Hindawi
Security and Communication Networks
Volume 2021, Article ID 5310545, 9 pages
https://doi.org/10.1155/2021/5310545

Research Article

Integral Distinguishers of the Full-Round Lightweight Block Cipher SAT_Jo

Xueying Qiu,[1] **Yongzhuang Wei**,[1,2] **Samir Hodzic**,[3] **and Enes Pasalic**[4]

[1]*Guilin University of Electronic Technology, Guilin 541004, China*
[2]*State Key Laboratory of Cryptology, P. O. Box 5159, Beijing 100878, China*
[3]*Technical University of Denmark, DTU Compute, Kongens Lyngby, Denmark*
[4]*University of Primorska, FAMNIT, Koper, Slovenia*

Correspondence should be addressed to Yongzhuang Wei; walker_wyz@guet.edu.cn

Received 10 May 2021; Revised 25 July 2021; Accepted 16 August 2021; Published 18 September 2021

Academic Editor: Chien Ming Chen

Integral cryptanalysis based on division property is a powerful cryptanalytic method whose range of successful applications was recently extended through the use of Mixed-Integer Linear Programming (MILP). Although this technique was demonstrated to be efficient in specifying distinguishers of reduced round versions of several families of lightweight block ciphers (such as SIMON, PRESENT, and few others), we show that this method provides distinguishers for a full-round block cipher SAT_Jo. SAT_Jo cipher is very similar to the well-known PRESENT block cipher, which has successfully withstood the known cryptanalytic methods. The main difference compared to PRESENT, which turns out to induce severe weaknesses of SAT_Jo algorithm, is its different choice of substitution boxes (S-boxes) and the bit-permutation layer for the reasons of making the cipher highly resource-efficient. Even though the designers provided a security analysis of this scheme against some major generic cryptanalytic methods, an application of the bit-division property in combination with MILP was not considered. By specifying integral distinguishers for the full-round SAT_Jo algorithm using this method, we essentially disapprove its use in intended applications. Using a 30-round distinguisher, we also describe a subkey recovery attack on the SAT_Jo algorithm whose time complexity is about 2^{66} encryptions (noting that SAT_Jo is designed to provide 80 bits of security). Moreover, it seems that the choice of bit-permutation induces weak division properties since replacing the original bit-permutation of SAT_Jo by the one used in PRESENT immediately renders integral distinguishers inefficient.

1. Introduction

Lightweight block ciphers play an important role in providing the security in various constrained environments (referring to different applications of Internet of Things). In recent years, many resource-efficient block ciphers have been proposed, such as MIDORI [1], PICCOLO [2], MIBS [3], PRIDE [4], PRESENT [5], and LBLOCK [6]. Recently, many new lightweight ciphers (candidates) in the second round of NIST's lightweight cryptography standardization process were also proposed [7]. However, because of restricted design rationales, certain lightweight designs sometimes fail to deliver a reasonable resistance to certain cryptanalytic methods. Although designers of new schemes provide a security analysis against the well-known attacks (e.g., integral attacks [8], differential attacks [9], and linear attacks [10]), it may happen that not all attacks are taken into consideration.

In this work, we consider a lightweight block cipher SAT_Jo [11] (proposed in 2018) and search for integral distinguishers based on division property using the MILP technique [12] introduced in [13]. Before describing the contribution of this work in more detail, we briefly summarize a development of integral attack and division property.

Namely, in 1997, Daemen et al. [14] proposed a square attack on block cipher SQUARE. In 2001, Lucks et al. [15] proposed a saturation attack on TWOFISH cipher, which

generalizes the square attack. Biryukov et al. [16] introduced a multiset attack on the SPN-based block ciphers. Then, in 2002, Knudsen et al. [8] proposed the so-called integral analysis, which generalizes the previous three attacks. In fact, from the point of view of Boolean functions, this attack is also closely related to higher-order differential attack proposed in [17]. Some further versions of this attack have been derived in 2008 by Z'aba et al. [18], who proposed the bit-pattern-based integral attack. It has been shown that one can derive integral distinguishers by analyzing the propagation of the integral property, where one tracks the positions of active, constant, and balanced bits. More specifically, the opponent selects a set of plaintexts having a portion of bits fixed at certain positions (called *constant bits*), whereas the remaining bits can take all possible values and are called *active*. Moreover, the XOR sum of their corresponding ciphertexts is computed (alternatively, a suitable subset is considered). Now, if the XOR sum at certain positions is always 0, regardless of the choice of secret key, such bits are called *balanced*. On the other hand, if the XOR sum changes at some positions (depending on the secret key value), such bits are commonly called *unknown*. This integral property can then be used to distinguish the real encryption algorithm form a random permutation.

A further generalization of integral attacks has been introduced by Todo [19] at EUROCRYPT 2015, by developing a cryptanalytic framework based on the so-called *division property*. Later, Todo and Morii [20] proposed *bit-based division property*, which was utilized for construction of a 15-round integral distinguisher for SIMON32 [21]. Finally, at ASIACRYPT 2016, Xiang et al. [13] proposed a method which combines the bit-based division property and searches for the division trails by employing the MILP method. Consequently, this combination successfully overcomes the main issue of the bit-based division property reflected in relatively high time and memory complexity which is bounded above by 2^n, where n is the block length. In what follows, we describe the contribution and structure of the subsequent sections.

Our contribution: in this paper, we analyze the lightweight block cipher SAT_Jo, which is built as a substitution-permutation (SP) network and processes plaintext blocks of length 64 bits through an iterative application of 31 identical rounds, using the secret key of size 80 bits. We emphasize that the designers of this algorithm provided the security evaluation [22] of the cipher by considering some main cryptanalytic tools such as differential and linear cryptanalysis, as well as the resistance against algebraic attacks. However, to the best of our knowledge, the robustness of this scheme with respect to integral attacks has not been evaluated so far.

We consider the three basic operations used in the SAT_Jo algorithm which then give rise to a set of linear inequalities that characterize the propagation of bit-based division property for SAT_Jo algorithm. Similar to the analysis performed in [13], by employing the open-source Gurobi MILP solver, an automated search for integral distinguishers is performed. Most notably, this MILP solver returns an integral distinguisher for the full-round SAT_Jo algorithm within a few seconds on a standard personal computer. Consequently, the bit-permutation of SAT_Jo algorithm (linear layer) appears not to be well designed and its increased efficiency turns out to be traded-off against lower security margins. Though our cryptanalysis does not substantially differ from the security evaluation in [13] (performed on SIMON, PRESENT, and a few more lightweight block ciphers), the results are quite dramatic due to the possibility of specifying integral distinguishers for a full-round block cipher which is not quite common. Moreover, we show that an efficient subkey recovery attack, whose time complexity corresponds to 2^{66} encryptions, can be easily mounted using our distinguisher.

Outline of the paper: Section 2 mainly introduces notations and definitions related to the division property. In Section 3, we discuss the MILP method and propagation rules of division property. In Section 4, an MILP model for SAT_Jo algorithm is derived, and its application is summarized in Section 5. In Section 6, the conclusion is given.

2. Preliminaries

By F_2^n, we denote the binary vector space of all n-tuples $\mathbf{x} = (x[0], \ldots, x[n-1]) \in F_2^n$, where $x[i]$ denotes the i-th coordinate of x. Throughout this work, the following definitions will be used.

Definition 1 (bit product function $\boldsymbol{\pi}_u$ [20]). Let $\mathbf{x} = (x[0], \ldots, x[n-1]) \in F_2^n$. A bit product function π_u: $F_2^n \longrightarrow F_2$ is defined as

$$\pi_{\mathbf{u}}(x) = \prod_{i=0}^{n-1} x[i]^{u[i]}, \quad \mathbf{u} \in F_2^n. \tag{1}$$

Here, we have that $x[i]^1 = x[i]$ and $x[i]^0 = 1$ if $u[i] = 1$ and $u[i] = 0$, respectively.

Definition 2 (algebraic normal form (ANF) [19]). A Boolean function f: $F_2^n \longrightarrow F_2$ can be uniquely represented by its algebraic normal form (ANF) as

$$f(x) = \bigoplus_{\mathbf{u} \in F_2^n} a_{\mathbf{u}}^f \left(\prod_{i=0}^{n-1} x[i]^{u[i]} \right) = \bigoplus_{u \in F_2^n} a_{\mathbf{u}}^f \pi_{\mathbf{u}}(x), \tag{2}$$

where $a_{\mathbf{u}}^f \in F_2$ are the binary constants that depend on $\boldsymbol{u}$ and specify f.

In 2015, Todo [19] introduced the division property (as a generalization of the integral property), which was utilized to efficiently construct integral distinguishers (mainly applicable to S-box-oriented block ciphers). This concept was later refined in [20] by introducing the bit-based division property, which applies to block ciphers that do not necessarily employ S-boxes. The following definitions capture the essence of the bit-based division property.

Definition 3 (ordering $''\prec''$). For two binary vectors $k = (k_1, \ldots, k_n) \in F_2^n$ and $k^* = (k_1^*, \ldots, k_n^*) \in F_2^n$, the inequality $u \prec u$ between $\mathbf{k}$ and $\mathbf{k}^*$ is defined as $\mathbf{k} \prec \mathbf{k}^*$ if and only if $k_i \prec k_i^*$ for all $i \in \{1, \ldots, n\}$.

Definition 4 (bit-based division property [20]). Let X be a multiset whose elements belong to the space F_2^n. Then, X is said to satisfy the division property $D^{1^n}_{\mathbf{k}^{(0)},\mathbf{k}^{(1)},\ldots,\mathbf{k}^{(q-1)}}$, if the parity of $\pi_{\mathbf{u}}(x)$ for all $\mathbf{x} \in X$ is always even. Equivalently, the following conditions must be satisfied:

$$\mathbf{u} \in \left\{(u_0, u_1, \ldots, u_{n-1}) \in F_2^n | \mathbf{u} \succeq \mathbf{k}^{(0)}, \ldots, \mathbf{u} \succeq \mathbf{k}^{(q-1)}\right\}. \tag{3}$$

By 1^n, we denote the binary all-one vector of size n (i.e., $1^n = (1, 1, \ldots, 1)$), where for simplicity, the all-one vector of size one will be simply denoted by 1 instead of 1^1. To provide more clarity about the bit-based division property, we give the following example.

Example 1. Assume that a multiset $X \subset F_2^4$ satisfies the division property $D_K^{1^4}$, where $K = \{1100, 1010, 0011\}$. This means that $\bigoplus_{\mathbf{x}\in X} \pi_{\mathbf{u}}(x)$ is exactly equal to 0 for any $\mathbf{u} \in \{0000, 1000, 0100, 0010, 0001, 1001, 0110, 0101\}$.

In addition, the propagation rules for the bit-division property in SPN schemes were also derived in [19, 20]. Nevertheless, since these rules are not relevant in our context, we omit their specification.

Definition 5 (division trail [13]). Let f_r denote a round function of an iterated block cipher. Assume that an input multiset to the block cipher has initial division property $D^{1^n}_{\{\mathbf{k}\}}$, and denote the division property after propagating through f_r for i rounds by $D^{1^n}_{K_i}$. Thus, we have the following chain of division property propagations:

$$\{k\} \longrightarrow^{\text{def}} K_0 \xrightarrow{f_r} K_1 \xrightarrow{f_r} K_2 \xrightarrow{f_r} \cdots. \tag{4}$$

Moreover, for any vector $\mathbf{k}_i^* \in K_i (i \geq 1)$, there must exist a vector $\mathbf{k}_{i-1}^* \in K_{i-1}$ such that $\mathbf{k}_{\mathbf{i-1}}^*$ can propagate to $\mathbf{k}_{\mathbf{i}}^*$ by division property propagation rules. Furthermore, for $(k_0, k_1, \ldots, k_r) \in K_0 \times K_1 \times \cdots \times K_r$, if $\mathbf{k}_{i-1}$ can propagate to $\mathbf{k}_i$ for all $i \in \{1, 2, \ldots, r\}$, then $(k_0, k_1, \ldots, k_r)$ is called an *r-round division trail.*

Example 2 (Proposition 5 in [13]). Denote by $D^{1^n}_{\{\mathbf{k}\}}$ the division property of the input multiset of an iterated block cipher, and let f_r be its round function. Denote also by

$$\{k\} \longrightarrow^{\text{def}} K_0 \xrightarrow{f_r} K_1 \xrightarrow{f_r} K_2 \xrightarrow{f_r} \cdots \xrightarrow{f_r} K_r, \tag{5}$$

the r-round propagation of division property. Thus, the set of the last vectors in this chain of all r-round division trails which start with $\mathbf{k}$ is equal to K_r.

3. MILP Combined with Bit-Based Division Property

3.1. A Brief Overview of the MILP Method. Many classical cryptanalytic methods can be converted into optimization problems, where the main goal is to achieve an optimal solution (minimum or maximum) of the objective function under certain constraints. The mixed-integer linear programming is a well-known optimization method also used in the field of cryptanalysis and in particular for finding division trails in block ciphers [13, 20]. In general, the objective function can be defined as

$$\text{Obj}\frac{\max}{\min} = \sum_{i=0}^{n-1} c_i x_i, \tag{6}$$

where the linear constraints (including the requirement on variables x_i) are given as follows:

$$\begin{cases} \sum_{i=0}^{n-1} a_{ij} x_i \overset{>}{=}\overset{<}{(><)} b_i, & j = 0, 1, \ldots, m-1, b_i \in Z, \\ x_i \geq 0, & i = 0, 1, \ldots, n-1, \\ x_i \in Z, & i = 0, 1, \ldots, I \text{ and } I \leq n-1. \end{cases} \tag{7}$$

Notice that the MILP problem can be transformed into an integer programming (IP) problem if $I = n - 1$. In particular, it has been verified that IP problems, in general, are somewhat easier to solve than MILP problems of similar kind [12].

For our purpose, the parameters involved in the MILP method are all positive integers. An MILP model is denoted by M, the variables involved are denoted by M.var, the constraints are denoted by M.con, and the objective function is denoted by M.obj. A simple example of an MILP instance can be described as follows. The set of linear inequalities, denoted by L, is given by

$$\begin{cases} x + 2y + 3z \leq 4, \\ x + y \geq 1, \end{cases} \tag{8}$$

where $x, y, z \in \mathrm{Z}^+$ and the objective function is $q = x + y + 2z$. The goal is then to find the maximum value of q. In this example, the domain of the objective function is determined by the two inequalities and constraints that $x, y, z \in \mathrm{Z}^+$, and then the feasible solutions of the objective function in this domain are obtained. The maximum value of q is 3, and it corresponds to $(x, y, z) = (1, 0, 1)$. On the other hand, a closely related problem is to provide a set of points, say A, and to obtain the set of linear inequalities L (using for instance the inequality_generator () function in the Sage software) for which all the solutions satisfying L are included in this set of points A. For further details on how this method works, the reader is referred to Appendix A in [13], where a detailed example is elaborated. As noticed in [13], the main problem with this approach is that the number of linear inequalities returned can be quite large which then makes the MILP instance computationally infeasible. The solution to this was provided by Sun et al. through a greedy algorithm which selects a subset of linear inequalities in L that still efficiently describes A (see [23] and Algorithm 1 in [13]).

Usually, the goal of an MILP problem is to quickly find a feasible (or optimal) solution to the given problem. In the context of bit-based division property, one constructs an MILP model such that it describes the propagation trails of the integral property. This procedure then represents an automatic search for integral distinguishers, where solutions of the MILP problem are interpreted as follows (see also [13]):

(i) Each feasible solution to the system of linear inequalities corresponds to a division trail. In other words, these feasible solutions do not contain any impossible division trail.

(ii) Conversely, each division trail must satisfy all linear inequalities in the system. That is, each division trail corresponds to a feasible solution of the linear inequality system.

Note that, in our work, the constructed MILP model will be solved by the open-source mathematical optimization software Gurobi (https://www.gurobi.com).

3.2. Bit-Based Division Property in terms of MILP. The main reason behind the use of MILP tools in context of the bit-based division property is to improve the time complexity when searching for division trails. In essence, a division trail of an encryption algorithm is obtained by converting the basic operations (involved in the round function) into corresponding linear inequalities, which satisfy the propagation rules of the division property.

Initial division property and stopping rule: let us consider a multiset X with division property $D_K^{1^n}$ and let $\mathbf{e}_i$ denote the vector of length n (also called a *unit vector*) whose i-th coordinate is the only nonzero coordinate. In [13], it was illustrated how to determine the existence of r-round integral distinguisher by checking whether K_{r+1} contains all $\mathbf{e}_i$ $(i \in \{1, 2, \ldots, n\})$. More precisely, if one can find all the unit vectors $\mathbf{e}_i$ in the set K_{r+1} (thus, each $\mathbf{e}_i \in K_{r+1}$), then it means that there does not exist any r-round division trail. Equivalently, if there exists $\mathbf{e}_i$ such that $\mathbf{e}_i \in K_{r+1}$, then it means that one can find an r-round integral distinguisher. In terms of Definition 4, the previously described termination test (condition) for the division property can be explained as follows. Let Y denote the output of r encryption rounds performed on the input set X. If Y does not have any useful integral property, then the XOR sum of all vectors of Y is unknown for each bit position. This means that $\oplus_{y \in Y} \pi_{\mathbf{e}_i}(y)$ is unknown for any unit vector $\mathbf{e}_i \in F_2^n$, where $i \in \{1, 2, \ldots, n\}$. On the contrary, if there exists at least one unit vector $\mathbf{e}_i$ which does not belong to K, then the value at the i-th position of $\oplus_{y \in Y} \pi_{\mathbf{e}_i}(y)$ is always equal to zero, i.e., we can find an r-round integral distinguisher.

For an iterated block cipher with a round function f_r, let $D_{\{\mathbf{k}\}}^{1^n}$ denote the division property of an input multiset. Also, let

$$\{k\} \longrightarrow^{\text{def}} K_0 \xrightarrow{f_r} K_1 \xrightarrow{f_r} K_2 \xrightarrow{f_r} \cdots \xrightarrow{f_r} K_r, \tag{9}$$

be the r-round division property propagation, where K_r denotes the set of vectors of all r-round division trails which start with $\mathbf{k}$.

Now, if we denote an r-round division trail by $(a_0^0, \ldots, a_{n-1}^0) \longrightarrow \cdots \longrightarrow (a_0^r, \ldots, a_{n-1}^r)$, then the set of linear inequalities (which constitute the MILP model) depends on variables $\mathbf{a}_i^j \in F_2$ $(i \in [0, n-1],\ j \in [0, r])$. In addition, the objective function is set to be Obj: Min $\{a_0^r + a_1^r + \cdots + a_{n-1}^r\}$.

Notice that feasible solutions of the given MILP model are all division trails, and furthermore, if K_i does not contain all-zero vector, then the objective function will never take the zero value. At the end of the search, the *balanced* and *unknown* positions of the integral distinguisher can be determined. More precisely, those unit vectors $\mathbf{e}_i$ which are not in K_r will indicate the balanced positions in the distinguisher.

When performing integral analysis on a given block cipher based on the division property and using the MILP model (whose round functions consist of a composition of the S-box and linear layer), the search for *effective* integral distinguisher is the main goal of the attack. In general, this analysis can be roughly divided into the following three steps:

Step 1: determine the division property of the initial input, that is, the specific number of active and constant bits of the input.

Step 2: using the division property mentioned in Step 1, the MILP model of the division path through the round function is constructed according to the structural characteristics of the cryptographic algorithm itself, including both linear and nonlinear layer.

Step 3: let the bit-based division property of r identical encryption rounds of a given block cipher, using the MILP model, be denoted by M. In order to obtain M, one needs to consider r-round propagation of the bit-based division property in the MILP model of the single round function operation.

This is basically done by using the division trail specified by $(a_0^0, \ldots, a_{n-1}^0) \longrightarrow \cdots \longrightarrow (a_0^r, \ldots, a_{n-1}^r)$. As previously mentioned, the system of linear inequalities L will depend on the binary variables $\mathbf{a}_i^j$, where $i = 0, \ldots, n-1$ and $j = 0, \ldots, n-1$ (thus, MILP becomes a 0-1 integer programming problem). However, many of these variables are automatically removed (assigned to a constant value 0) when running Algorithm 3 in [13]. This algorithm uses the set of inequalities L and the objective function to find feasible solutions of the MILP instance M and is constantly updated by adding new constraints with respect to $\mathbf{a}_i^j$, more precisely by setting $\mathbf{a}_i^j = 0$ when needed. The reader is referred to [13] for further details on how Algorithm 3 works. Notice, however, that the MILP instance that models the search for bit-based distinguishers is executed several times (this is an intrinsic property of Algorithm 3 in [13]) since we need to check whether all the unit vectors are included in K_{r+1}, as a stopping rule. Finally, if the solver can find a feasible solution for a particular MILP instance, then the existence of an r-round distinguisher for a given cipher is established (in our case for the SAT_Jo encryption algorithm).

Since some specific cryptographic operations such as key addition and adding a round constant do not affect the propagation of division property, these operations will not be considered here.

4. An MILP Model of SAT_Jo Algorithm

In this section, we describe the process of modelling SAT_Jo algorithm as an MILP instance for the purpose of specifying integral distinguishers.

4.1. A Description of SAT_Jo. The schematic structure of SAT_Jo block cipher is shown in Figure 1, whereas a precise description of its encryption process is given in Algorithm 1. The round function of SAT_Jo is similar to the one in the PRESENT block cipher, and it is defined as a composition of the S-box layer (applying 16 times the S-box defined in Table 1) and the bit-permutation function defined in Table 2. As mentioned earlier, SAT_Jo iterates the round function 31 times, where in addition the round key is applied at the end (as a postwhitening step). We omit the definition of the newRoundKey function because it is not important for the division property.

Remark 1. Notice that the permutation layer uses a simple rule $P(x+i) = P(x) + 8i \bmod 64$ which simplifies design but at the same time induces serious security issues (bad diffusion properties).

4.2. An Integral Attack on SAT_Jo Using Division Property. In order to apply the MILP method, one firstly has to derive a set of linear inequalities $Ax \le b$ (defined in Section 3.1, where $A = [a_{ij}]$) to describe the propagation of division property based on the structure of the round function. We note that both the S-box and permutation layer (P-box) affect the division property when deriving the MILP model. On the other hand, the division property is not affected by the AddRoundKey step in Algorithm 1, and thus the MILP model of a round function is constructed without considering this operation.

4.2.1. Modelling S-Box of SAT_Jo. Now, in order to derive the set of inequalities for the S-box layer of SAT_Jo, we only have to consider the S-box defined in Table 1. Let $x = (x_0, x_1, x_2, x_3)$ denote the input of this S-box and $y = (y_0, y_1, y_2, y_3)$ denote its corresponding output. The ANF of the S-box (given in Table 1) is given by

$$\begin{aligned} y_0 &= 1 + x_0 + x_1 + x_0x_2 + x_0x_1x_2 + x_3 + x_1x_2x_3, \\ y_1 &= x_1 + x_0x_1 + x_2 + x_0x_2 + x_1x_2 + x_3 + x_0x_1x_3, \\ y_2 &= x_1 + x_0x_1 + x_0x_2 + x_0x_3 + x_2x_3 + x_0x_2x_3, \\ y_3 &= x_1 + x_2 + x_0x_3 + x_1x_3 + x_2x_3 + x_1x_2x_3, \end{aligned} \tag{10}$$

where modulo two addition is performed. Then, utilizing Algorithm 2 in [13], we obtain 45 division trails (shown in Table 3) of the SAT_Jo S-box.

Each division trail of a 4-bit S-box can be viewed as an 8-dimensional vector in F_2^8. Thus, 45 division trails form a subset T of F_2^8. Next, by taking T as an input to the inequality_ generator () function of SageMath software, a set of 162 linear inequalities is returned. The following SageMath software code is used for this purpose:

$$\begin{aligned} C &= [\,], \\ T &= [[0,0,0,0,0,0,0,0],[0,0,0,1,0,0,0,1],\ldots,[1,1,1,1,1,1,1,1]], \\ \text{triangle} &= \text{Polyhedron}(\text{vertices} = T), \\ \text{for L in triangle. inequality\_generator}(\,): \; C &= C \cup L, \\ &\cdot \text{return} C. \end{aligned} \tag{11}$$

The output of the above SageMath code, consisting of 162 linear inequalities, can be further reduced to 10 linear inequalities by using the greedy algorithm in [[13], Algorithm 1], originally proposed in [23]. If the division path through the S-box is described by $(a_0, a_1, a_2, a_3) \longrightarrow^{S\text{-box}} (b_0, b_1, b_2, b_3)$, these 10 inequalities are given as follows:

$$S-\text{box} = \begin{cases} a_0 + a_1 + 4a_2 + a_3 - 2b_0 - 2b_1 - 2b_2 - 2b_3 \ge 0, \\ 3a_3 - b_0 - b_1 - b_2 - b_3 \ge 0, \\ -a_0 - 2a_1 - 3a_2 - a_3 + 6b_0 + 4b_1 + 5b_2 + 6b_3 \ge 0, \\ a_0 - a_1 - 2a_2 - 3a_3 - 2b_0 + b_1 + b_3 \ge 0, \\ -a_0 - a_3 + 3b_0 - b_1 - b_2 - 2b_3 \ge 0, \\ a_0 + 4a_1 + a_2 + a_3 - 2b_0 - 2b_1 - 2b_2 - 2b_3 \ge 0, \\ -2a_1 - a_2 - 2a_3 - b_0 + b_2 + b_3 \ge 0, \\ -2a_0 - 2a_1 - a_2 + b_0 - b_2 + b_3 \ge 0, \\ -a_2 - a_3 - b_0 + b_1 \ge 0, \\ -a_0 - a_2 + b_0 - b_1 \ge 0. \end{cases} \tag{12}$$

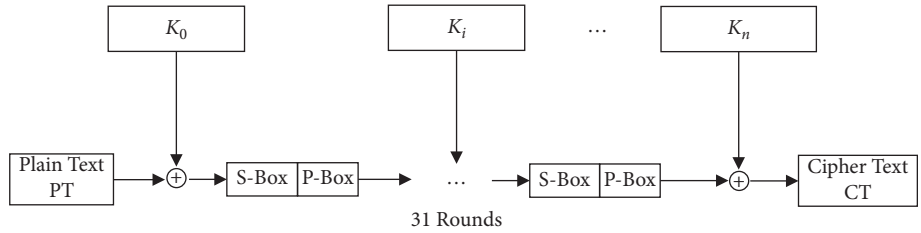

Figure 1: Structure of SAT_Jo.

Input: Plaintext PT [64], Key K [80]
Output: Ciphertext CT [64]
(1) $R \leftarrow 1$
(2) **while** $R \leq 32$ **do**
(3) Round $\text{key}_R \leftarrow$ new Round key (K, R)
(4) $PT \leftarrow PT \oplus \text{Round key}_R$
(5) $PT \leftarrow S - \text{Box}(PT)$
(6) $PT \leftarrow P - \text{Box}(PT)$
(7) $R \leftarrow R + 1$
(8) **end while**
(9) Round $\text{key}_{32} \leftarrow$ new Round key (K, R)
(10) $PT \leftarrow PT \oplus \text{Round key}_{32}$
(11) $CT \leftarrow PT$

Algorithm 1: The SAT_Jo encryption algorithm.

Table 1: S-box used in SAT_Jo cipher.

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

Table 2: Bit-permutation of SAT_Jo (P-box).

X	0	1	2	3	4	5	6	7
$P(X)$	0	8	16	24	32	40	48	56
X	8	9	10	11	12	13	14	15
$P(X)$	1	9	17	25	33	41	49	57
X	16	17	18	19	20	21	22	23
$P(X)$	2	10	18	26	34	42	50	58
X	24	25	26	27	28	29	30	31
$P(X)$	3	11	19	27	35	43	51	59
X	32	33	34	35	36	37	38	39
$P(X)$	4	12	20	28	36	44	52	60
X	40	41	42	43	44	45	46	47
$P(X)$	5	13	21	29	37	45	53	61
X	48	49	50	51	52	53	54	55
$P(X)$	6	14	22	30	38	46	54	62
X	56	57	58	59	60	61	62	63
$P(X)$	7	15	23	31	39	47	55	63

Table 3: 45 division trails in S-box of SAT_Jo algorithm.

Input $D_{\{k\}}^{1^4}$	Output $D_K^{1^4}$
(0,0,0,0)	(0,0,0,0)
(0,0,0,1)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,0,1,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,0,1,1)	(0,0,0,1) (0,0,1,0) (0,1,0,0)
(0,1,0,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(0,1,0,1)	(0,0,0,1) (0,0,1,0) (0,1,0,0)
(0,1,1,0)	(0,0,0,1) (0,0,1,0) (1,0,0,0)
(0,1,1,1)	(0,0,0,1) (0,1,1,0)
(1,0,0,0)	(0,0,0,1) (0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,0,0,1)	(0,0,1,0) (0,1,0,0) (1,0,0,0)
(1,0,1,0)	(0,0,1,0) (0,0,1,0) (1,0,0,0)
(1,0,1,1)	(0,0,1,0) (1,1,0,0)
(1,1,0,0)	(0,0,0,1) (0,1,0,0) (1,0,0,0)
(1,1,0,1)	(0,1,0,0) (1,0,0,1) (1,0,1,0)
(1,1,1,0)	(0,0,0,1) (1,0,0,0)
(1,1,1,1)	(1,1,1,1)

In order to obtain the solutions of linear inequalities restricted to F_2^8, we only need to specify that all variables can only take values in {0, 1}.

4.2.2. Modelling the Permutation Layer of SAT_Jo. In order to describe the permutation layer as an MILP instance, some intermediate variables are introduced to describe the basic operations in the permutation layer. Since the design of the permutation layer of the SAT_Jo encryption algorithm is relatively simple and described on the bit level in [5] (the bit i of the internal state is moved to bit position j in accordance with Table 2 and follows the rule given in Remark 1), the division path of input/output

Input: MILP model consisting of linear inequalities L and an objective function
Output: balanced bit position of S
(1) Initialization $S = \{a_0^r, \ldots, a_{63}^r\}$
(2) **for** $r = 0$; $r \leq 31$; $r++$ **do**
(3) **if** M has a feasible solution **then**
(4) **if** M.ObjValue = 1 **then**//$M(L, \text{Obj})$ value after optimizing M, it returns the current value of the objective function
(5) Obj = obj. Objective ()//obj.Objective represents the objective function of the returned model
(6) **for** $r = 0$; $r \leq 31$; $r++$ **do**
(7) var = obj.getValue (i)//Return the i-th variable of the objective function
(8) value = obj.getAttr ($\imath x\imath$)//Get the var value of the current solution
(9) **if** value = 1 **then**
(10) delete/{var}in S//Delete the var value in S
(11) M.addConstraint (var = 0)
(12) M.update ()
(13) break
(14) **end if**
(15) **end for**
(16) **end if**
(17) **end if**
(18) end for
(19) return S_{all}//Represent the S value of all outputs

Algorithm 2: An automatic search of integral distinguishers for SAT_Jo algorithm.

Table 4: 30-and 31-round distinguishers for SAT_Jo.

Active bits	Balanced bits	Rounds	Distinguisher
2	64	30	$a^2c^{14}, c^{16}, c^{16}, c^{16} \longrightarrow^3 0r\ b^{16}, b^{16}, b^{16}, b^{16}$
4	48	31	$a^4c^{12}, c^{16}, c^{16}, c^{16} \longrightarrow^{31r} ?^4b^4?^4b^4, ?^4b^4?^4b^4, b^{16}, b^{16}$
5	64	31	$c^{16}, c^{16}, c^{16}, c^{11}a^5 \longrightarrow^3 1r\ b^{16}, b^{16}, b^{16}, b^{16}$

through the permutation layer is easily embedded in the MILP model.

4.2.3. A Search Algorithm for Integral Distinguishers for SAT_Jo Algorithm. To summarize the whole procedure, an automatic search algorithm for integral distinguishers of SAT_Jo is given by Algorithm 2 (which is similar to Algorithm 3 in [13]). Note that the notation $M(L, \text{Obj})$ (used in Algorithm 2) denotes the MILP model M for r rounds composed of the set of inequalities L and an objective function Obj. Also, the set of output bits after r rounds is denoted by $S = \{a_0^r, \ldots, a_{63}^r\}$.

5. The Results

By specifying and solving the MILP instance that models the full-round SAT_Jo algorithm (having 31 encryption rounds), we can specify different integral distinguishers. Table 4 shows how many active bits can be set in the input and how many balanced bits are obtained in the output for the SAT_Jo algorithm. Note that all these results are practically confirmed on a personal computer within a few seconds. Moreover, integral distinguishers could be found for up to 151 encryption rounds, which indicates a serious design flow regarding the choice of bit-permutation employed in the SAT_Jo algorithm.

Recall that, for *active bits* at the input, denoted by "a," we essentially take all possible input values at these positions. For instance, if we have 5 active bits in the input, then in total we require 2^5 plaintexts that cover all the possible values at these specific positions. Other input bits that are kept fixed are denoted by "c." The *balanced bits* at the output, denoted by "b," simply correspond to those positions of the ciphertext having the same number of zeros and ones, whereas unbalanced cases are denoted by "?."

Table 5 shows other cryptanalytic results for SAT_Jo.

The key recovery attack on SAT_Jo: in order to perform a key recovery attack on the full-round SAT_Jo cipher, one can use the 30-round distinguisher specified in the first row in Table 4. More precisely, a set of 2^2 plaintexts which satisfies the input of the integral distinguisher is selected. Moreover, one needs to guess the last round subkey bits (64 bits in total) which are then used together with the ciphertexts to calculate the output of the 30^{th} round (the so-called one round partial decryption). For a guessed 64 bit subkey k_{31}, if the XOR sum of the state bits at the output of the 30^{th} round is zero, then it is considered as a valid candidate for the correct subkey; otherwise, the guessed value is considered incorrect. In order to achieve the correct one among these candidates, one selects another set of four plaintexts $P_1, \ldots, P_4$ (again varying the first two bits) and obtains the corresponding ciphertexts $C_1, \ldots, C_4$. For each candidate subkey, the decryption of the

Table 5: The results' comparison of different attacks on SAT_Jo cipher.

Cipher	Analytical method	Round number	References
SAT_Jo	Differential cryptanalysis	24	[22]
SAT_Jo	Linear cryptanalysis	24	[22]
SAT_Jo	Algebraic attack	27	[22]
SAT_Jo	Integral attack	31	New

Table 6: A bit-permutation for SAT_Jo allowing integral distinguishers for at most 9 rounds.

X	0	1	2	3	4	5	6	7
$P(X)$	0	4	8	12	16	20	24	28
X	8	9	10	11	12	13	14	15
$P(X)$	32	36	40	44	48	52	56	60
X	16	17	18	19	20	21	22	23
$P(X)$	1	5	9	13	17	21	25	29
X	24	25	26	27	28	29	30	31
$P(X)$	33	37	41	45	49	53	57	61
X	32	33	34	35	36	37	38	39
$P(X)$	2	6	10	14	18	22	26	30
X	40	41	42	43	44	45	46	47
$P(X)$	34	38	42	46	50	54	58	62
X	48	49	50	51	52	53	54	55
$P(X)$	3	7	11	15	19	23	27	31
X	56	57	58	59	60	61	62	63
$P(X)$	35	39	43	47	51	55	59	63

ciphertexts C_i can be performed and the resulting values (corresponding to the encryption of 30 rounds) are XORed together. The correct subkey is identified when this XOR sum is the all-zero vector of length 64. Thus, the 31-round attack process of SAT_Jo is as follows:

Step 1: after 31 rounds of encryption are performed on the 4 selected plaintexts according to the 30-round integral distinguisher, the opponent can attain 4 ciphertexts.

Step 2: by guessing the 64 bit $k^{0\sim31}, k^{32\sim63}$, the opponent can decrypt the 31-st round data status.

Step 3: similarly, the opponent guesses the 4-bit $k^{0\sim1}, k^{2\sim3}$ so that she can decrypt the 30^{th}-round data status. In this case, she can further calculate the XOR sum of the state bits at the output of the 30^{th} round.

This attack requires 8 chosen plaintexts, and its time complexity is about $2^{64+2} = 2^{66}$ encryption operations. The success rate of this attack is 1. Notice that the master key is of length 80 bits, and after recovering 64 bits of k_{31}, the similar procedure can be performed to retrieve other subkeys.

Remark 2. The simulations have been conducted using the computer with the following specification: Intel(R) Core (T-M) i5-8300H CPU@ 2.30 GHz, RAM-8 GB, x64 Windows 10. In addition, the Python programming language, Sage software, and Gurobi solver have been used to implement the search algorithm.

6. Conclusion

We remark that the choice of bit-permutation used in the SAT_Jo algorithm appears to be the main reason for the existence of full-round integral distinguishers. Indeed, replacing the bit-permutation used in the SAT_Jo algorithm by the one employed in the PRESENT block cipher implies that there are no integral distinguishers for the full-round SAT_Jo. In particular, if the original permutation layer of SAT_Jo is replaced by the bit-permutation given in Table 6, one can verify that the SAT_Jo variant cipher achieves quite good integral property. More precisely, an integral distinguisher can then be specified for at most 9 encryption rounds. The main weakness of SAT_Jo algorithm, as already mentioned in Remark 1, is an inappropriate choice of its bit-permutation which does not provide sufficient diffusion. The permutation layer uses a simple rule $P(x+i) = P(x) + 8i \bmod 64$ for SAT_Jo, which simplifies design but at the same time induces serious security issues (bad diffusion properties). However, the new permutation layer (see Table 6) uses the different rule $P(x+i) = P(x) + 4i \bmod 64$).

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there are no conflicts of interest.

Acknowledgments

The works of Y. Wei and X. Qiu were supported in part by the National Natural Science Foundation of China (No. 61872103), in part by the Guangxi Science and Technology Foundation (No. Guike AB18281019), in part by the Innovation Research Team Project of Guangxi (No. 2019GXNSFGA245004), in part by the Foundation of Science and Technology on Communication Security Laboratory (No. 6142103190103), and in part by the Foundation of Key Laboratory of Cognitive Radio and Information Processing, Ministry of Education (Guilin University of Electronic Technology) (No. CRKL180107). S. Hodzic was supported by a grant from the Independent Research Fund Denmark for Technology and Production (No. 8022-00348A).

References

[1] S. Banik, A. Bogdanov, T. Isobe, and K. Shibutani, "Midori: a block cipher for low energy," *Advances in Cryptology-ASIACRYPT*, vol. 9453, pp. 411–436, 2015.

[2] K. Shibutani, T. Isobe, H. Hiwatari, T. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: an ultra-lightweight block cipher," *Cryptographic Hardware and Embedded Systems-CHES*, vol. 6917, pp. 342–357, 2011.

[3] M. Izadi, B. Sadeghiyan, S. Sadeghian, and H. A. Khanooki, "MIBS: a new lightweight block cipher," *Cryptology and Network Security-CANS*, vol. 5888, pp. 334–348, 2009.

[4] M. R. Albrecht, B. Driessen, E. B. Kavun, G. Leader, C. C. Paar, and T. Yalçın, "Block ciphers-focus on the linear layer (feat. PRIDE)," *Advances in Cryptology-CRYPTO*, vol. 8616, pp. 57–76, 2014.

[5] A. Bogdanov, L. Knudsen, G. Leander et al., "PRESENT: an ultra-lightweight block cipher," *Cryptographic Hardware and Embedded Systems-CHES*, vol. 4727, pp. 450–466, 2007.

[6] W. Wu and L. Zhang, "Lblock: a lightweight block cipher," *Applied Cryptography and Network Security-ACNS*, vol. 6715, pp. 327–344, 2011.

[7] M. S. Turan (Nist), K. McKay (Nist), D. Chang (NIST) et al., *Status Report on the Second Round of the NIST Lightweight Cryptography Standardization Process: NISTIR 8369*, National Insititute of Standards and Technology, Gaithersburg, MD, USA, 2021.

[8] L. Knudsen and D. Wagner, "Integral cryptanalysis," *Fast Software Encryption-FSE*, vol. 2365, pp. 112–127, 2002.

[9] E. Biham and A. Shamir, "Differential cryptanalysis of DES variants," *Differential Cryptanalysis of the Data Encryption Standard*, Springer, New York, NY, USA, 1993.

[10] M. Matsui, "Linear cryptanalysis method for DES cipher," *Advances in Cryptology-EUROCRYPT*, vol. 765, pp. 386–397, 1993.

[11] M. Shantha and L. Arockiam, "Sat_Jo: an enhanced lightweight block cipher for the internet of things," in *Proceedings of the IEEE International Conference on Intelligent Computing and Control Systems-ICICCS2018*, pp. 1146–1150, Madurai, India, June 2018.

[12] N. Mouha, Q. Wang, D. Gu, and B. Preneel, "Differential and linear cryptanalysis using mixed-integer linear programming," *Information Security and Cryptology-Inscrypt*, vol. 7537, pp. 57–76, 2011.

[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," *Advances in Cryptology-ASIACRYPT*, vol. 10031, pp. 648–678, 2016.

[14] J. Daemen, L. Knudsen, and V. Rijmen, "The block cipher SQUARE," *Fast Software Encryption-FSE*, vol. 1267, pp. 149–165, 1997.

[15] S. Lucks, "The saturation attack-a bait for Twofish," *Fast Software Encryption-FSE*, vol. 2355, pp. 1–15, 2001.

[16] A. Biryukov and A. Shamir, "Structural cryptanalysis of SASAS," *Advances in Cryptology-EUROCRYPT*, vol. 2045, pp. 395–405, 2001.

[17] X. Lai, "Higher order derivatives and differential cryptanalysis," *Communications and Cryptography*, vol. 276, pp. 227–233, 1994.

[18] M. Z'aba, H. Raddum, M. Henricksen, and E. D. Dawson, "Bit-pattern based integral attack," *Fast Software Encryption-FSE*, vol. 5086, pp. 363–381, 2008.

[19] Y. Todo, "Structural evaluation by generalized integral property," *Advances in Cryptology-EUROCRYPT*, vol. 9056, pp. 287–314, 2015.

[20] Y. Todo and M. Morii, "Bit-based division property and application to Simon family," *Fast Software Encryption-FSE*, vol. 9783, pp. 357–377, 2016.

[21] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, *The SIMON and SPECK families of lightweight block ciphers*, IACR Cryptology ePrint Archive, 2013, https://eprint.iacr.org/2013/404.

[22] R. Joshitta, L. Arockiam, and P. Malarchelvi, "Security analysis of SAT_Jo lightweight block cipher for data security in healthcare IoT," in *Proceedings of the 2019 3rd International Conference on Cloud and Big Data Computing-ICCBDC 2019*, pp. 111–116, Oxford, England, August 2019.

[23] S. Sun, L. Hu, M. Wang et al., *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, https://eprint.iacr.org/2014/747.pdf.