IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 17 September 2022, accepted 4 October 2022, date of publication 13 October 2022, date of current version 21 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3214665

RESEARCH ARTICLE

Integral Cryptanalysis of Lightweight Block Cipher PIPO

SUNYEOP KIM[1], JESEONG KIM[1], SEONGGYEOM KIM[1], DEUKJO HONG[2], JAECHUL SUNG[3], AND SEOKHIE HONG[1]
[1]Institute of Cyber Security and Privacy (ICSP), Korea University, Seoul 02841, South Korea
[2]Department of Information Technology and Engineering, Jeonbuk National University, Jeonju 54896, South Korea
[3]Department of Mathematics, University of Seoul, Seoul 02504, South Korea

Corresponding author: Deukjo Hong (deukjo.hong@jbnu.ac.kr)

This work was supported in part by the Agency for Defense Development by the Korean Government under Grant UD210027XD, and in part by the Research Funds of Jeonbuk National University in 2021.

ABSTRACT PIPO is a lightweight block cipher proposed at ICISC 2020, which has a byte-oriented structure suitable for bit-sliced implementation and allows for efficient higher-order masking implementations. In this study, we use bit-based division property techniques to construct 6-round integral distinguishers, and propose key-recovery attacks on 8 rounds of PIPO-64/128 and 10 rounds of PIPO-64/256. The data complexity of both attacks is 2^{63} chosen plaintexts and the time complexities are 2^{125} and $2^{253.8}$ respectively. Our results complement the security analysis of PIPO, and show that the PIPO structure is resistant to recently researched cryptanalysis methods. Because only differential and linear attacks were carefully considered to determine the number of rounds of PIPO, our work, based on division property, is important for verifying the security margin.

INDEX TERMS Division Property, integral cryptanalysis, PIPO.

I. INTRODUCTION

PIPO is a lightweight block cipher proposed at ICISC 2020 [1]. It has a byte-oriented structure suitable for bit-sliced implementation, and provides good performance on an 8-bit AVR platform. It also allows for efficient higher-order masking implementations. The designers claimed that differential, linear, impossible differential, boomerang, and meet-in-the-middle attacks work at most 9, 9, 6, 8, and 6 rounds for PIPO-64/128, and at most 11, 11, 8, 10, and 10 rounds for PIPO-64/256, respectively.

Integral cryptanalysis [2] exploits a distinguisher causing a zero sum for a target structure, similar to higher-order cryptanalysis [3] and square attack [4]. Todo [5] proposed a remarkable approach of division property, allowing the construction of many rounds of integral distinguishers for target structures. His work led to the first attack on full-round MISTY cipher [6] and developed into bit-based techniques [7], [8].

The associate editor coordinating the review of this manuscript and approving it for publication was Tony Thomas.

TABLE 1. Integral attacks on PIPO-64/128 and PIPO-64/256.

Target	Attacked Rounds	Data Complexity	Time Complexity
PIPO-64/128	8	2^{63}	2^{125}
PIPO-64/256	10	2^{63}	$2^{253.8}$

In this study, we examine the division property for PIPO and find that the division property can propagate up to 6 rounds. Then, we construct 6-round integral distinguishers [2] based on the observations and perform a key-recovery attack on 8-round PIPO-64/128 and 10 rounds of PIPO-64/256. The attack on 8-round PIPO-64/128 recovers a 128-bit key with 2^{63} chosen plaintexts and 2^{125} encryptions, whereas the attack on 10-round PIPO-64/256 recovers a 256-bit key with 2^{63} chosen plaintexts and $2^{253.8}$ encryptions. Our results are summarized in Table 1. Integral cryptanalysis is an important tool for analyzing the security of block ciphers; however, to the best of our knowledge, the resistance of PIPO to integral cryptanalysis has never been published,

TABLE 2. Comparisons of attacks on reduced-round PIPO-64/128 and PIPO-64/256.

• Full-Round PIPO-64/128 : 13 Rounds • Full-Round PIPO-64/256 : 17 Rounds

Cryptanalysis	Type	Key Size	Distinguisher	Key Recovery	Source
Differential	Chosen-Plaintext Attack	128-bit	6 Rounds	9 Rounds	[1]
		256-bit		11 Rounds	
Linear	Known-Plaintext Attack	128-bit	6 Rounds	9 Rounds	
		256-bit		11 Rounds	
Impossible differential	Chosen-Plaintext Attack	128-bit	4 Rounds	6 Rounds	
		256-bit		8 Rounds	
Boomerang/Rectangle	Adaptive Chosen-Ciphertext Attack	128-bit	6 Rounds	8 Rounds	
		256-bit		10 Rounds	
Meet-in-the-Middle	Chosen-Plaintext Attack	128-bit	6 Rounds	6 Rounds	
		256-bit	10 Rounds	10 Rounds	
Integral	Chosen-Plaintext Attack	128-bit	6 Rounds	8 Rounds	This paper
		256-bit		10 Rounds	

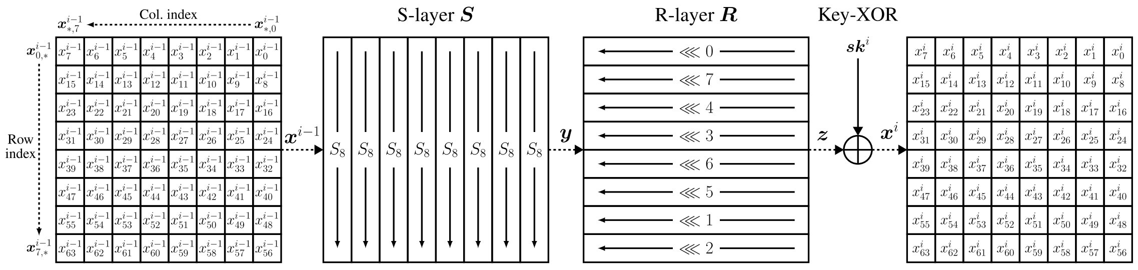


FIGURE 1. ***i*-th round of PIPO.**

even in [1]. Although our results do not weaken the security claim of full-round PIPO as presented in Table 2, these complement the security analysis by conducting attacks on the reduced-round versions.

The remainder of this paper is organized as follows. In Section II, we present the basic background and related work. Section III discusses how the PIPO structure is modeled as suitable for an MILP solver. In Section IV, we analyze the division properties of PIPO structure. Section V presents the integral distinguishers and attacks on reduced rounds of PIPO. In Section VI, we present our conclusions.

II. PRELIMINARIES

A. SYMBOLS AND NOTATIONS

An n-bit binary vector $\boldsymbol{x} \in \mathbb{F}_2^n$ is defined as $(x_{n-1}, x_{n-2}, \ldots, x_0)$, where $x_i \in \mathbb{F}_2$ for $0 \leq i < n$. This can also be denoted by $\boldsymbol{x} = x_{n-1}x_{n-2}\cdots x_0$. We define $\boldsymbol{x} \lll i$ as an operation rotating a binary vector $\boldsymbol{x}$ in the left direction by i bits. We denote the concatenation of the two binary vectors $\boldsymbol{x}$ and $\boldsymbol{y}$ by $\boldsymbol{x}\|\boldsymbol{y}$. We represent a sequence of consecutive identical bits with the superposition of a single bit. For example, a 7-bit string 1111000 or a 7-bit binary vector (1, 1, 1, 1, 0, 0, 0) can be denoted by $1^4 0^3$.

Let $\mathbb{X}$ be a multiset of n-bit vectors. We denote the output multiset of a map $\boldsymbol{f} : \mathbb{F}_2^n \to \mathbb{F}_2^m$ by $\boldsymbol{f}(\mathbb{X}) := \{\boldsymbol{f}(\boldsymbol{x}) : \boldsymbol{x} \in \mathbb{X}\}$, where $\boldsymbol{x} \in \mathbb{F}_2^n$ and $\boldsymbol{y} \in \mathbb{F}_2^n$. We define $w(\boldsymbol{x}) = \sum_{i=0}^{n-1} x_i$ as the Hamming weight of $\boldsymbol{x}$, $\boldsymbol{x} \succeq \boldsymbol{y}$ as $x_i \geq y_i$ for $0 \leq i < n$, and $\boldsymbol{x} \cdot \boldsymbol{y} = \bigoplus_{i=0}^{n-1} x_i y_i$ as the inner product of $\boldsymbol{x}$ and $\boldsymbol{y}$, where $x_i y_i$ is the AND of $x_i \in \mathbb{F}_2$ and $y_i \in \mathbb{F}_2$. In addition, we define $\boldsymbol{x}^{\boldsymbol{y}}$ as a monomial $\prod_{i=0}^{n-1} x_i^{y_i}$.

Let f be a Boolean function from $\mathbb{F}_2^n$ to $\mathbb{F}_2$ and the algebraic normal form (ANF) of f be $f(\boldsymbol{x}) = \bigoplus_{\boldsymbol{y} \in \mathbb{F}_2^n} \alpha_{\boldsymbol{y}} \boldsymbol{x}^{\boldsymbol{y}}$ with $\alpha_{\boldsymbol{y}} \in \mathbb{F}_2$. We define a set ANF_f of all the terms of f as $\mathrm{ANF}_f = \{\boldsymbol{y} \in \mathbb{F}_2^n \mid \alpha_{\boldsymbol{y}} = 1\}$.

Let $E : \mathbb{F}_2^k \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be a block cipher with k-bit key and n-bit block. $\boldsymbol{c} = E_{\boldsymbol{\kappa}}(\boldsymbol{p})$ indicates that plaintext $\boldsymbol{p} \in \mathbb{F}_2^n$ is encrypted to ciphertext $\boldsymbol{c} \in \mathbb{F}_2^n$ through block cipher E with key $\boldsymbol{\kappa} \in \mathbb{F}_2^k$. Furthermore, $\mathbb{Y} = E_{\boldsymbol{\kappa}}(\mathbb{X})$ implies that $\mathbb{Y}$ is the (multi)set of ciphertexts to which the block cipher E encrypts all plaintexts in the (multi)set $\mathbb{X}$ with the key $\boldsymbol{\kappa} \in \mathbb{F}_2^k$.

B. BLOCK CIPHER PIPO

Block cipher PIPO was proposed at ICISC 2020 [1]. The block length of PIPO is 64 bits. PIPO is denoted by PIPO-64/128 for 128-bit keys and by PIPO-64/256 for 256-bit keys, respectively. PIPO-64/128 and PIPO-64/256 have the SPN (Substitution-Permutation Network) structure with 13 and 17 rounds, respectively.

As Fig. 1 shows, it is convenient to represent a 64-bit state vector $\boldsymbol{x}$ of PIPO as an 8×8 binary matrix $\{x_{i,j}\}$ whose (i,j)-th entry $x_{i,j}$ is equal to x_{8i+j} for $0 \leq i, j < 8$. Its i-th row $\boldsymbol{x}_{i,*}$ is $(x_{i,7}, x_{i,6}, \ldots, x_{i,0}) = (x_{8i+7}, \ldots, x_{8i+1}, x_{8i})$, and its j-th column $\boldsymbol{x}_{*,j}$ is defined as $(x_{7,j}, x_{6,j}, \ldots, x_{0,j})^t = (x_{56+j}, x_{48+j}, \ldots, x_j)^t$. Note that the column index starts on the right.

The key schedule of PIPO-64/128 splits a 128-bit master key $\boldsymbol{\kappa}$ into two 64-bit parts $\boldsymbol{\kappa} = \boldsymbol{\kappa}_1 \| \boldsymbol{\kappa}_0$. Subsequently, subkeys are defined as $\boldsymbol{sk}^i = \boldsymbol{\kappa}_{i \bmod 2} \oplus i$ for $0 \leq i \leq 13$. The key schedule of PIPO-64/256 splits a 256-bit $\boldsymbol{\kappa}$ into four

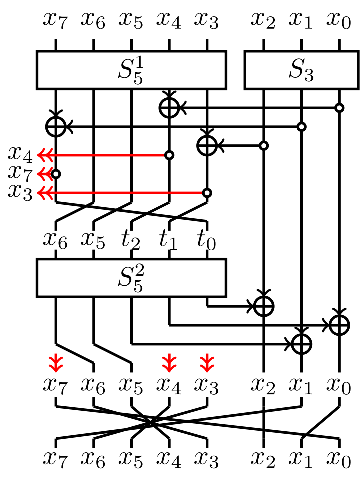

x_7 x_6 x_5 x_4 x_3 x_2 x_1 x_0
S_5^1
S_3
x_4
x_7
x_3
x_6 x_5 t_2 t_1 t_0
S_5^2
x_7 x_6 x_5 x_4 x_3 x_2 x_1 x_0
x_7 x_6 x_5 x_4 x_3 x_2 x_1 x_0

FIGURE 2. **Overall structure of S_8.**

64-bit parts $\kappa = \kappa_3 \| \kappa_2 \| \kappa_1 \| \kappa_0$. Subsequently, the subkeys are defined as $sk^i = \kappa_{i \bmod 4} \oplus i$ for $0 \le i \le 17$.

The round function of PIPO consists of an S-layer S for nonlinear operation, an R-layer R for linear operation, and Key-XOR for adding round keys (see Fig. 1). The input x of the first round is the XOR of the plaintext p and whitening subkey sk^0: $x^0 = p \oplus sk^0$. In the first round, S applies the 8-bit S-box S_8 to each column of x^0, and the output y of S is the concatenation of the outputs of the S-boxes. The output z of R is the concatenation of the left rotation of each row of y. The numbers of rotated bits are 0, 7, 4, 3, 6, 5, 1, and 2 from 0-th row to 7-th row of y, respectively. The output of the first round is $x^1 = z \oplus sk^1$ which is the input of the second round. Each of the remaining rounds has the same process: S-layer, R-layer, and Key-XOR (with sk^i for $i = 2, 3, \ldots$).

1) · 8-BIT S-BOX of PIPO

The 8-bit S-box S_8 of S-layer is constructed with a 3-bit S-box S_3 and two 5-bit S-boxes S_5^1 and S_5^2. Fig. 2 illustrates the structure of S_8. The 3-bit input $x = (x_2, x_1, x_0)$ can be updated to the output $S_3(x)$ of S_3 as follows:

$$\begin{aligned} x_2 &\leftarrow x_2 \oplus (x_1 \wedge x_0); \\ x_0 &\leftarrow x_0 \oplus (x_2 \vee x_1); \\ x_1 &\leftarrow x_1 \oplus (x_2 \vee x_0); \\ x_2 &\leftarrow x_2 \oplus 1. \end{aligned}$$

The 5-bit input $x = (x_4, x_3, x_2, x_1, x_0)$ can be updated to the output $S_5^1(x)$ of S_5^1 as follows:

$$\begin{aligned} x_2 &\leftarrow x_2 \oplus (x_4 \wedge x_3); \\ x_1 &\leftarrow x_1 \oplus (x_2 \wedge x_0); \\ x_4 &\leftarrow x_4 \oplus x_1; \\ x_3 &\leftarrow x_3 \oplus x_0; \\ x_0 &\leftarrow x_0 \oplus (x_2 \vee x_1); \\ x_2 &\leftarrow x_2 \oplus x_4; \\ x_1 &\leftarrow x_1 \oplus (x_3 \wedge x_2). \end{aligned}$$

The 5-bit input $x = (x_4, x_3, x_2, x_1, x_0)$ can be updated to the output $S_5^2(x)$ of S_5^2 as follows:

$$\begin{aligned} x_4 &\leftarrow x_4 \oplus (x_3 \wedge x_0); \\ x_0 &\leftarrow x_0 \oplus x_4; \\ x_4 &\leftarrow x_4 \oplus (x_2 \vee x_1); \\ x_1 &\leftarrow x_1 \oplus x_3; \\ x_3 &\leftarrow x_3 \oplus (x_4 \vee x_2); \\ x_2 &\leftarrow x_2 \oplus (x_1 \wedge x_0). \end{aligned}$$

Finally, for the 8-bit input $x = (x_7, \ldots, x_0)$, the output of 8-bit S-box S_8 is computed as follows:

$$\begin{aligned} (x_7, x_6, x_5, x_4, x_3) &\leftarrow S_5^1(x_7, x_6, x_5, x_4, x_3); \\ (x_2, x_1, x_0) &\leftarrow S_3(x_2, x_1, x_0); \\ t_2 &\leftarrow x_4 \leftarrow x_4 \oplus x_0; \\ t_0 &\leftarrow x_7 \leftarrow x_7 \oplus x_1; \\ t_1 &\leftarrow x_3 \leftarrow x_3 \oplus x_2; \\ (x_6, x_5, t_2, t_1, t_0) &\leftarrow S_5^2(x_6, x_5, t_2, t_1, t_0); \\ x_2 &\leftarrow x_2 \oplus t_0; \\ x_0 &\leftarrow x_0 \oplus t_1; \\ x_1 &\leftarrow x_1 \oplus t_2; \\ (x_7, \ldots, x_0) &\leftarrow (x_1, x_3, x_4, x_5, x_6, \\ &\qquad x_2, x_0, x_7). \end{aligned}$$

The unbalanced-bridge structure, which combines S_3, S_5^1, and S_5^2, provides high differential and linear branch numbers as well as efficient masking implementations [1].

C. INTEGRAL CRYPTANALYSIS

Integral cryptanalysis stemmed from the security evaluation of block cipher Square [4] and was formalized in [2]. This method uses integral distinguishers.

We denote the state of an active bit variable, on which 0 and 1 both appear, by ‘a’ and the state of a constant bit variable, on which the value is fixed as constant, by ‘c’. For example, if the state of the 4-bit variable (x_3, x_2, x_1, x_0) is (ccaa), four 4-bit values can appear with $(x_1, x_0) = (0, 0), (0, 1), (1, 0)$, and $(1, 1)$ for a certain constant value of (x_3, x_2). An integral distinguisher requires an input multiset whose state consists of active and constant bits, and exploits the fact that the XOR-sum of the corresponding output multiset is always zero at some bits.

Definition 1 (Integral Distinguisher): Let $E : \mathbb{F}_2^k \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an r-round block cipher with k-bit key and n-bit block. Let $\mathbb{X}$ and $\mathbb{Y} = E_\kappa(\mathbb{X})$ be a plaintext multiset and ciphertext multiset under a key $\kappa \in \mathbb{F}_2^k$, respectively. If there exists any index i such that

$$\bigoplus_{y \in \mathbb{Y}} y_i = \bigoplus_{x \in \mathbb{X}} E_\kappa(x)_i = 0 \quad \forall \kappa \in \mathbb{F}_2^k,$$

we say that the i-th bit variable y_i of the ciphertext is *balanced*, and call the transition from $\mathbb{X}$ to $\mathbb{Y}$ an *r-round integral distinguisher* for E.

Assuming that an integral distinguisher has m balanced bits, the probability that random permutation $\mathcal{P}$ on $\mathbb{F}_2^n$ satisfies m balanced bits is 2^{-m}. Hence, we can use such an integral distinguisher to distinguish block cipher E from $\mathcal{P}$.

D. DIVISION PROPERTY

The notion of the division property was proposed by Todo at EUROCRYPT 2015 [5] as an efficient method for constructing integral distinguishers, and was subsequently generalized to bit-based division property [7]. In this study, we focus on the conventional bit-based division property. The definition is given in Definition 2.

Definition 2 (Conventional Bit-Based Division Property [6]): Let $\mathbb{X}$ be a multiset whose elements take the value of $\mathbb{F}_2^n$, and let $\boldsymbol{k}$ be an n-dimensional vector whose i-th element takes 0 or 1. When multiset $\mathbb{X}$ has the *conventional bit-based division property* $\mathcal{D}_{\mathbb{K}}^n$, it satisfies the following conditions:

$$\bigoplus_{\boldsymbol{x}\in\mathbb{X}} \boldsymbol{x}^{\boldsymbol{u}} = \begin{cases} \text{unknown} & \text{if } \exists \boldsymbol{k} \in \mathbb{K} \text{ s.t. } \boldsymbol{u} \succeq \boldsymbol{k}, \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

For simplicity, the conventional bit-based division property is mentioned as a division property in the remainder of this paper. If $\boldsymbol{k} \in \mathbb{K}$ and $\boldsymbol{k}' \in \mathbb{K}$ satisfy $\boldsymbol{k} \succeq \boldsymbol{k}'$, we can remove $\boldsymbol{k}$ from $\mathbb{K}$ because $\boldsymbol{k}$ does not affect the condition (1). In [8], Xiang et al. defined operation SizeReduce($\mathbb{K}$) by removing redundant vectors from $\mathbb{K}$ and returning the reduced set of $\mathbb{K}$.

1) DIVISION PROPERTY PROPAGATION RULE

Todo [7] demonstrated how the division property is propagated through `copy`, `and`, and `xor`. In this section, we briefly present propagation rules. In the following rules, the notation $A \Leftarrow B$ for sets A, B denotes $A = A \cup B$.

a: · RULE 1 (`copy`)

Let $\boldsymbol{f} : \mathbb{F}_2 \rightarrow \mathbb{F}_2^2$ be a `copy` function, where the input $(x_0) \in \mathbb{F}_2$ and the output is calculated as (x_0, x_0). Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multisets of $\boldsymbol{f}$. If $\mathbb{X}$ has $\mathcal{D}_{\mathbb{K}}^1$, $\mathbb{Y}$ has $\mathcal{D}_{\mathbb{K}'}^2$, where $\mathbb{K}'$ is computed for all $\boldsymbol{k} \in \mathbb{K}$ as

$$\mathbb{K}' \Leftarrow \begin{cases} \{(0,0)\} & \text{if } k_0 = 0, \\ \{(0,1),(1,0)\} & \text{if } k_0 = 1. \end{cases} \quad (2)$$

b: · RULE 2 (`and`)

Let $f : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2$ be an `and` function, where the input $(x_1, x_0) \in \mathbb{F}_2^2$ and the output is calculated as $(x_1 \wedge x_0)$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multisets of f, respectively. If $\mathbb{X}$ has $\mathcal{D}_{\mathbb{K}}^2$, $\mathbb{Y}$ has $\mathcal{D}_{\mathbb{K}'}^1$, where $\mathbb{K}'$ is computed for all $\boldsymbol{k} \in \mathbb{K}$ as

$$\mathbb{K}' \Leftarrow \left\{ \left(\left\lceil \frac{(k_1 + k_0)}{2} \right\rceil \right) \right\}. \quad (3)$$

c: · RULE 3 (`xor`)

Let $f : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2$ be an `xor` function, where the input $(x_1, x_0) \in \mathbb{F}_2^2$ and the output is calculated as $(x_1 \oplus x_0)$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multisets of f, respectively.

Algorithm 1 Calculating $DP(\boldsymbol{f}, \boldsymbol{k})$

```
Input: The input division property $\mathcal{D}_{\boldsymbol{k}}^n$ of $\boldsymbol{f}$ where $\boldsymbol{k} \in \mathbb{F}_2^n$
Output: A set $\mathbb{K}$ of vectors such that the output multiset has
        the division property $\mathcal{D}_{\mathbb{K}}^m$
1: $\mathbb{S} \leftarrow \{\boldsymbol{k}' \mid \boldsymbol{k}' \succeq \boldsymbol{k}\}$
2: $\mathbb{K} \leftarrow \varnothing$
3: for $\boldsymbol{u} \in \mathbb{F}_2^m$ do
4:     if $\mathrm{ANF}_{\boldsymbol{f}^{\boldsymbol{u}}} \cap \mathbb{S} \neq \varnothing$ then
5:         $\mathbb{K} \Leftarrow \{\boldsymbol{u}\}$
6:     end if
7: end for
8: $DP(\boldsymbol{f}, \boldsymbol{k}) = \mathrm{SizeReduce}(\mathbb{K})$
9: return $DP(\boldsymbol{f}, \boldsymbol{k})$
```

If $\mathbb{X}$ has $\mathcal{D}_{\mathbb{K}}^2$, $\mathbb{Y}$ has $\mathcal{D}_{\mathbb{K}'}^1$, where $\mathbb{K}'$ is computed for all $\boldsymbol{k} \in \mathbb{K}$ as

$$\mathbb{K}' \Leftarrow \{(\max\{k_1, k_0\})\}. \quad (4)$$

d: · RULE 4 (S-BOX)

In addition to the above basic operations, the division property propagation through the S-box can be derived by analyzing its ANF [8].

Let $\boldsymbol{f} : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m$ be a function of the S-box, where the input $\boldsymbol{x} \in \mathbb{F}_2^n$ and the output $\boldsymbol{y} \in \mathbb{F}_2^m$. Let $\mathbb{X}$ and $\mathbb{Y}$ be the input and output multisets of $\boldsymbol{f}$. If $\mathbb{X}$ has $\mathcal{D}_{\mathbb{K}}^n$, $\mathbb{Y}$ has $\mathcal{D}_{\mathbb{K}'}^m$, where $\mathbb{K}'$ is computed for all $\boldsymbol{k} \in \mathbb{K}$ as

$$\mathbb{K}' \Leftarrow DP(\boldsymbol{f}, \boldsymbol{k}).$$

For each $\boldsymbol{k} \in \mathbb{K}$, $DP(\boldsymbol{f}, \boldsymbol{k}) \subset \mathbb{F}_2^m$ is defined as

$$\{\boldsymbol{k}' \mid \boldsymbol{f}^{\boldsymbol{k}'} \text{contains any term } \boldsymbol{x}^{\boldsymbol{u}} \text{ satisfying } \boldsymbol{u} \succcurlyeq \boldsymbol{k}\},$$

where $\boldsymbol{f}^{\boldsymbol{k}'}$ is $\prod_{i=0}^{m-1} f_i(\boldsymbol{x})^{k_i'}$. We can calculate $DP(\boldsymbol{f}, \boldsymbol{k})$ using Algorithm 1, which was introduced in [8]. As mentioned above, the redundant vectors of $\mathbb{K}'$ do not affect the division property. Therefore, Algorithm 1 considers the reduced set by applying SizeReduce($\mathbb{K}$) in Line 8.

2) DIVISION TRAIL

As shown in [8], the propagation of the division property can be regarded as a transition of vectors, from $\boldsymbol{k} \in \mathbb{K}$ of the division property $\mathcal{D}_{\mathbb{K}}^n$ to $\boldsymbol{k}' \in \mathbb{K}'$ of the division property $\mathcal{D}_{\mathbb{K}'}^m$. In [8], Xiang et al. defined a chain of propagation as a division trail.

Definition 3 (Division Trail [8]): Let $E : \mathbb{F}_2^k \times \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$ be an iterated block cipher, and let $\boldsymbol{f}_i$ denote the i-th round function of E. Assume that the input multiset to E has an initial division property $\mathcal{D}_{\boldsymbol{k}}^n$, and denote the division property after r-round propagation through $\boldsymbol{f}_i$ by $\mathcal{D}_{\mathbb{K}_r}^n$. Thus, we have the following chain of division property propagations.

$$\{\boldsymbol{k}\} := \mathbb{K}_0 \xrightarrow{f_1} \mathbb{K}_1 \xrightarrow{f_2} \mathbb{K}_2 \xrightarrow{f_3} \cdots \xrightarrow{f_r} \mathbb{K}_r.$$

Moreover, for any vector $\boldsymbol{k}_i^*$ in $\mathbb{K}_i$ $(i \geq 1)$, there exists a vector $\boldsymbol{k}_{i-1}^*$ in $\mathbb{K}_{i-1}$ such that $\boldsymbol{k}_{i-1}^*$ can propagate to $\boldsymbol{k}_i^*$

by the division property propagation rules. Furthermore, for $(\boldsymbol{k}_0, \boldsymbol{k}_1, \ldots, \boldsymbol{k}_r) \in \mathbb{K}_0 \times \mathbb{K}_1 \times \cdots \times \mathbb{K}_r$, if $\boldsymbol{k}_{i-1}$ can propagate to $\boldsymbol{k}_i$ for all $i \in \{1, 2, \ldots, r\}$, then we call $(\boldsymbol{k}_0, \boldsymbol{k}_1, \ldots, \boldsymbol{k}_r)$ an *r-round division trail*.

Definition 3 implies that the set of last vectors of all r-round division trails starting with $\boldsymbol{k}$ is equal to $\mathbb{K}_r$. Therefore, checking for the existence of a useful integral distinguisher after r-round encryption (i.e., obtaining $\mathbb{K}_r$ such that there exists any unit vector $e \notin \mathbb{K}_r$) is equivalent to finding all r-round division trails starting with $\boldsymbol{k}$. Based on this observation, Xiang et al. proposed an approach for finding all division trails by constructing a linear inequality system whose feasible solutions represent all division trails.

E. MILP-AIDED DIVISION PROPERTY

Mixed-integer linear programming (MILP) has been applied to cryptanalytic problems. An MILP model $\mathcal{M}$ comprises a variable set $\mathcal{M}.var$, a constraint set $\mathcal{M}.con$, and the objective function $\mathcal{M}.obj$.

The above propagation rules for MILP should be adjusted to determine the division property. To determine the division property of $\boldsymbol{f} : \mathbb{F}_2^n \to \mathbb{F}_2^m$, we should create an MILP model $\mathcal{M}$ such that $\mathcal{M}$ has only division trails of $\boldsymbol{f}$ as solutions. There are two requirements for it for $\boldsymbol{a}\|\boldsymbol{b} \in \mathbb{F}_2^{n+m}$:

1) If $\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$ is a division trail of $\boldsymbol{f}$, $\boldsymbol{a}\|\boldsymbol{b}$ is a solution of $\mathcal{M}$;
2) If $\boldsymbol{a}\|\boldsymbol{b}$ s a solution of $\mathcal{M}$, $\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$ is a division trail of $\boldsymbol{f}$.

a: · MILP MODEL FOR `copy`

When $(a_0) \xrightarrow{\texttt{copy}} (b_1, b_0)$ is a division trail of $\texttt{copy}(x_0) = (x_0, x_0)$, then the MILP model $\mathcal{M}$ can be

$$\begin{cases} \mathcal{M}.con \Leftarrow a_0 - b_1 - b_0 = 0; \\ \mathcal{M}.var \Leftarrow a_0, b_1, b_0 \ : \ \text{binaries} \end{cases}$$

to satisfy Rule 1 of Equation (2).

b: · MILP MODEL FOR `and`

When $(a_1, a_0) \xrightarrow{\texttt{and}} (b_0)$ is a division trail of $\texttt{and}(x_1, x_0) = x_1 \wedge x_0$, then the MILP model $\mathcal{M}$ can be

$$\begin{cases} \mathcal{M}.con \Leftarrow b_0 - a_1 \geq 0; \\ \mathcal{M}.con \Leftarrow b_0 - a_0 \geq 0; \\ \mathcal{M}.con \Leftarrow b_0 - a_1 - a_0 \leq 0; \\ \mathcal{M}.var \Leftarrow a_1, a_0, b_0 \ : \ \text{binaries} \end{cases}$$

to satisfy Rule 2 of Equation (3).

c: · MILP MODEL FOR `xor`

When $(a_1, a_0) \xrightarrow{\texttt{xor}} (b_0)$ is a division trail of $\texttt{xor}(x_1, x_0) = x_1 \oplus x_0$, then the MILP model $\mathcal{M}$ can be

$$\begin{cases} \mathcal{M}.con \Leftarrow a_1 + a_0 - b_0 = 0; \\ \mathcal{M}.var \Leftarrow a_1, a_0, b_0 \ : \ \text{binaries} \end{cases}$$

to satisfy Rule 3 of Equation (4).

d: · MILP MODEL FOR S-BOX

Compared with the basic operations `copy`, `and`, and `xor`, various approaches can be considered to construct an MILP model $\mathcal{M}$ for the S-box. Constructing $\mathcal{M}$ for S-box $\boldsymbol{f} : \mathbb{F}_2^n \to \mathbb{F}_2^m$ is equivalent to converting a set of $(n+m)$-bit vectors

$$\{\boldsymbol{a}\|\boldsymbol{b} \mid \boldsymbol{b} \in DP(\boldsymbol{f}, \boldsymbol{a})\}$$

into a set of linear inequalities, $\mathcal{M}.con$. The conversion can be conducted in two ways: using the product-of-sum representation of Boolean functions [9] and the *Inequality_generator()* function in Sagemath software.[1] Each of the two conversions is detailed in the following section when constructing MILP models for the S-box of PIPO.

III. MILP MODEL FOR PIPO BLOCK CIPHER

In this section, we propose three methods of constructing MILP models for the S-box S_8 of PIPO and compare them. Moreover, we introduce a method for exploiting the rotational symmetry of PIPO to analyze the division properties more efficiently.

A. MILP MODEL FOR S-BOX OF PIPO

We attempted to construct MILP models for S-box S_8 of PIPO in three ways.

1) BY H-REPRESENTATION: $\mathcal{M}^{\text{H-repre}}$

First, we applied Rule 4 (S-box) directly to S_8 and obtained the set

$$P_8 = \bigcup_{\boldsymbol{a} \in \mathbb{F}_2^8} \{\boldsymbol{a}\|\boldsymbol{b} \mid \boldsymbol{b} \in DP(S_8, \boldsymbol{a})\}$$

of division trails for S_8. We convert P_8 into the corresponding linear inequalities using the *Inequality_generator()* function in the Sagemath software. Specifically, the function *Inequality_generator()* determines an H-representation (a set of inequalities) of the convex hull of P_8. We denote this model for S_8 by $\mathcal{M}^{\text{H-repre}}$. Although the greedy approaches in [10], [11] can optimize $\mathcal{M}^{\text{H-repre}}$ by computing a small number of inequalities that exactly describe P_8, this reduction is only possible when the original H-representation is given.

2) BY PRODUCT-OF-SUM REPRESENTATION: $\mathcal{M}^{\text{QM}}$

Second, we applied the conversion of [9] to P_8. We define the Boolean function $g : \mathbb{F}_2^{n+m} \to \mathbb{F}_2$ as

$$g(\boldsymbol{x}) = \begin{cases} 1 & \text{if } \boldsymbol{x} \in P_8, \\ 0 & \text{if } \boldsymbol{x} \notin P_8. \end{cases}$$

This gives the product-of-sum representation of $g(\boldsymbol{x})$ as

$$g(\boldsymbol{x}) = \bigwedge_{\boldsymbol{u} \notin P_8} \left(\bigvee_{u_i=0} x_i \vee \bigvee_{u_i=1} \overline{x_i} \right).$$

[1] Available at http://www.sagemath.org/

The product-of-sum representation trivially corresponds to a set of inequalities which exactly describe P_8 as

$$\left\{\sum_{u_i=0} x_i + \sum_{u_i=1}(1-x_i) \geq 1 \mid \boldsymbol{u} \notin P_8\right\}.$$

Therefore, we can simplify the set of inequalities for S_8 by minimizing the number of terms in the product-of-sum representation. We apply the Quine-McClusky algorithm to minimize and obtain the MILP model $\mathcal{M}^{\mathrm{QM}}$ for S_8.

3) BY CONSIDERING STRUCTURE OF S_8: $\mathcal{M}^{\mathrm{struct}}$

Finally, considering the structure of S_8, we derive an MILP model $\mathcal{M}^{\mathrm{struct}}$ for S_8 from the sets

$$P_3 = \bigcup_{\boldsymbol{a}\in\mathbb{F}_2^3} \{\boldsymbol{a}\|\boldsymbol{b} \mid \boldsymbol{b} \in DP(S_3, \boldsymbol{a})\}$$

$$P_5^1 = \bigcup_{\boldsymbol{a}\in\mathbb{F}_2^5} \{\boldsymbol{a}\|\boldsymbol{b} \mid \boldsymbol{b} \in DP(S_5^1, \boldsymbol{a})\}$$

$$P_5^2 = \bigcup_{\boldsymbol{a}\in\mathbb{F}_2^5} \{\boldsymbol{a}\|\boldsymbol{b} \mid \boldsymbol{b} \in DP(S_5^2, \boldsymbol{a})\}$$

of division trails for S_3, S_5^1, and S_5^2 respectively. As explained in Section II-B and described in Fig. 2, S_8 is constructed with an unbalanced-bridge structure with S_3, S_5^1 and S_5^2. We obtain the corresponding MILP models for P_3, P_5^1, and P_5^2 by applying the Quine-McClusky algorithm. We then combine them with the MILP models for `copy` and `xor` operations explained in Section II-E to obtain $\mathcal{M}^{\mathrm{struct}}$ for S_8.

4) COMPARISON OF MILP MODELS FOR S_8

$\mathcal{M}^{\text{H-repre}}$ and $\mathcal{M}^{\mathrm{QM}}$ allow accurate analysis of S_8. However, $\mathcal{M}^{\text{H-repre}}$ is efficient only for S-boxes whose sizes are less than 8 bits, because the computational complexity required to obtain linear inequalities and optimize them increases in proportion to the size of the S-box. $\mathcal{M}^{\mathrm{QM}}$ also does not guarantee its efficiency over 8-bit S-boxes, but fortunately, we obtained it on S_8 of PIPO around one hour.

However, $\mathcal{M}^{\mathrm{struct}}$ does not guarantee analysis as accurate as $\mathcal{M}^{\text{H-repre}}$ and $\mathcal{M}^{\mathrm{QM}}$ because it does not cover monomials cancelled through XORs in the ANF of S_8. For some input division property $\boldsymbol{k}$, $\mathcal{M}^{\mathrm{struct}}$ occurs a larger unknown set[2] of Equation (1)

$$\{\boldsymbol{u} \succeq \boldsymbol{k}' \mid \boldsymbol{k}\|\boldsymbol{k}' \text{ is feasible in}\|\text{out in } \mathcal{M}^{\mathrm{struct}}\} \tag{5}$$

than $\mathcal{M}^{\text{H-repre}}$ and $\mathcal{M}^{\mathrm{QM}}$. Nevertheless, we proceeded to obtain $\mathcal{M}^{\mathrm{struct}}$ because of its efficiency in modeling simple operations and small S-boxes. Note that modeling simple operations, such as `copy` and `xor` costs, is negligible. See Table 3 for a comparison of the time complexities for modeling S_8.

[2]Note that the unknown set of $\mathcal{M}^{\mathrm{QM}}$ is included in that of $\mathcal{M}^{\mathrm{struct}}$ for any input division property $\boldsymbol{k}$. This implies some integral distinguishers may not be found in the model $\mathcal{M}^{\mathrm{struct}}$.

TABLE 3. Comparison of MILP models for S_8.

MILP Model	Modeling time	# of $\boldsymbol{k} \to \boldsymbol{u}$ in Equation (5)
$\mathcal{M}^{\text{H-repre}}$	Infeasible	-
$\mathcal{M}^{\mathrm{QM}}$	About 1 hour	55832
$\mathcal{M}^{\mathrm{struct}}$	Less than 1 sec.	55832 + 829

B. ROTATIONAL SYMMETRY OF PIPO

We can find an MILP model $\mathcal{M}$ for r rounds of PIPO, based on the analysis given in Section III-A. Then, we solve $\mathcal{M}$ to construct any r-round division trail $(\boldsymbol{a}^0, \boldsymbol{a}^1, \ldots, \boldsymbol{a}^r)$. To obtain an integral distinguisher from the trail, we need to start the trail with $\boldsymbol{k}$ of the division property $\mathcal{D}_{\boldsymbol{k}}^{64}$ for the plaintext multiset. We can achieve this by adding the following constraints to $\mathcal{M}.con$.

$$a_j^0 = k_j \quad \text{for } j = 0, 1, \ldots, n-1.$$

Moreover, we should set $w(\boldsymbol{k}) = 63$ to obtain the longest integral distinguishers, for which we can search.

PIPO has the rotational property stated in Theorem 4.

Theorem 4 (Rotational Symmetry of PIPO): Let $\boldsymbol{k}$ be an 8×8 array of 64-bit binary vector. Let $\tau(\boldsymbol{k})$ be the 64-bit vector in which each row of $\boldsymbol{k}$ is right rotated by one bit. For the round function $\boldsymbol{f}$ of PIPO, if $\boldsymbol{k} \xrightarrow{f} \boldsymbol{k}'$ is a division trail of $\boldsymbol{f}$, $\tau(\boldsymbol{k}) \xrightarrow{f} \tau(\boldsymbol{k}')$ is a division trail of $\boldsymbol{f}$ as well.

Proof: We omit the Key-XOR operation considering the components of the round function $\boldsymbol{f}$ of PIPO because it does not have any impact on the division property. Subsequently, a division trail $\boldsymbol{k} \xrightarrow{f} \boldsymbol{k}'$ is regarded as $\boldsymbol{k} \xrightarrow{\boldsymbol{S}} \boldsymbol{k}^s \xrightarrow{\boldsymbol{R}} \boldsymbol{k}'$.

First, we demonstrate that $\tau(\boldsymbol{k}) \xrightarrow{\boldsymbol{S}} \tau(\boldsymbol{k}^s)$. We have $\tau(\boldsymbol{k})_{*,j} = \boldsymbol{k}_{*,(j-1) \bmod 8}$ and $\boldsymbol{R}(\boldsymbol{k}^s)_{*,j} = \boldsymbol{k}^s_{*,(j-1) \bmod 8}$. Because $\boldsymbol{k} \xrightarrow{\boldsymbol{S}} \boldsymbol{k}^s$ is a division trail of the S-layer, $\boldsymbol{k}_{*,(j-1) \bmod 8}$ can be propagated to $\boldsymbol{k}^s_{*,(j-1) \bmod 8}$ through the S-box S_8. Therefore, we have $\tau(\boldsymbol{k})_{*,j} \xrightarrow{S_8} \tau(\boldsymbol{k}^s)_{*,j}$ for $0 \leq j \leq 7$, and $\tau(\boldsymbol{k}) \xrightarrow{\boldsymbol{S}} \tau(\boldsymbol{k}^s)$.

Finally, we demonstrate that $\tau(\boldsymbol{k}^s) \xrightarrow{\boldsymbol{R}} \tau(\boldsymbol{k}')$. It is trivial from the assumption $\boldsymbol{k}^s \xrightarrow{\boldsymbol{R}} \boldsymbol{k}'$, because both τ and the R-layer belong to rotation operations on 8×8 arrays of 64-bit values. This completes this proof. □

Rotational symmetry can be used to reduce the number of initial division properties to be considered for search, because searching for trails starting with $\boldsymbol{k}$ covers trails starting with $\tau(\boldsymbol{k}), \tau^2(\boldsymbol{k}), \ldots,$ or $\tau^7(\boldsymbol{k})$.

IV. DIVISION PROPERTY ANALYSIS WITH LINEAR TRANSFORMATIONS

A. EXTENDED INTEGRAL DISTINGUISHERS

Lambin et al. [12] presented a method for identifying more integral distinguishers. Their approach involves searching for $L_{out} \circ E \circ L_{in}$ instead of a block cipher $E : \mathbb{F}_2^k \times \mathbb{F}_2^n \to \mathbb{F}_2^n$, where L_{in} and $L_{out} \in GL_n(\mathbb{F}_2)$ and where we regard E as a nonlinear permutation on $\mathbb{F}_2^n$, a block cipher with a randomly selected secret key over $\mathbb{F}_2^k$. Generally, their method finds an extended integral distinguisher. This is defined as Definition 5.

Definition 5 ((Extended) Integral Distinguisher): Let $E : \mathbb{F}_2^k \times \mathbb{F}_2^n \to \mathbb{F}_2^n$ be an r-round block cipher with k-bit key and n-bit block. Let $\mathbb{X}$ and $\mathbb{Y}$ be the plaintext and ciphertext multisets of E, respectively. For any key $\kappa \in \mathbb{F}_2^k$, if there exists some $\boldsymbol{v} \in \mathbb{F}_2^n \setminus \{\boldsymbol{0}\}$ such that

$$\bigoplus_{\boldsymbol{y}\in\mathbb{Y}} \boldsymbol{v}\cdot\boldsymbol{y} = \bigoplus_{\boldsymbol{x}\in\mathbb{X}} \boldsymbol{v}\cdot E_\kappa(\boldsymbol{x}) = 0,$$

$(\mathbb{X}, \boldsymbol{v})$ is called an r-round integral distinguisher of E, and $\boldsymbol{v}\cdot\boldsymbol{y}$ is called a balanced bit.

B. LINEAR TRANSFORMATIONS ON INPUT AND OUTPUT

We consider L_{in} only as a concatenation of eight 8×8 matrices $L_{in}^j \in GL_8(\mathbb{F}_2)$ for $0 \leq j < 8$, because it is computationally impossible to try all the 64×64 binary linear matrices. Similarly, we consider L_{out} only as a concatenation of eight 8×8 matrices $L_{out}^j \in GL_8(\mathbb{F}_2)$ for $0 \leq j < 8$.

Each output bit of $L_{out}^j \circ S_8$ for $0 \leq j < 8$ has the form of $\boldsymbol{v}_{out} \cdot S_8$ for the corresponding row $\boldsymbol{v}_{out}$ of L_{out}^j. Therefore, we only need to check whether there exists $\boldsymbol{v}_{out}$ such that $\boldsymbol{v}_{out} \cdot S_8$ is balanced for the j-th S-box in the last round function in order to find integral distinguishers with L_{out}^j.

Considering the rotational symmetry of PIPO, we can force the initial division property $\mathcal{D}_{\boldsymbol{k}}^{64}$ to have a single zero bit at the least significant position of $\boldsymbol{k}$. In other words, we assume that the initial division property is $\mathcal{D}_{1^{63}0}^{64}$ and that the initial multiset is $(\mathtt{a}\cdots\mathtt{ac})$ where the least significant bit is constant, and the other bits are active. Under this assumption, Theorem 6 implies that L_{in}^j for $1 \leq j < 8$ do not change its initial division properties.

Theorem 6: If the input division property is $\mathcal{D}_{1^n}^n$, for any invertible $\boldsymbol{f} : \mathbb{F}_2^n \to \mathbb{F}_2^n$, the output division property is $\mathcal{D}_{1^n}^n$.

Proof: Assume $\boldsymbol{f}(\boldsymbol{x}) = \boldsymbol{y}$. According to Proposition 1 in [13], $\deg(\boldsymbol{y}^{\boldsymbol{u}}) = n$ only when $\boldsymbol{u}$ is an n-bit all-one vector 1^n. Therefore, $DP(\boldsymbol{f}, 1^n) = \{1^n\}$, and the output division property is $\mathcal{D}_{1^n}^n$. $\square$

Now, we can consider only L_{in}^0 with the given input division property $\mathcal{D}_{1^70}^8$. Because $DP(S_8 \circ L_{in}^0, 1^70)$ depends only on linear combinations of bits that become constant, we can classify 8×8 invertible matrices into $2^8 - 1$ classes, in which each matrix instantiating L_{in}^0 has the same $DP(S_8 \circ L_{in}^0, 1^70)$.

For $\mathcal{D}_{\mathbb{K}}^n$, we define $Succ(\boldsymbol{k}) := \{\boldsymbol{u} \in \mathbb{F}_2^n \mid \boldsymbol{u} \succeq \boldsymbol{k}\}$ for $\boldsymbol{k} \in \mathbb{K}$ and $Succ(\mathbb{K}) := \bigcup_{\boldsymbol{k}\in\mathbb{K}} Succ(\boldsymbol{k})$. Let $\mathcal{D}_{\mathbb{K}_1}^{64}$ and $\mathcal{D}_{\mathbb{K}_1'}^{64}$ be the output division properties of the first S-box in the first round when two different matrices M and M' instantiate L_{in}^0. If $Succ(\mathbb{K}_1) \subseteq Succ(\mathbb{K}_1')$, we can exclude M' from the search for extended integral distinguishers. We only have four candidates for L_{in}^0 after applying this observation.

Finally, we propose a search algorithm for integral distinguishers considering L_{in} and L_{out}. Algorithm 2 takes an MILP model $\mathcal{M}$ for $(r-2)$-round division trail and a linear transformation L_{in}^0 as inputs to provide r-round extended integral distinguishers.

TABLE 4. Search results for 6-round integral distinguishers of PIPO.

Model for S_8	# of Distinguishers	Running Time
$\mathcal{M}^{\text{QM}}$	136	7.25h
$\mathcal{M}^{\text{struct}}$	64	16.5h

In Lines 2 - 4, Algorithm 2 first computes the division property for the first round by considering $\mathcal{D}_{1^70}^8$ for $S_8 \circ L_{in}^0$ and $\mathcal{D}_{1^8}^8$ for $S_8 \circ L_{in}^i$, $1 \leq i \leq 7$. Then, through the loop covering Lines 5 - 31, it searches for balanced bits in the r-th round output on the division property after the first round. $\mathbb{U}$ is the set of all $\boldsymbol{v}$, such that the parity of $\boldsymbol{x}^{\boldsymbol{v}}$ is unknown for the output $\boldsymbol{x}$ of the $(r-1)$-th round. In Lines 7 - 24, we use the MILP model $\mathcal{M}$ for $r-2$ rounds of PIPO to collect all possible entries of $\mathbb{U}$. In Lines 25 - 30, it computes $\text{ANF}_{\boldsymbol{v}_{out}\circ S_8}$ for the j-th S-box in the r-th round, and checks whether $\text{ANF}_{\boldsymbol{v}_{out}\circ S_8}$ contains monomials whose parities are unknown. If $\text{ANF}_{\boldsymbol{v}_{out}\circ S_8}$ contains no such monomials, $\boldsymbol{v}_{out} \circ S_8$ is a balanced bit of an r-round extended integral distinguisher. Consequently, all balanced bits after r rounds are stored in $\mathbb{S}$ in the form of $(j, \boldsymbol{v}_{out})$.

V. INTEGRAL DISTINGUISHERS AND ATTACKS

A. SEARCHING FOR DISTINGUISHERS

We attempted two ways to search for distinguishers by constructing two MILP models for PIPO combining the S-box models, namely $\mathcal{M}^{\text{struct}}$ and $\mathcal{M}^{\text{QM}}$ obtained in Section III-A. We used Gurobi MILP Solver and performed every experiment on the platform of AMD Ryzen Threadipper 3970X CPU 3.7GHz, 256GB RAM and Ubuntu 20.04.1 LTS x86_64.

As a result, we found seventeen 6-round integral distinguishers for PIPO by searching with $\mathcal{M}^{\text{QM}}$, of which we can also find eight through a search with $\mathcal{M}^{\text{struct}}$. This implies 136 6-round distinguishers due to the rotational symmetry in the PIPO structure. Both search approaches did not find any integral distinguishers for more than 6 rounds of PIPO.

The 6-round integral distinguishers are split into two classes depending on the form of constant bit information in the input. Considering rotational symmetry with $0 \leq i < 8$, the distinguishers in the first class have the constant bit information:

$$x_{6,i}^0 \oplus x_{3,i}^0$$

in the input. The corresponding balanced bit information in the output is one of seven:

$$\begin{aligned}\mathcal{B}_0 = \{&x_{0,1+i}^6 \oplus x_{1,i}^6 \oplus x_{6,2+i}^6,\\ &x_{0,2+i}^6 \oplus x_{1,1+i}^6 \oplus x_{6,3+i}^6,\\ &x_{0,3+i}^6 \oplus x_{1,2+i}^6 \oplus x_{6,4+i}^6,\\ &x_{0,4+i}^6 \oplus x_{1,3+i}^6 \oplus x_{6,5+i}^6,\\ &x_{0,5+i}^6 \oplus x_{1,4+i}^6 \oplus x_{6,6+i}^6,\\ &x_{0,6+i}^6 \oplus x_{1,5+i}^6 \oplus x_{6,7+i}^6,\\ &x_{0,7+i}^6 \oplus x_{1,6+i}^6 \oplus x_{6,i}^6\}.\end{aligned}$$

Algorithm 2 Extended Integral Distinguisher Search

```
Input: MILP model $\mathcal{M}$ for $r-2$ rounds of PIPO, linear transformation $L_{in}^0$
Output: A set $\mathbb{S}$ of $r$-round extended integral distinguishers
1:  $\mathbb{S} \leftarrow \varnothing$
2:  $\mathbb{K}_{in} \leftarrow DP(S_8 \circ L_{in}^0, 1^7 0)$        ▷ Consider only $L_{in}^0$ due to the rotational symmetry
3:  $\mathbb{K}_1 \leftarrow \{(1^7k_7\|1^7k_6\|\dots\|1^7k_0) \in \mathbb{F}_2^{64} \mid \boldsymbol{k} = (k_7, k_6, \dots, k_0) \in \mathbb{K}_{in}\}$
4:  $\mathbb{K}_1 \leftarrow \boldsymbol{R}(\mathbb{K}_1)$        ▷ $\boldsymbol{R}$ is the R-layer function
5:  for $j = 0, 1, \dots, 7$ do
6:      $\mathbb{U} \leftarrow \varnothing$        ▷ $\mathbb{U}$ implies the division property on the position of $j$-th S-box
7:      for $\boldsymbol{k} \in \mathbb{K}_1$ do
8:          for $\boldsymbol{v} \in \mathbb{F}_2^8 \setminus \{\boldsymbol{0}\}$ do
9:              if $\boldsymbol{v} \notin \mathbb{U}$ then
10:                 $\mathcal{M}' \leftarrow \mathcal{M}$
11:                 $\mathcal{M}'.con \Leftarrow \boldsymbol{a}^0 = \boldsymbol{k}$
12:                 for $i = 0, 1, \dots, 7$ do        ▷ Locate $\boldsymbol{v}$ at the position of $j$-th S-box
13:                     if $i = j$ then
14:                         $\mathcal{M}'.con \Leftarrow \boldsymbol{a}_{*,i}^{r-2} = \boldsymbol{v}$
15:                     else
16:                         $\mathcal{M}'.con \Leftarrow \boldsymbol{a}_{*,i}^{r-2} = \boldsymbol{0}$
17:                     end if
18:                 end for
19:                 if $\mathcal{M}'$ has any feasible solution then
20:                     $\mathbb{U} \Leftarrow \{\boldsymbol{v}' \mid \boldsymbol{v}' \succeq \boldsymbol{v}\}$
21:                 end if
22:             end if
23:         end for
24:     end for
25:     for $\boldsymbol{v}_{out} \in \mathbb{F}_2^8 \setminus \{\boldsymbol{0}\}$ do
26:         Compute $\mathrm{ANF}_{\boldsymbol{v}_{out} \cdot S_8}$
27:         if $\mathrm{ANF}_{\boldsymbol{v}_{out} \cdot S_8} \cap \mathbb{U} = \varnothing$ then
28:             $\mathbb{S} \Leftarrow (j, \boldsymbol{v}_{out})$
29:         end if
30:     end for
31: end for
32: return $\mathbb{S}$
```

Algorithm 3 Key-Recovery Attack on 8-Round **PIPO**-64/128

- **Data Collection Phase** - Choose 2^{63} plaintexts, $\boldsymbol{p}$'s in which p_{56} is fixed as constant, and obtain the corresponding ciphertexts, $\boldsymbol{c}$'s.
- **Key Filtering Phase** - Guess a 64-bit value of the last subkey $\boldsymbol{sk}^8$ and do the followings:
  1) Perform one-round decryption for all ciphertexts with the guessed value of $\boldsymbol{sk}^8$,
  $$\boldsymbol{x}^7 = \boldsymbol{S}^{-1}(\boldsymbol{R}^{-1}(\boldsymbol{c} \oplus \boldsymbol{sk}^8)).$$
  2) Consider $\boldsymbol{x}^6 = \boldsymbol{S}^{-1}(\boldsymbol{R}^{-1}(\boldsymbol{x}^7) \oplus \boldsymbol{rk}^7)$ instead of $\boldsymbol{x}^6 = \boldsymbol{S}^{-1}(\boldsymbol{R}^{-1}(\boldsymbol{x}^7 \oplus \boldsymbol{sk}^7))$, and let $\boldsymbol{y} = \boldsymbol{R}^{-1}(\boldsymbol{x}^7)$. For all 2^{63} values of $\boldsymbol{x}^7$, count each number $t_i[\boldsymbol{y}_{*,i}]$ of times $\boldsymbol{y}_{*,i}$ appear for $i \in \{0, 1, 6, 7\}$.
  3) Guess 4-byte values of $(\boldsymbol{rk}_7^7, \boldsymbol{rk}_6^7, \boldsymbol{rk}_1^7, \boldsymbol{rk}_0^7)$ and do the followings:
     a) For all 2^{63} values of $\boldsymbol{y}$, compute each parity of α_j for $0 \le j \le 3$ in Table 5 considering $t_i[\boldsymbol{y}_{*,i}]$ for $i \in \{0, 1, 6, 7\}$ (See Fig. 3).
     b) If one of the α_i's parities is odd, exclude the guessed values of $\boldsymbol{sk}^8$ and $(\boldsymbol{rk}_6^7, \boldsymbol{rk}_7^7, \boldsymbol{rk}_1^7, \boldsymbol{rk}_0^7)$ from the space of key candidates.
- **Exhaustive Searching Phase** - Perform an exhaustive search for the 128-bit key $\boldsymbol{k}$ over the key space.

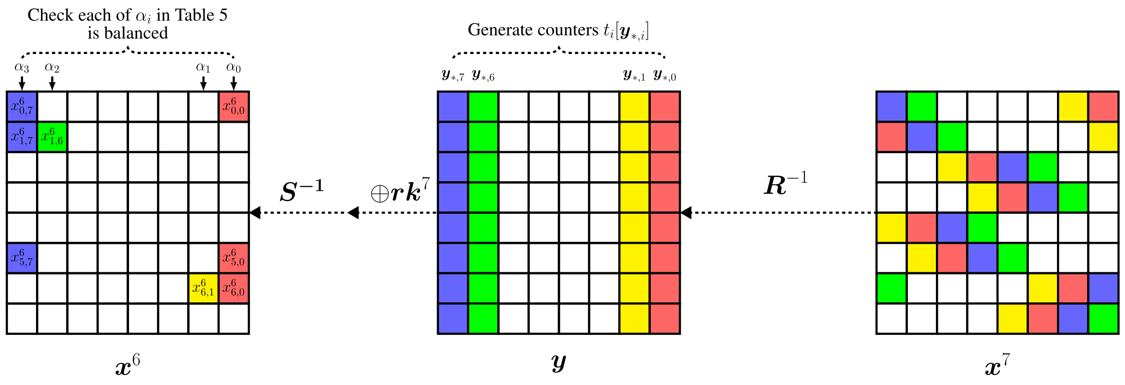


FIGURE 3. 7-th round partial decryption of Key Filtering Phase in Algorithm 3: Colored bits in y and x^7 are related to the bits of the same color in x^6. Only four inverse S-box operations of S^{-1} are required with a 4-byte partial guessed key $(rk^7_7, rk^7_6, rk^7_1, rk^7_0)$.

Similarly, with $0 \le i < 8$, the distinguishers in the second class have the constant bit information:

$$x^0_{7,i}$$

in the input. The corresponding balanced bit information in the output is one of ten:

$$\begin{aligned}\mathcal{B}_1 = \{&\underline{x^6_{5,i}},\\ &\underline{x^6_{5,7+i}},\\ &\underline{x^6_{0,i} \oplus x^6_{1,7+i} \oplus x^6_{6,1+i}},\\ &\underline{x^6_{0,1+i} \oplus x^6_{1,i} \oplus x^6_{6,2+i}},\\ &x^6_{0,2+i} \oplus x^6_{1,1+i} \oplus x^6_{6,3+i},\\ &x^6_{0,3+i} \oplus x^6_{1,2+i} \oplus x^6_{6,4+i},\\ &x^6_{0,4+i} \oplus x^6_{1,3+i} \oplus x^6_{6,5+i},\\ &x^6_{0,5+i} \oplus x^6_{1,4+i} \oplus x^6_{6,6+i},\\ &x^6_{0,6+i} \oplus x^6_{1,5+i} \oplus x^6_{6,7+i},\\ &x^6_{0,7+i} \oplus x^6_{1,6+i} \oplus x^6_{6,i}\}.\end{aligned}$$

Except for $x^6_{5,i}$ and $x^6_{5,7+i}$, distinguishers with the balanced bit information in $\mathcal{B}_1$ can be found by searching with both $\mathcal{M}^{\mathrm{QM}}$ and $\mathcal{M}^{\mathrm{struct}}$.

B. KEY-RECOVERY ATTACK ON 8-ROUND PIPO-64/128

We can use four 6-round integral distinguishers, underlined in $\mathcal{B}_1$, to mount a key-recovery attack on 8 rounds of PIPO-64/128. The distinguishers are applied from the first round to the sixth round, with the same active bits in the input and various balanced bits in the output. The plaintext is denoted as $\boldsymbol{p} = (p_{63}, \ldots, p_1, p_0)$. We use 2^{63} plaintexts in which p_{56} is fixed as a constant. In the attack, the attacker should try all possible 2^{64} candidates of the last subkey $\boldsymbol{sk}^8$ and guess four bytes of $\boldsymbol{rk}^7$, where $\boldsymbol{rk}^7 = \boldsymbol{R}^{-1}(\boldsymbol{sk}^7)$. Table 5 lists the balanced bits in the output and the key bytes of $\boldsymbol{rk}^7$ related to the distinguishers. The attack process is presented in Algorithm 3. During the attack, the 7-th round partial decryption of **Key Filtering Phase** requires only the 32-bit intermediate values $\boldsymbol{y}_{*,i}$ for $i \in \{0, 1, 6, 7\}$ and a 4-byte guessed key $(\boldsymbol{rk}^7_7, \boldsymbol{rk}^7_6, \boldsymbol{rk}^7_1, \boldsymbol{rk}^7_0)$ as Fig. 3 describes.

TABLE 5. Balanced bits and key bytes related to distinguishers.

No.	Balanced bit	Key bytes
0	$\alpha_0 = x^6_{5,0}$	$\boldsymbol{rk}^7_0$
1	$\alpha_1 = x^6_{5,7}$	$\boldsymbol{rk}^7_1$
2	$\alpha_2 = x^6_{0,0} \oplus x^6_{1,7} \oplus x^6_{6,1}$	$\boldsymbol{rk}^7_0, \boldsymbol{rk}^7_1, \boldsymbol{rk}^7_7$
3	$\alpha_3 = x^6_{0,7} \oplus x^6_{1,6} \oplus x^6_{6,0}$	$\boldsymbol{rk}^7_0, \boldsymbol{rk}^7_6, \boldsymbol{rk}^7_7$

We expect that the key space can be reduced by the ratio of 2^{-4} after **Key Filtering Phase** because the α_i for $1 \le i \le 3$ are even with the probability of 2^{-4} if the guessed keys are not correct. Thus, **Exhaustive Search Phase** requires a time complexity of 2^{124} 8-round PIPO-64/128 encryptions. The time complexity of **Key Filtering Phase** is estimated as $2^{63} \times 2^{64} \times 2^{-3} = 2^{124}$ 8-round PIPO-64/128 encryptions, because it is dominated by step 1) of **Key Filtering Phase**. Therefore, the total time complexity of the attack is 2^{125}.

C. KEY-RECOVERY ATTACK ON 10-ROUND PIPO-64/256

PIPO-64-128 and PIPO-64/256 has the same structure except the key schedule. The difference between the key schedules allows a 10-round attack on PIPO-64/256. In the attack, we use the same distinguishers, guess the same bits of $\boldsymbol{rk}^7$ and $\boldsymbol{sk}^8$ as in the attack on 8-round PIPO-64/128, and additionally guess the whole bits of $\boldsymbol{sk}^9$ and $\boldsymbol{sk}^{10}$. Therefore, the time complexity of **Key Filtering Phase** is estimated as $2^{253.3} \approx 2^{63} \times 2^{64 \cdot 3} \times 3/10$ 10-round PIPO-64/256 encryptions, while the time complexity of the final exhaustive search phase is 2^{252}. Therefore, the total time complexity of the attack is approximately $2^{253.8} \approx 2^{253.3} + 2^{252}$.

VI. CONCLUSION

In this paper, we analyzed the division property of the lightweight block cipher PIPO proposed at ICISC 2020 based on three MILP models with different modeling time and accuracy. As a result, we could find 136 6-round integral distinguishers. Among them, 120 distingusihers were derived by adding linear transformations into the S-box. We performed key-recovery attacks on 8 rounds of PIPO-64/128 and 10 rounds of PIPO-64/256 based on four of the obtained distinguishers with 2^{125} and $2^{253.8}$ time complexities, respectively. Although our results do not weaken the security claim of full-round PIPO, these complement the security analysis. Moreover, we expect that our search approach[3] can be used to find the best choice of R-layer in terms of resistance against integral attack.

REFERENCES

[1] H. Kim, Y. Jeon, G. Kim, J. Kim, B.-Y. Sim, D.-G. Han, H. Seo, J. Sung, and D. Hong, "*PIPO*: A lightweight block cipher with efficient higher-order masking software implementations," in *Proc. Int. Conf. Inf. Secur. Cryptol.* Cham, Switzerland: Springer, 2020, pp. 99–122.

[2] L. Knudsen and D. Wagner, "Integral cryptanalysis," in *Pro. Int. Workshop Fast Softw. Encryption*. Cham, Switzerland: Springer, 2002, pp. 112–127.

[3] L. R. Knudsen, "Truncated and higher order differentials," in *Proc. Int. Workshop Fast Softw. Encryption*. Cham, Switzerland: Springer, 1994, pp. 196–211.

[4] J. Daemen, L. Knudsen, and V. Rijmen, "The block cipher square," in *Proc. Int. Workshop Fast Softw. Encryption*. Cham, Switzerland: Springer, 1997, pp. 149–165.

[5] Y. Todo, "Structural evaluation by generalized integral property," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.* Cham, Switzerland: Springer, 2015, pp. 287–314.

[6] Y. Todo, "Integral cryptanalysis on full MISTY1," *J. Cryptol.*, vol. 30, no. 3, pp. 920–959, Jul. 2017.

[7] Y. Todo and M. Morii, "Bit-based division property and application to SIMON family," in *Proc. Int. Conf. Fast Softw. Encryption*. Cham, Switzerland: Springer, 2016, pp. 357–377.

[8] 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 *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Cham, Switzerland: Springer, 2016, pp. 648–678.

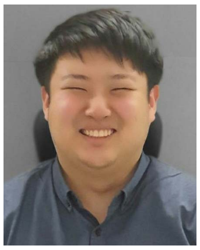
[9] A. Abdelkhalek, Y. Sasaki, Y. Todo, M. Tolba, and A. M. Youssef, "MILP modeling for (large) S-boxes to optimize probability of differential characteristics," *IACR Trans. Symmetric Cryptol.*, vol. 2017, no. 4, pp. 99–129, Dec. 2017.

[10] S. Sun, L. Hu, P. Wang, K. Qiao, X. Ma, and L. Song, "Automatic security evaluation and (related-key) differential characteristic search: Application to SIMON, PRESENT, LBlock, DES(L) and other bit-oriented block ciphers," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Secur.* Cham, Switzerland: Springer, 2014, pp. 158–178.

[11] S. Sun, L. Hu, M. Wang, P. Wang, K. Qiao, X. Ma, D. Shi, L. Song, and K. Fu, "Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties," Cryptol. ePrint Arch., 2014.

[12] B. Lambin, P. Derbez, and P.-A. Fouque, "Linearly equivalent S-boxes and the division property," *Des. Codes Cryptogr.*, vol. 88, no. 10, pp. 2207–2231, Oct. 2020.

[13] C. Boura, A. Canteaut, and C. D. Cannière, "Higher-order differential properties of KECCAK and *Luffa*," in *Proc. Int. Workshop Fast Softw. Encryption*. Cham, Switzerland: Springer, 2011, pp. 252–269.



SUNYEOP KIM received the B.S. degree in mathematics from Korea University, in 2019, where he is currently pursuing the Ph.D. degree with the Graduate School of Cyber Security. His research interest includes symmetric cryptography.

[3]Available at https://github.com/SUNYEOP/PIPO_DP

JESEONG KIM received the M.S. degree in information security from Korea University, in 2022. His research interest includes symmetric cryptography.

SEONGGYEOM KIM received the M.S. degree in information security from Korea University, in 2018, where he is currently pursuing the Ph.D. degree with the Graduate School of Cyber Security. His research interests include symmetric cryptography and random number generators.

DEUKJO HONG received the B.S. and M.S. degrees in mathematics and the Ph.D. degree in information security from Korea University, in 1999, 2002, and 2006, respectively. From 2007 to 2015, he was employed at ETRI. He is currently an Associate Professor with the Department of Information Technology and Engineering, Jeonbuk National University. His research interest includes symmetric cryptography.

JAECHUL SUNG received the Ph.D. degree in mathematics from Korea University, in 2002. He was a Senior Researcher at the Korea Information Security Agency (KISA), from July 2002 to January 2004. He is currently a Professor with the Department of Mathematics, University of Seoul. His research interests include cryptography, symmetric cryptosystems, hash functions, and MACs.

SEOKHIE HONG received the M.S. and Ph.D. degrees in mathematics from Korea University, in 1997 and 2001, respectively. He was at SECURITY Technologies Inc., from 2000 to 2004. He conducted postdoctoral research with COSIC at KU Leuven, Belgium, from 2004 to 2005. He joined the Graduate School of Cyber Security with Korea University. His research interests include cryptography, public and symmetric cryptosystems, hash functions, and MACs.