Integral Cryptanalysis of Reduced-Round SAND-64 Based on Bit-Based Division Property

Atiyeh Mirzaie
Information Systems and Security Lab (ISSL), Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran
atiyeh.mirzaie@ee.sharif.edu

Siavash Ahmadi
Electronics Research Institute, Sharif University of Technology, Tehran, Iran
s.ahmadi@sharif.edu

Mohammad Reza Aref
Information Systems and Security Lab (ISSL), Department of Electrical Engineering, Sharif University of Technology, Tehran, Iran
aref@sharif.edu

***Abstract*—Conventional Bit-based Division Property (CBDP), as a generalization of integral property, has been a powerful tool for integral cryptanalysis of many block ciphers. Exploiting a Mixed Integral Linear Programming (MILP) optimizer, an alternative approach to searching integral distinguishers was proposed, which has overcome the bottleneck of the cipher block length. The MILP-aided method starts by modeling CBDP propagation by a system of linear inequalities. Then by choosing an appropriate objective function, the problem of searching distinguisher transforms into an MILP problem. As an application of this technique, we focused on a newly proposed lightweight block cipher SAND. SAND is a family of two AND-RX block ciphers SAND-64 and SAND-128, which was designed to overcome the difficulty regarding security evaluation. For SAND-64, we found a 12-round distinguisher with 23 balanced bits and a data complexity of 2^{63}, with the superiority of a higher number of balanced bits than the designers' one. Furthermore, we applied an integral attack on a 15 and 16-round SAND-64, including the key recovery step which resulted in time complexity of 2^{105} and $2^{109.91}$ and memory complexity of 2^{52} and 2^{85} bytes, respectively.**

***Index Terms*—Division property, integral distinguisher, MILP, SAND block cipher.**

I. Introduction

So far, block ciphers that do not use S-boxes have been the best candidates in lightweight environments because of simple implementation using logic gates instead of lookup tables. According to various requirements, many lightweight block ciphers have been proposed. Midori [1], PRESENT [2], CRAFT [3], CLEFIA [4], SKINNY [5], and GIFT [6] are some instances among this great family of block ciphers. Additionally, two lightweight block ciphers SIMON and SPECK, which are AND-RX and ARX respectively, were proposed by NSA [7]. A newly proposed SIMON-based block cipher family is SAND [8]. Its designers claimed that SAND supports S-box-based security evaluation approaches due to admitting an equivalent S-box-based representation, unlike SIMON. They also showed that SAND has better efficiency and stronger security bounds even in the related-key setting, in comparison to similar AND-RX block ciphers.

Integral cryptanalysis, introduced by Knudsen and Wagner [9], has been powerful cryptanalysis against many block ciphers. In Eurocrypt 2015, a new technique to find an integral distinguisher was proposed by Todo [10] by defining "division property", which is a generalization of integral property, and was applied to full round MISTY1 for the first time [11]. A year later, Todo et al. [12] introduced bit-based division property to analyze each bit independently to gain higher precision. Conventional Bit-based Division Property (CBDP) and Bit-based Division Property using Three subsets (BDPT) were two kinds of division property defined in [12]. This technique was applied to SIMON32 and a 14-round distinguisher was found as a result. However, this method is impractical for block ciphers with block length (represented by n) greater than 32 bits due to the 2^n complexity requirement. Xiang et al. [13] decreased this complexity significantly by presenting an automated search utilizing a Mixed Integral Linear Programming (MILP) optimizer which formerly had been applied to search for differential and linear characteristics [14]. Xiang defined a new notation called "division trail" as an illustration of CBDP propagation. Furthermore, he showed how to describe CBDP propagation through a cipher using a system of linear inequalities, as the MILP constraints. Also, he proposed stopping rules, and thus transformed the search for the CBDP into an MILP problem. Afterward, many innovations have risen, from MILP-aided methods for primitives with non-bit-permutation linear layers [15]–[17] to MILP-aided method of searching BDPT [18], [19].

In spite of proposed improvements in MILP-aided searching methods, Xiang's method is among the most powerful techniques for AND-RX block ciphers like SAND, since these ciphers have no S-box or complex linear layer. As a result, in this paper we exploited his approach and found a 12-round integral distinguisher with 23 balanced bits for SAND-64, which is the best integral distinguisher found so far, to the best of our knowledge, subject to the number of rounds and balanced bits. Utilizing this characteristic, we proposed an integral cryptanalysis of 15 and 16-round SAND-64 including the key recovery step. The results are illustrated in Table I.

This paper is organized as follows: Section II represents SAND specifications briefly. Section III provides the background on division property, propagation rules and the MILP-aided method. We propose the 12-round SAND-64 distinguisher in section IV. In section V, the detailed procedure of

TABLE I: Integral cryptanalysis results of SAND-64

#Attack rounds	#Distinguisher rounds	#Balanced bits	Data complexity	Time complexity	Memory complexity (Bytes)	Reference
-	12	1	2^{63} CP	-	-	[8]
15	12	23	2^{63} CP	2^{105}	2^{52}	This paper
16	12	23	2^{63} CP	$2^{109.91}$	2^{85}	This paper

reduced-round attack has been explained. Eventually, section VI concludes the paper.

II. A Brief Description of SAND-64

SAND [8] is a family of two Feistel block ciphers SAND-64 and SAND-128 with the parameters in Table II.

TABLE II: Parameters of SAND

Cipher	**Block size $2n$**	**Branch size n**	**Key size m**	**Rounds R**
SAND-64	64	32	128	48
SAND-64	128	64	128	54

The round function (F) is illustrated in Figure 1. For the input state $(x^r, y^r) = (x_0^r, ..., x_{n-1}^r, y_0^r, ..., y_{n-1}^r)$ and the subkey $sk^r = (sk_0^r, ..., sk_{n-1}^r)$ of the r-th round (for $0 \le r < R$), the output state (x^{r+1}, y^{r+1}) is computed with

$$\begin{aligned}&(x^{r+1}, y^{r+1}) = F_{sk^r}(x^r, y^r)\\ &= \left(P_n(G_0(x^r \lll_{\frac{n}{4}} \alpha) \oplus G_1(x^r \lll_{\frac{n}{4}} \beta)) \oplus y^r \oplus sk^r, x^r\right).\end{aligned} \tag{1}$$

More details of SAND-64 are provided as follows (for SAND-64 $n = 32$):

- Shift boxes ($x^r \lll_{\frac{n}{4}} \alpha$ and $x^r \lll_{\frac{n}{4}} \beta$): The n-bit input variable x^r can be assumed as a concatenation of four $\frac{n}{4}$-bit words $x^r = (x_0^r, ..., x_{n-1}^r) = x^r\{0\}||x^r\{1\}||x^r\{2\}||x^r\{3\}$ and each word $x^r\{i\}$ is rotated by α (or β) to the left within the word circularly, i.e. $x^r \lll_{\frac{n}{4}} \alpha = (x^r\{0\} \lll \alpha)||(x^r\{1\} \lll \alpha)||(x^r\{2\} \lll \alpha)||(x^r\{3\} \lll \alpha)$. The tuple of rotation constants (α, β) is fixed to $(0, 1)$.
- Non-linear functions (G_0 and G_1): Let the n-bit variable x be the input value of G_0 and G_1 which can be assumed as a concatenation of four $\frac{n}{4}$-bit words $= x\{0\}||x\{1\}||x\{2\}||x\{3\}$. Let $y = y\{0\}||y\{1\}||y\{2\}||y\{3\}$ represents the output value. The output of G_0 and G_1 is calculated as (the symbols $\oplus$ and $\odot$ denote bitwise-XOR and bitwise-AND, respectively):

$$G_0 : \begin{cases} y\{0\} = y\{3\} \odot x\{2\} \oplus x\{0\} \\ y\{1\} = x\{1\} \\ y\{2\} = x\{2\} \\ y\{3\} = x\{0\} \odot x\{1\} \oplus x\{3\} \end{cases} \tag{2}$$

$$G_1 : \begin{cases} y\{0\} = x\{0\} \\ y\{1\} = x\{0\} \odot x\{2\} \oplus x\{1\} \\ y\{2\} = x\{3\} \odot y\{1\} \oplus x\{2\} \\ y\{3\} = x\{3\} \end{cases}. \tag{3}$$

- Bit permutation (P_n): Let the n-bit variables x and y be the input and output values. Then the n-bit permutation P_n is a parallel application of a $\frac{n}{4}$-bit permutation $p_{\frac{n}{4}}$ on four $\frac{n}{4}$-bit words. In other words

$$y_{\frac{n}{4}.i + p_{\frac{n}{4}}(j)} = x_{\frac{n}{4}.i+j}, \text{ for } 0 \le j < \frac{n}{4} \text{ and } 0 \le i < 4, \tag{4}$$

where $p_{\frac{n}{4}}$ is given in Table III.

TABLE III: p_8 for SAND-64

j	0	1	2	3	4	5	6	7
$p_8(j)$	5	2	7	4	1	6	3	0

- Key Schedule: The 128-bit master key is viewed as four 32-bit words $K = K^3||K^2||K^1||K^0$, which are the initial states of a LFSR. The r-th round subkey sk^r ($0 \le r < R$) is loaded from r-th state of the LFSR, i.e. K^r. As a result, discovery of four consecutive subkeys (i.e. sk^r, sk^{r+1}, sk^{r+2} and sk^{r+3}) reveals the master key uniquely.

Remark. Variable indexing utilized for SAND-64 in this paper is rather different from [8]. Thus, some differences in equations describing SAND-64 are noticeable. For more details about SAND-64, see [8].

III. Division Property

A. Notations

Let $\mathbb{F}_2$ denotes the binary finite field and $\mathbf{a} = (a_0, a_1, \cdots a_{n-1}) \in \mathbb{F}_2^n$ be an n-bit vector. For an n-bit vectors $\mathbf{x}$ and $\mathbf{u}$, we define $\mathbf{x}^{\mathbf{u}} = \Pi_{i=0}^{n-1} x_i^{u_i}$. Also, for any $\mathbf{k}, \mathbf{k'} \in \mathbb{F}_2^n$, $\mathbf{k} \ge \mathbf{k'}$ means $k_i \ge k'_i$ for all $i = 0, 1, \cdots, n-1$.

B. Bit-Based Division Property: Definition and Propagation Rules

In the following, we will define conventional bit-based division property (CBDP) and present its propagation rules.

Definition 1 (CBDP [11]): Let $\mathbb{X}$ be a multiset whose elements take a value of $\mathbb{F}_2^n$. When the multiset $\mathbb{X}$ has the CBDP $\mathcal{D}_{\mathbb{K}}^{1^n}$, where $\mathbb{K}$ denotes a set of n-dimensional vectors whose i-th element takes a value between 0 and 1, it fulfills the condition

$$\bigoplus_{\mathbf{x} \in \mathbb{X}} \mathbf{x}^{\mathbf{u}} = \begin{cases} \text{unknown}, & \text{if there exists } \mathbf{k} \in \mathbb{K} \text{ where } \mathbf{u} \ge \mathbf{k} \\ 0, & \text{o.w.} \end{cases}. \tag{5}$$

Propagation Rule 1 (Copy [12]): Let $\mathbf{x}$ and $\mathbf{y}$ denote the input and output of the Copy function, i.e. $\mathbf{x} = (x_0, x_1, ..., x_{n-1}) \in \mathbb{F}_2^n$ and $\mathbf{y} = (x_0, x_0, x_1, ..., x_{n-1}) \in \mathbb{F}_2^{n+1}$.

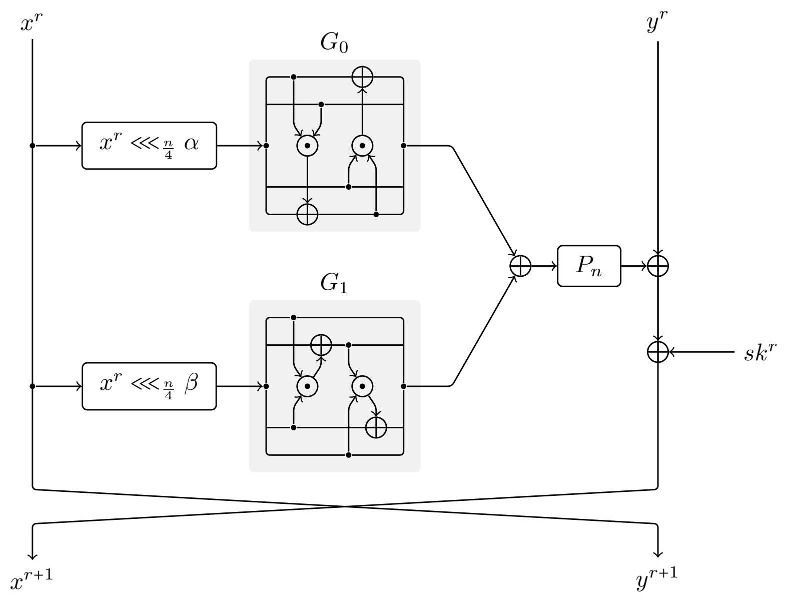
x^r
y^r
G_0
G_1
$x^r \lll_{\frac{n}{4}} \alpha$
$x^r \lll_{\frac{n}{4}} \beta$
P_n
sk^r
x^{r+1}
y^{r+1}

Fig. 1: SAND round function [8]

If the input multiset $\mathbb{X}$ has $\mathcal{D}^{1^n}_{\mathbb{K}}$, then the output multiset $\mathbb{Y}$ has $\mathcal{D}^{1^{n+1}}_{\mathbb{K}'}$ where

$$\mathbb{K}' \leftarrow \begin{cases} (0,0,k_1,...,k_{n-1}), & k_0 = 0 \\ (1,0,k_1,...,k_{n-1}),(0,1,k_1,...,k_{n-1}), & k_0 = 1 \end{cases}. \tag{6}$$

Propagation Rule 2 (And [12]): Let $\mathbf{x}$ and $\mathbf{y}$ denote the input and output of the And function, i.e. $\mathbf{x} = (x_0, x_1, ..., x_{n-1}) \in \mathbb{F}_2^n$ and $\mathbf{y} = (x_0 \wedge x_1, x_2, ..., x_{n-1}) \in \mathbb{F}_2^{n-1}$. If the input multiset $\mathbb{X}$ has $\mathcal{D}^{1^n}_{\mathbb{K}}$, then the output multiset $\mathbb{Y}$ has $\mathcal{D}^{1^{n-1}}_{\mathbb{K}'}$ where

$$\mathbb{K}' \leftarrow \left(\lceil \frac{k_0 + k_1}{2} \rceil, k_2, ..., k_{n-1} \right), \text{ for all } \mathbf{k} \in \mathbb{K}. \tag{7}$$

Propagation Rule 3 (XOR [12]): Let $\mathbf{x}$ and $\mathbf{y}$ denote the input and output of the XOR function, i.e. $\mathbf{x} = (x_0, x_1, ..., x_{n-1}) \in \mathbb{F}_2^n$ and $\mathbf{y} = (x_0 \oplus x_1, x_2, ..., x_{n-1}) \in \mathbb{F}_2^{n-1}$. If the input multiset $\mathbb{X}$ has $\mathcal{D}^{1^n}_{\mathbb{K}}$, then the output multiset $\mathbb{Y}$ has $\mathcal{D}^{1^{n-1}}_{\mathbb{K}'}$ where

$$\mathbb{K}' \leftarrow (k_0 + k_1, k_2, ..., k_{n-1}), \quad \text{for all } \mathbf{k} \in \mathbb{K} \text{ where } (k_0, k_1) \neq (1,1). \tag{8}$$

Please refer to [11] for proofs and further details.

C. The MILP-Aided Method

In spite of the bit-based division property's high precision in finding integral distinguisher, the exponential complexity of using this tool makes this method almost impossible for block ciphers with block length greater than 32 bits. Xiang et al. [13] decreased this complexity significantly by utilizing an MILP optimizer. As the first step of Xiang's method, we describe an MILP problem as follows.

Definition 2 (MILP problem [15]): Find a vector $\mathbf{x} \in \mathbb{F}_2^n$ subject to $\mathbf{A}\mathbf{x} \leq \mathbf{b}$, so that the objective linear function

$$c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

is minimized, where $(c_1, c_2, \cdots, c_n) \in \mathbb{R}^n$, $\mathbf{b} \in \mathbb{R}^m$ and $\mathbf{A} \in \mathbb{R}^{m \times n}$.

Now we define division trail:

Definition 3 (division trail [13]): Let $\{\mathbf{k}\} := \mathbb{K}_0 \to \mathbb{K}_1 \to \cdots \to \mathbb{K}_r$ be a chain of CBDP propagation through r rounds of encryption. For any vector $\mathbf{k}_i^* \in \mathbb{K}_i$ $(i \geq 1)$, there must exist a vector $\mathbf{k}_{i-1}^* \in \mathbb{K}_{i-1}$ such that $\mathbf{k}_{i-1}^*$ can propagate to $\mathbf{k}_i^*$ according to the propagation rules. Additionally, for $(\mathbf{k}_0, \mathbf{k}_0, \cdots, \mathbf{k}_r) \in \mathbb{K}_0 \times \mathbb{K}_1 \times \cdots \times \mathbb{K}_r$, if $\mathbf{k}_{i-1}^*$ can propagate to $\mathbf{k}_i^*$ for all $i \in \{1, 2, \cdots, r\}$, we call $(\mathbf{k}_0, \mathbf{k}_0, \cdots, \mathbf{k}_r)$ an r-round division trail.

In the next step, propagation rules of CBDP are modeled with a system of linear inequalities:

Model 1 (Copy [13]): Let $(a) \xrightarrow{\text{Copy}} (b_0, b_1)$ be the Copy function division trail. Then, the inequalities

$$\begin{cases} a - b_0 - b_1 = 0 \\ a,\ b_0,\ b_1 \text{ are binaries} \end{cases} \tag{9}$$

are sufficient to describe the propagation through Copy function.

Model 2 (And [13]): Let $(a_0, a_1) \xrightarrow{\text{And}} (b)$ be the And function division trail. Then, the inequalities

$$\begin{cases} b - a_0 \geq 0 \\ b - a_1 \geq 0 \\ b - a_0 - a_1 \leq 0 \\ a_0,\ a_1,\ b \text{ are binaries} \end{cases} \tag{10}$$

are sufficient to describe the propagation through And function.

Model 3 (XOR [13]): Let $(a_0, a_1) \xrightarrow{\text{XOR}} (b)$ be the XOR function division trail. Then, the inequalities

$$\begin{cases} a_0 + a_1 - b = 0 \\ a_0,\ a_1,\ b \text{ are binaries} \end{cases} \tag{11}$$

are sufficient to describe the propagation through XOR function.

So far, constructing a system of linear inequalities describing one round propagation has been explained. By iterating this procedure r times, a system of inequalities for r round propagation, named $\mathcal{L}$, is gained. Let's represent the r round division trail by $(a_0^0, a_1^0, \cdots, a_{n-1}^0) \to \cdots \to (a_0^r, a_1^r, \cdots, a_{n-1}^r)$. Apparently, $\mathcal{L}$ is defined on variables a_i^j ($i = 0, 1, \cdots n-1$, $j = 0, 1, \cdots, r$) and some auxiliary variables. Also, the constrains corresponding to the initial division property should be added to $\mathcal{L}$, i.e. $a_i^0 = k_i$ ($i = 0, 1, \cdots, n-1$) where $\mathcal{D}_{\{\mathbf{k}\}}^{1^n}$ is the input division property with $\mathbf{k} = (k_0, k_1, \cdots, k_{n-1})$.

If the set $\mathbb{K}_r$ doesn't have the unit vector $\mathbf{e}_m \in \mathbb{F}_2^n$, whose elements are 0 except m-th element which is 1, the m-th output bit of r-round ciphertexts is balanced (i.e. has the zero-sum property). As a result, the objective function of the MILP model is set as

$$Obj : Min\{a_0^r + a_1^r + \cdots a_{n-1}^r\}.$$

If $\mathbb{K}_{r+1}$ contains all the n-bit unit vectors for the first time, the division property propagation should stop, and an r-round distinguisher can be derived.

IV. Integral Characteristic for 12 Rounds of SAND-64

We apply the MILP-aided method to SAND-64 as follows: Denote one round division trail of SAND-64 by $(a_0^i, ..., a_{n-1}^i, b_0^i, ..., b_{n-1}^i) \to (a_0^{i+1}, ..., a_{n-1}^{i+1}, b_0^{i+1}, ..., b_{n-1}^{i+1})$. In order to get the inequalities representing all possible division trails of one round SAND-64, we define some auxiliary vectors. These vectors denote the division properties of interior values. As an example, we denote $(t_0^i, ..., t_{n-1}^i)$ and $(u_0^i, ..., u_{n-1}^i)$ the input division property of the first and second shift box. According to model 1, the following inequalities are sufficient to model the Copy operation:

$$\mathcal{L}_1 : a_j^i - t_j^i - u_j^i - b_j^{i+1} = 0, \qquad \text{for } j \in \{0, 1, ..., n-1\}. \tag{12}$$

Now, consider the XOR just after the permutation box; We represent its inputs and output division properties by $(p_0^i, ..., p_{n-1}^i)$, $(y_0^i, ..., y_{n-1}^i)$ and $(q_0^i, ..., q_{n-1}^i)$, respectively. We model the XOR operation by the following inequalities according to model 3:

$$\mathcal{L}_2 : p_j^i + y_j^i - q_j^i = 0, \qquad \text{for } j \in \{0, 1, ..., n-1\}. \tag{13}$$

At last, we denote $(f_0^i, ..., f_{n-1}^i)$, $(g_0^i, ..., g_{n-1}^i)$ and $(h_0^i, ..., h_{n-1}^i)$ the inputs and output division properties of the left AND in G_0 illustration, respectively. The following inequalities represent the AND operation model according to model 2:

$$\mathcal{L}_3 : \begin{cases} h_j^i - f_j^i >= 0, & \text{for } j \in \{0, 1, ..., \frac{n}{4} - 1\} \\ h_j^i - g_j^i >= 0, & \text{for } j \in \{0, 1, ..., \frac{n}{4} - 1\} \\ h_j^i - f_j^i - g_j^i <= 0, & \text{for } j \in \{0, 1, ..., \frac{n}{4} - 1\} \end{cases}. \tag{14}$$

Other And, Copy and XOR operations of SAND-64 round function are modeled similarly. Additionally, since we consider bit-based division property, division property propagation through circular shift or permutation is just a circular shift or permutation of the coordinates of the vector. This results in a system of constraints, containing 288 inequalities for each round of SAND-64.

So, 1-round SAND-64 division trails has been modeled completely. For r-round model, this process should be used r times. Furthermore, the initial condition (input division property) should be added to system of inequalities. By applying Gurobi MILP solver [20], a 12-round distinguisher with 23 balanced bits is found which starts with division property $(1_{31}0_1, 1_{32})$ (i.e. all bits are active except x_{31}^0) at the plaintext. It is remarkable that total number of constrains given to the solver equals to $12 \times 288 + 64 = 3520$ inequalities. The balanced bits are

$$\{y_i^{12} : i \in \{2, 3, 4, 6, 7, 9, 10, 11, 13, 14, 15, 17, 18, 19, \\ 21, 22, 23, 24, 26, 27, 28, 30, 31\}\}.$$

We will use this distinguisher for the 15 and 16 rounds attack. So the data complexity would be 2^{63} chosen plaintexts, which all of the bits take all possible values except x_{31}^0.

V. Integral Cryptanalysis of Reduced-Round SAND-64

In this section, we propose an integral attack on 15 and 16-round SAND-64. The attack utilizes the 12-round characteristic described in section IV from round 1 to 12. To the best of our knowledge, this is the first attack on reduced-round SAND-64 ever presented.

In the partial decryption step, the subkey sk^{12} has no effect on whether y_i^{12} is balanced or not; so, it isn't involved in attack procedure and should be guessed individually.

In Table IV, required bits of subkeys for calculating each balanced y_i^{12} (or $P_n^{-1}(y_i^{12})$ equivalently) from the ciphertext, is represented. For example, according to row number 1, to calculate the bit $P_n(v_1^{12}) = y_2^{12}$ we require 5 bits of sk^{13}, 13 bits of sk^{14} and 25 bits of sk^{15} ($v^i = (v_0^i, ..., v_{31}^i)$ denotes the input to the permutation box in i-th round). Apparently, the columns corresponding to sk_i^{13} and sk_i^{14} are used for 15-round attack and sk_i^{13}, sk_i^{14} and sk_i^{15} for 16-round attack.

A. 15 Rounds Attack

The procedure of the key recovery step for 15-round SAND-64 is summarized in Table V. We explain some steps in detail and the rest is similar:

Step 1. According to row number 12 of Table IV, calculating the bit $P_n(v_{16}^{12}) = y_{21}^{12}$ requires 15 bits of sk^{13} and sk^{14} in total ($\{sk_i^{13} : i \in \{16, 1, 9, 17\}\}$ and $\{sk_i^{14} : i \in \{0, 4, 5, 8, 12, 13, 16, 20, 21, 23, 28\}\}$). So partial decryption of 2^{63} ciphertexts with 2^{15} different keys, results in a time complexity of $2^{63+15} \times$ 3-round En.

So far, 15 key bits are involved in the attack. Checking whether the bit $P_n(v_{16}^{12}) = y_{21}^{12}$ is balanced for each key guess, provides a filter of 2^{-1} on the space of 2^{15} keys, on average. So, 2^{14}

TABLE IV: Required bits of subkeys for each balanced bit

Row number	$P_n^{-1}(y_i^{12})$	$\{sk_i^{13}\}$	$\#\{sk_i^{13}\}$ (bit)	$\{sk_i^{14}\}$	$\#\{sk_i^{14}\}$ (bit)	$\{sk_i^{15}\}$	$\#\{sk_i^{15}\}$ (bit)
1	1	$\{1,2,9,17,25\}$	5	$\{1,2,4,5,9,12,13, 17,20,21,25,27,28\}$	13	$\{0,1,2,3,4,5,6,8,9,11, 12,13,14,16,17,19,20,21, 24,25,26,27,28,29,30\}$	25
2	2	$\{2,3,10,18,26\}$	5	$\{1,2,6,7,9,10,14,17, 18,22,24,25,30\}$	13	$\{1,2,3,4,5,6,7,9,10,12,13, 14,15,17,18,20,21,22,25, 26,27,28,29,30,31\}$	25
3	3	$\{3,4,11,19,27\}$	5	$\{3,4,6,7,11,14,15,19, 22,23,27,29,30\}$	13	$\{0,2,3,4,5,6,7,8,10,11,13, 14,15,18,19,21,22,23,24\ 26,27,28,29,30,31\}$	25
4	5	$\{5,6,13,21,29\}$	5	$\{0,1,5,6,8,9,13,16, 17,21,24,29,31\}$	13	$\{0,1,2,4,5,6,7,8,9,10,12, 13,15,16,17,20,21,23, 24,25,26,28,29,30,31\}$	25
5	6	$\{6,7,14,22,30\}$	5	$\{2,3,5,6,10,13,14, 18,21,22,26,28,29\}$	13	$\{0,1,2,3,5,6,7,8,9,10,11, 13,14,16,17,18,21,22, 24,25,26,27,29,30,31\}$	25
6	8	$\{1,8,9,17,24\}$	5	$\{0,4,5,7,8,12,13,15, 16,20,21,28,30,31\}$	14	$\{0,1,2,3,4,5,7,8,9,10,11, 12,13,15,16,17,18,19,20, 23,24,25,26,27,28,29,31\}$	27
7	9	$\{2,9,10,18,25\}$	5	$\{1,2,4,5,9,10,12,13, 17,18,21,25,27,28\}$	14	$\{0,1,2,3,4,5,6,8,9,10,11, 12,13,14,16,17,18,19,20, 21,24,25,26,27,28,29,30\}$	27
8	10	$\{3,10,11,19,26\}$	5	$\{1,2,6,7,9,10,14,15, 18,22,23,24,25,30\}$	14	$\{1,2,3,4,5,6,7,9,10,11,12, 13,14,15,17,18,19,20,21, 22,25,26,27,28,29,30,31\}$	27
9	12	$\{5,12,13,21,28\}$	5	$\{0,1,3,4,8,9,11,12, 16,17,20,24,26,27\}$	14	$\{0,1,3,4,5,6,7,8,9,11,12, 13,14,15,16,19,20,21,22, 23,24,25,27,28,29,30,31\}$	27
10	13	$\{6,13,14,22,29\}$	5	$\{0,1,5,6,8,9,13,14, 17,21,22,24,29,31\}$	14	$\{0,1,2,4,5,6,7,8,9,10,12, 13,14,15,16,17,20,21,22, 23,24,25,26,28,29,30,31\}$	27
11	14	$\{7,14,15,23,30\}$	5	$\{2,3,5,6,10,11,13,14, 18,19,22,26,28,29\}$	14	$\{0,1,2,3,5,6,7,8,9,10,11, 13,14,15,16,17,18,21,22, 23,24,25,26,27,29,30,31\}$	27
12	16	$\{16,1,9,17\}$	4	$\{0,4,5,8,12,13, 16,20,21,23,28\}$	11	$\{0,1,3,4,7,8,9,11,12,15,16, 17,18,19,20,23,24,26,27,31\}$	20
13	17	$\{18,17,2,10\}$	4	$\{1,2,5,9,10,13, 17,18,20,21,25\}$	11	$\{0,1,2,4,5,8,9,10,12,13,16, 17,18,19,20,21,24,25,27,28\}$	20
14	18	$\{11,19,18,3\}$	4	$\{2,6,7,10,14,15, 17,18,22,23,30\}$	11	$\{1,2,3,5,6,9,10,11,13,14,17, 18,19,20,21,22,25,26,28,29\}$	20
15	20	$\{13,21,20,5\}$	4	$\{0,1,4,8,9,12,16, 17,19,20,24\}$	11	$\{0,3,4,5,7,8,11,12,13,15,16, 19,20,21,22,23,27,28,30,31\}$	20
16	21	$\{22,21,6,14\}$	4	$\{1,5,6,9,13,14,16, 17,21,22,29\}$	11	$\{0,1,4,5,6,8,9,12,13,14,16, 17,20,21,22,23,24,28,29,31\}$	20
17	22	$\{15,23,22,7\}$	4	$\{2,3,6,10,11,14,18, 19,21,22,26\}$	11	$\{1,2,5,6,7,9,10,13,14,15,16, 17,18,21,22,23,24,25,29,30\}$	20
18	25	$\{24,1,9,25\}$	4	$\{4,5,7,12,13,15,20, 21,25,27,28,30,31\}$	13	$\{0,1,2,3,4,5,6,8,9,10,11, 12,13,14,16,17,18,19,20, 24,25,26,27,28,29,30,31\}$	27
19	26	$\{26,25,2,10\}$	4	$\{1,2,4,9,10,12,17, 18,24,25,26,27,28\}$	13	$\{1,2,3,4,5,6,7,9,10,11,12, 13,14,15,17,18,19,20,21, 25,26,27,28,29,30,31\}$	26
20	27	$\{11,27,26,3\}$	4	$\{1,6,7,9,14,15,22, 23,24,25,27,29,30\}$	13	$\{0,2,3,4,5,6,7,8,10,11,12, 13,14,15,18,19,20,21,22, 24,25,26,27,28,29,30,31\}$	27
21	29	$\{13,29,28,5\}$	4	$\{0,1,3,8,9,11,16,17, 24,26,27,29,31\}$	13	$\{0,1,2,4,5,6,7,8,9,10,12, 13,14,15,16,20,21,22,23, 24,25,26,27,28,29,30,31\}$	27
22	30	$\{30,29,6,14\}$	4	$\{0,5,6,8,13,14,21, 22,24,28,29,30,31\}$	13	$\{0,1,2,3,5,6,7,8,9,10,11, 13,14,15,16,17,21,22,23, 24,25,26,27,29,30,31\}$	26
23	31	$\{15,31,30,7\}$	4	$\{2,3,5,10,11,13,18, 19,25,26,28,29,31\}$	13	$\{0,1,2,3,4,6,7,8,9,10,11, 12,14,15,16,17,18,22,23, 24,25,26,27,28,29,30,31\}$	27

keys remain after filtering out the wrong ones.

Step 2. According to row number 13 of Table IV, we need 15 key bits to calculate the bit $P_n(v_{17}^{12}) = y_{18}^{12}$ ($\{sk_i^{13} : i \in \{18,17,2,10\}\}$ and $\{sk_i^{14} : i \in \{1,2,5,9,10,13,17,18,20,21,25\}\}$). So, the time complexity of partial decryption would be $2^{63+15} \times$ 3-round En. Checking whether the bit $P_n(v^{12}_{17}) = y^{12}_{18}$ is balanced for each of 2^{15} key guesses, provides a filter of 2^{-1} on the space of 2^{15} keys, on average and 2^{14} keys remain. So far, 25 key bits are involved in attack (5 key bits of step 2 were already involved in previous step i.e. sk^{13}_{17} and $\{sk^{14}_i : i \in \{5, 13, 20, 21\}\}$). Now remaining keys from steps 1 and 2 should be merged properly; 5 key bits sk^{13}_{17} and $\{sk^{14}_i : i \in \{5, 13, 20, 21\}\}$ are the same which results in 2^5 different keys. Then multiplied by 2^9 and 2^9 endured by steps 1 and 2, respectively. So, this procedure takes a time complexity of 2^{23} (i.e. merging complexity of step 2 equals to 2^{23}) and the set of remaining keys contains 2^{23} different keys.

Step 3. We need 15 key bits $\{sk^{13}_i : i \in \{11, 19, 18, 3\}\}$ and $\{sk^{14}_i : i \in \{2, 6, 7, 10, 14, 15, 17, 18, 22, 23, 30\}\}$ to calculate the bit $P_n(v^{12}_{18}) = y^{12}_{23}$ and the time complexity of partial decryption would be $2^{63+15} \times$ 3-round En. By a filter of 2^{-1} on the space of 2^{15} keys, 2^{14} keys remain. So far, 34 key bits are involved in the attack (6 key bits of step 3 were already involved in previous steps i.e sk^{13}_{18} and $\{sk^{14}_i : i \in \{2, 10, 17, 18, 23\}\}$). Merging 2^{14} remaining keys of step 3 and 2^{23} keys of previous steps needs a time complexity of 2^{31} (2^6 different keys from 6 identical key bits i.e. i.e sk^{13}_{18} and $\{sk^{14}_i : i \in \{2, 10, 17, 18, 23\}\}$, multiplied by 2^8 and 2^{17} because of step 3 and the previous ones, respectively). Finally, 2^{31} keys remain.

Attack complexity: The partial decryption time complexity of each step is denoted in Table V. The summation equals to

$$T_1 \approx 2^{63+22.375} \times \frac{3}{15} \approx 2^{83.053}.$$

Also, the total time complexity of merging the remaining keys would be $T_2 \approx 2^{50.558}$, which is calculated by adding up the merging complexities of each step.

As mentioned in section II, the discovery of four consecutive subkeys (i.e. sk^r, sk^{r+1}, sk^{r+2} and sk^{r+3}), reveals the master key uniquely. The key recovery step guessed 63 key bits (sk^{14} and $\{sk^{13}_i : i \in \{1, 2, ..., 31\}\}$) and 65 key bits remain from the master key which should be guessed. Additionally, 2^{40} keys remained from the key recovery step. Therefore, the total number of master key candidates would be $2^{40+65} = 2^{105}$, where brute-force takes a time complexity of $T_3 = 2^{105}$. Eventually, the total time complexity becomes

$$T = T_1 + T_2 + T_3 \approx 2^{105}.$$

The data complexity, as mentioned in section IV, is 2^{63} chosen plaintexts. Also, the memory complexity is dominated by steps 10, 11, and 12 which needs 2^{52} bytes.

B. 16 Rounds Attack

Since the procedure of the key recovery step for 16-round SAND-64 is similar to the 15-round one, we only represent it in Table VI and discuss attack complexity.

Attack complexity: The partial decryption time complexity of each step is denoted in Table VI. The summation equals to

$$T_1 \approx 2^{63+48.865} \times \frac{4}{16} \approx 2^{109.865}.$$

TABLE V: Key recovery of 15-round attack

Step	$P_n^{-1}(y_i^{12})$	#Key bits required	#Total guessed key bits	Time complexity	#Total keys after filter
1	16	15	15	2^{63+15}	2^{14}
2	17	15	25	2^{63+15}	2^{23}
3	18	15	34	2^{63+15}	2^{31}
4	20	15	40	2^{63+15}	2^{36}
5	21	15	44	2^{63+15}	2^{39}
6	22	15	50	2^{63+15}	2^{44}
7	25	17	54	2^{63+17}	2^{47}
8	26	17	55	2^{63+17}	2^{47}
9	27	17	56	2^{63+17}	2^{47}
10	29	17	58	2^{63+17}	2^{48}
11	30	17	59	2^{63+17}	2^{48}
12	31	17	60	2^{63+17}	2^{48}
13	1	18	60	2^{63+18}	2^{47}
14	2	18	60	2^{63+18}	2^{46}
15	3	18	61	2^{63+18}	2^{46}
16	5	18	61	2^{63+18}	2^{45}
17	6	18	61	2^{63+18}	2^{44}
18	8	19	62	2^{63+19}	2^{44}
19	9	19	62	2^{63+19}	2^{43}
20	10	19	62	2^{63+19}	2^{42}
21	12	19	63	2^{63+19}	2^{42}
22	13	19	63	2^{63+19}	2^{41}
23	14	19	63	2^{63+19}	2^{40}

Also, the total time complexity of merging the remaining keys would be $T_2 \approx 2^{84.219}$, which is calculated by adding up the merging complexities of each step.

The key recovery step guessed 95 key bits (sk^{15}, sk^{14} and $\{sk^{13}_i : i \in \{1, 2, ..., 31\}\}$) and 33 key bits remain from the master key which should be guessed. Additionally, 2^{72} keys remained from the key recovery step. Therefore, the total number of master key candidates would be $2^{72+33} = 2^{105}$, where brute-force takes a time complexity of $T_3 = 2^{105}$. Eventually, the total time complexity becomes

$$T = T_1 + T_2 + T_3 \approx 2^{109.91}.$$

The data complexity, as mentioned in section IV, is 2^{63} chosen plaintexts. Also, the memory complexity is dominated by step 6 which needs 2^{85} bytes.

VI. Conclusion

In this paper, we analyzed the security of SAND-64 against integral cryptanalysis based on conventional bit-based division property. SAND is a newly proposed family of two AND-RX block ciphers SAND-64 and SAND-128, which was designed to overcome the difficulty regarding security evaluation. The attack was applied using the MILP-aided method of searching integral distinguisher proposed by Xiang et al. [13]. In this technique, firstly, division property propagation of three basic operations (copy, bitwise AND, XOR) is modeled by linear inequalities. So, a linear inequality system that can accurately describe the division property propagation of a block cipher given an initial division property, can be constructed. Secondly, by choosing an appropriate objective function, the search algorithm is converted into an MILP problem and feasible solutions to this problem determine whether a useful distinguisher exists.

TABLE VI: Key recovery of 16-round attack

Step	$P_n^{-1}(y_i^{12})$	#Key bits required	#Total guessed key bits	Time complexity	#Total keys after filter
1	1	43	43	2^{63+43}	2^{42}
2	2	43	61	2^{63+43}	2^{59}
3	3	43	72	2^{63+43}	2^{69}
4	5	43	81	2^{63+43}	2^{77}
5	6	43	86	2^{63+43}	2^{81}
6	8	46	88	2^{63+46}	2^{82}
7	9	46	88	2^{63+46}	2^{81}
8	10	46	88	2^{63+46}	2^{80}
9	12	46	90	2^{63+46}	2^{81}
10	13	46	90	2^{63+46}	2^{80}
11	14	46	92	2^{63+46}	2^{81}
12	16	35	93	2^{63+35}	2^{81}
13	17	35	93	2^{63+35}	2^{80}
14	18	35	93	2^{63+35}	2^{79}
15	20	35	94	2^{63+35}	2^{79}
16	21	35	94	2^{63+35}	2^{78}
17	22	35	94	2^{63+35}	2^{77}
18	25	44	94	2^{63+44}	2^{76}
19	26	43	94	2^{63+43}	2^{75}
20	27	44	94	2^{63+44}	2^{74}
21	29	44	94	2^{63+44}	2^{73}
22	30	43	94	2^{63+43}	2^{72}
23	31	44	95	2^{63+44}	2^{72}

As a result, we presented a 12-round integral characteristic with 23 balanced bits and a data complexity of 2^{63}, which is better than what the designers claimed in the number of balanced bits (the distinguisher proposed by the designers covers 12 rounds with 1 balanced bit and a data complexity of 2^{63}). Furthermore, we provided a 15 and 16 round attack including key recovery step, resulting in time complexity of 2^{105} and $2^{109.91}$ and memory complexity of 2^{52} and 2^{85} bytes, respectively. The results are also summarized in Table I. Notably, the practical security of full-round SAND-64 is not dramatically gotten to risk by the proposed attack.

References

[1] S. Banik, A. Bogdanov, T. Isobe, K. Shibutani, H. Hiwatari, T. Akishita, and F. Regazzoni, "Midori: A block cipher for low energy," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2015, pp. 411–436.

[2] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. Robshaw, Y. Seurin, and C. Vikkelsoe, "Present: An ultra-lightweight block cipher," in *International workshop on cryptographic hardware and embedded systems*. Springer, 2007, pp. 450–466.

[3] C. Beierle, G. Leander, A. Moradi, and S. Rasoolzadeh, "Craft: lightweight tweakable block cipher with efficient protection against dfa attacks," *IACR Transactions on Symmetric Cryptology*, vol. 2019, no. 1, pp. 5–45, 2019.

[4] T. Shirai, K. Shibutani, T. Akishita, S. Moriai, and T. Iwata, "The 128-bit blockcipher clefia," in *International workshop on fast software encryption*. Springer, 2007, pp. 181–195.

[5] C. Beierle, J. Jean, S. Kölbl, G. Leander, A. Moradi, T. Peyrin, Y. Sasaki, P. Sasdrich, and S. M. Sim, "The skinny family of block ciphers and its low-latency variant mantis," in *Annual International Cryptology Conference*. Springer, 2016, pp. 123–153.

[6] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "Gift: a small present," in *International conference on cryptographic hardware and embedded systems*. Springer, 2017, pp. 321–345.

[7] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The simon and speck families of lightweight block ciphers," *cryptology eprint archive*, 2013.

[8] S. Chen, Y. Fan, L. Sun, Y. Fu, H. Zhou, Y. Li, M. Wang, W. Wang, and C. Guo, "Sand: an and-rx feistel lightweight block cipher supporting s-box-based security evaluations," *Designs, Codes and Cryptography*, vol. 90, no. 1, pp. 155–198, 2022.

[9] L. Knudsen and D. Wagner, "Integral cryptanalysis," in *International Workshop on Fast Software Encryption*. Springer, 2002, pp. 112–127.

[10] Y. Todo, "Structural evaluation by generalized integral property," in *Annual International Conference on the Theory and Applications of Cryptographic Techniques*. Springer, 2015, pp. 287–314.

[11] ——, "Integral cryptanalysis on full misty1," *Journal of Cryptology*, vol. 30, no. 3, pp. 920–959, 2017.

[12] Y. Todo and M. Morii, "Bit-based division property and application to simon family," in *International Conference on Fast Software Encryption*. Springer, 2016, pp. 357–377.

[13] Z. Xiang, W. Zhang, Z. Bao, and D. Lin, "Applying milp method to searching integral distinguishers based on division property for 6 lightweight block ciphers," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2016, pp. 648–678.

[14] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "Differential and linear cryptanalysis using mixed-integer linear programming," in *International Conference on Information Security and Cryptology*. Springer, 2011, pp. 57–76.

[15] W. Zhang and V. Rijmen, "Division cryptanalysis of block ciphers with a binary diffusion layer," *IET Information Security*, vol. 13, no. 2, pp. 87–95, 2019.

[16] L. Sun, W. Wang, and M. Q. Wang, "Milp-aided bit-based division property for primitives with non-bit-permutation linear layers," *IET Information Security*, vol. 14, no. 1, pp. 12–20, 2020.

[17] K. Hu, Q. Wang, and M. Wang, "Finding bit-based division property for ciphers with complex linear layers," *IACR Transactions on Symmetric Cryptology*, pp. 396–424, 2020.

[18] S. Wang, B. Hu, J. Guan, K. Zhang, and T. Shi, "Milp-aided method of searching division property using three subsets and applications," in *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, 2019, pp. 398–427.

[19] ——, "Exploring secret keys in searching integral distinguishers based on division property," *IACR Transactions on Symmetric Cryptology*, pp. 288–304, 2020.

[20] http://www.gurobi.com/.