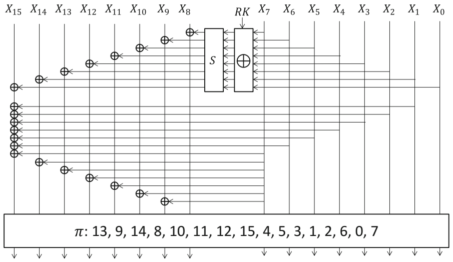

Fig. 2. Round function of LILLIPUT.

Table 3. S-box.

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.

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

Non-linear Layer $\mathcal{F}$. At first, the state input and round key are xored. Then, a 4-bit S-box is applied to each of eight nibbles in the right half of the state, and the results are xored to the left half of the state. Let RK_i^j and X_i^j be the i-th nibble of the j-th round key RK^j and j-th round state X^j, respectively. Then, the nonlinear layer can be defined as $X_{8+i}^j \leftarrow X_{8+i}^j \oplus S(X_{7-i}^j \oplus RK_i^j)$, $i = 0, 1, \ldots, 7$, where $S(\cdot)$ is a 4-bit to 4-bit S-box defined in Table 3.

Linear Layer $\mathcal{L}$. The idea in $\mathcal{L}$ is, along with diffusion by $\mathcal{F}$, having X_7^j propagate to all nibbles in the left half of the state and having X_{15}^j be propagated from all nibbles from the right half of the state. $\mathcal{L}$ is defined as follows.

$$X_{15}^j \leftarrow X_{15}^j \oplus X_7^j \oplus X_6^j \oplus X_5^j \oplus X_4^j \oplus X_3^j \oplus X_2^j \oplus X_1^j,$$
$$X_{15-i}^j \leftarrow X_{15-i}^j \oplus X_7^j \text{ for } i = 1, 2, \ldots, 6.$$

Permutation Layer $\mathcal{P}$. Nibble positions are permuted with permutation π defined in Table 4. The designers chose π to achieve the highest number of active S-boxes after 18, 19 and 20 rounds.

3 Difficulties of Analyzing LILLIPUT Round Function

In Sect. 4, we will show that the lower bounds of the number of active S-boxes provided by the authors are wrong. However, this is not because of careless mistakes. In Sect. 5, we will present a current best attack against LILLIPUT using division property. Before explaining details, in this section, we extract overview of the essential difficulties of analyzing EGFN and LILLIPUT with respect to differential cryptanalysis and division property.

Differential Cryptanalysis. Evaluating security of EGFN and Lilliput against differential cryptanalysis is quite difficult owing to their unique computation structure, $\mathcal{L}$. The previous truncated differential search, both dedicated search or more structural approach such as wide trail strategy in AES [10], yields a correct result only if the cipher can be assumed to be Markov cipher [11] with respect to truncated differential. Namely, the probability to achieve a truncated differential in round $i+1$ needs to be determined only depending on a truncated differential in round i (or possibly in any fixed round before round $i+1$).

A main obstacle for EGFN and Lilliput is that this assumption does not hold after a few rounds because of the linear layer $\mathcal{L}$. Let us discuss the Lilliput round function (Fig. 2).

- For some round j, X_{15}^{j} easily gets active thanks to $\mathcal{L}$, then X_{15}^{j} moves to X_{7}^{j+1} after $\mathcal{P}$.
- X_{7}^{j+1} duplicates *an identical difference* to X_{9}^{j+1} to X_{14}^{j+1}, and those will propagate to subsequent rounds.
- In a truncated differential, we only remember active/inactive of each nibble, thus we lose information that those differences are identical, which with high probability causes contradiction after a few rounds. (In Markov cipher, difference in round $j+2$ or later rounds should not depend on difference in round j.)

An example of contradicting truncated differential is shown in Figs. 3 and 4. The differential is 3 middle rounds of 16-round differential evaluated by the basic nibble-wise MILP model, which will be explained later. Figure 3 shows that the truncated differential is valid under the assumption that difference of all nibbles are reset to be a random difference in every round. Meanwhile, Fig. 4 traces the impact of linear diffusion. It shows that the difference of $x_{14}^{i+2}, x_{13}^{i+2}$, and x_{9}^{i+2} are the same as the one in x_{7}^{i}, which are denoted by Δ in Fig. 4. Here, we denote the difference of x_{7}^{i+2} by α, Then, the difference of the 9th, 13th, and 14th branches after the linear layer in round $i+2$ are denoted by $\Delta \oplus \alpha$. It is unknown if $\Delta \oplus \alpha$ is 0 or not, however, differences in those three branches must be identical. As one can see, Fig. 4 assumes that the 13th and 14th branches are inactive while the 9th branch is active. Thus this differential is contradicted.

Even with contradiction, it is still possible to provide lower bounds. However, the derived bounds are not tight as the linear layer $\mathcal{L}$, a source of contradiction, diffuses many truncated differential at once. Alternative approach is simulating differential propagation bit-by-bit precisely instead of truncated differential. However, this approach requires a very expensive search cost, and simulating all rounds is infeasible. All in all, evaluating security of EGFN and Lilliput against differential cryptanalysis is challenging work.

Integral Cryptanalysis. The designers of EGFN and Lilliput already showed the security against the integral cryptanalysis in [5,7], and the propagation characteristic of the integral property [8] was used to search for the integral distinguisher. When a d-round EGFN reaches the full diffusion, the integral

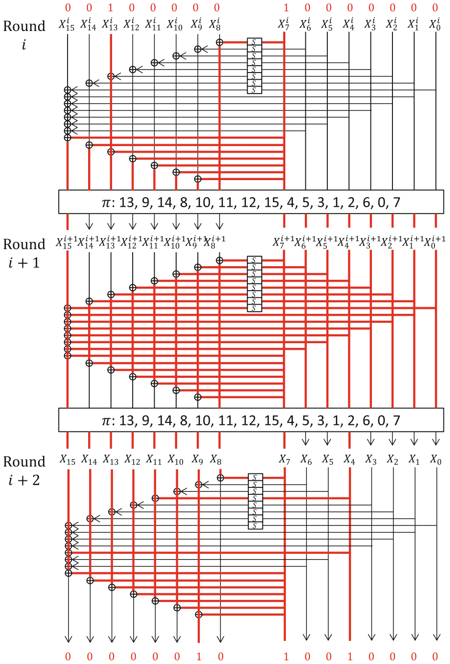

Fig. 3. Valid if differential is reset in every round.

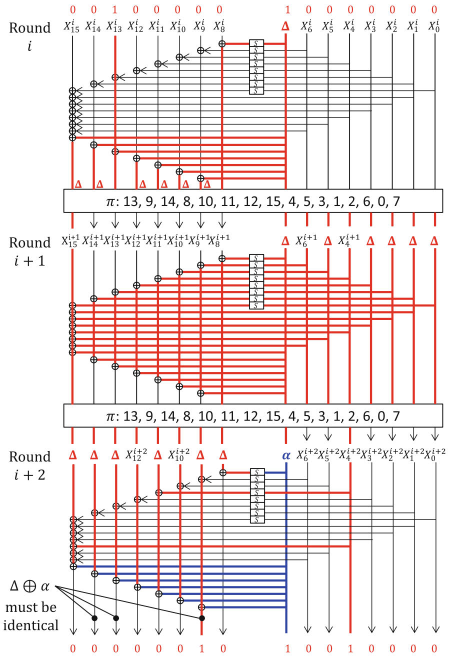

Fig. 4. Contradiction if linear propagation is considered.

distinguisher of the EGFN covers at most $2d + 2$ rounds. Moreover, Lilliput, which is a specific block cipher based EGFN with $d = 4$, has the 9-round integral distinguisher. Compared with 15-round integral distinguishers of TWINE and LBlock, it implies that the linear layer enhances the security against the integral cryptanalysis by $6(= 15 - 9)$ rounds. On the other hand, if we construct the integral distinguisher by estimating the algebraic degree, which is often called the higher order differential cryptanalysis, the security is not likely to dramatically improve because the linear layer does not increase the algebraic degree.

The division property is a new method to find integral distinguishers, and it is the generalization of the integral property so that can exploit the algebraic degree in the same time. Therefore, we can more accurately evaluate the contribution of the linear layer by using the division property. In Sect. 5, we will show a new integral distinguisher with the division property, and it covers 13 rounds, which is beyond $2d + 2 = 10$. Very recently, Zhang and Wu showed that TWINE and LBlock have 16-round integral distinguishers by using the division property [12]. Therefore, the true contribution by the linear layer is $3(= 16 - 13)$ rounds. Moreover, this 13-round integral distinguisher leads to a 17-round attack, which is a current best attack against Lilliput.

4 New Differential Bounds

We search for lower bounds of number of active S-boxes of Lilliput with MILP. Section 4.1 explains background of MILP based search. Section 4.2 explains nibble-wise search and proves the 16-round truncated differential shown by the designers are incorrect. Section 4.3 explains bit-wise search, which proves better bounds than the evaluation by the designers up to 11 rounds.

4.1 Background of Mixed Integer Linear Programming (MILP)

An MILP-based search was proposed by Mouha et al. [13]. The approach has two stages; (1) describing valid active byte/nibble/bit propagation patterns with a system of linear inequalities, and (2) solving the system with an MILP solver. Cryptographer's task is for (1) to efficiently describe active byte/nibble/bit patterns. Regarding stage (2), many softwares are available, some are license-free and other are in commerce. In this research, we used Gurobi Optimizer [14] for stage (2). Hereafter we explain stage (1).

The following discussion focuses on nibble-oriented ciphers. The goal is counting the number of active S-boxes, thus truncated differential is analyzed. Each nibble in each round is represented by a binary variable x_i meaning that the nibble is active when $x_i = 1$ and inactive when $x_i = 0$. Then, we specify an object to be optimized, called *objective function*. Our goal is finding a minimal number of active S-boxes, thus if S-box is applied to all nibbles, the objective function is "minimize $\sum_i x_i$." The main task is giving *constraint inequalities* to specify valid differential propagations with linear inequalities.

Inequations to Describe XOR by Mouha et al. Suppose that the nibble corresponding to x_3 is computed by other two nibbles corresponding to x_1 and x_2, i.e. $x_1 \oplus x_2 = x_3$. Mouha et al. describe all possible differential patterns by introduced a dummy binary variable d as follows.

$$\begin{aligned} x_1 + x_2 + x_3 - 2d &\geq 0, \\ x_1 - d &\leq 0, \\ x_2 - d &\leq 0, \\ x_3 - d &\leq 0. \end{aligned}$$

Bit-Wise Model by Sun et al. Several nibble-oriented ciphers cannot be evaluated with the approach by Mouha et al. An example is PRESENT, in which 4 bits output from a 4-bit S-box will be input to different S-box in the next round. Thus, it is necessary to look inside the S-box. Sun et al. proposed MILP-based search in a bit-wise model to simulate such a case [15], in which each binary variable x_i represents active/inactive of each bit. This approach is more advantageous for versatility, while it loses efficiency (the number of evaluated rounds is less).

A notable technique in [15] is to rule out impossible differential patterns from a feasible region of MILP. Recall the XOR case explained above, $x_3 = x_2 \oplus x_1$. We need to rule out $(x_1, x_2, x_3) = (1,0,0), (0,1,0), (0,0,1)$ and Sun et al. showed each impossible pattern can be ruled out with 1 inequality. For example, $(x_1, x_2, x_3) = (1,0,0)$ is ruled out with $-x_1 + x_2 + x_3 \geq 0$. Indeed, any other value of (x_1, x_2, x_3) satisfy this inequality, and thus only $(x_1, x_2, x_3) = (1,0,0)$ is ruled out. $(0,1,0)$ and $(0,0,1)$ can be ruled out similarly.

4.2 Nibble-Wise Search

We first explain a basic method which assumes that the difference of each active nibble is reset to a random difference in every round. This assumption is clearly incorrect for the real specification because the linear layer $\mathcal{L}$ diffuses difference only linearly (difference in round j uniquely determines difference in round $j+1$ in $\mathcal{L}$). Hence, the derived lower bounds are loose. We then show that equivalently transforming the cipher's description helps us to improve the model that can derive tighter lower bounds.

Constructing Basic Model. We assign a binary variable to each nibble in every round. Thus we use $16r$ variables for r rounds; $x_0, \ldots, x_{15}$ for round 1, $x_{16}, \ldots, x_{31}$ for round 2, and so on.

As for the objective function, our goal is minimizing the number of active S-boxes, thus we minimize the sum of x_i in the right half of the state, i.e. "minimize $\sum_r \sum_{j=0}^{7} x_{16r+j}$."

Constraint inequalities can be derived round-by-round. For simplicity, we explain constraints between $x_0, \ldots, x_{15}$ and $x_{16}, \ldots, x_{31}$, which are depicted in Fig. 5. The other rounds can be modeled just by replacing indices. S-box and key addition do not impact to truncated differential, thus we omit them in Fig. 5. First, we list variables before the permutation layer, which are $\pi^{-1}(x_{16}, \ldots, x_{31})$. Here the right most one, x_{29}, can be represented by $x_{\pi(0)+16}$. Similarly, the other 15 variables can be represented by $x_{\pi(1)+16}, x_{\pi(2)+16}, \ldots, x_{\pi(15)+16}$. This representation is useful to systematically construct MILP models. We then derive constraint inequalities between $x_0, x_1, \ldots x_{15}$ and $x_{\pi(0)+16}, x_{\pi(1)+16}, \ldots, x_{\pi(15)+16}$ dividing them into four types.

Type 1: Right half of the state is not updated. Constraints are $x_{\pi(i)+16} = x_i$ for $i = 0, 1, \ldots, 7$.

Type 2: $x_{\pi(8)+16}, x_8$ and x_7 must be a valid xor, i.e. $(x_{\pi(8)+16}, x_8, x_7) = (1,0,0), (0,1,0), (0,0,1)$ are impossible. We rule out those three patterns with the following three inequalities;

$$\begin{aligned}
-x_{\pi(8)+16} + x_8 + x_7 &\geq 0,\\
x_{\pi(8)+16} - x_8 + x_7 &\geq 0,\\
x_{\pi(8)+16} + x_8 - x_7 &\geq 0.
\end{aligned}$$

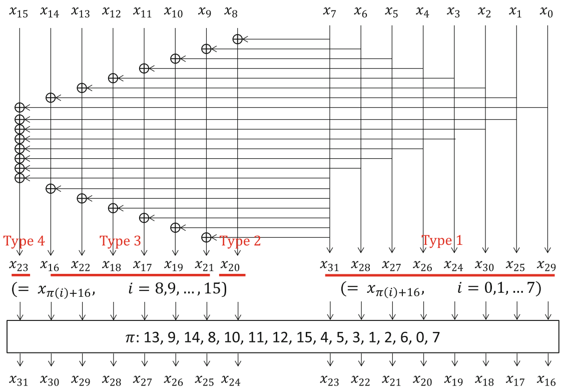
x_{15} x_{14} x_{13} x_{12} x_{11} x_{10} x_9 x_8
x_7 x_6 x_5 x_4 x_3 x_2 x_1 x_0
Type 4
Type 3
Type 2
Type 1
x_{23} x_{16} x_{22} x_{18} x_{17} x_{19} x_{21} x_{20}
x_{31} x_{28} x_{27} x_{26} x_{24} x_{30} x_{25} x_{29}
$(= x_{\pi(i)+16}, \quad i = 8,9,\ldots,15)$
$(= x_{\pi(i)+16}, \quad i = 0,1,\ldots 7)$
π: 13, 9, 14, 8, 10, 11, 12, 15, 4, 5, 3, 1, 2, 6, 0, 7
x_{31} x_{30} x_{29} x_{28} x_{27} x_{26} x_{25} x_{24}
x_{23} x_{22} x_{21} x_{20} x_{19} x_{18} x_{17} x_{16}

Fig. 5. Nibble based MILP model for LILLIPUT.

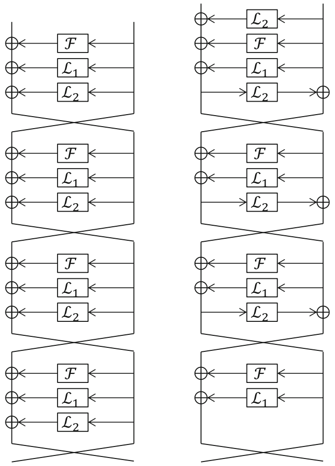

Fig. 6. Equivalent descriptions.

Type 3: For $j = 9, 10, \ldots, 14$, $x_{\pi(j)+16}, x_{8+j}, x_{7-j}, x_7$ must be a valid xor. We rule out $(x_{\pi(j)+16}, x_{8+j}, x_{7-j}, x_7) = (1,0,0,0), (0,1,0,0), (0,0,1,0), (0,0,0,1)$. Similarly to Type 2, this can be done with four inequalities for each j.

Type 4: $x_{\pi(15)+16}$ and other 9 input variables must be a valid xor. Similarly to Type 2 and Type 3, differential propagation is impossible if and only if exactly one variable is active. There are ten impossible patterns, and these are ruled out with ten inequalities.

In total, we use $8 + 3 + (6 * 4) + 10 = 45$ inequalities per round, thus $45r$ for r rounds. In addition we use 1 inequalities $\sum_{j=0}^{15} x_i > 0$ to ensure at least one nibble is active in plaintext.

Results of Basic Model. Execution time is reasonably short. The system for 16 rounds was solved in a few minutes by a standard PC. The results are shown in Table 1. At first glance, the derived bounds are worse than the designers' evaluation. However this is not right. The designers claimed that the best 16-round characteristic activates 31 S-boxes [7, Sect. 7].

> *we provide here the best truncated differential and linear masks we found for 16 rounds of LILLIPUT with 31 active S-boxes · The best truncated differential path is given by an input of the form* $(\alpha_0, 0, \alpha_0, \alpha_0, \alpha_0, \alpha_0, \alpha_0, \alpha_1, \alpha_0, 0, 0, 0, 0, 0, 0, 0,)$ *that gives after 16 rounds an output of the form* $(0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, \beta, 0) \cdots$

We tested their input and output differential masks. We obtained that the (loose) lower bound for those masks is 34 for 16 rounds, thus their claim is wrong.[1]

[1] We communicated to the designers and asked to provide more details, in particular differential masks for every round. The designers have not provide us the details.

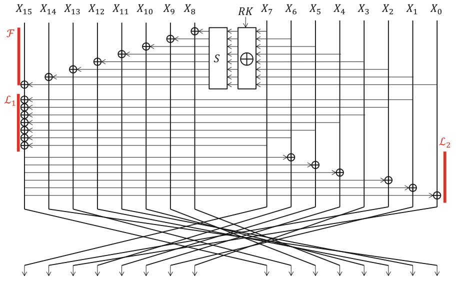

Fig. 7. Equivalently Transformed Round Function Analyzed in Advanced MILP Model.

Constructing Advanced Model. The drawback in the basic model is that the truncated differential is assumed to be reset in every round, while it is not in the actual specification. Indeed, we manually verified several optimal solutions returned by a solver, but they always include contradiction. Namely, the bound is not tight (though 30 rounds seem sufficient to resist differential cryptanalysis). Let us analyze more details. We divide the linear layer $\mathcal{L}$ into two layers $\mathcal{L}_1$ and $\mathcal{L}_2$, in which $\mathcal{L}_1$ is the diffusion from $X_1, X_2, \ldots, X_7$ to X_{15} and $\mathcal{L}_2$ is the diffusion from X_7 to $X_9, X_{10}, \ldots, X_{15}$ defined below (illustrated in Fig. 7).

$$\begin{aligned}
&\mathcal{L}_1 : X_{15} \leftarrow X_{15} \oplus X_7 \oplus X_6 \oplus X_5 \oplus X_4 \oplus X_3 \oplus X_2 \oplus X_1, \\
&\mathcal{L}_2 : X_{15-i} \leftarrow X_{15-i} \oplus X_7 \text{ for } i = 1, 2, \ldots, 6.
\end{aligned}$$

Our observation is that the impact of linear diffusion with $\mathcal{L}_1$ and $\mathcal{L}_2$ never interact within one round. X_{15} is (easily) activated through $\mathcal{L}_1$, and this moves to x_7 after $\mathcal{P}$, and in the next round, x_7 diffuses with $\mathcal{L}_2$. In the basic MILP model, the above combination effect via $\mathcal{L}_1$ and $\mathcal{L}_2$ over two rounds cannot be captured due to the difference reset in every round.

Our improving idea is moving the position of the linear layer $\mathcal{L}_2$ so that the cancellation through $\mathcal{L}_1$ and $\mathcal{L}_2$ can be simulated within one round. In details, we move $\mathcal{L}_2$ for round i (diffusion from X_7 in round i) to round $i-1$ (diffusion from $\pi^{-1}(X_7) = X_{15}$ in round $i-1$). The converted computation structure is shown in the right-half of Fig. 6. Note that the original $\mathcal{L}_2$ in the first round can be regarded as a preprocessing and $\mathcal{L}_2$ in the last round is removed.

Results of Advanced Model. Execution time of the advanced model is almost the same as the basic one. The results are shown in Table 1. Compared to the basic model, the lower bounds are improved when the number of rounds is 12, 14, 15 and 16. Compared to the designers' original expectation, the lower bounds are improved, meaning that Lilliput is more secure than it was expected. In particular, proving 32 active S-boxes for 16 rounds is important owing to the 64-bit block size and the maximum differential probability of the S-box, 2^{-2}.

Even with the advanced model, contradiction via $\mathcal{L}$ over 3 rounds cannot be simulated, thus the bounds are not tight. This motivates us to generate tight bounds in the next section.

4.3 Bit-Wise Search

The bit-wise model traces active/inactive of each bit. The main advantage is that the cancellation by the xor operation, which is the main cause of the contradiction in the nibble-wise model, can be simulated precisely and solving the system becomes equivalent to finding the best characteristic. Meanwhile, S-box is not bit-wise thus cannot be ignored as the nibble-wise model, which requires a large number of constraint inequalities to describe valid differential propagations.

Variables in One Round. We assign a binary variable x_i bit by bit. To reduce a total number of variables, we introduce new variables only for updated 32 bits (right half of the state) in every round. Besides, active/inactive of each bit changes through S-box, thus we introduce a binary variable y_i to describe active/inactive of each bit of S-box output.

Permutation π needs to be adjusted to be bitwise, π_{bw}. The conversion is straightforward, thus we omit it.

Number of Active S-Boxes in Bitwise Model. We need to convert active-bit information into active-nibble one to count the number of active S-boxes. Here, we introduce a dummy binary variable, n. Suppose that n_{4i} is a nibble whose corresponding 4 input bits are $x_i, x_{i+1}, x_{i+2}, x_{i+3}$. We set constrains so that n_{4i} becomes 1 when at least one of $x_i, \ldots, x_{i+3}$ are active and $n_{4i} = 0$ if all of $x_i, \ldots, x_{i+3}$ are inactive. This can be done by borrowing the idea of simulating XOR by Mouha et al. [13], and we set the following five inequalities;

$$\begin{aligned} x_i + x_{i+1} + x_{i+2} + x_{i+3} - n_{4i} &\geq 0, \\ n_{4i} - x_i &\geq 0, \\ n_{4i} - x_{i+1} &\geq 0, \\ n_{4i} - x_{i+2} &\geq 0, \\ n_{4i} - x_{i+3} &\geq 0. \end{aligned}$$

If all of $x_i, x_{i+1}, x_{i+2}, x_{i+3}$ are inactive $(= 0), n_{4i}$ becomes 0. If at least one of $x_i, x_{i+1}, x_{i+2}, x_{i+3}$ is active $(= 1), n_{4i}$ becomes 1. Thus, n_{4i} represents active/inactive of the S-box.

Each round computes 8 S-boxes. The objective function for r rounds is "minimize $\sum_{i=0}^{8r-1} n_i$."

Constraints for S-Box. We first generate differential distribution table (DDT). DDT consists of 150 zero entries (impossible propagations). With the approach by Sun et al. [15], we can rule out each impossible propagation with one inequality.

For example, $x_{i+3} \| x_{i+2} \| x_{i+1} \| x_i = \mathtt{0010}$ and $y_{i+3} \| y_{i+2} \| y_{i+1} \| y_i = \mathtt{0011}$ is an impossible propagation and this can be ruled out by

$$x_{i+3} + x_{i+2} - x_{i+1} + x_i + y_{i+3} + y_{i+2} - y_{i+1} - y_i \geq -2.$$

All the impossible differential propagations can be ruled out with at most 150 inequalities. Sun *et al.* showed that several impossible propagations may be ruled out with 1 inequality.

For example, $x_{i+3}\|x_{i+2}\|x_{i+1}\|x_i\|y_{i+3}\|y_{i+2}\|y_{i+1}\|y_i = \texttt{*00**101}$ is impossible for any choice of $* \in \{0,1\}$. Those 8 patterns are ruled out by

$$x_{i+2} + x_{i+1} - y_{i+2} + y_{i+1} - y_i \geq -1.$$

We exhaustively searched for such compact representations. The number of total constraint inequalities should be minimized. We followed the approach by Sun et al. [16] using the greedy algorithm to choose constraint inequalities. In the end, we rule out all 150 impossible differential patterns with 46 inequalities.

Constraints Other Than S-Box. Update on 28 bits, from bit positions 32 to 59, is rather simple. If the computation is the 2-input xor, e.g. $a \oplus b = c$, the number of impossible propagations is 4; $(a, b, c) = (1, 0, 0), (0, 1, 0), (0, 0, 1), (1, 1, 1)$, which can be ruled out with 4 inequalities. Note that differently from nibble-wise search, $(a, b, c) = (1, 1, 1)$ is impossible in the bitwise computation. Similarly, for 3-input xor, e.g. $a \oplus b \oplus c = d$, there are 8 impossible patterns, and we can rule them out with 8 inequalities.

The last 4 bits are updated with 9-input xor, thus the number of impossible propagations is $2^9 = 512$ per bit. Using 512 inequalities is too expensive. Here, we focus on the property that the sum of involved variables must be even. We introduce an *integer* dummy variable e, where $e \in \{0, 1, 2, 3, 4\}$. Let 9 input bits be $y_0, x_4, x_8, \ldots, x_{28}$ and 1 output bit be $x_{\pi_{bw}(60)}$. We set the following constraint;

$$y_0 + x_4 + x_8 + x_{12} + x_{16} + x_{20} + x_{24} + x_{28} - x_{\pi_{bw}(60)} = 2e.$$

Result of Bitwise Model. Owing to the expensive computational cost, the machine performance is an important factor for this research. We executed Gurobi Optimizer with Xeon Processor E5-2699 (18 cores) in 128 GB RAM. The results are shown in Table 1. It provides the best bound from 8 rounds and we confirmed the tightness. Namely the optimal solutions can be used for attacks. The running time for 8, 9, 10, and 11 rounds is 2746 s, 5512 s, 53099 s (≈14 h), and about 1 week, respectively. Because of the complicated algorithm of the MILP solver, it is difficult to predict the running time for more rounds.

5 Attacks Based on Division Property

5.1 Background of Division Property

The division property proposed in [9] is a new method to find integral distinguishers. This section briefly shows the definition and propagation rules to understand this paper. Please refer to [9] for details.

The division property of a multiset is evaluated by using the bit product function defined as follows. Let $\pi_{\boldsymbol{u}} : (\mathbb{F}_2^n)^m \to \mathbb{F}_2$ be a bit product function for any $\boldsymbol{u} \in (\mathbb{F}_2^n)^m$. Let $\boldsymbol{x} \in (\mathbb{F}_2^n)^m$ be the input, and $\pi_{\boldsymbol{u}}(\boldsymbol{x})$ is defined as

$$\pi_{\boldsymbol{u}}(\boldsymbol{x}) := \prod_{i=1}^{m} \left(\prod_{j=1}^{n} x_i[j]^{u_i[j]} \right).$$

Notice that $x_i[j]^1 = x_i[j]$ and $x_i[j]^0 = 1$.

Definition 1 (Division Property [9]**).** *Let $\mathbb{X}$ be a multiset whose elements take a value of $(\mathbb{F}_2^n)^m$. When the multiset $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^{n^m}$, where $\mathbb{K}$ denotes a set of m-dimensional vectors whose elements take a value between 0 and n, it fulfills the following conditions:*

$$\bigoplus_{\boldsymbol{x} \in \mathbb{X}} \pi_{\boldsymbol{u}}(\boldsymbol{x}) = \begin{cases} \textit{unknown} & \textit{if there are } \boldsymbol{k} \in \mathbb{K} \textit{ s.t. } W(\boldsymbol{u}) \succeq \boldsymbol{k}, \\ 0 & \textit{otherwise}, \end{cases}$$

where $W(\boldsymbol{u}) = (w(u_m), \ldots, w(u_1)) \in \mathbb{Z}^m$ and $w(u_j) = \sum_{i=1}^{n} u_j[i]$. Moreover, $\boldsymbol{k} \succeq \boldsymbol{k}'$ denotes $k_i \geq k_i'$ for all $i \in \{1, 2, \ldots, m\}$.

If there are $\boldsymbol{k} \in \mathbb{K}$ and $\boldsymbol{k}' \in \mathbb{K}$ satisfying $\boldsymbol{k} \succeq \boldsymbol{k}'$ in the division property $\mathcal{D}_{\mathbb{K}}^{n^m}$, $\boldsymbol{k}$ can be removed from $\mathbb{K}$ because the vector $\boldsymbol{k}$ is redundant. Let $\mathbb{X}$ be the set of texts encrypted by r rounds, and $e_i \in \mathbb{Z}^m$ denotes an unit vector whose ith element is one and the others are zero. Assuming that $\mathbb{X}$ fulfills the division property $\mathcal{D}_{\mathbb{K}}^{n^m}$ and e_i does not belong to $\mathbb{K}$, the cipher has the r-round integral distinguisher, where the ith element is balanced.

We summarize propagation rules that we use in this paper as follows.

Rule 1 (Substitution). Let F be a function that consists of m S-boxes, where the bit length and the algebraic degree of S-boxes is n bits and d, respectively. The input and the output take a value of $(\mathbb{F}_2^n)^m$ and $\mathbb{X}$ and $\mathbb{Y}$ denote the input multiset and the output multiset, respectively. Assuming that the multiset $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^{n^m}$, the multiset $\mathbb{Y}$ has the division property $\mathcal{D}_{\mathbb{K}'}^{n^m}$, where $\mathbb{K}'$ is calculated as follows: First, $\mathbb{K}'$ is initialized to ϕ. Then, for all $\boldsymbol{k} \in \mathbb{K}$,

$$\mathbb{K}' = \mathbb{K}' \cup \left[\left\lceil \frac{k_1}{d} \right\rceil, \left\lceil \frac{k_2}{d} \right\rceil, \ldots, \left\lceil \frac{k_m}{d} \right\rceil \right],$$

is calculated. Here, when the ith S-box is bijective and $k_i = n$, the ith element of the propagated property becomes n not $\lceil n/d \rceil$.

Rule 2 (Copy). Let F be a copy function, where the input x takes a value of $\mathbb{F}_2^n$ and the output is calculated as $(y_1, y_2) = (x, x)$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input multiset and output multiset, respectively. Assuming that the multiset $\mathbb{X}$ has the division property $\mathcal{D}_k^n$, the multiset $\mathbb{Y}$ has the division property $\mathcal{D}_{\mathbb{K}'}^{n,n}$, where $\mathbb{K}'$ is calculated as follows: First, $\mathbb{K}'$ is initialized to ϕ. Then, for all i $(0 \leq i \leq k)$,

$$\mathbb{K}' = \mathbb{K}' \cup [k - i, i],$$

is calculated.

Rule 3 (Compression by XOR). Let F be a function compressed by an XOR, where the input (x_1, x_2) takes a value of $(\mathbb{F}_2^n \times \mathbb{F}_2^n)$ and the output is calculated as $y = x_1 \oplus x_2$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input multiset and output multiset, respectively. Assuming that the multiset $\mathbb{X}$ has the division property $\mathcal{D}_{\mathbb{K}}^{n,n}$, the division property of the multiset $\mathbb{Y}$ is $\mathcal{D}_{k'}^{n}$ as

$$k' = \min_{[k_1,k_2] \in \mathbb{K}} \{k_1 + k_2\}.$$

Here, if the minimum value of k' is larger than n, the propagation characteristic of the division property is aborted. Namely, a value of $\oplus_{y \in \mathbb{Y}} \pi_v(y)$ is 0 for all $v \in \mathbb{F}_2^n$.

These propagation rules are proven in [9,17].

5.2 Integral Distinguisher on Lilliput

The state of Lilliput is represented as sixteen 4-bit values, and the use of the division property $\mathcal{D}_{\mathbb{K}}^{4^{16}}$ is appropriate. Let $|\mathbb{K}|$ be the number of elements in $\mathbb{K}$, and the upper bound of $|\mathbb{K}|$ is $5^{16} \approx 2^{37.15}$. Since we can reduce $|\mathbb{K}|$ by removing redundant vectors in general, we can practically evaluate the propagation characteristic of $\mathcal{D}_{\mathbb{K}}^{4^{16}}$.

Propagation Characteristic. The round function of EGFN consists of three layers: the non-linear layer, the linear layer, and the permutation layer. In the non-linear layer of EGFN, the core operation is

$$x_i = x_i \oplus F(x_j)$$

for appropriate i and j. We only focus on the case that F is permutation because the most important instantiation Lilliput uses a bijective S-box. Let $\mathcal{D}_k^4$ and $\mathcal{D}_{k'}^4$ be the input and output division property for the S-box, respectively. As the algebraic degree of F is at most three, it holds

$$k' = D_S(k) = \begin{cases} 4 & \text{if } k = 4, \\ 1 & \text{if } k = 1, 2, 3, \\ 0 & \text{if } k = 0. \end{cases}$$

Assuming $\mathcal{D}_{(k_i,k_j)}^{4^2}$ be the input division property of the Feistel structure, the output division property $\mathcal{D}_{\mathbb{K}}^{4^2}$ is

$$\mathbb{K} = \{(k_i + D_S(x), k_j - x) \mid 0 \le x \le k_j, D_S(x) \le 4 - k_i\}.$$

The propagation characteristic for the non-linear layer is shown in `nonLinear` of Algorithm 1.

Algorithm 1. Propagation from $\mathcal{D}^{4^{16}}_{\mathbb{K}}$ for the round function of Lilliput

```
procedure nonLinear($\mathbb{K}, i, j$)
    $\mathbb{K}' \Leftarrow \phi$
    for all $\boldsymbol{k} \in \mathbb{K}$ do
        $\boldsymbol{k}' \Leftarrow \boldsymbol{k}$
        for $x = 0$ to $k_j$ do
            $k'_i \Leftarrow k_i + D_S(x)$
            $k'_j \Leftarrow k_j - x$
            if $k'_i \leq 4$ then
                $\mathbb{K}' \Leftarrow \mathbb{K}' \cup \{\boldsymbol{k}'\}$
            end if
        end for
    end for
    remove redundant vectors from $\mathbb{K}'$
    return $\mathbb{K}'$
end procedure
```

```
procedure linear($\mathbb{K}, i, j$)
    $\mathbb{K}' \Leftarrow \phi$
    for all $\boldsymbol{k} \in \mathbb{K}$ do
        $\boldsymbol{k}' \Leftarrow \boldsymbol{k}$
        for $x = 0$ to $k_j$ do
            $k'_j \Leftarrow k_j - x$
            $k'_i \Leftarrow k_i + x$
            if $k'_i \leq 4$ then
                $\mathbb{K}' \Leftarrow \mathbb{K}' \cup \{\boldsymbol{k}'\}$
            end if
        end for
    end for
    remove redundant vectors from $\mathbb{K}'$
    return $\mathbb{K}'$
end procedure
```

The linear layer of EGFN consists of the iteration of XORs as

$$x_i = x_i \oplus x_j$$

for appropriate i and j. Therefore, assuming $\mathcal{D}^{4^2}_{(k_i,k_j)}$ be the input division property of the Feistel structure, the output division property $\mathcal{D}^{4^2}_{\mathbb{K}}$ is

$$\mathbb{K} = \{(k_i + x, k_j - x) \mid 0 \leq x \leq \min\{k_j, 4 - k_i\}\}.$$

The propagation characteristic for the linear layer is shown in `linear` of Algorithm 1.

About the permutation layer, the propagation characteristic is the only modification of the corresponding index. The entire algorithm to evaluate the propagation characteristic of the round function is shown in `roundFunction` of Algorithm 2.

New Integral Distinguisher. As the number of exploiting chosen plaintexts increases, the integral distinguisher can analyze more rounds in general. Therefore, we evaluate all integral distinguishers with 2^{63} chosen plaintexts where only one bit in the right half is constant. Note that these distinguishers are always better than distinguishers whose only one bit in the left half is constant. We choose one 4-bit value from X_0 to X_7, and we prepare chosen plaintexts such that any one bit in the chosen value is constant and the others are active.

We implemented Algorithm 2 and searched non-trivial integral distinguishers. Let $\mathcal{D}^{4^{16}}_{\boldsymbol{k}}$ be the plaintext division property. When we choose one-bit constant from X_p, we use $\boldsymbol{k}$ as

$$k_i = \begin{cases} 4 & \text{if } i \neq p \\ 3 & \text{if } i = p \end{cases}$$

Algorithm 2. Propagation from $\mathcal{D}_{\mathbb{K}}^{4^{16}}$ for the round function of Lilliput

```
procedure roundFunction($\mathbb{K}$)
    for all $(i, j) \in \{(8,7), (9,6), (10,5), (11,4), (12,3), (13,2), (14,1), (15,0)\}$ do
        $\mathbb{K}$ = nonLinear($\mathbb{K}, i, j$)
    end for
    for all $(i, j) \in \{(15,1), (15,2), (15,3), (15,4), (15,5), (15,6), (15,7)\}$ do
        $\mathbb{K}$ = linear($\mathbb{K}, i, j$)
    end for
    for all $(i, j) \in \{(14,7), (13,7), (12,7), (11,7), (10,7), (9,7)\}$ do
        $\mathbb{K}$ = linear($\mathbb{K}, i, j$)
    end for
    $\mathbb{K}' \Leftarrow \phi$
    for all $\boldsymbol{k} \in \mathbb{K}$ do
        for $i = 0$ to $16$ do
            $k'_{\pi(i)} \Leftarrow k_i$
        end for
        $\mathbb{K}' \Leftarrow \mathbb{K}' \cup \{\boldsymbol{k}'\}$
    end for
    return $\mathbb{K}'$
end procedure
```

Table 5. Propagation from $\mathcal{D}_{\{[4,4,\ldots,4,3]\}}^{4^{16}}$

#rounds	0	1	2	3	4	5	6 ⋆
$\|\mathbb{K}\|$	1	1	3	14	377	33948	5513237
$\max_w(\mathbb{K})$	63	63	63	63	63	55	≤57
$\min_w(\mathbb{K})$	63	63	61	59	55	19	35
#rounds	7 ⋆	8 ⋆	9 ⋆	10	11	12	13
$\|\mathbb{K}\|$	266813452	70804820	1385951	16960	572	52	16
$\max_w(\mathbb{K})$	≤51	≤43	≤25	13	6	4	2
$\min_w(\mathbb{K})$	22	9	6	3	2	1	1

In rounds labeled ⋆, the set $\mathbb{K}$ includes redundant vectors.

for $i \in \{0, 1, \ldots, 16\}$. We coded our algorithm with C++, and we executed it in Xeon Processor E5-2699 (18 cores) in 128 GB RAM. As a result, our algorithm found 13-round integral distinguishers for $p = 0$ and $p = 6$. For other p, our algorithm found 12-round integral distinguishers.

When $p = 0$ i.e., $\boldsymbol{k} = [4,4,4,4,4,4,4,4,4,4,4,4,4,4,4,3]$, we find a 13-round integral distinguisher, and the position X_9^{13} is balanced. Table 5 shows the propagation characteristic, where $\min_w(\mathbb{K})$ and $\max_w(\mathbb{K})$ are calculated as

$$\min_w(\mathbb{K}) = \min_{\boldsymbol{k} \in \mathbb{K}} \left\{ \sum_{i=1}^{16} k_i \right\}, \ \max_w(\mathbb{K}) = \max_{\boldsymbol{k} \in \mathbb{K}} \left\{ \sum_{i=1}^{16} k_i \right\}.$$

Round 0 denotes the division property of the plaintext set, and we perfectly remove redundant vectors except for 6, 7, 8, and 9 rounds.

5.3 Key Recovery

Let X_i^j be the j-round nibble value in X_i, where the plaintext is represented as $(X_{15}^0, \ldots, X_0^0)$. Moreover, let Y_i^j be the output of the S-box as $Y_i^j = S(X_i^j \oplus RK_i^j)$. We prepare 2^{63} chosen plaintexts such that any one bit of X_0^0 is constant and the other 63 bits are active. Then, it holds $\bigoplus X_9^{13} = 0$, and we can attack 17-round LILLIPUT by using the 13-round integral distinguisher. In our attack, let $c = (c_{15}, \ldots, c_0)$ be the ciphertext, where the linear layer of the last round is removed. Note that the last round of LILLIPUT has the linear layer but this c is equivalent with the ciphertext of 17-round LILLIPUT because the linear layer is public.

Since LILLIPUT has many XORs in the round function, the procedure of the key recovery is very complicating. For simplicity, we use the following strategy. We first decompose four rounds of LILLIPUT into five subfunctions denoted by $f_{13}, f_{14}, f_{15}, f_{16}$, and L. Here the output of f_i is the XOR of Y^i involved in X_9^{13}, and the output of L is the linear part to compute X_9^{13} from ciphertext. Then

$$X_9^{13} = f_{13}(c, \mathcal{K}_{13}) \oplus f_{14}(c, \mathcal{K}_{14}) \oplus f_{15}(c, \mathcal{K}_{15}) \oplus f_{16}(c, \mathcal{K}_{16}) \oplus L(c),$$

where $\mathcal{K}_i$ is the set of round keys involved in f_i. The bit sizes of $\mathcal{K}_{13}, \mathcal{K}_{14}, \mathcal{K}_{15}$, and $\mathcal{K}_{16}$ are 44, 16, 48, and 28 bits, respectively. Then,

$$\begin{aligned}
f_{13}(c, \mathcal{K}_{13}) &= Y_6^{13}, \\
f_{14}(c, \mathcal{K}_{14}) &= Y_0^{14}, \\
f_{15}(c, \mathcal{K}_{15}) &= Y_0^{15} \oplus Y_1^{15} \oplus Y_3^{15} \oplus Y_5^{15} \oplus Y_6^{15} \oplus Y_7^{15}, \\
f_{16}(c, \mathcal{K}_{16}) &= Y_0^{15} \oplus Y_1^{15} \oplus Y_3^{15} \oplus Y_4^{15} \oplus Y_5^{15} \oplus Y_6^{15} \oplus Y_7^{15}.
\end{aligned}$$

We compute the sum of $f_i(c, \mathcal{K}_i)$ by guessing $\mathcal{K}_i$ independently of i. Then, we compute keys satisfying

$$\bigoplus_{X^0} f_{13}(c, \mathcal{K}_{13}) \oplus f_{14}(c, \mathcal{K}_{14}) \oplus f_{16}(c, \mathcal{K}_{16}) = \bigoplus_{X^0} f_{15}(c, \mathcal{K}_{15}) \oplus L(c) \qquad (1)$$

Note that we do not need to guess round keys to compute the sum of $L(c)$. Note that $\mathcal{K}_{13} \cup \mathcal{K}_{14} \cup \mathcal{K}_{15} \cup \mathcal{K}_{16}$ is 72 bits, and the probability that Eq. (1) holds randomly is 2^{-4}. Therefore, we reduce the space of key candidates from 2^{72} to 2^{68}. Finally, we recover the correct key by additionally guessing the remaining 8 bits. It is enough to determine the correct key by using two known plaintexts. Thus, the total time complexity is $2^{76} \times 2 = 2^{77}$.

Note that the time complexity that we evaluate whether Eq. (1) holds or not is less than 2^{61} and it is negligible because of [18,19]. Due to the limited space, we omit the detailed procedure.

6 Concluding Remarks

In this paper, we showed security evaluation of LILLIPUT. The linear layer $\mathcal{L}$, which is the main feature introduced by EGFN, gives several security concerns to be carefully discussed. By using MILP, we proved that the lower bounds of number of active S-boxes provided by the designers were incorrect. Then, we derived new bounds in two approaches; nibble-wise and bitwise models. Interestingly, it turned out that security of Lilliput is better than the original expectation. Further improving the lower bounds and deriving tight bounds for more rounds will be interesting future research directions. Meanwhile, we showed that the security enhance by the linear layer $\mathcal{L}$, which applies many xors without increasing S-box, is not so strong against division property, and improved the previous best key recovery attacks by three rounds. EGFN is a relatively new design approach. We believe that this paper leads to better understanding of EGFN.

References

1. Biryukov, A., Johann Großschädl, Y.L.C.: CryptoLUX, Lightweight Cryptography (2015). https://www.cryptolux.org/index.php/Lightweight_Cryptography
2. Suzaki, T., Minematsu, K.: Improving the generalized Feistel. In: Hong, S., Iwata, T. (eds.) FSE 2010. LNCS, vol. 6147, pp. 19–39. Springer, Heidelberg (2010). doi:10.1007/978-3-642-13858-4_2
3. 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
4. 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
5. 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
6. Zhang, L., Wu, W.: Differential analysis of the extended generalized Feistel networks. Inf. Process. Lett. **114**(12), 723–727 (2014)
7. 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. **65**, 2074–2089 (2015)
8. Knudsen, L., Wagner, D.: Integral cryptanalysis. In: Daemen, J., Rijmen, V. (eds.) FSE 2002. LNCS, vol. 2365, pp. 112–127. Springer, Heidelberg (2002). doi:10.1007/3-540-45661-9_9
9. Todo, Y.: Structural evaluation by generalized integral property. In: Oswald, E., Fischlin, M. (eds.) EUROCRYPT 2015. LNCS, vol. 9056, pp. 287–314. Springer, Heidelberg (2015). doi:10.1007/978-3-662-46800-5_12
10. Daemen, J., Rijmen, V.: The Design of Rijndeal: AES - The Advanced Encryption Standard (AES). Springer, Heidelberg (2002). doi:10.1007/978-3-662-04722-4
11. Lai, X., Massey, J.L., Murphy, S.: Markov ciphers and differential cryptanalysis. In: Davies, D.W. (ed.) EUROCRYPT 1991. LNCS, vol. 547, pp. 17–38. Springer, Heidelberg (1991). doi:10.1007/3-540-46416-6_2
12. Zhang, H., Wu, W.: Structural evaluation for generalized Feistel structures and applications to LBlock and TWINE. In: Biryukov, A., Goyal, V. (eds.) INDOCRYPT 2015. LNCS, vol. 9462, pp. 218–237. Springer, Cham (2015). doi:10.1007/978-3-319-26617-6_12

13. 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
14. Gurobi Optimization Inc.: Gurobi optimizer 6.5 (2015). Official webpage http://www.gurobi.com/
15. 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
16. Sun, S., Hu, L., Wang, M., Wang, P., Qiao, K., Ma, X., Shi, D., Song, L.: Automatic enumeration of (related-key) differential and linear characteristics with predefined properties and its applications. IACR Cryptol. ePrint Arch. **2014**, 747 (2014)
17. Todo, Y.: Integral cryptanalysis on full MISTY1. In: Gennaro, R., Robshaw, M. (eds.) CRYPTO 2015. LNCS, vol. 9215, pp. 413–432. Springer, Heidelberg (2015). doi:10.1007/978-3-662-47989-6_20
18. Ferguson, N., Kelsey, J., Lucks, S., Schneier, B., Stay, M., Wagner, D., Whiting, D.: Improved cryptanalysis of Rijndael. In: Goos, G., Hartmanis, J., van Leeuwen, J., Schneier, B. (eds.) FSE 2000. LNCS, vol. 1978, pp. 213–230. Springer, Heidelberg (2001). doi:10.1007/3-540-44706-7_15
19. Sasaki, Y., Wang, L.: Meet-in-the-middle technique for integral attacks against Feistel ciphers. In: Knudsen, L.R., Wu, H. (eds.) SAC 2012. LNCS, vol. 7707, pp. 234–251. Springer, Heidelberg (2013). doi:10.1007/978-3-642-35999-6_16