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\(^4\)

\(^1\)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].

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,
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. 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 $x \in \mathbb{F}_2^n$ 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 $x = x_{n-1}x_{n-2} \cdots x_0$. We define $x \lll i$ as an operation rotating a binary vector $x$ in the left direction by $i$ bits. We denote the concatenation of the two binary vectors $x$ and $y$ by $x|y$. 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 $1^40^3$.

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

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

Let $E : \mathbb{F}_2^2 \times \mathbb{F}_2^2 \to \mathbb{F}_2^2$ be a block cipher with $k$-bit key and $n$-bit block. $c = E_k(p)$ indicates that plaintext $p \in \mathbb{F}_2^n$ is encrypted to ciphertext $c \in \mathbb{F}_2^n$ through block cipher $E$ with key $k \in \mathbb{F}_2^k$. Furthermore, $Y = E_k(X)$ implies that $Y$ is the (multi)set of ciphertexts to which the block cipher $E$ encrypts all plaintexts in the (multi)set $X$ with the key $k \in \mathbb{F}_2^k$.

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 $x$ of PIPO as an $8 \times 8$ 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 $x_{i,*}$ is $(x_{8i+7}, x_{8i+6}, \ldots, x_{8i}) = (x_{8i+7}, x_{8i+6}, \ldots, x_{8i})$. Its $j$-th column $x_{*,j}$ is defined as $(x_{i,7}, x_{i,6}, \ldots, x_{i,0}) = (x_{i,7}, x_{i,6}, \ldots, x_{i,0})$. Note that the column index starts on the right.

The key schedule of PIPO-64/128 splits a 128-bit master key $k$ into two 64-bit parts $k = k_1||k_0$. Subsequently, subkeys are defined as $sk_i = k_{i \mod 2} \oplus i$ for $0 \leq i < 13$. The key schedule of PIPO-64/256 splits a 256-bit $k$ into four
The 5-bit input \( x = (x_4, x_3, x_2, x_1, x_0) \) can be updated to the output \( S^2_S(x) \) of \( S^2_S \) as follows:

\[
\begin{align*}
x_4 &\leftarrow x_4 \oplus (x_3 \land x_0); \\
x_0 &\leftarrow x_0 \oplus x_4; \\
x_4 &\leftarrow x_4 \oplus (x_2 \lor x_1); \\
x_1 &\leftarrow x_1 \oplus x_3; \\
x_3 &\leftarrow x_3 \oplus (x_4 \lor x_2); \\
x_2 &\leftarrow x_2 \oplus (x_1 \land x_0).
\end{align*}
\]

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:

\[
\begin{align*}
(x_7, x_6, x_5, x_4, x_3) &\leftarrow S^1_S(x_7, x_6, x_5, x_4, x_3); \\
(x_2, x_1, x_0) &\leftarrow S_3(x_2, x_1, x_0); \\
t_2 &\leftarrow x_4 \oplus x_0; \\
t_0 &\leftarrow x_7 \oplus x_1; \\
t_1 &\leftarrow x_3 \oplus x_2; \\
(x_6, x_5, t_2, t_1, t_0) &\leftarrow S^2_S(x_6, x_5, t_2, t_1, t_0); \\
x_2 &\leftarrow x_2 \oplus t_0; \\
x_0 &\leftarrow x_0 \oplus t_1; \\
x_1 &\leftarrow x_1 \oplus t_2; \\
(x_7, \ldots, x_0) &\leftarrow (x_1, x_3, x_4, x_5, x_6, x_2, x_0, x_7).
\end{align*}
\]

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

**C. INTEGRAL CRYPTOANALYSIS**

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 \((\text{cca}a)\), four 4-bit values can appear with \((x_1, x_0) = (0, 0), (0, 1), (1, 0), \text{and} (1, 1)\) 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 \( E : \mathbb{F}_2^n \times \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n \) be an r-round block cipher with k-bit key and n-bit block. Let \( X \) and \( Y = E_k(X) \) be a plaintext multiset and ciphertext multiset under a key \( k \in \mathbb{F}_2^k \), respectively. If there exists any index \( i \) such that

\[ \bigoplus_{x \in X} E_k(x)_i = 0 \quad \forall k \in \mathbb{F}_2^k, \]

we say that the \( i \)-th bit variable \( y_i \) of the ciphertext is balanced, and call the transition from \( X \) to \( Y \) 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 \(X\) be a multiset whose elements take the value of \(\mathbb{F}_2^n\), and let \(k\) be an \(n\)-dimensional vector whose \(i\)-th element takes 0 or 1. When multiset \(X\) has the *conventional bit-based division property* \(D_{X,2}^m\), it satisfies the following conditions:

\[
\bigoplus_{x \in X} x^u = \begin{cases} 
\text{unknown} & \text{if } \exists k \in K \text{ s.t. } u \geq k, \\
0 & \text{otherwise.}
\end{cases}
\]  

(1)

For simplicity, the conventional bit-based division property is mentioned as a division property in the remainder of this paper. If \(k \in \mathbb{K}\) and \(k' \in \mathbb{K}\) satisfy \(k \geq k'\), we can remove \(k\) from \(\mathbb{K}\) because \(k\) 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 \(f : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2^2\) be a *copy* function, where the input \((x_0, x_0) \in \mathbb{F}_2^2\) and the output is calculated as \((x_0, x_0)\). Let \(X\) and \(Y\) be the input and output multisets of \(f\). If \(X\) has \(D_{X,2}^m\), \(Y\) has \(D_{Y,2}^m\), where \(\mathbb{K}'\) is computed for all \(k \in \mathbb{K}\) as

\[
\mathbb{K}' = \begin{cases} 
\{(0,0)\} & \text{if } k_0 = 0, \\
\{(0,1), (1,0)\} & \text{if } k_0 = 1.
\end{cases}
\]  

(2)

**b:** · **RULE 2 (and)**

Let \(f : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2^2\) be an *and* function, where the input \((x_1, x_0) \in \mathbb{F}_2^2\) and the output is calculated as \((x_1 \land x_0)\). Let \(X\) and \(Y\) be the input and output multisets of \(f\), respectively. If \(X\) has \(D_{X,2}^m\), \(Y\) has \(D_{Y,2}^m\), where \(\mathbb{K}'\) is computed for all \(k \in \mathbb{K}\) as

\[
\mathbb{K}' = \left\{ \left\lfloor \left( k_1 + k_0 \right) / 2 \right\rfloor \right\}.
\]  

(3)

**c:** · **RULE 3 (xor)**

Let \(f : \mathbb{F}_2^2 \rightarrow \mathbb{F}_2^2\) 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 \(X\) and \(Y\) be the input and output multisets of \(f\), respectively.

![Algorithm 1 Calculating DP(f, k)](image)

**Input:** The input division property \(D_{K,2}^m\) of \(f\) where \(k \in \mathbb{F}_2^n\)

**Output:** A set \(\mathbb{K}\) of vectors such that the output multiset has the division property \(D_{Y,2}^m\)

1: \(\mathbb{S} \leftarrow \{k' \mid k' \geq k\}\)
2: \(\mathbb{K} \leftarrow \emptyset\)
3: for \(u \in \mathbb{F}_2^m\) do
4: if \(\text{ANF}_{f^u} \cap \mathbb{S} \neq \emptyset\) then
5: \(\mathbb{K} \leftarrow \{u\}\)
6: end if
7: end for
8: \(\text{DP}(f, k) = \text{SizeReduce}(\mathbb{K})\)
9: return \(\text{DP}(f, k)\)

If \(X\) has \(D_{X,2}^m\), \(Y\) has \(D_{Y,2}^m\), where \(\mathbb{K}'\) is computed for all \(k \in \mathbb{K}\) as

\[
\mathbb{K}' = \{(\max(k_1, k_0))\}.
\]  

(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 \(f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n\) be a function of the S-box, where the input \(x \in \mathbb{F}_2^n\) and the output \(y \in \mathbb{F}_2^n\). Let \(X\) and \(Y\) be the input and output multisets of \(f\). If \(X\) has \(D_{X,2}^m\), \(Y\) has \(D_{Y,2}^m\), where \(\mathbb{K}'\) is computed for all \(k \in \mathbb{K}\) as

\[
\mathbb{K}' \leftarrow \text{DP}(f, k).
\]

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

\[
\{k' \mid f^k \text{ contains any term } x^u \text{ satisfying } u \geq k\},
\]

where \(f^k\) is \(\prod_{i=0}^{m-1} f_i(x)^{k_i}\). We can calculate \(\text{DP}(f, k)\) using Algorithm 1, which was introduced in [8]. As mentioned above, the redundant vectors of \(K\) 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 \(k \in \mathbb{K}\) of the division property \(D_{X,2}^m\) to \(k' \in \mathbb{K}'\) of the division property \(D_{Y,2}^m\). 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 \(f_i\) denote the \(i\)-th round function of \(E\). Assume that the input multiset to \(E\) has an initial division property \(D_{X,2}^m\), and denote the division property after \(r\)-round propagation through \(f_i\) by \(D_{Y,2}^m\). Thus, we have the following chain of division property propagations.

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

Moreover, for any vector \(k_i^*\) in \(\mathbb{K}_i\) (\(i \geq 1\)), there exists a vector \(k_{i-1}^*\) in \(\mathbb{K}_{i-1}\) such that \(k_{i-1}^*\) can propagate to \(k_i^*\).
to satisfy Rule 3 of Equation (4).

When \((k_0, k_1, \ldots, k_r) \in \mathbb{K}_0 \times \mathbb{K}_1 \times \cdots \times \mathbb{K}_r\), if \(k_{i-1}\) can propagate to \(k_i\) for all \(i \in \{1, 2, \ldots, r\}\), then we call \((k_0, k_1, \ldots, k_r)\) an \(r\)-round division trail.

Definition 3 implies that the set of last vectors of all \(r\) round division trails starting with \(K\) 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 \(K\). 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 \(f : \mathbb{F}_2^n \rightarrow \mathbb{F}_2^m\), we should create an MILP model \(\mathcal{M}\) such that \(\mathcal{M}\) has only division trails of \(f\) as solutions. There are two requirements for it for \(a\|b \in \mathbb{F}_2^{n+m}\):

1. If \(a \xrightarrow{\text{copy}} b\) is a division trail of \(f\), \(a\|b\) is a solution of \(\mathcal{M}\);
2. If \(a\|b\) is a solution of \(\mathcal{M}\), then \(a \xrightarrow{\text{copy}} b\) is a division trail of \(f\).

### a: MILP MODEL FOR \text{copy}

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

\[
\begin{align*}
\mathcal{M}.con &\leftarrow a_0 - b_1 - b_0 = 0; \\
\mathcal{M}.var &\leftarrow a_0, b_1, b_0 : \text{binaries}
\end{align*}
\]

\(\text{satisfy Rule 1 of Equation (2)}\).

### b: MILP MODEL FOR \text{and}

When \((a_1, a_0) \xrightarrow{\text{and}} (b_1, b_0)\) is a division trail of \(\text{and}(x_1, x_0) = x_1 \land x_0\), then the MILP model \(\mathcal{M}\) can be

\[
\begin{align*}
\mathcal{M}.con &\leftarrow b_0 \land a_1 \geq 0; \\
\mathcal{M}.con &\leftarrow b_0 - a_0 \geq 0; \\
\mathcal{M}.con &\leftarrow b_0 - a_1 - a_0 \leq 0; \\
\mathcal{M}.var &\leftarrow a_1, a_0, b_0 : \text{binaries}
\end{align*}
\]

\(\text{satisfy Rule 2 of Equation (3)}\).

### c: MILP MODEL FOR \text{xor}

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

\[
\begin{align*}
\mathcal{M}.con &\leftarrow b_1 = a_1 + a_0 - b_0 = 0; \\
\mathcal{M}.var &\leftarrow a_1, a_0, b_0 : \text{binaries}
\end{align*}
\]

\(\text{satisfy Rule 3 of Equation (4)}\).

### d: MILP MODEL FOR \text{S-BOX}

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

\[
\{a\|b \mid b \in DP(f, a)\}
\]

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 \text{Inequality_generator()} function in Sagemath software. Each of the two conversions is detailed in the following section when constructing MILP models for the S-box of \text{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 \text{PIPO} and compare them. Moreover, we introduce a method for exploiting the rotational symmetry of \text{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 \text{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

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

of division trails for \(S_8\). We convert \(P_8\) into the corresponding linear inequalities using the \text{Inequality_generator()} function in the Sagemath software. Specifically, the function \text{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 \(P_8\) and obtained the set

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

This gives the product-of-sum representation of \(g(x)\) as

\[
g(x) = \bigcap_{u \notin P_8} \left( \bigvee_{u_i = 0} x_i \bigvee_{u_i = 1} x_i \right).
\]

\[1\] Available at http://www.sagemath.org/
The product-of-sum representation trivially corresponds to a set of inequalities which exactly describe $P_8$ as
\[
\sum_{u=0}^{k} x_i + \sum_{u=1}^{k} (1 - x_i) \geq 1 \mid u \notin P_8
\]

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 $M^{\text{QM}}$ for $S_8$.

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

Finally, considering the structure of $S_8$, we derive an MILP model $M^{\text{struct}}$ for $S_8$ from the sets
\[
P_3 = \bigcup_{a \in S^3_2} \{a\|b \mid b \in DP(S_3, a)\}
\]
\[
P_5^{1} = \bigcup_{a \in S^3_2} \{a\|b \mid b \in DP(S^1_2, a)\}
\]
\[
P_5^{2} = \bigcup_{a \in S^3_2} \{a\|b \mid b \in DP(S^2_2, a)\}
\]
of division trails for $S_3$, $S^1_2$, and $S^2_2$ respectively. As explained in Section II-B and described in Fig. 2, $S_8$ is constructed with an unbalanced-bridge structure with $S_3$, $S^1_2$ and $S^2_2$. 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 $\text{copy}$ and $\text{xor}$ operations explained in Section II-E to obtain $M^{\text{struct}}$ for $S_8$.

4) COMPARISON OF MILP MODELS FOR $S_8$

$M^{\text{H-repre}}$ and $M^{\text{QM}}$ allow accurate analysis of $S_8$. However, $M^{\text{H-repre}}$ 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. $M^{\text{QM}}$ 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, $M^{\text{struct}}$ does not guarantee analysis as accurate as $M^{\text{H-repre}}$ and $M^{\text{QM}}$ because it does not cover monomials cancelled through XORs in the ANF of $S_8$. For some input division property $k$, $M^{\text{struct}}$ occurs a larger unknown set\(^2\) of Equation (1)
\[
\{u \geq k' \mid k\|k' \text{ is feasible in } \text{out in } M^{\text{struct}}\}
\]
(5)

than $M^{\text{H-repre}}$ and $M^{\text{QM}}$. Nevertheless, we proceeded to obtain $M^{\text{struct}}$ 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 $M^{\text{QM}}$ is included in that of $M^{\text{struct}}$ for any input division property $k$. This implies some integral distinguishers may not be found in the model $M^{\text{struct}}$.

### Table 3. Comparison of MILP models for $S_8$

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

**B. ROTATIONAL SYMMETRY OF PIPO**

We can find an MILP model $M$ for $r$ rounds of PIPO, based on the analysis given in Section III-A. Then, we solve $M$ to construct any $r$-round division trail $(a^0, a^1, \ldots, a^r)$. To obtain an integral distinguisher from the trail, we need to start the trail with $k$ of the division property $D_k^8$ for the plaintext multiset. We can achieve this by adding the following constraints to $M$.

\[
da^0_j = k_j \quad \text{for } j = 0, 1, \ldots, n - 1.
\]

Moreover, we should set $w(k) = 63$ 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 $k$ be an $8 \times 8$ array of 64-bit binary vector. Let $\tau(k)$ be the 64-bit vector in which each row of $k$ is right rotated by one bit. For the round function $f$ of PIPO, if $k \xrightarrow{\tau} k'$ is a division trail of $f$, $\tau(k) \xrightarrow{f} \tau(k')$ is a division trail of $f$ as well.

**Proof:** We omit the Key-XOR operation considering the components of the round function $f$ of PIPO because it does not have any impact on the division property. Subsequently, a division trail $k \xrightarrow{f} k'$ is regarded as $k \xrightarrow{\tau} k' \xrightarrow{R} k'$. First, we demonstrate that $\tau(k) \xrightarrow{S} \tau(k')$. We have $\tau(k)_{s, j} = k_{s, (j - 1) \mod 8}$ and $R(k')_{s, j} = k'_{8s, (j - 1) \mod 8}$. Because $k \xrightarrow{S} k'$ is a division trail of the S-layer, $k_{s, (j - 1) \mod 8}$ can be propagated to $k'_{8s, (j - 1) \mod 8}$ through the S-box $S_6$. Therefore, we have $\tau(k)_{s, j} \xrightarrow{8} \tau(k ')_{s, j}$ for $0 \leq j \leq 7$, and $(k) \xrightarrow{S} \tau(k')$.

Finally, we demonstrate that $\tau(k') \xrightarrow{R} \tau(k')$. It is trivial from the assumption $k \xrightarrow{R} k'$, because both $\tau$ and the R-layer belong to rotation operations on $8 \times 8$ 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 $k$ covers trails starting with $\tau(k), \tau^2(k), \ldots, \tau^7(k)$.

### 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 $L_{\text{out}} \circ \odot L_{\text{in}}$ instead of a block cipher $E : \mathbb{F}_2^n \times \mathbb{F}_2^n \rightarrow \mathbb{F}_2^n$, where $L_{\text{in}}$ and $L_{\text{out}} \in GL_n(\mathbb{F}_2)$ and where we regard $E$ as a nonlinear permutation on $\mathbb{F}_2^n$. A block cipher with a randomly selected secret key over $\mathbb{F}_2^n$. Generally, their method
finds an extended integral distinguisher. This is defined as Definition 5.

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

\[
\bigoplus_{y \in Y} v \cdot y = \bigoplus_{x \in X} v \cdot E_k(x) = 0,
\]

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

### B. Linear Transformations on Input and Output

We consider \( L_{in} \) only as a concatenation of eight \( 8 \times 8 \) matrices \( L_{in}^j \in \text{GL}_8(\mathbb{F}_2) \) for \( 0 \leq j < 8 \), because it is computationally impossible to try all the \( 64 \times 64 \) binary linear matrices. Similarly, we consider \( L_{out} \) only as a concatenation of eight \( 8 \times 8 \) matrices \( L_{out}^j \in \text{GL}_8(\mathbb{F}_2) \) for \( 0 \leq j < 8 \).

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

Considering the rotational symmetry of \( \text{PIPO} \), we can force the initial division property \( D_8^{64} \) to have a single zero bit at the least significant position of \( k \). In other words, we assume that the initial division property is \( D_8^{64} \) and that the initial set is \( (a \cdots a c) \) 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 \( D_8^{64} \), for any invertible \( f : \mathbb{F}_2^n \to \mathbb{F}_2^n \), the output division property is \( D_8^{64} \).

**Proof:** Assume \( f(x) = y \). According to Proposition 1 in [13], \( \text{deg}(y^j) = n \) only when \( u \) is an \( n \)-bit all-one vector \( 1^n \). Therefore, \( \text{DP}(f, 1^n) = \{ 1^n \} \), and the output division property is \( D_8^{64} \). \( \Box \)

Now, we can consider only \( L_{in}^0 \) with the given input division property \( D_8^{64} \). Because \( \text{DP}(S_8 \circ L_{in}^0, 1^n) \) depends only on linear combinations of bits that become constant, we can classify \( 8 \times 8 \) invertible matrices into \( 2^8 - 1 \) classes, in which each matrix instantiating \( L_{in}^0 \) has the same \( \text{DP}(S_8 \circ L_{in}^0, 1^n) \).

For \( D_8^{64} \), we define \( \text{Succ}(k) := \{ u \in \mathbb{F}_2^n | u \geq k \} \) for \( k \in \mathbb{F}_2^k \) and \( \text{Succ}(\mathbb{F}_2^k) := \bigcup_{k \in \mathbb{F}_2^k} \text{Succ}(k) \). Let \( D_8^{64} \) and \( D_8^{64} \) 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 \( \text{Succ}(\mathbb{F}_2^n) \subseteq \text{Succ}(\mathbb{F}_2^n) \), 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.

### V. Integral Distinguishers and Attacks

#### A. Searching for Distinguishers

We attempted two ways to search for distinguishers by constructing two MILP models for \( \text{PIPO} \) 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 Threadripper 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 \( \text{PIPO} \) 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 \( \text{PIPO} \) structure. Both search approaches did not find any integral distinguishers for more than 6 rounds of \( \text{PIPO} \).

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:

\[
x_{i,0}^0 \oplus x_{j,0}^0
\]

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

\[
\mathcal{B}_0 = \{ x_{0,1+i}^6 \oplus x_{i,0}^6 \oplus x_{6,2+i}^6, x_{6,2+i}^6 \oplus x_{i,1+i}^6 \oplus x_{6,3+i}^6, x_{6,3+i}^6 \oplus x_{0,2+i}^6 \oplus x_{6,4+i}^6, x_{6,4+i}^6 \oplus x_{1,3+i}^6 \oplus x_{6,5+i}^6, x_{6,5+i}^6 \oplus x_{1,4+i}^6 \oplus x_{6,6+i}^6, x_{6,6+i}^6 \oplus x_{1,5+i}^6 \oplus x_{6,7+i}^6, x_{6,7+i}^6 \oplus x_{1,6+i}^6 \oplus x_{6,0}^6 \}.
\]

### Table 4. Search results for 6-round integral distinguishers of \( \text{PIPO} \).

| Model for \( S_{8r} \) | # 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 \( D_8^8 \circ L_{in}^0 \) and \( D_8^8 \circ L_{in}^1 \) for \( S_8 \circ L_{in}^1 \), \( 1 \leq i \leq 7 \). Then, through the loop covering Lines 5 - 31, it searches for balanced bits in the \((r - 1)\)-round output on the division property after the first round. \( \mathbb{U} \) is the set of all \( v \), such that the parity of \( x^v \) is unknown for the output \( x \) of the \((r - 1)\)-th round. In Lines 7 - 24, we use the MILP model \( \mathcal{M} \) for \( r - 2 \) rounds of \( \text{PIPO} \) to collect all possible entries of \( \mathbb{U} \). In Lines 25 - 30, it computes \( \text{ANF}_{v_{out} \circ S_{8r}} \) for the \( j \)-th S-box in the \((r - 1)\)-round, and checks whether \( \text{ANF}_{v_{out} \circ S_{8r}} \) contains monomials whose parities are unknown. If \( \text{ANF}_{v_{out} \circ S_{8r}} \) contains no such monomials, \( v_{out} \circ S_{8r} \) is a balanced bit of an \((r - 1)\)-round extended integral distinguisher. Consequently, all balanced bits after \( r \) rounds are stored in \( \mathbb{S} \) in the form of \( (j, v_{out}) \).
Algorithm 2 Extended Integral Distinguisher Search

Input: MILP model $\mathcal{M}$ for $r - 2$ rounds of PIPO, linear transformation $L^0_{in}$
Output: A set $\mathcal{S}$ of $r$-round extended integral distinguishers

1: $\mathcal{S} \leftarrow \emptyset$
2: $\mathcal{K}_{in} \leftarrow DP(S_8 \circ L^0_{in}, I_{17}^0)$
3: $\mathcal{K}_1 \leftarrow \{ (1^7 k_7 \parallel 1^7 k_6 \parallel \ldots \parallel 1^7 k_0) \in \mathbb{F}_2^{64} \mid k = (k_7, k_6, \ldots, k_0) \in \mathcal{K}_{in} \}$
4: $\mathcal{K}_1 \leftarrow R(\mathcal{K}_1)$
5: for $j = 0, 1, \ldots, 7$ do
6: $\cup \leftarrow \emptyset$
7: for $k \in \mathcal{K}_1$ do
8: for $v \in \mathbb{F}_2^8 \setminus \{0\}$ do
9: if $v \notin \cup$ then
10: $\mathcal{M}' \leftarrow \mathcal{M}$
11: $\mathcal{M}'.con \leftarrow a^0 = k$
12: for $i = 0, 1, \ldots, 7$ do
13: if $i = j$ then
14: $\mathcal{M}'.con \leftarrow a^{r-2}_{s,j} = v$
15: else
16: $\mathcal{M}'.con \leftarrow a^{r-2}_{s,j} = 0$
17: end if
18: end for
19: if $\mathcal{M}'$ has any feasible solution then
20: $\cup \leftarrow \{v' \mid v' \geq v\}$
21: end if
22: end if
23: end for
24: end for
25: for $v_{out} \in \mathbb{F}_2^8 \setminus \{0\}$ do
26: Compute $\text{ANF}_{v_{out}, S_8}$
27: if $\text{ANF}_{v_{out}, S_8} \cap \cup = \emptyset$ then
28: $\mathcal{S} \leftarrow (j, v_{out})$
29: end if
30: end for
31: end for
32: return $\mathcal{S}$

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

- **Data Collection Phase** - Choose $2^{63}$ plaintexts, $p$’s in which $p_{S_6}$ is fixed as constant, and obtain the corresponding ciphertexts, $c$’s.
- **Key Filtering Phase** - Guess a 64-bit value of the last subkey $sk^8$ and do the followings:
  1. Perform one-round decryption for all ciphertexts with the guessed value of $sk^8$,
     $$x^7 = S^{-1}(R^{-1}(c \oplus sk^8)).$$
  2. Consider $x^6 = S^{-1}(R^{-1}(x^7) \oplus rk^7)$ instead of $x^6 = S^{-1}(R^{-1}(x^7 \oplus sk^7))$, and let $y = R^{-1}(x^7)$. For all $2^{63}$ values of $x^7$, count each number $t_i[y_{s,i}]$ of times $y_{s,i}$ appear for $i \in \{0, 1, 6, 7\}$.
  3. Guess 4-byte values of $(rk^7_6, rk^7_3, rk^7_1, rk^7_0)$ and do the followings:
     a) For all $2^{63}$ values of $y$, compute each parity of $\alpha_i$ for $0 \leq j \leq 3$ in Table 5 considering $t_i[y_{s,i}]$ for $i \in \{0, 1, 6, 7\}$ (See Fig. 3).
     b) If one of the $\alpha_i$’s parities is odd, exclude the guessed values of $sk^8$ and $(rk^7_6, rk^7_3, rk^7_1, rk^7_0)$ from the space of key candidates.
- **Exhaustive Searching Phase** - Perform an exhaustive search for the 128-bit key $k$ over the key space.
Similarly, with $0 \leq i < 8$, 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:

$$B_1 = \{x^6_{5,i},$$

$$x^6_{6,7+i},$$

$$x^6_{6,0} \oplus x_{1,7+i} \oplus x_{6,1+i},$$

$$x^6_{0,1+i} \oplus x_{1,6+i} \oplus x_{6,2+i},$$

$$x^6_{0,2+i} \oplus x_{1,1+i} \oplus x_{6,3+i},$$

$$x^6_{0,3+i} \oplus x_{1,2+i} \oplus x_{6,4+i},$$

$$x^6_{0,4+i} \oplus x_{1,3+i} \oplus x_{6,5+i},$$

$$x^6_{0,5+i} \oplus x_{1,4+i} \oplus x_{6,6+i},$$

$$x^6_{0,6+i} \oplus x_{1,5+i} \oplus x_{6,7+i},$$

$$x^6_{0,7+i} \oplus x_{1,6+i} \oplus x_{6,8+i}\}.$$

Except for $x^6_{5,i}$ and $x^6_{6,7+i}$, the distinguishers with the balanced bit information in $B_1$ can be found by searching with both $M^}\text{OM}$ and $M^}\text{struct}$.

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

We use four 6-round integral distinguishers, underlined in $B_1$, 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 $p = (p_{63}, \ldots, p_1, p_0)$. 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 $sk^8$ and guess four bytes of $rk^7$, where $rk^7 = R^{-1}(sk^8)$. Table 5 lists the balanced bits in the output and the key bytes of $rk^7$ 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 $y_{x,i}$ for $i \in \{0, 1, 6, 7\}$ and a 4-byte guessed key $(rk^7, rk^6, rk^5, rk^4)$ as Fig. 3 describes.

We expect that the key space can be reduced by the ratio of 2$^{-4}$ after Key Filtering Phase because the $\alpha_i$ for $1 \leq i \leq 3$ 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 $rk^7$ and $sk^8$ as in the attack on 8-round PIPO-64/128, and additionally guess the whole bits of $sk^9$ and $sk^{10}$. Therefore, the time complexity of Key Filtering Phase is estimated as $2^{253.3} = 2^{63} \times 2^{64} \times 3/10 10$-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 distinguishers 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 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. Abdulkhalik, Y. Sasaki, Y. Todo, M. Tolba, and A. M. Youssef, “MILP modeling for (large) S-boxes to optimize probability of differential characteristics,” JACR 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.

JESEONG 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 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 interest includes symmetric cryptography.

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.

Available at https://github.com/SUNYEOP/PIPO_DP