Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.31, NO.5, Oct. 2021 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2021.31.5.875

경량 블록 암호 PIPO의 MILP-Aided 디비전 프로퍼티 분석 및 인테그랄 공격*

김 제 성,[1†] 김 성 겸,[1] 김 선 엽,[1] 홍 득 조,[2‡] 성 재 철,[3] 홍 석 희[4]
[1,4]고려대학교 (대학원생, 교수), [2]전북대학교 (교수), [3]서울시립대학교 (교수)

MILP-Aided Division Property and Integral Attack on Lightweight Block Cipher PIPO*

Jeseong Kim,[1†] Seonggyeom Kim,[1] Sunyeop Kim,[1]
Deukjo Hong,[2‡] Jaechul Sung,[3] Seokhie Hong[4]
[1,4]Korea University (Graduate student, Professor),
[2]Chonbuk National University (Professor), [3]University of Seoul (Professor)

요 약

본 논문에서는 경량 블록 암호 PIPO에 대한 인테그랄 구별자(integral distinguisher)을 탐색한 결과를 통해 8-라운드 PIPO-64/128에 대한 키 복구 공격을 수행한다. ICISC 2020에서 제안된 경량 블록 암호 PIPO는 고차 마스킹 구현을 고려한 설계를 통해 부채널 공격에 대한 저항성을 갖는 효율적인 구현이 가능하다. 동시에 차분 분석, 선형 분석 등의 다양한 분석법을 적용하여 PIPO의 안전성을 보였다. 그러나 인테그랄 공격에 대해, 5-라운드 이상의 인테그랄 구별자가 존재하지 않을 것이라고 제안되었을 뿐 인테그랄 공격에 대한 안전성 분석은 현재까지 수행된 바 없다. 본 논문에서는 MILP 기반 Division Property를 통해 PIPO에 대한 인테그랄 구별자를 탐색하는 방법을 제시하고, 기존의 결과와 달리 6-라운드 인테그랄 구별자가 존재함을 보인다. 뿐만 아니라, PIPO의 라운드 함수 구조를 활용하여 입출력에 대한 선형 변환을 고려하는 인테그랄 구별자 탐색 방법을 통해 총 136개의 6-라운드 인테그랄 구별자를 제시한다. 마지막으로, 획득한 6-라운드 인테그랄 구별자 중 4개를 이용하여 $2^{124.5849}$의 시간 복잡도와 2^{93}의 메모리 복잡도를 가지는 8-라운드 PIPO-64/128 키 복구 공격을 제안한다.

ABSTRACT

In this paper, we search integral distinguishers of lightweight block cipher PIPO and propose a key recovery attack on 8-round PIPO-64/128 with the obtained 6-round distinguishers. The lightweight block cipher PIPO proposed in ICISC 2020 is designed to provide the efficient implementation of high-order masking for side-channel attack resistance. In the proposal, various attacks such as differential and linear cryptanalyses were applied to show the sufficient security strength. However, the designers leave integral attack to be conducted and only show that it is unlikely for PIPO to have integral distinguishers longer than 5-round PIPO without further analysis on Division Property. In this paper, we search integral distinguishers of PIPO using a MILP-aided Division Property search method. Our search can show that there exist 6-round integral distinguishers, which is different from what the designers insist. We also consider linear operation on input and output of

Received(09. 23. 2021), Modified(10. 08. 2021), Accepted(10. 08. 2021)
* 본 연구는 2019년도 정부(과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (No. 2017-0-00520, (ICT 기초연구실) SCR-Friendly 대칭키 암호 및 응용모드 개발)
† 주저자, wptjd1012@korea.ac.kr
‡ 교신저자, deukjo.hong@jbnu.ac.kr(Corresponding author)

distinguisher, respectively, and manage to obtain totally 136 6-round integral distinguishers. Finally, we present an 8-round PIPO-64/128 key recovery attack with time complexity $2^{124.5849}$ and memory complexity of 2^{93} with four 6-round integral distinguishers among the entire obtained distinguishers.

Keywords: Integral Attack, PIPO, Division Property, MILP Modeling, Linear combination

I. 서 론

경량 블록 암호 PIPO[1]는 ICISC 2020에서 제안된 SPN구조의 블록 암호이다. PIPO는 한 개 블록에 대한 비트 슬라이스 구현이 가능하여 소프트웨어 구현에 좋은 성능을 제공한다. 또한, 부채널 공격 저항성을 위한 고차 마스킹 구현에 효율적인 Sbox를 사용하였다. PIPO 설계자들은 다양한 암호 분석 기법을 PIPO에 적용하여 각 공격에 대한 안전성을 제시하였다. 그러나 PIPO에 대한 인테그랄 분석은 대수적 차수의 상한을 계산하여 5-라운드 이상의 인테그랄 구별자를 쉽게 찾을 수 없을 것으로 보인 결과만 [2]에서 제안하였다. 이러한 방법은 인테그랄 공격에 대한 안전성을 보이기에 적합하지 않다.

인테그랄 분석[3]은 SPN 구조의 블록 암호에 효과적인 암호 분석 이론으로 1997년 Daemen에 의해 처음 제안되었다. 이 분석은 평문을 멀티셋 단위로 활용하며 해당 멀티셋을 암호화한 암호문을 모두 더하여(xor-sum) 랜덤 순열과 구별되는 인테그랄 구별자를 기반으로 수행된다. 따라서, 인테그랄 분석의 가장 핵심이 되는 것은 인테그랄 구별자를 찾는 것으로, 이를 위한 많은 연구가 진행되고 있다.

디비전 프로퍼티(division proeperty)는 Todo[4]에 의해 처음 제안된 일반화된 인테그랄 구별자이며 동시에 인테그랄 구별자 탐색 방법이다. 처음 제안 시에는 Sbox의 대수적 차수만 고려하였으나, 비트 기반 디비전 프로퍼티(bit-based division property)로 확장되며 Sbox의 대수적인 구조를 활용하는 방법이 제안되었다. 현재 디비전 프로퍼티는 이러한 비트 기반 디비전 프로퍼티로 통용되고 있다. 디비전 프로퍼티는 주어진 멀티셋에 대한 단항식의 xor-sum 값이 0 또는 unknown인 지 나타내며, 연산 이후 하나 이상의 디비전 프로퍼티로 전파되는 특징이 있다. 이를 이용해 Todo와 Morii[16]는 블록 암호 SIMON32의 14-라운드 인테그랄 구별자를 찾아내었다. Todo와 Morii의 디비전 프로퍼티 탐색 방법은 그 자체로 높은 메모리 복잡도를 갖고 있기에 블록 크기가 32-비트 이하인 블록 암호에만 적용되었다. 그러나 ASIACRYPT'16에서 Xiang el al.[5]은 이러한 탐색 복잡도 문제를 해결하기 위해 디비전 트레일(division trail) 개념을 제시하고, 이를 통해 디비전 프로퍼티 도출 방법을 제시하였다. 제시 방법은 MILP(Mixed-Integer Linear Programming) Solver를 통해 구체화되었다. 이에, Xiang et al.은 디비전 프로퍼티의 전파를 부등식으로 변환하여 MILP 모델링을 하였으며 이를 통해 블록 크기가 64-비트인 6개 경량 블록 암호의 인테그랄 구별자를 도출하였다.

Table 1. Comparison of cryptanalyses on PIPO-64/128

Cryptanalysis	Best Dist.	Key Recovery	Ref
Differential	6-Round	9-Round	[2]
Linear	6-Round	9-Round	
Impossible differential	4-Round	6-Round	
Boomerang /Rectangle	6-Round	8-Round	
Meet-in the-Middle	6-Round	6-Round	
Integral	**6-Round**	**8-Round**	**This Paper**

본 논문에서는 PIPO의 구조를 활용하여 효율적인 MILP 기반 Division Property 탐색 방법을 제시한다. PIPO의 Sbox S_8은 작은 크기의 Sbox 3개를 unbalanced-bridge로 만든 것으로 그 설계 구조가 알려져 있다. 이러한 설계 구조를 활용한 MILP 모델링($\mathcal{M}^{struct}$) 방법을 제시하며 기존 방법과 비교한다. 그리고 PIPO의 비트 순열을 통해 회전 대칭성(rotational symmetry)을 보이며, 이를 통해 인테그랄 구별자 탐색 시 고려해야 할 입력의 수를 8배 감소시킨다. 그뿐만 아니라, 디비전 프로퍼티를 통한 인테그랄 구별자 탐색을 향상시키기 위해 입출력에 대한 선형 변환 L_{in}, L_{out}까지 고려하여 탐색한다. 제시한 탐색 방법은 PIPO의 6-라운드 인테그랄 구별자 136개를 도출하였으며, 이 중 4개의 6-라운드 인테그랄 구별자를 이용하여 8-라운드 PIPO-64/128에 대한 키 복구 공격을 제시한다.

본 논문의 구성은 다음과 같다. 2장에서 본 논문에 사용된 표기 방법과 배경 지식을 설명한다. 그리고 PIPO의 MILP 모델링 방법들과 PIPO의 회전 대칭성을 3장에서 소개한다. 4장에서는 입출력에 대한 선형 변환 L_{in}, L_{out}과 이를 활용한 디비전 프로퍼티 분석 알고리즘에 대해 설명한다. 5장에서는 제안한 알고리즘을 PIPO에 적용한 결과를 제시하며 획득한 구별자를 통해 PIPO-64/128 키 복구 공격을 수행한다. 마지막으로, 6장에서 결론을 맺으며 논문을 마무리 한다.

II. 배경 지식

2.1 표기법

다음은 본 논문에서 사용된 표기법을 나타낸다.

$\boldsymbol{x}\in\mathbb{F}_2^n$	$(x_{n-1},x_{n-2},...,x_0)$를 나타내는 n-비트 벡터
$\boldsymbol{e_i},\boldsymbol{0}$	i-번째 단위 벡터와 영벡터
$\boldsymbol{x}\lll i$	n-비트 벡터 $\boldsymbol{x}$를 i만큼 좌회전한 벡터
$\mathbb{X}$	n-비트 벡터의 멀티셋
$f(\mathbb{X})$	$f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^m$에 대해, 입력 멀티셋 $\mathbb{X}$의 출력 멀티셋 즉, $f(\mathbb{X}):=\{f(\boldsymbol{x})\|\boldsymbol{x}\in\mathbb{X}\}$.
$hw(\boldsymbol{x})$	n-비트 벡터 $\boldsymbol{x}$의 해밍 웨이트, $\sum_{i=0}^{n-1}x_i$
$\boldsymbol{x}\succcurlyeq\boldsymbol{x'}$	모든 i에 대해, $x_i\geq x'_i$이 성립함을 의미
$\boldsymbol{x}\bullet\boldsymbol{y}$	두 n-비트 벡터의 내적, $\bigoplus_{i=0}^{n-1}x_i\cdot y_i$
$\boldsymbol{x^u}$	$\prod_{i=0}^{n-1}x_i^{u_i}$로 정의된 $\boldsymbol{x}$의 단항식(monomial) e.g., $\boldsymbol{u}=(0,...,0,1,1)\Rightarrow\boldsymbol{x^u}=x_1x_0$ $\boldsymbol{u}=(0,...,0,0,0)\Rightarrow\boldsymbol{x^u}=1$
ANF_f	불 함수 $f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2$의 ANF(Algebraic Normal Form)에서 계수가 1인 항의 집합 즉, $f=\bigoplus_{\boldsymbol{u}\in\mathbb{F}_2^n}\alpha_{\boldsymbol{u}}\boldsymbol{x^u}$, $\alpha_{\boldsymbol{u}}\in\mathbb{F}_2$일 때, $ANF_f:=\{\boldsymbol{u}\in\mathbb{F}_2^n\mid\alpha_{\boldsymbol{u}}=1\}$.
$D_{\mathbb{K}}^n$	n-비트 디비전 프로퍼티(division property)
$\mathbb{K}\Leftarrow\{\boldsymbol{k}\}$	집합 $\mathbb{K}$에 원소 $\boldsymbol{k}$를 추가, $\mathbb{K}\cup=\{\boldsymbol{k}\}$
$\boldsymbol{k}\xrightarrow{f}\boldsymbol{k'}$	$\boldsymbol{k}$에서 $\boldsymbol{k'}$으로 함수 f를 통해 전파 가능한 디비전 트레일
$\mathbb{K}\xrightarrow{f}\mathbb{K'}$	함수 f를 통한 집합 $\mathbb{K}$의 전파 $\mathbb{K'}:=\left\{\boldsymbol{k'}\mid\boldsymbol{k}\xrightarrow{f}\boldsymbol{k'}\text{ for }\boldsymbol{k}\in\mathbb{K}\right\}$

2.2 블록 암호 PIPO

블록 암호 PIPO는 ICISC 2020에서 제안된 SPN 구조의 블록 암호이다[1]. PIPO의 블록 크기는 64-비트이고 키 크기에 따라 PIPO-64/128와 PIPO-64/256으로 나뉘며 각각 13, 17 라운드를 가진다. PIPO는 암호화 과정 동안, 64-비트 벡터 $\boldsymbol{x}$를 8×8 크기의 비트 행렬로 본다. 본 논문에서 $\boldsymbol{x}_{i,j}$는 $\boldsymbol{x}$를 8×8 행렬로 보았을 때, i-행 j-열의 비트를 의미한다. 또, $\boldsymbol{x}_{i,*}$와 $\boldsymbol{x}_{*,j}$는 각각 8*8 행렬의 i-행과 j-열을 의미하는 8-비트 벡터이다. 즉 $0\leq i,j\leq 7$에 대해, $\boldsymbol{x}_{i,*}$와 $\boldsymbol{x}_{*,j}$는 각각 $(x_{8i+7},\cdots,x_{8i+1},x_{8i+0})$와 $(x_{56+j},\cdots,x_{8+j},x_{0+j})$을 나타낸다.

◉ **암호화.** PIPO의 라운드 함수는 비선형 연산을 의미하는 S-layer와 선형 연산을 의미하는 P-layer, 그리고 라운드 키와 상수를 Xor하는 Key-xor(K-layer)로 이루어진다. S-layer는 행렬의 각 열에 8-비트 Sbox S_8을 적용하며 P-Layer는 행렬의 각 행을 특정한 수만큼 좌회전시킨다. 이때, 블록 암호 PIPO의 8-비트 Sbox S_8는 Fig. 1의 설계 구조를 가진다.

◉ **키 스케줄.** PIPO-64/128의 라운드 키 $\boldsymbol{RK}_i$는 128-비트 마스터 키 $\boldsymbol{K}(=\boldsymbol{K}_1\|\boldsymbol{K}_0)$를 2개의 64-비트 서브키 $\boldsymbol{K}_0$와 $\boldsymbol{K}_1$로 나눈 후, $\boldsymbol{RK}_i=\boldsymbol{K}_{i\bmod 2}$을 통해 구할 수 있다. 마찬가지로 PIPO-64/256의 라운드 키 $\boldsymbol{RK}_i$는 256-비트 마스터 키 $\boldsymbol{K}(=\boldsymbol{K}_3\|\boldsymbol{K}_2\|\boldsymbol{K}_1$

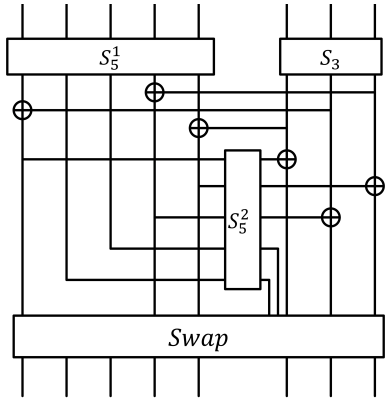

Fig. 1. Structure of S_8

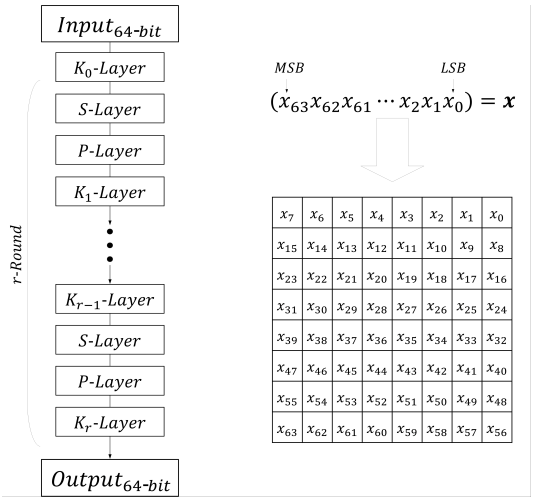

Fig. 2. Overall structure (left), intermediate state (right) of PIPO

$\|\boldsymbol{K}_0$)를 4개의 64-비트 서브키 $\boldsymbol{K}_0,\boldsymbol{K}_1,\boldsymbol{K}_2$ 그리고 $\boldsymbol{K}_3$로 나눈 후 $\boldsymbol{RK}_i=\boldsymbol{K}_{i \bmod 4}$을 통해 구한다.

2.3 Integral Cryptanalysis and Division Property

2.3.1 Integral Cryptanalysis

인테그랄 공격은 블록 암호 Square의 안전성 평가를 위해 처음 제안되었으며[3], 이후 Knudsen과 Wagner에 의해 형식화된 블록 암호 공격법이다[7]. 인테그랄 공격은 인테그랄 구별자를 찾는 것으로부터 시작한다.

모든 값이 가능한 비트를 활성 비트(a), 상수인 비트를 상수 비트(c)로 표기한다. 예를 들어 $(ccaa)$는 0~1번째 비트에서 모든 값이 가능하지만 2~3번째 비트는 상수인 멀티셋들을 지칭할 수 있다. 즉, $(ccaa)$를 만족하는 가장 작은 멀티셋은 집합 $\{\boldsymbol{x}\in\mathbb{F}_2^4 \mid x_3,x_2 : constant\}$이다. 이때, 인테그랄 구별자는 활성 비트와 상수 비트로 표현 가능한 멀티셋 $\mathbb{X}$을 이용하여 xor-sum을 계산했을 때, 그 값이 0인 비트의 위치를 활용한다. 인테그랄 구별자에 대한 구체적인 정의는 다음과 같다.

정의 1 (인테그랄 구별자). r-라운드 블록 암호 $E_{\boldsymbol{sk}}:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^n$의 평문과 암호문의 멀티셋을 각각 $\mathbb{X}$, $\mathbb{Y}(=E_{\boldsymbol{sk}}(\mathbb{X}))$라 하자. 이때, 임의의 키 $\boldsymbol{sk}$에 대해 조건 (1)을 만족하는 i가 있다면 $(\mathbb{X},i)$를 블록 암호 $E_{\boldsymbol{sk}}$의 ***r-라운드 인테그랄 구별자***라 한다. 또한, 암호문의 i-번째 비트 y_i를 ***밸런스 비트***(b)라고 한다.

$$\bigoplus_{\boldsymbol{y}\in\mathbb{Y}} y_i=\bigoplus_{\boldsymbol{x}\in\mathbb{X}} E_{\boldsymbol{sk}}(\boldsymbol{x})_i=0 \tag{1}$$

블록 암호의 인테그랄 구별자가 m개 있다고 가정했을 때, 랜덤 순열이 이러한 특성을 만족할 확률은 2^{-m}이 된다. 그러므로 인테그랄 구별자는 블록 암호와 랜덤 순열을 구별하는 구별 공격(distinguish attack)에도 바로 사용할 수 있다.

2.3.2 Division Property & Propagation Rule

디비전 프로퍼티는 EUROCRYPT'15에서 새롭게 제안된 성질이다[4]. 이후 디비전 프로퍼티는 비트 기반 디비전 프로퍼티로 일반화되었으며 본 논문의 디비전 프로퍼티는 모두 비트 기반 디비전 프로퍼티를 칭한다. 디비전 프로퍼티는 멀티셋 $\mathbb{X}$에서 정의되며 단항식을 xor-sum이 0인 공간과 unknown인 공간으로 구분하는 성질이다. 구체적인 정의는 다음과 같다.

정의 2. $\mathbb{F}_2^n$ 원소들의 멀티셋 $\mathbb{X}$와 집합 $\mathbb{K}$에 대해, 멀티셋 $\mathbb{X}$가 다음 조건을 만족하면 ***디비전 프로퍼티*** $D_{\mathbb{K}}^n$를 가진다고 한다.

$$\bigoplus_{\boldsymbol{x}\in\mathbb{X}} \boldsymbol{x}^{\boldsymbol{u}} = \begin{cases} unknown & \text{if } \exists \boldsymbol{k}\in\mathbb{K}\ s.t.\ \boldsymbol{u}\succcurlyeq\boldsymbol{k} \\ 0 & otherwise \end{cases} \quad (2)$$

만약 $\boldsymbol{k}\succcurlyeq\boldsymbol{k}'$를 만족하는 $\boldsymbol{k}$와 $\boldsymbol{k}'$가 모두 $\mathbb{K}$에 존재할 경우, 벡터 $\boldsymbol{k}$는 디비전 프로퍼티의 조건 (2)에 영향을 주지 않으므로 $\mathbb{K}$에서 제거해도 무방하다. 이런 불필요한 벡터 $\boldsymbol{k}$를 제거하는 것을 $SizeReduce$라고 한다.

디비전 프로퍼티는 각 연산을 통과하면서 하나 이상의 디비전 프로퍼티로 전파가 가능하다. 이러한 전파에 대한 규칙을 전파 규칙(propagation rule)이라 한다. 일반적으로 연산의 영향을 받는 비트들만 고려하여 전파되며 다른 비트들은 동일한 값으로 전파된다. 또한, 비트의 위치만 바꾸는 비트 순열의 경우 디비전 프로퍼티에서도 위치를 바꾼 값으로 전파되며 비밀키 또는 상수 Xor의 경우 디비전 프로퍼티에 영향을 주지 않는다.

◉ **함수 f에 대한 전파 규칙**[5]. 특정 연산의 함수 $f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^m$에 대한 디비전 프로퍼티의 전파는 **DPT**(Division Property Table)를 통해 표 형식으로 나타낼 수 있다. DPT의 $\boldsymbol{k}$행은 입력 디비전 프로퍼티가 $D_{\{\boldsymbol{k}\}}^n$일 때 출력 디비전 프로퍼티 $D_{\mathbb{K}'}^n$의 $\mathbb{K}'$를 나타내며 **알고리즘 1**을 통해 구할 수 있다. 따라서, $f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^m$의 입력 디비전 프로퍼티 $D_{\mathbb{K}}^n$로부터 출력 디비전 프로퍼티 $D_{\mathbb{K}'}^m$를 다음과 같이 구할 수 있다.

Algorithm 1. Calculating $DPT_f[\boldsymbol{k}]$

Input : Input Division Property $D_{\{\boldsymbol{k}\}}^n$ of $f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^m$

Output : A set $\mathbb{K}$ of output Division Property $D_{\mathbb{K}}^m$

1. $\mathbb{S}=\{\boldsymbol{k}' \mid \boldsymbol{k}'\succcurlyeq\boldsymbol{k}\}$.
2. $\overline{\mathbb{K}}=\varnothing$
3. **for** $\boldsymbol{u}\in\mathbb{F}_2^m$ **do**
4. if $ANF_{f^{\boldsymbol{u}}}\cap\mathbb{S}\neq\varnothing$ then
5. $\overline{\mathbb{K}}\Leftarrow\{\boldsymbol{u}\}$
6. $\mathbb{K}=SizeReduce(\overline{\mathbb{K}})$
7. **Return** $\mathbb{K}$

Fig. 3. Algorithm to calculate $DPT_f[\boldsymbol{k}]$

모든 $\boldsymbol{k}\in\mathbb{K}$에 대해, $\mathbb{K}'\Leftarrow DPT_f[\boldsymbol{k}]$
(단, $DPT_f[\boldsymbol{k}]$는 f의 DPT의 $\boldsymbol{k}$행을 의미한다.)

따라서, 모든 비밀키 $\boldsymbol{sk}$에 대한 블록암호 $E_{\boldsymbol{sk}}$의 DPT의 분석은 블록암호의 입/출력 디비전 프로퍼티를 정확히 도출해 준다. 그러나, 블록암호 $E_{\boldsymbol{sk}}$의 입출력 크기와 비밀키 $\boldsymbol{sk}$ 개수로 인해 해당하는 모든 DPT의 분석은 불가능하다. 이에 따라, 차분분석과 선형분석의 구별자 구성방법과 유사하게 블록암호의 구성요소들의 디비전 프로퍼티 전파를 분석하여 모든 비밀키 $\boldsymbol{sk}$에 대해 만족하는 (즉, 각 라운드 키가 어떤 값이 오더라도 만족하는) 인테그랄 구별자를 획득하는 방법이 일반적이다.

디비전 프로퍼티의 전파는 결국 $\boldsymbol{k}\in\mathbb{K}$에서 $\boldsymbol{k}'\in\mathbb{K}'$의 대응으로 볼 수 있으며 연속된 디비전 프로퍼티의 전파를 하나의 체인으로 나타낼 수 있다. [5]에서는 이러한 체인을 디비전 트레일이라 정의하였으며 그 정의는 다음과 같다.

정의 3 (디비전 트레일)[5]. 블록 암호 $E_{\boldsymbol{sk}}$의 i-번째 라운드 함수를 f_i, i-라운드 후의 디비전 프로퍼티를 $D_{\mathbb{K}_i}^n$라고 하자. 즉, 초기 디비전 프로퍼티가 $D^{n\{\boldsymbol{k}\}}$라고 가정했을 때 다음의 체인을 나타낸다.

$$\{\boldsymbol{k}\} := \mathbb{K}_0 \xrightarrow{f_1} \mathbb{K}_1 \xrightarrow{f_2} \mathbb{K}_2 \xrightarrow{f_3} \cdots \xrightarrow{f_r} \mathbb{K}_r \quad (3)$$

추가적으로, $\boldsymbol{k}\xrightarrow{f_i}\boldsymbol{k}'$는 함수 f_i의 입출력 디비전 프로퍼티가 $D_{\{\boldsymbol{k}\}}^n, D_{\mathbb{K}'}^n$일 때 $\boldsymbol{k}'\in\mathbb{K}'$임을 뜻하며 ***전파 가능***하다고 한다. 만약 $(\boldsymbol{k}_0,\boldsymbol{k}_1,\cdots,\boldsymbol{k}_r)\in\mathbb{K}_0\times\mathbb{K}_1\times\cdots\mathbb{K}_r$가 모든 $i\geq 1$에 대해 $\boldsymbol{k}_{i-1}\xrightarrow{f_i}\boldsymbol{k}_i$를 만족하면, $(\boldsymbol{k}_0,\boldsymbol{k}_1,\cdots,\boldsymbol{k}_r)$를 r-라운드 ***디비전 트레일***이라 한다.

◉ **암호 구성요소의 디비전 트레일**. 함수 $f:\mathbb{F}_2^n\rightarrow\mathbb{F}_2^m$에 대해, 만약 $\boldsymbol{b}\in DPT_f[\boldsymbol{a}]$를 만족하면 $\boldsymbol{a}\xrightarrow{f}\boldsymbol{b}$는 함수 f의 디비전 트레일이다. 또한, copy, xor 함수 $copy(x_0)=(x_0,x_0)$와 $xor(x_1,x_0)=(x_1\oplus x_0)$에

대한 디비전 트레일은 다음과 같다.

Table 2. Division Trail of Copy, Xor

Rule	Operation	Division Trail
Sbox	$\boldsymbol{x} \mapsto f(\boldsymbol{x})$	$\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$ if $\boldsymbol{b} \in DPT_f[\boldsymbol{a}]$
COPY	$x \mapsto (x,x)$	$(0) \xrightarrow{copy} (0,0)$ $(1) \xrightarrow{copy} (1,0)$ or $(0,1)$
XOR	$(x,y) \mapsto x \oplus y$	$(0,0) \xrightarrow{xor} (0)$ $(1,0)$ or $(0,1) \xrightarrow{xor} (1)$

2.3.3 MILP-Aided Division Property

선형 계획법은 주어진 선형 제약식들을 만족시키면서 선형 목적함수를 최적화하는 문제이다. MILP(Mixed-Integer Linear Programming)은 선형 계획법의 변수 중 일부 또는 전체가 정수형인 문제를 의미한다. 암호 분석 분야에서 MILP는 [8]에서 활성 Sbox의 최소 개수를 구하기 위해 처음 사용되었으며 이후, 다양한 암호 분석 방법에 성공적으로 활용되었다[5, 9-12]. MILP 모델 $\mathcal{M}$은 변수, 제약식 그리고 목적함수로 구성되며 각각 $\mathcal{M}.var$, $\mathcal{M}.con$, $\mathcal{M}.obj$으로 표기한다.

MILP를 활용하여 디비전 프로퍼티를 구하기 위해선 각 연산을 모델링하는 과정을 거쳐야 한다. 연산 $f:\mathbb{F}_2^n \to \mathbb{F}_2^m$를 모델링 하는 것은 f의 디비전 트레일만을 해로 갖는 MILP 모델 $\mathcal{M}$을 구하는 과정으로 이는 다음 2가지 조건으로 나타낼 수 있다.

◉ MILP 모델링 조건

1) $\boldsymbol{a} \| \boldsymbol{b} \in \mathbb{F}_2^{n+m}$에 대해, 만약 $\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$가 함수 f의 디비전 트레일이면 $\boldsymbol{a} \| \boldsymbol{b}$는 MILP 모델 $\mathcal{M}$의 해이다.

2) 만약 $\boldsymbol{a} \| \boldsymbol{b}$가 MILP 모델 $\mathcal{M}$의 해이면 $\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$가 함수 f의 디비전 트레일이다.

앞서 소개한 COPY, XOR 연산에 대한 MILP 모델링은 다음과 같이 나타낼 수 있다.

Copy 모델링. $(a_0) \xrightarrow{copy} (b_1,b_0)$를 함수 $f(x_0) = (x_0,x_0)$의 디비전 트레일이라 하자. 이때, 함수 f를 모델링한 MILP 모델 $\mathcal{M}$은 다음과 같다.

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

Xor 모델링. $(a_1,a_0) \xrightarrow{xor} (b_0)$를 함수 $f(x_1,x_0) = (x_1 \oplus x_0)$의 디비전 트레일이라 하자. 이때, 함수 f를 모델링한 MILP 모델 $\mathcal{M}$은 다음과 같다.

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

III. MILP Modeling for PIPO

본 장에서는 경량 블록 암호 PIPO의 디비전 프로퍼티 분석을 위한 Sbox 모델링과 모델 $\mathcal{M}$을 통한 디비전 프로퍼티 분석 방법 그리고 PIPO의 회전 대칭성을 소개한다.

3.1 Sbox 모델링

3.1.1 기존의 Sbox 모델링 - $\mathcal{M}^{H-repre}$, $\mathcal{M}^{QM}$

Sbox $f:\mathbb{F}_2^n \to \mathbb{F}_2^m$의 임의의 디비전 트레일 $\boldsymbol{a} \xrightarrow{f} \boldsymbol{b}$은 $n+m$-비트 벡터 $\boldsymbol{a} \| \boldsymbol{b}$로 볼 수 있으며 이러한 디비전 트레일은 집합 $P \subset \mathbb{F}_2^{n+m}$를 형성한다. Sbox 모델링은 오직 집합 P만을 표현하는 부등식을 구하는 것으로 Sagemath 패키지를 이용하는 방법($\mathcal{M}^{H-repre}$)과 원소 제거 부등식을 이용하는 방법($\mathcal{M}^{QM}$)이 있다.

Sagemath 패키지의 Inequality_generator() 메소드는 볼록 껍질의 $H-representation$을 구하는 메소드로 이를 통해 P만을 표현하는 부등식들을 구할 수 있다.

부등식 (4)은 $\mathbb{F}_2^{n+m}$의 원소 중 오직 $\boldsymbol{x}$만 성립하지 않는 부등식으로, 부등식 (4)을 $\mathcal{M}.con$에 추가함으로써 $\boldsymbol{x}$를 $\mathcal{M}$의 해 집합에서 제거할 수 있다. 따라서 P^C의 원소 수가 m일 때, m개의 부등식 통해 Sbox를 모델링할 수 있다.

$$\sum_{i\in\{j|x_j=0\}} x_i + \sum_{i\in\{j|x_j=1\}} (1-x_i) \geq 1 \quad (4)$$

그러나 대부분은 부등식의 수가 많아져 MILP Solver의 해결 속도에 영향을 주기 때문에 간소화 과정이 요구된다. 본 논문에서는 P^C를 통해 불함수 $F:\mathbb{F}_2^{n+m}\rightarrow\mathbb{F}_2$ (5)를 정의한 후, 이를 논리식 간소화 알고리즘 QM(Quine-Mccluskey)을 통해 간소화하여 S_8을 모델링하였다.

$$F(\boldsymbol{x})\begin{cases}1 \text{ if } \boldsymbol{x}\in P^C\\ 0\ else\end{cases} \quad (5)$$

3.1.2 Sbox 설계 구조를 활용한 모델링 - $\mathscr{M}^{struct}$

$\mathscr{M}^{struct}$는 Sbox의 설계 구조를 활용한다. Sbox의 설계 구조에는 Feistel, SPN등이 있으며 PIPO Sbox S_8은 3-비트 Sbox S_3와 5-비트 Sbox S_5^1, S_5^2를 사용한 Unbalanced-Bridge 구조로 설계되었다. $\mathscr{M}^{struct}$는 Sbox의 설계 구조를 활용하기 위해 먼저 내부의 작은 Sbox를 3.1.1의 방법으로 모델링한다. 이후, 블록 암호를 모델링하는 것과 같이 Copy, Xor 그리고 내부 Sbox 모델링을 설계 구조에 맞게 적용하여 MILP 모델 $\mathscr{M}$을 도출한다.

3.1.3 모델링 방법 비교

Sbox의 디비전 트레일, 즉 DPT를 활용하는 $\mathscr{M}^{H-repre}$와 $\mathscr{M}^{QM}$ 모델링 방법은 DPT의 정의에 의해 가장 정확한 분석이 가능하다. 그러나 DPT의 부등식을 만들고 이를 간소화하는 복잡도로 인해, 8-비트 이하의 Sbox에만 적용되고 있다.

Sbox의 설계 구조를 활용하는 $\mathscr{M}^{struct}$의 경우 [13]에서 제안한 구조 기반 DPT의 트레일을 나타낸다. 구조 기반 DPT는 Sbox의 ANF에서 XOR을 통해 사라지는 단항식을 고려하지 못하기 때문에 기존 DPT보다 정확성 측면에서 불리한 단점이 있다. 하지만 Copy, Xor, And 모델링은 코스트가 미미하며 내부의 Sbox만 모델링하기 때문에 전체 모델링에 걸리는 시간이 적은 장점이 있다.

Table 3. Comparison of each method when applied to S_8

Method	Modeling Time	# improper propagations
$\mathscr{M}^{H-repre}$	infeasible	-
$\mathscr{M}^{QM}$	about an hour	0
$\mathscr{M}^{struct}$	less than 1sec	892

3.2 Rotational Symmetry of PIPO

PIPO는 Sbox와 비트 순열, Key-xor로 이루어져 있기 때문에 앞선 모델링 방법을 통해 PIPO의 라운드 함수를 충분히 모델링할 수 있으며 라운드 함수를 r-번 적용하여 PIPO의 r-라운드 디비전 트레일을 구하는 MILP 모델 $\mathscr{M}$을 구성할 수 있다. 본 소문단에서는 입력 디비전 프로퍼티 설정과 회전 대칭성에 대해 설명한다.

3.2.1 입력 디비전 프로퍼티

모델링한 MILP 모델 $\mathscr{M}$의 해를 구함으로써 임의의 r-라운드 디비전 트레일 $(\boldsymbol{a}^0,\cdots,\boldsymbol{a}^r)$을 구할 수 있다. 하지만, 인테그랄 구별자를 구하기 위해선 평문 멀티셋의 디비전 프로퍼티(입력 디비전 프로퍼티) $D_{\{\boldsymbol{k}\}}^{64}$의 $\boldsymbol{k}$로부터 시작하는 디비전 트레일을 탐색해야 한다. $\mathscr{M}.con$에 다음 제약식 (6)을 추가함으로써 $\boldsymbol{k}$에서 시작하는 디비전 트레일을 구할 수 있으며, 가장 긴 인테그랄 구별자를 탐색하기 위해 $hw(\boldsymbol{k})=63$을 만족한다.

$$a_j^0 = k_j \quad \text{for } j=0,\ldots,n-1 \quad (6)$$

3.2.2 Rotational Symmetry

블록 암호 PIPO는 열에 대한 Sbox 연산과 행에 대한 쉬프트 연산으로 이루어져 있으므로 다음 정리 1을 보일 수 있다.

정리 1 (Rotational Symmetry). 64-비트 벡터 $\boldsymbol{k}$를 8×8 행렬로 보았을 때, $\boldsymbol{k}$의 각 행을 1-비트만큼 원형 좌회전시킨 벡터를 $R(\boldsymbol{k})$라 하자. PIPO의 i-번째 라운드 함수 f_i에 대해, 만약 $\boldsymbol{k}\xrightarrow{f_i}\boldsymbol{k}'$이 f_i의 디비전 트레일이라면 $R(\boldsymbol{k})\xrightarrow{f_i}R(\boldsymbol{k}')$ 또한 f_i의 디비전 트레일이다.

증명. 라운드 함수의 K-layer는 디비전 프로퍼티에 영향을 주지 않으므로 제외한다. 디비전 트레일 $\boldsymbol{k} \xrightarrow{f_i} \boldsymbol{k}'$가 $\boldsymbol{k} \xrightarrow{S} \boldsymbol{k}^s \xrightarrow{P} \boldsymbol{k}'$로 이루어졌다고 가정했을 때 다음 2개의 과정을 통해 정리 1을 증명할 수 있다.

1. $R(\boldsymbol{k}) \xrightarrow{S} R(\boldsymbol{k}^s)$

2. $R(\boldsymbol{k}^s) \xrightarrow{P} R(\boldsymbol{k}')$

$$R(\boldsymbol{k})_{*,j} \xrightarrow{S_8} R(\boldsymbol{k}^s)_{*,j}, (0 \le j \le 7) \tag{7}$$

1번을 보이기 위해선 수식 (7)가 성립함을 보여야 한다. 함수 R의 정의에 의해 $R(\boldsymbol{k})_{*,j}$와 $R(\boldsymbol{k}^s)_{*,j}$는 각각 $\boldsymbol{k}_{*,(j-1)\bmod 8}$, $\boldsymbol{k}^s{}_{*,(j-1)\bmod 8}$와 같다. $\boldsymbol{k} \xrightarrow{S} \boldsymbol{k}^s$가 S-layer의 디비전 트레일이므로 $\boldsymbol{k}_{*,(j-1)\bmod 8}$에서 $\boldsymbol{k}^s{}_{*,(j-1)\bmod 8}$로 S_8을 통해 전파 가능하다. 따라서 수식 (7)이 성립함을 보일 수 있다. 2번의 경우 P와 R이 모두 8×8 행렬을 좌회전시키는 함수이기 때문에 가정 $\boldsymbol{k}^s \xrightarrow{P} \boldsymbol{k}'$으로부터 자명하게 보일 수 있다. □

회전 대칭성을 활용할 시, $\boldsymbol{k}$에서 시작하는 트레일을 탐색하는 것으로 $R(\boldsymbol{k}), \cdots, R^7(\boldsymbol{k})$에 대한 탐색까지 포괄할 수 있으므로 탐색 시 고려해야할 입력 디비전 프로퍼티의 수를 줄일 수 있다. 여기서, $R^7(\boldsymbol{k})$는 $\boldsymbol{k}$에 함수 R을 7번 적용한 것을 의미한다.

IV. 선형 변환을 고려한 디비전 프로퍼티 분석

4.1 확장된 인테그랄 구별자

더 많은 인테그랄 구별자 탐색을 위해 Lambin et al.[6]은 새로운 방안을 제시하였다. 핵심 아이디어는 블록 암호 $E_{\boldsymbol{sk}}: \mathbb{F}_2^n \to \mathbb{F}_2^n$에 대한 인테그랄 구별자 탐색 대신, $L_{out} \circ E_{\boldsymbol{sk}} \circ L_{in}$에 대한 인테그랄 구별자 탐색을 진행함으로써 인테그랄 구별자를 확장할 수 있다는 것이다. 여기서 L_{in}, L_{out}은 각각 $GL_n(\mathbb{F}_2)$의 원소이다. 선형 변환을 고려한 디비전 프로퍼티 분석을 위해, 먼저 확장된 인테그랄 구별자를 정의한다.

정의 4 (확장된 인테그랄 구별자). r-라운드 블록 암호 $E_{\boldsymbol{sk}}: \mathbb{F}_2^n \to \mathbb{F}_2^n$의 평문과 암호문의 멀티셋을 각각 $\mathbb{X}$, $\mathbb{Y}(=E_{\boldsymbol{sk}}(\mathbb{X}))$라 하자. 이때, 임의의 키 $\boldsymbol{sk}$에 대해 조건 (8)을 만족하는 $\boldsymbol{\lambda} \in \mathbb{F}_2^n \backslash \{\boldsymbol{0}\}$가 있다면 $(\mathbb{X}, \boldsymbol{\lambda})$를 블록 암호 $E_{\boldsymbol{sk}}$의 ***r-라운드 인테그랄 구별자***라 한다. 또한, 암호문의 $\boldsymbol{\lambda} \cdot \boldsymbol{y}$를 ***밸런스 비트***$(b)$라고 한다.

$$\bigoplus_{\boldsymbol{y} \in \mathbb{Y}} \boldsymbol{\lambda} \cdot \boldsymbol{y} = \bigoplus_{\boldsymbol{x} \in \mathbb{X}} \boldsymbol{\lambda} \cdot E_{\boldsymbol{sk}}(\boldsymbol{x}) = 0 \tag{8}$$

4.2 입출력에 대한 선형 변환 L_{in}, L_{out}

4.2.1 선형 변환의 형태

블록 크기 64인 PIPO에 대해, 가능한 선형 변환의 수는 $GL_{64}(\mathbb{F}_2)$의 원소 수와 같으므로 모두 고려하는 것은 불가능하다. 따라서 S-layer의 각 Sbox에 대응되는 8개의 $L_{in}^j, L_{out}^j \in GL_8(\mathbb{F}_2)(0 \le j \le 7)$으로 L_{in}과 L_{out}을 구성한다.

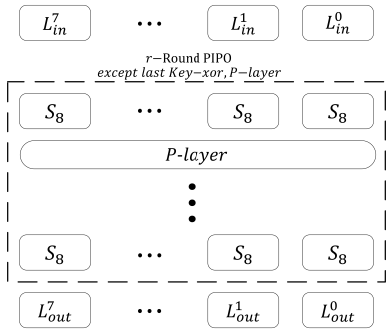

Fig. 4. Diagram of L_{in}, L_{out}

4.2.2 출력에 대한 선형 변환 L_{out}

$L_{out}^j \circ S_8 (0 \le j \le 7)$의 각 출력 비트는 특정 $\boldsymbol{\lambda}_{out} \in \mathbb{F}_2^8 \backslash \{\boldsymbol{0}\}$에 대해 $\boldsymbol{\lambda}_{out} \cdot S_8$의 형태를 띤다. 따라서 마지막 라운드의 각 Sbox에 대해 출력의 선형 변환 $\boldsymbol{\lambda}_{out} \cdot S_8$ 중 밸런스 비트가 있는지 확인하는 것으로 출력 부분의 밸런스 비트 유무를 판별할 수 있다.

4.2.3 입력에 대한 선형 변환 L_{in}

PIPO의 회전 대칭성을 고려할 경우, 입력 디비전 프로퍼티 $D^{64}_{\{\boldsymbol{k}\}}$를 첫 번째 Sbox에 0이 있는 것으로 제한할 수 있다. 또한, 비트 순열도 선형 변환 중 하나이기 때문에 입력 디비전 프로퍼티와 멀티셋을 $D^{64}_{\{(1,\cdots,1,0)\}}$와 $(a\cdots ac)$로 제한하여도 무방하다. 이렇게 제한할 경우, 8개의 $L^j_{in}\,(0\le j\le 7)$이 모두 L_{in}과 S-layer 연산 후의 디비전 프로퍼티에 영향을 주는 것은 아니다. L_{in}과 S-layer를 함께 생각하면 S_8 대신 새로운 Sbox $S_8\circ L^j_{in}$를 적용하는 것으로 볼 수 있다. 이 때, 7개의 $S_8\circ L^j_{in}\,(1\le j\le 7)$에 대해 정리 2를 적용할 수 있으며 결과적으로 L^j_{in}에 관계없이 항상 같은 디비전 프로퍼티를 갖는다. 뿐만 아니라, L^0_{in}이 가역행렬이고 입력 멀티셋이 오직 1-비트만 상수이므로, L^0_{in}을 같은 디비전 프로퍼

Algorithm 2. Extended Integral Distinguisher Search

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

1. $\mathbb{S}=\varnothing$
2. $\mathbb{K}_{in} = DPT_{S_8\,\bullet\, L^0_{in}}[11111110]$
3. $\mathbb{K}_1 = \{(1111111k_7 \,||\, \cdots \,||\, 1111111k_0)\in\mathbb{F}_2^{64} \mid \boldsymbol{k}\in\mathbb{K}_{in}\}$
4. $\mathbb{K}_1 = P(\mathbb{K}_1)$ //P : P-layer funtion
5. for j from 0 to 7 do:
6. $\mathbb{U}=\varnothing$ //Step 1
7. for $\boldsymbol{k}$ in $\mathbb{K}_1$ do: //r-2 round trail search from 2 to r-1
8. for $\boldsymbol{\lambda}$ in $\mathbb{F}_2^8\backslash\{\mathbf{0}\}$. do:
9. if $\boldsymbol{\lambda}$ not in $\mathbb{U}$ do:
10. $\mathcal{M}'=\mathcal{M}$
11. $\mathcal{M}'.con \Leftarrow \boldsymbol{a}^0=\boldsymbol{k}$
12. $\mathcal{M}'.con \Leftarrow \boldsymbol{a}^{r-2}_{*,i}=\begin{cases}\boldsymbol{\lambda} & \text{if } i=j\\ \mathbf{0} & else\end{cases}$
13. if $\mathcal{M}'$ has not feasible solution do:
14. $\mathbb{U}\Leftarrow\{\boldsymbol{\lambda}' \mid \boldsymbol{\lambda}'\succcurlyeq\boldsymbol{\lambda}\}$
15. for $\boldsymbol{\lambda}_{out}$ in $\mathbb{F}_2^8\backslash\{\mathbf{0}\}$ do: //Step 2
16. Compute $ANF_{\boldsymbol{\lambda}_{out}\,\bullet\, S_8}$
17. if $ANF_{\boldsymbol{\lambda}_{out}\,\bullet\, S_8}\cap\mathbb{U}=\varnothing$ do:
18. $\mathbb{S}\Leftarrow(j,\boldsymbol{\lambda}_{out})$
19. **Return** $\mathbb{S}$

- **line 2-3.** Compue $(S_8\circ L^0_{in})$ DPT to calculate the division property $\mathbb{K}_1$ after 1-round.
- **line 5.** Search whether the distinguishers exist on output of r-round j-th Sbox
- **line 6.** For input $\boldsymbol{x}$ of j-th Sbox, $\mathbb{U}$ is a set of $\boldsymbol{\lambda}$ where xor-sum of $\boldsymbol{x}^{\boldsymbol{\lambda}}$ is unknown
- **line 7-14.** To distinguish xor-sum of $\boldsymbol{x}^{\boldsymbol{\lambda}}$, search a division trail from $\mathbb{K}_1$ to $\mathbf{0}\|\boldsymbol{\lambda}$
  If there is a trail from $\mathbb{K}_1$ to $\mathbf{0}\|\boldsymbol{\lambda}$, add $\boldsymbol{\lambda}'$ into $\mathbb{U}$ since xor-sum of $\boldsymbol{x}^{\boldsymbol{\lambda}'}$ is unknown for all $\boldsymbol{\lambda}'\succcurlyeq\boldsymbol{\lambda}$
- **line 15-18.** Calculate ANF of $\boldsymbol{\lambda}_{out}\,\bullet\, S_8$ to see if there is a monomial of which xor-sum is unknown
  If not, the linear transformation $\boldsymbol{\lambda}_{out}\,\bullet\, S_8$ of j-th Sbox output is balanced bit.

Fig. 5. Extended Intergral Distinguisher Search Algorithm

티로 전파되는 (2^8-1)개의 클래스로 나눌 수 있다.

정리 2. 임의의 가역 함수 $f:\mathbb{F}^{n_2}\rightarrow\mathbb{F}^{n_2}$에 대해, 입력 디비전 프로퍼티가 $D^n_{\{(1,...,1)\}}$이면 출력 디비전 프로퍼티 또한 $D^n_{\{(1,...,1)\}}$이다.

증명. $f(\boldsymbol{x})=\boldsymbol{y}$라고 가정하자. [14]의 Proposition 1에 따르면 $\deg(\boldsymbol{y}^{\boldsymbol{v}})=n$은 오직 $\boldsymbol{v}=(1,...,1)$일 때만 성립한다. 따라서 $DPT_f[(1,...,1)]=\{(1,...,1)\}$가 성립하며 출력 디비전 프로퍼티는 $D^n_{\{(1,...,1,1)\}}$이 된다. □

디비전 프로퍼티는 unknown인 단항식을 기반으로 전파되기 때문에, $D^n_{\mathbb{K}}$의 unknown 단항식 집합 $Succ(\mathbb{K})$가 포함 관계가 있을 경우 전파 후에도 그 관계가 유지된다. 예를 들어, L^0_{in}과 L'^0_{in}을 선형 변환으로 사용한 1-라운드 후의 디비전 프로퍼티를 각각 $D^{64}_{\mathbb{K}_1}$, $D^{64}_{\mathbb{K}'_1}$라고 하자. 만약 $Succ(\mathbb{K}_1)\subseteq Succ(\mathbb{K}'_1)$가 성립한다면 r-라운드 후의 디비전 프로퍼티에 대해서도 해당 관계가 성립한다. unknown 단항식이 적어야 인테그랄 구별자를 찾을 수 있으므로 L'^0_{in}에 대한 탐색을 배제하여도 무방하다.

이를 통해 탐색 시 고려할 경우의 수를 줄일 수 있으며 S_8에 적용한 결과, 총 4개의 L^0_{in}만이 남았다.

4.3 PIPO의 확장된 인테그랄 구별자 탐색

본 소단원에서는 각 L_{in}, L_{out}를 고려한 인테그랄 구별자 탐색 알고리즘에 관해 설명한다. 알고리즘 2는 입력으로 $(r-2)$-라운드 디비전 트레일을 모델링한 MILP 모델 $\mathscr{M}$과 L^0_{in}를 받아 확장된 r-라운드 인테그랄 구별자를 출력하는 알고리즘이다.

V. Integral Distinguisher of PIPO

5.1 인테그랄 구별자 탐색 결과

$\mathscr{M}^{struct}$와 $\mathscr{M}^{QM}$를 이용하여 MILP 모델 $\mathscr{M}$을 구성한 후, **알고리즘 2**를 활용하여 6-라운드 인테그랄 구별자를 각각 탐색한다. 본 논문에서는 Gurobi MILP Solver를 사용하였으며, 모든 실험은 AMD Ryzen Threadripper 3970X CPU @3.7GHz, 256G RAM, Ubuntu 20.04.1 LTS x86_64 환경에서 진행되었다.

Table 4. 136 integral distinguishers of 6-round PIPO with $\mathscr{M}^{QM}$

Constant input bit	Balanced 6-Round output bit	$\mathscr{M}^{struct}$
$\boldsymbol{x}_{6,0+i}\oplus\boldsymbol{x}_{3,0+i}$	$\boldsymbol{x}_{0,1+i}\oplus\boldsymbol{x}_{1,0+i}\oplus\boldsymbol{x}_{6,2+i}$	
	$\boldsymbol{x}_{0,2+i}\oplus\boldsymbol{x}_{1,1+i}\oplus\boldsymbol{x}_{6,3+i}$	
	$\boldsymbol{x}_{0,3+i}\oplus\boldsymbol{x}_{1,2+i}\oplus\boldsymbol{x}_{6,4+i}$	
	$\boldsymbol{x}_{0,4+i}\oplus\boldsymbol{x}_{1,3+i}\oplus\boldsymbol{x}_{6,5+i}$	
	$\boldsymbol{x}_{0,5+i}\oplus\boldsymbol{x}_{1,4+i}\oplus\boldsymbol{x}_{6,6+i}$	
	$\boldsymbol{x}_{0,6+i}\oplus\boldsymbol{x}_{1,5+i}\oplus\boldsymbol{x}_{6,7+i}$	
	$\boldsymbol{x}_{0,7+i}\oplus\boldsymbol{x}_{1,6+i}\oplus\boldsymbol{x}_{6,0+i}$	
$\boldsymbol{x}_{7,0+i}$	$\boldsymbol{x}_{5,0+i}$	
	$\boldsymbol{x}_{5,7+i}$	
	$\boldsymbol{x}_{0,0+i}\oplus\boldsymbol{x}_{1,7+i}\oplus\boldsymbol{x}_{6,1+i}$	✓
	$\boldsymbol{x}_{0,1+i}\oplus\boldsymbol{x}_{1,0+i}\oplus\boldsymbol{x}_{6,2+i}$	✓
	$\boldsymbol{x}_{0,2+i}\oplus\boldsymbol{x}_{1,1+i}\oplus\boldsymbol{x}_{6,3+i}$	✓
	$\boldsymbol{x}_{0,3+i}\oplus\boldsymbol{x}_{1,2+i}\oplus\boldsymbol{x}_{6,4+i}$	✓
	$\boldsymbol{x}_{0,4+i}\oplus\boldsymbol{x}_{1,3+i}\oplus\boldsymbol{x}_{6,5+i}$	✓
	$\boldsymbol{x}_{0,5+i}\oplus\boldsymbol{x}_{1,4+i}\oplus\boldsymbol{x}_{6,6+i}$	✓
	$\boldsymbol{x}_{0,6+i}\oplus\boldsymbol{x}_{1,5+i}\oplus\boldsymbol{x}_{6,7+i}$	✓
	$\boldsymbol{x}_{0,7+i}\oplus\boldsymbol{x}_{1,6+i}\oplus\boldsymbol{x}_{6,0+i}$	✓

The addition + in each subscript denotes addition modular 8.
✓ refers to a distinguisher that also can be searched throught $\mathscr{M}^{struct}$.

탐색 결과, $\mathcal{M}^{struct}$를 사용한 탐색에선 6-라운드 인테그랄 구별자 8개를 찾을 수 있었으며 $\mathcal{M}^{QM}$를 사용한 탐색에선 이를 포함하는 17개의 구별자가 발견되었다. 17개의 구별자에 회전 대칭성을 적용할 경우, 총 136개의 6-라운드 인테그랄 구별자가 존재함을 보일 수 있다.

6-라운드 탐색 결과, 1-비트 만으로 구별자가 되는 것은 $\boldsymbol{x}_{5,0+i}$, $\boldsymbol{x}_{5,7+i}$ 둘 뿐이며 7 라운드에 $\boldsymbol{x}_{5,0+i}$와 $\boldsymbol{x}_{5,7+i}$ 변수만으로 표현가능한 비트가 없기 때문에 제안한 방법으로 7 라운드 이상의 구별자를 탐색할 수 없다.

Table 5. Integral distinguisher search results

S-box Modeling	Longest Round	# Obtained 6-R Dist.	Time
$\mathcal{M}^{QM}$	6-round	136	7.25h
$\mathcal{M}^{struct}$	6-round	64	16.5h

5.2 8-Round Key-Recovery Attack

본 소단원에서는 5.1에서 제시한 6-라운드 인테그랄 구별자를 이용하여 8-라운드로 축소된 PIPO-64/128의 키 복구 공격을 진행한다.

각 인테그랄 구별자는 2^{63}개의 평문을 이용하므로, 데이터 복잡도를 위해 같은 평문셋을 사용하는 인테그랄 구별자를 선택한다. 본 공격에서는 평문셋 $\{\boldsymbol{x}\in\mathbb{F}_2^n | x_{56}=0\}$와 이에 해당하는 10개의 인테그랄 구별자 중 부분 복호화 시 필요한 키를 최대한 겹치게하기 위해 4개의 구별자를 선택하였으며 공격 과정에서 다음의 표기 방법을 사용한다.

- $\boldsymbol{x}^i$: PIPO-64/128의 i-라운드 후의 텍스트
- $\mathbb{X}^i$: i-라운드 후의 텍스트 멀티셋
- $|\mathbb{X}^i|$: 멀티셋 $\mathbb{X}^i$의 원소 수
- P^{-1}: P-layer 역연산
- sk'_j : $P^{-1}(K_1)_{*,j}$

5.2.1 Partial-Sum Technique

Partial-sum Technique은 FSE 2000에서 암호 Rijndael의 공격을 위해 처음으로 제안되었다[15]. Partial-sum Technique은 모든 키 비트를 추측하는 것이 아니라 키 일부분만 추측하여 부분 복호화를 진행한다. 라운드 함수의 입출력의 모든 비트가 서로 상관관계가 있는 것은 아니므로 키 일부분만 추측하여 원하는 부분을 부분 복호화할 수 있으며 이를 통해 복잡도를 낮출 수 있다.

본 공격에서는 7-라운드의 라운드 키 K_1의 각 sk'_j에 해당하는 8-비트씩 추측하여 부분 복호화를 진행한다.

Table 6. 4-distinguishers used for attack

no.	Balanced bit	Bits for partial decryption
1	$\boldsymbol{x}^6_{5,0}$	sk'_0
2	$\boldsymbol{x}^6_{5,7}$	sk'_1
3	$\boldsymbol{x}^6_{0,0}\oplus\boldsymbol{x}^6_{1,7}\oplus\boldsymbol{x}^6_{6,1}$	sk'_0, sk'_1, sk'_7
4	$\boldsymbol{x}^6_{0,7}\oplus\boldsymbol{x}^6_{1,6}\oplus\boldsymbol{x}^6_{6,0}$	sk'_0, sk'_6, sk'_7

5.2.2 공격 시나리오

$\{\boldsymbol{x}\in\mathbb{F}_2^n | x_{56}=0\}$을 만족하는 평문 집합 $\mathbb{X}^0$과 그에 해당하는 암호문 집합 $\mathbb{X}^8$을 준비한다. 또, 256-비트 $table_i (i=0,1,6,7)$을 준비한다. $table_i[\boldsymbol{j}\in\mathbb{F}_2^8]$

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

1. for each K_0 in $\mathbb{F}^{64_2}$:
2. for each $\boldsymbol{j}$ in $\mathbb{F}_2^8$:
3. for i in $[0,1,6,7]$:
4. $table_i[\boldsymbol{j}]=0$
5. for each $\boldsymbol{x}$ in $\mathbb{X}_8$:
6. $\boldsymbol{c}=D_1(\boldsymbol{x},K_0)$
7. for i in $[0,1,6,7]$:
8. $table_i[P^{-1}(\boldsymbol{c})_{*,i}]\oplus=1$
9. for each 32-bit $sk'_0 \| sk'_1 \| sk'_6 \| sk'_7$:
10. for i in $[0,1,6,7]$:
11. $\mathbb{X}^6_{*,i}=\{S_8^{-1}(\boldsymbol{j}\oplus sk_i) \mid table_i[\boldsymbol{j}]=1\}$
12. if Xor-sum of each balance-bit $\neq$ 0:
13. remove $(K_0, sk'_0, sk'_1, sk'_6, sk'_7)$ in CK

- **line 11.** Partial decryption process of i-th Sbox in 7-round
- **line 13.** Since it is a wrong key, remove it from the candidate key set $CK(\subseteq\mathbb{F}^{96_2})$

Fig. 6. Key-Recovery attack on 8-round PIPO-64/128

은 7-라운드 복호화 과정에서 i-번째 Sbox에 j 값이 짝수 번 나오면 0, 홀수 번 나오면 1을 갖는다.

5.3 공격 복잡도

5.3.1 시간 복잡도

본 공격 시나리오에서 가장 많은 시간 복잡도를 가지는 곳은 line 5-8이며 다른 과정의 경우 시간 복잡도가 line 5-8에 비해 매우 미미하다. line 5-8 과정은 2^{64+63}번의 라운드 함수 연산이 필요하므로 $2^{127}/8=2^{123}$번의 8-라운드 PIPO-64/128 암호화 연산을 통해 후보 키의 크기를 줄일 수 있다. 각 구별자가 성립할 확률은 약 1/2이므로 후보 키의 크기를 2^{96}에서 2^{92}로 줄인다. 후보 키와 나머지 32-비트를 함께 추측하면 2^{92+32}개의 경우만 남으므로 총, $2^{123}+2^{92+32} \fallingdotseq 2^{124.5849}$의 8-라운드 PIPO 암호화 연산을 통해 마스터 키를 복구할 수 있다.

5.3.2 데이터 복잡도, 메모리 복잡도

먼저, 본 공격 시나리오에 필요한 데이터의 수는 2^{63}이므로 데이터 복잡도는 2^{63}이다.

메모리 복잡도를 가장 많이 필요로 하는 부분은 후보 키 CK이며 다른 부분은 후보 키 CK에 비해 매우 미미하다. 후보 키 CK는 2^{96}-비트만큼 필요하므로 공격 시나리오의 총 메모리 복잡도는 $2^{96}/8=2^{93}$ 바이트이다.

VI. 결 론

본 논문에서는 Sbox 설계 구조를 이용한 모델링 방법($\mathcal{M}^{struct}$)을 제안하였으며 경량 블록 암호 PIPO에 대한 확장된 인테그랄 구별자 탐색과 인테그랄 공격을 진행하였다.

PIPO의 제안 논문에서 제안한 것과 달리, PIPO에 대한 인테그랄 구별자 탐색 결과 6-라운드 인테그랄 구별자가 존재하였으며 $\mathcal{M}^{QM}$ 모델링을 통한 탐색을 통해 최대 136개의 구별자를 보일 수 있었다. 그중 120개의 경우, Sbox 출력의 선형 변환을 통해 도출할 수 있었다. 이는 PIPO Sbox S_8의 대수적 차수(algebraic degree)가 3인 반면 최소 대수적 차수(minimum degree)가 2인 특징이 있기 때문으로 볼 수 있다. 구체적으로, $\boldsymbol{\lambda}=(0,1,0,0,0,0,1,1)$에 대해 $\boldsymbol{\lambda} \bullet S_8$의 대수적 차수는 2로 더 낮아지며 이러한 점을 통해 인테그랄 구별자를 더 도출한 것으로 볼 수 있다.

Table 7. Complexity of attack

Round	Time (Enc)	Data (Plaintext)	Memory (Byte)
8	$2^{124.5849}$	2^{63}	2^{93}

결과적으로 공격 복잡도를 고려하여 인테그랄 구별자 중 4개의 인테그랄 구별자를 공격에 사용하였으며, $2^{124.5849}$번의 8-라운드 PIPO-64/128 암호화 연산을 요구하는 키 복구 공격을 수행하였다.

Sbox 설계 구조를 이용한 모델링 방법은 기존 방법과 비교했을 때, 최대 8개의 6 라운드 구별자까지 밖에 도출하지 못함으로써 정확성 측면에서 단점이 있다. 하지만 MILP 모델링에 걸리는 복잡도가 내부의 작은 Sbox 크기에 의존하기 때문에 빠른 모델링이 가능하며 Superbox[17]와 같은 큰 Sbox를 통한 디비전 프로퍼티 분석 시 활용될 것을 기대한다.

References

[1] Kim, Hangi, et al. "PIPO: A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations," International Conference on Information Security and Cryptology, pp. 99-122, Dec. 2020.

[2] Kim, Hangi, et al. "A New Method for Designing Lightweight S-boxes with High Differential and Linear Branch Numbers, and Its Application," IACR ePrint 2020-1582, Dec. 2020.

[3] Daemen, Joan, Lars Knudsen, and Vincent Rijmen, "The block cipher Square," International Workshop on Fast Software Encryption, pp. 149-165, Jan. 1997.

[4] Todo, Yosuke, "Structural evaluation by generalized integral property,"

Annual International Conference on the Theory and Applications of Cryptographic Techniques, pp. 287-314, Apr. 2015.
[5] Xiang, Zejun, et al. "Applying MILP method to searching integral distinguishers based on division property for 6 lightweight block ciphers," International Conference on the Theory and Application of Cryptology and Information Security, pp. 648-678, Dec. 2016.
[6] Derbez, Patrick, Pierre-Alain Fouque, and Baptiste Lambin, "Linearly equivalent S-boxes and the Division Property," IACR ePrint 2019-97, Nov. 2019.
[7] Knudsen, Lars and David Wagner, "Integral cryptanalysis," International Workshop on Fast Software Encryption, pp. 112-127, Feb. 2002.
[8] Mouha, Nicky, et al. "Differential and linear cryptanalysis using mixed-integer linear programming," International Conference on Information Security and Cryptology, pp. 57-76, Nov. 2011.
[9] Sun, Siwei, et al. "Towards finding the best characteristics of some bit-oriented block ciphers and automatic enumeration of (related-key) differential and linear characteristics with predefined properties," IACR ePrint 2014-45, Feb. 2015.
[10] Sun, Siwei, et al. "Automatic security evaluation and (related-key) differential characteristic search: application to SIMON, PRESENT, LBlock, DES (L) and other bit-oriented block ciphers," International Conference on the Theory and Application of Cryptology and Information Security, pp. 158-178, Dec. 2014
[11] Tingting, Cui, et al. "New automatic search tool for impossible differentials and zero-correlation linear approximations," IACR ePrint 2016-689, Jul. 2016.
[12] Sasaki, Yu, and Yosuke Todo, "New impossible differential search tool from design and cryptanalysis aspects," Annual International Conference on the Theory and Applications of Cryptographic Techniques. pp. 185-215, Apr. 2017.
[13] Je-seong Kim, et al. "Study on Division Property Analysis exploiting S-Box Construction," CISC-S'21, pp. 488-491, June 2021
[14] Boura, Christina, Anne Canteaut, and Christophe De Canniere. "Higher-order differential properties of Keccak and Luffa," International Workshop on Fast Software Encryption. pp. 252-269, Feb. 2011.
[15] Ferguson, Niels, et al. "Improved cryptanalysis of Rijndael," International Workshop on Fast Software Encryption. pp. 213-230, Apr. 2000.
[16] Todo, Yosuke, and Masakatu Morii. "Bit-based division property and application to simon family," International Conference on Fast Software Encryption. pp. 357-377, Mar. 2016.
[17] Gilbert, Henri, and Thomas Peyrin. "Super-Sbox cryptanalysis: Improved attacks for AES-like permutations," International Workshop on Fast Software Encryption. pp. 365-383, Feb. 2010.

〈저자소개〉

김 제 성 (Jeseong Kim) 학생회원
2020년 2월: 고려대학교 수학과 졸업
2020년 3월~현재: 고려대학교 정보보호대학원 석사과정
〈관심분야〉 암호 알고리즘 설계 및 분석, 대칭키 암호

김 성 겸 (Seonggyeom Kim) 학생회원
2016년 8월: 한양대학교 수학과 졸업
2016년 9월~2018년 8월: 고려대학교 정보보호대학원 석사
2019년 3월~현재: 고려대학교 정보보호대학원 박사과정
〈관심분야〉 암호 알고리즘 설계 및 분석, 대칭키 암호, 난수발생기

김 선 엽 (Sunyeop Kim) 학생회원
2019년 8월: 고려대학교 수학과 졸업
2019년 9월~현재: 고려대학교 정보보호대학원 석박사통합과정
〈관심분야〉 암호 알고리즘 설계 및 분석, 대칭키 암호

홍 득 조 (Deukjo Hong) 종신회원
2006년 2월: 고려대학교 정보보호대학원 박사
2006년 3월~2007년 12월: 고려대학교 정보보호기술연구센터 연구교수
2007년 12월~2015년 8월: 국가보안기술연구소 선임연구원
2015년 9월~현재: 전북대학교 IT정보공학과 부교수
〈관심분야〉 암호 알고리즘 설계 및 분석

성 재 철 (Jaechul Sung) 종신회원
2002년 8월: 고려대학교 수학과 박사
2002년 8월~2004년 1월: 한국정보보호진흥원 선임연구원
2004년 2월~현재: 서울시립대학교 수학과 전임강사, 조교수, 부교수, 교수
〈관심분야〉 암호 알고리즘 설계 및 분석

홍 석 희 (Seokhie Hong) 종신회원
2001년: 고려대학교 수학과 박사
1999년 8월~2004년 2월: ㈜시큐리티 테크놀로지 선임연구원
2003년 3월~2004년 2월: 고려대학교 정보보호기술연구센터 선임연구원
2004년 4월~2005년 2월: K.U. Leuven ESAT/SCD-COSIC 박사후 연구원
2005년 3월~2013년 8월: 고려대학교 정보보호대학원 부교수
2013년 9월~현재: 고려대학교 정보보호대학원 정교수
〈관심분야〉 대칭키 및 공개키 암호 알고리즘, 부채널 공격 및 대응기법, 디지털 포렌식