Journal of the Korea Academia-Industrial cooperation Society
Vol. 23, No. 2 pp. 751-759, 2022

https://doi.org/10.5762/KAIS.2022.23.2.751
ISSN 1975-4701 / eISSN 2288-4688

부채널 분석에 강인한 블록 암호 PIPO 구현을 위한 비선형 마스킹 기법

장세창[1], 장재원[1], 배대현[2], 하재철[1*]
[1]호서대학교 컴퓨터공학부, [2]호서대학교 정보보호학과

Non-Linear Masking Method for Secure Implementation of Block Cipher PIPO Resistant to Side-Channel Analysis

Sechang Jang[1], Jaewon Jang[1], Daehyeon Bae[2], Jaecheol Ha[1*]
[1]Division of Computer Engineering, Hoseo University
[2]Department of Information Security, Hoseo University

요 약 최근 IoT 환경에서의 암호 연산을 고려하여 경량 블록 암호 알고리즘인 PIPO(Plug-In Plug-Out)가 제안되었다. PIPO는 고속 구현을 위해 비트 슬라이스(bit slice) 기법을 사용하였고 비선형 연산을 최소화하는 데 중점을 두고 설계되었다. 이러한 경량 암호 알고리즘은 IoT 환경에서 동작하므로 부채널 분석에 대한 안전성이 필수로 검증되어야 한다. 본 논문에서는 ChipWhisperer 플랫폼과 Atmel ATxmega128 마이크로컨트롤러가 탑재된 타겟 보드를 대상으로 대응책이 없는 PIPO는 물론 ISW 기반 대응책이 적용된 구현체가 전력 분석 공격에 의해 비밀키가 충분히 누출될 수 있음을 실험적으로 증명하고, 1차 전력 분석 공격에 대응하기 위한 비선형 연산인 논리적 AND 및 OR 연산에 적용할 수 있는 마스킹 기법을 제안하였다. 제안하는 마스킹과 셔플링(shuffling) 기법을 적용해 PIPO 암호 알고리즘을 구현한 결과, 비밀키 누출을 방어할 수 있음을 확인하였다. 또한, 대응책으로 인한 오버헤드를 줄이기 위해 축소 마스킹을 적용한 결과 적정 수준의 추가 연산만으로 효율적으로 부채널 분석에 대응할 수 있음을 확인하였다.

Abstract Recently, a lightweight block cipher PIPO (Plug-In Plug-Out) has been proposed considering cryptographic operations in the IoT environment. The PIPO used a bit slice method for high-speed implementation and was designed to emphasize minimizing non-linear operations. Since the lightweight block cipher algorithm operates in the IoT environment, the resistance of the side-channel analysis must be verified. This paper shows that a power-based side-channel analysis can sufficiently leak the secret key of unprotected/protected PIPO. The side-channel analysis uses the ChipWhisperer platform and target board with Atmel ATxmega128 MCU. This study also proposes two non-linear masking methods applied to logical AND and OR operations, which can defeat the first-order power analysis attack on PIPO. As a result of implementing the proposed masking methods and shuffling technology, we confirmed that the leakage of the secret key could be prevented. In addition, we confirmed that the proposed countermeasure could be utilized without a huge overhead by using the masking reduction method.

Keywords : PIPO, Side-Channel Analysis, Power Analysis Attack, Boolean Masking, Non-linear Masking

이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1F1A1074358).
*Corresponding Author : Jaecheol Ha(Hoseo Univ.)
email: jcha@hoseo.edu
Received December 14, 2021 Revised January 5, 2022
Accepted February 4, 2022 Published February 28, 2022

1. 서론

최근 사물 인터넷(IoT, Internet of Things) 기술이 발달함에 따라 스마트 홈이나 스마트 공장 및 제어 시설에서 사용할 수 있는 안전한 보안 장치 및 단말이 필요하게 되었다. 그러나 대부분의 IoT 기기들은 저용량 저성능의 프로세서를 탑재하고 있으므로 여기에 사용되는 정보보호 기능도 경량화될 필요가 있다. 이와 같은 정보보호 요구 조건을 고려하여 저성능 IoT 기기에도 사용할 수 있는 LEA나 SPECK과 같은 경량 블록 암호들이 제안되기도 하였다[1,2].

PIPO(Plug-In Plug-Out)는 IoT 환경 등을 고려하여 연산 부하가 적고 효율적인 마스킹 적용을 위해 2020년에 제안된 국내 대표적인 경량 블록 암호 알고리즘 중 하나이다[3]. 특히, PIPO는 최소한의 메모리 사용과 고속 구현을 위해 비트 슬라이스 기법을 사용하였으며, 부채널 공격에 대한 효과적인 대응을 위해 비선형 연산을 최소화하는 데 중점을 두고 설계되었다. 현재까지 실험 결과, PIPO는 8비트 마이크로프로세서 상에서 128비트의 비밀키를 사용하는 다른 64비트 블록 암호보다 성능이 우수한 것으로 평가되고 있다.

한편 암호 알고리즘에 대한 부채널 공격은 암호용 장비에서 비밀키와 연관된 연산을 수행하는 동안 누출되는 전력, 전자기파, 수행 시간 등과 같은 부채널 신호를 분석하여 기기 내부의 비밀 정보를 복구해 내는 공격 기술이다[4,5]. 그동안 블록 암호나 공개키 암호 시스템을 대상으로 단순 전력 분석 공격(SPA: Simple Power Analysis,), 차분 전력 분석 공격(DPA: Differential Power Analysis, 이하 DPA) 그리고 상관 전력 분석 공격(CPA: Correlation Power Analysis, 이하 CPA) 등과 같은 분석 기법을 이용하여 내부 비밀키를 수 분 내에 찾을 수 있는 수준에 이르고 있다[6-8].

한편, 블록 암호 알고리즘에 대한 부채널 분석을 무력화시키기 위해 여러 대응 기법들이 제안되었는데, 이중 가장 대표적인 것이 임의의 값을 이용해 암호 알고리즘의 중간 연산 값을 무작위화시키는 마스킹(masking) 기법이 있다[9-12]. 1차(first-order) 전력 분석 공격에 대응하기 위한 마스킹 기법을 사용하더라도 고차(high-order) 분석 공격에 취약할 수 있다. 그러나 현재는 연산의 효율성을 고려하여 대부분 1차 마스킹 기법까지 적용하고 있다.

본 논문에서는 마이크로컨트롤러에 PIPO 암호 알고리즘을 구현하고 이를 구동할 때 누설되는 전력 부채널 신호를 분석하면 비밀키가 누출될 수 있음을 보이고, 1차 전력 분석에 대응하기 위해 비선형 마스킹 기법을 제안하고자 한다[13,14]. 제안하는 마스킹 기법은 XOR 마스킹을 적용한 상태에서 비선형 연산인 AND나 OR 연산을 효과적으로 수행하는 방법이다. 제안된 마스킹 대응 기법과 셔플링(shuffling) 기술을 적용하면 PIPO에 대한 1차 전력 분석 공격에 대응할 수 있음을 실험적으로 확인할 수 있었다[15].

2. 경량 블록 암호 PIPO

PIPO 블록 암호 알고리즘은 8-바이트 블록 사이즈와 128-비트(혹은 256-비트) 키 사이즈를 사용하는 경량 블록 암호 알고리즘이다. 암・복호화 과정에서 중간 연산 값들은 8×8 크기의 비트로 구성된 상태 배열로 연산이 수행된다. SPN 구조인 PIPO는 128-비트(혹은 256-비트) 키를 가질 경우 13 라운드(혹은 17라운드)로 구성된다. 각 라운드는 비선형 연산인 S-layer와 선형 연산인 P-layer 그리고 Key-XOR 연산으로 구성된다. 다음 Fig. 1은 PIPO의 세부 구조를 나타낸 것이다.

PIPO 블록 암호 알고리즘은 S-layer를 구현하는 형태에 따라 테이블 참조 방식과 비트 슬라이스 방식으로 구분될 수 있다. 테이블 참조 방식은 SBox 연산 결과를 사전 연산 테이블(배열)로 만들어 놓은 후, 이를 참조하는 형태로 S-layer 연산을 수행한다. 반면 비트 슬라이스 방식은 비트 연산으로만 구성된 S-layer 연산을 직접 수행하며 이외 모든 연산은 테이블 참조 방식과 동일하게 구성되어 있다.

PIPO에서 평문은 테이블 참조와 비트 슬라이스를 이용한 구현체 모두 Fig. 2와 같은 내부 상태 배열로 표현된다. 이때 마이크로컨트롤러의 레지스터는 행(row)단위

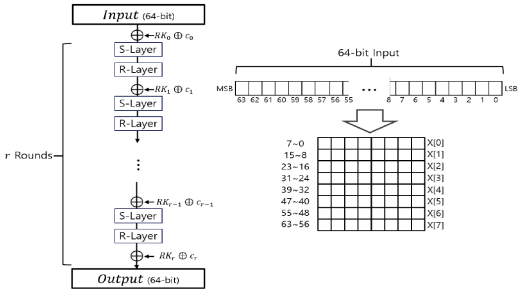

Fig. 1. Structure of PIPO block cipher.

	Plain text(Row)								Hex
X[0]	0	0	1	0	0	1	1	0	0x26
X[1]	0	0	0	0	0	0	0	0	0x00
X[2]	0	0	1	0	0	1	1	1	0x27
X[3]	0	0	0	1	1	1	1	0	0x1E
X[4]	1	1	1	1	0	1	1	0	0xF6
X[5]	0	1	0	1	0	0	1	0	0x52
X[6]	1	0	0	0	0	1	0	1	0x85
X[7]	0	0	0	0	1	0	0	1	0x09
Hex	50	30	15	38	88	5D	3D	C4	

Fig. 2. State array of PIPO encryption.

로 값이 저장되며, 평문이 입력되면 먼저 화이트닝 키(whitening Key) wk와 XOR 연산을 수행한다. 이후 1라운드 S-layer가 수행되기 전 입력 값의 상태 배열은 두 구현체 모두 동일하다. 그러나 S-Layer는 열 단위의 테이블 참조 연산이 수행되어야 하므로 상태 배열의 행과 열을 바꿔주는 작업이 선행되어야 한다. 반면 비트 슬라이스 구현체는 이러한 변환 없이 S-layer 연산을 모든 비트에 대해 병렬적으로 연산을 수행할 수 있다. 이후 수행되는 P-layer와 Key-XOR 연산은 모두 행 단위로 연산이 수행된다.

PIPO에서는 64비트로 구성된 화이트닝 키와 라운드 키가 128비트 마스터키의 일부로 구성되어 있어 간단하게 생성할 수 있다. 즉, 각 라운드 키는 마스터키에 이미 알려진 라운드 상수(RCON) 값을 XOR하여 생성하게 된다. 따라서 PIPO에서는 홀수와 짝수 라운드의 키를 각각 1개씩만 복구하면 마스터키를 쉽게 복구할 수 있다.

3. PIPO에 대한 전력 분석 공격

본 논문에서는 먼저 비트 슬라이스로 구현된 PIPO에 대한 1차 CPA 공격 실험을 위해 ChipWhisperer-Lite와 XMEGA128D4 MCU가 탑재된 타겟보드를 대상으로 직접 전력 파형을 수집한다. 마이크로프로세서 XMEGA128D4 MCU는 ChipWhisperer-Lite에서 공급하는 7.37MHz의 클럭 신호로 동작하며, 이는 ADC의 클럭과 동기화되어있다. 본 논문의 실험에서는 동기화된 클럭 신호를 이용해 한 클럭당 4개의 샘플을 측정한다.

공격자는 타겟보드에 PIPO 암호 알고리즘을 구현하여 비밀키를 이용해 암호화를 수행한다고 가정하며, 이 과정에서 수많은 소비 전력 신호를 측정하게 된다. 소비 전력 신호는 ChipWhisperer-Lite 뿐만 아니라 표본화 율이 높고 고해상도를 가지는 오실로스코프를 사용할 수도 있다.

블록 암호 알고리즘을 대상으로 하는 전력 분석 공격은 비밀키를 가정한 상태에서 암호 알고리즘의 중간 연산 값을 예측하고 소비 전력 신호와의 상관관계를 등을 분석하여 키를 복구한다. 이때, 암호 알고리즘의 중간 연산값을 계산하기 위해서는 모든 이전 라운드의 키를 알고 있어야 한다. 예를 들어, 3라운드 공격을 위해서는 화이트닝 키와 1, 2라운드 키를 모두 알고 있어야 한다. 따라서 PIPO 암호 알고리즘을 대상으로 하는 공격에서는 암호화 과정에서 0라운드 키(화이트닝 키), 1라운드 키를 차례로 찾거나 복호화 과정에서 13라운드, 12라운드 키를 차례로 찾아 공격을 수행한다.

비트 슬라이스 버전은 테이블 참조 버전과 다르게 독립적인 각 비트에 대한 S-layer 연산을 병렬로 수행하며, 이 과정에서 레지스터에 저장된 8비트 값은 모두 독립적이다. 즉, 다음 Fig. 3에서 레지스터는 행 단위의 값을 저장하고 있어 해당 8비트 단위의 누출이 발생하지만, 실제 S-layer 연산은 열 단위로 수행되므로 공격자는 열 단위의 중간 연산 값만을 추측할 수 있다. 따라서 공격자는 PIPO 암호를 수행하는 과정에서 발생하는 누설 전력을 이용하며 1비트 단위로 CPA 공격을 수행해 64비트의 모든 비밀키를 복구한다.

공격 모델을 구체적으로 표현하기 위해, Fig. 3에서 평문의 각 열을 p_c, 화이트닝 키의 각 열을 wk_c, 1바이트 값의 특정 비트를 추출하는 함수를 $parseBIT$라 하자. 이때 1라운드 S-box 결괏값에 대한 각 비트열 부채널 누출 모델은 다음과 수식 (1)과 같이 정의할 수 있다.

$$L(p,k) = parseBIT(SBOX[p_c \oplus wk_c]) \quad (1)$$

이 공격 모델을 사용하여 비트 슬라이스 구현체에서

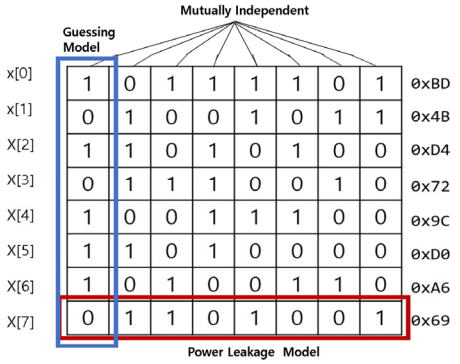

Fig. 3. Key guessing model and power leakage model for bit slice implementation.

wk의 특정 열에 해당하는 wk_c 단위로 복구할 수 있다. 다음 Fig. 4는 부채널 분석 대응책이 적용되지 않은 PIPO의 1라운드에 대하여 누출 모델 L을 사용해 CPA 공격을 수행한 결과이다. 이때, 누출 모델 L의 모든 조합 즉, 8종류의 대상 바이트, 8종류의 대상 비트로 총 64개의 비트를 대상으로 공격을 수행한다. 그리고 올바른 키를 가정했을 때의 상관계수를 나타낸 것이다. 분석 결과, 1라운드와 2라운드에서 모두 부채널 누출이 발생하며, 대부분 0.5의 높은 상관계수를 보이며 라운드 키 복구가 가능하다.

부채널 공격에 대한 대응책이 없는 PIPO 암호 알고리즘에 대해서는 설계 단계에서 DPA 및 CPA와 같은 공격이 가능할 것으로만 예상하였으나 구체적인 공격 모델이 제시되고 실험을 통해 공격이 성공했다는 문헌은 아직 없다. 다만, 고차 전력 분석 공격에 대비하여 비선형 마스킹을 위한 ISW-AND 및 ISW-OR 기법을 적용할 수 있음을 제시한 바 있다[3]. 본 논문에서는 1차 마스킹을 적용한 ISW 기법을 구현하여 공격 대응 실험을 진행하였으나 이 또한 CPA 공격에 안전하지 않음을 확인하였다.

4. 비선형 마스킹 대응 기법 및 실험 분석

블록 암호 알고리즘을 대상으로 하는 부채널 분석에 대응하는 방법 중 하나가 암호 알고리즘의 연산 값에 대해 난수로 XOR($\oplus$)연산을 수행해 무작위화하는 즉, 마스킹 기법이다. 마스킹 기법은 중간 연산 값들에 대한 XOR나 rotation와 같은 선형 연산을 수행할 때는 크게 문제가 되지 않으나 논리적 AND($\wedge$)나 OR($\vee$)와 같은 비선형 연산시에는 다른 방법이 적용되어야 한다.

PIPO 알고리즘도 테이블 참조 구현에 대한 1차 마스킹은 단순히 마스킹 테이블을 사용해 구현될 수 있다. 따라서 테이블 참조 구현에 대한 마스킹 기법 적용에 관한

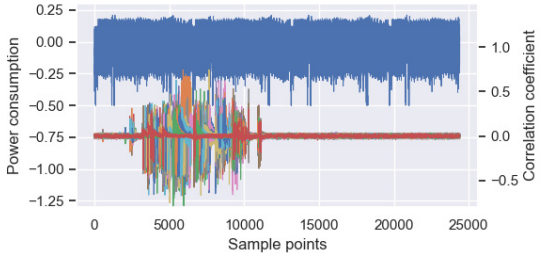

Fig. 4. A result of CPA on whitening key.

세부 내용은 생략한다. 논문에서는 앞으로 많이 사용될 것으로 예상되는 비트 슬라이스 구현체 버전에 대해서만 대응책을 설계하고자 한다. 이전 논문에서는 부채널 공격에 대응하는 PIPO를 설계하면서 비선형 마스킹을 위해 고차 마스킹에서 사용된 ISW-AND 및 ISW-OR 기법을 적용할 수 있음을 제시한 바 있다[16,17]. 그러나 실험 결과 커플링 효과 등으로 인해 ISW 방식도 공격에 취약하므로 본 논문에서는 중간 값 정보에 대한 누설없이 비선형 연산인 AND, OR의 출력 마스크값을 계산하는 새로운 마스킹 방법을 제안한다.

제안하는 방법을 설명하기 위해, AND 또는 OR에서 사용되는 암호 알고리즘의 중간 연산 값을 a, b라 하며, 각각 마스크를 m_1, m_2, 마스크가 XOR된 값을 $a_m = a \oplus m_1$, $b_m = b \oplus m_2$이라 하자. 문제의 핵심은 이항 논리 연산자 "$\cdot$"(AND 혹은 OR)에 대하여 다음 수식을 만족하는 마스크 M을 찾는 것이다.

$$a_m \cdot b_m = (a \cdot b) \oplus M \tag{2}$$

만약 이항 연산자 $\cdot$ 이 XOR이라면, 출력 마스크 M은 $m_1 \oplus m_2$로 쉽게 계산될 수 있으며 M을 계산하는 과정에서 a나 b의 누출은 없다. 즉, 다음과 같은 등식을 만족한다.

$$a_m \oplus b_m = (a \oplus b) \oplus M \tag{3}$$

$$M = m_1 \oplus m_2 \tag{4}$$

그런데 $\cdot$ 이 비선형 연산인 AND 혹은 OR 연산의 경우에도 M을 효과적으로 찾을 수 있어야 한다. 따라서 AND 연산에 대한 출력 마스크 M_{AND}과 OR에 대한 출력 마스크 M_{OR}을 찾기 위해서는 아래의 두 등식을 만족하는 M_{AND}과 M_{OR}을 각각 찾아야 한다. 여기서 중요한 점은 M_{AND}과 M_{OR}는 오직 a_m, b_m, m_1, m_2의 수식으로만 표현되어야 하고 계산 과정에서 a와 b는 직접 사용되지 않아야 정보 누설을 막을 수 있다.

$$a_m \wedge b_m = (a \wedge b) \oplus M_{AND} \tag{5}$$

$$a_m \vee b_m = (a \vee b) \oplus M_{OR} \tag{6}$$

4.1 AND 연산에 대한 마스크

먼저, AND 연산에 대한 마스크값을 계산하기 위해 a, b, m_1, m_2의 모든 비트의 경우를 고려해 보자. 모든 1-비트의 조합은 2^4개의 경우의 수가 존재하며, 이를 표현한 진리표가 다음 Table 1과 같다. 진리표의 우측 부분은 모든 비트 조합에 대하여 $M_{AND} = (a \wedge b) \oplus (a_m \wedge b_m)$

를 계산한 것이다. 여기서 $a_m \wedge b_m$는 암호 연산 과정에서 계산되는 값이며 M_{AND}은 $a \wedge b$의 마스크값이 된다.

a, b, m_1, m_2에 대한 M_{AND}의 조합을 진리표로 표현했으므로, 이를 새로운 논리식으로 표현하기 위해 카르노 맵(Karnaugh map)을 사용할 수 있다. 그러나 M_{AND} 마스크의 경우 마스크 결괏값 중 1을 가지는 경우가 6개, 0을 가지는 경우가 10개로 불균형적인 경우를 가지게 된다. 이처럼 출력 마스크값의 불균형 문제가 발생하면 이것이 누설 정보가 되어 통계학적 방법을 통해 비밀키와의 상관도를 추가로 파악할 수도 있다.

따라서 마스크 결괏값에 대해 0과 1의 개수가 균형을 이루도록 설계할 필요가 있다. 그러므로 최종적인 RM_{AND} 값은 M_{AND}에 새로운 난수 r을 XOR한 $M_{AND} \oplus r$로 변경시킬 수 있다. 그 결과 최종적인 마스크값 RM_{AND}의 0과 1의 수가 동일하게 되어 마스크값에 의한 정보 누출을 막을 수 있다. 따라서 실제 AND 연산은 마스킹된 다음 수식 (7)과 같이 두 값 a_m과 b_m에 대한 AND 연산을 수행하고 난수 r과 XOR를 수행해야 한다.

$$RAND_M = (a_m \wedge b_m) \oplus r \tag{7}$$

마스크값 RM_{AND}에 대한 최적 논리식은 $RM_{AND} = M_{AND} \oplus r$ 인데 먼저 M_{AND} 값을 Fig. 5의 카르노 맵을 이용하여 다음 수식 (8)과 같이 3개의 논리합 항으로 설계할 수 있다.

$$\begin{aligned} M_{AND} = {} & (a_m \wedge \overline{m_1} \wedge m_2) \\ & \vee (b_m \wedge m_1 \wedge \overline{m_2}) \\ & \vee ((a_m \oplus b_m \oplus m_1) \wedge m_1 \wedge m_2) \end{aligned} \tag{8}$$

위와 같은 마스크값을 계산할 때 가장 중요한 것은 중간 결괏값 a와 b의 누출이 없어야 한다는 점과 구현에 필요한 연산 수가 최소화되어야 한다는 것이다. 그림에서 보는 바와 같이 AND 연산에 대한 마스크값 M_{AND}를 계산하는데 2번의 NOT 연산, 2번의 OR 연산, 6번의 AND 연산, 2번의 XOR 연산이 필요하다.

Table 1. The truth table of M_{AND}.

Input				Output				
a	b	m_1	m_2	a_m	b_m	$a \wedge b$	$a_m \wedge b_m$	M_{AND}
0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0
0	0	1	0	1	0	0	0	0
0	0	1	1	1	1	0	1	1
0	1	0	0	0	1	0	0	0
0	1	0	1	0	0	0	0	0
0	1	1	0	1	1	0	1	1
0	1	1	1	1	0	0	0	0
1	0	0	0	1	0	0	0	0
1	0	0	1	1	1	0	1	1
1	0	1	0	0	0	0	0	0
1	0	1	1	0	1	0	0	0
1	1	0	0	1	1	1	1	0
1	1	0	1	1	0	1	0	1
1	1	1	0	0	1	1	0	1
1	1	1	1	0	0	1	0	1

$AND_M = (a_m \wedge b_m)$

$M_{AND} = AND_M \oplus (a \wedge b) = (a_m \wedge b_m) \oplus (a \wedge b)$

4.2 OR 연산에 대한 마스크

비선형 부울 연산 OR에 대한 마스킹 $M_{OR} = (a \vee b) \oplus (a_m \vee b_m)$을 표현한 진리표는 다음 Table 2와 같다. 진리표의 우측 부분은 모든 비트 조합에 대하여 M_{OR}를 계산한 것이다. 여기서 $a_m \vee b_m$는 암호 연산 과정에서 계산하는 값이며 $a \vee b$의 마스크값은

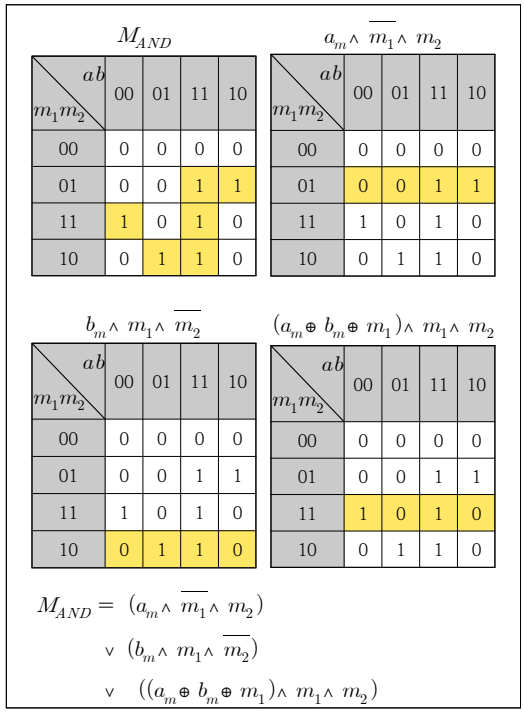

Fig. 5. Karnaugh map of random mask M_{AND} for the AND operation.

M_{OR}이 된다.

그러나 M_{OR} 마스크의 경우 마스크 결괏값 중 1을 가지는 경우가 6개, 0을 가지는 경우가 10개로 불균형적인 경우를 가지게 된다. 이 경우에도 추가적인 정보 누설을 막기 위해 마스크 결괏값에 대해 0과 1의 개수가 균형을 이루도록 설계할 필요가 있다. 따라서 최종적인 RM_{OR} 값은 M_{OR}에 r값을 XOR 연산한 $M_{AND}\oplus r$로 변경시킬 수 있다. 따라서 최종적인 마스크값 RM_{OR}의 0과 1의 수가 같게 되어 마스크값에 의한 정보 누출을 막을 수 있다. 또한, AND와 마찬가지로 실제 암호 연산시 OR 연산은 마스킹된 두 값 a_m과 b_m에 대한 OR 연산을 수행하고 난수 r과 XOR를 수행한다.

$$ROR_M = (a_m \vee b_m) \oplus r \tag{9}$$

최종적으로 $(a \vee b)$의 마스크값은 RM_{OR}가 되며 중간 마스크값 M_{OR}을 계산하는 것 외에 r값을 XOR하는 연산을 추가하여야 한다. 마스크값 RM_{OR}에 대한 최적 논리식은 $RM_{OR} = M_{OR}\oplus r$ 인데 먼저 M_{OR} 값을 Fig. 6의 카르노 맵을 이용하여 다음과 같이 3개의 논리합 항으로 설계할 수 있다. 즉, 다음 수식 (10)과 같이 표현될 수 있다.

Table 2. The truth table of M_{OR}.

Input				Output				
a	b	m_1	m_2	a_m	b_m	$a\vee b$	$a_m\vee b_m$	M_{OR}
0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	1	1
0	0	1	0	1	0	0	1	1
0	0	1	1	1	1	0	1	1
0	1	0	0	0	1	1	1	0
0	1	0	1	0	0	1	0	1
0	1	1	0	1	1	1	1	0
0	1	1	1	1	0	1	1	0
1	0	0	0	1	0	1	1	0
1	0	0	1	1	1	1	1	0
1	0	1	0	0	0	1	0	1
1	0	1	1	0	1	1	1	0
1	1	0	0	1	1	1	1	0
1	1	0	1	1	0	1	1	0
1	1	1	0	0	1	1	1	0
1	1	1	1	0	0	1	0	1

$$OR_M = (a_m \vee b_m)$$
$$M_{OR} = OR_M \oplus (a \vee b) = (a_m \vee b_m) \oplus (a \vee b)$$

$$M_{OR} = (\overline{a_m} \wedge \overline{m_1} \wedge m_2) \vee (\overline{b_m} \wedge m_1 \wedge \overline{m_2}) \vee ((a_m \oplus b_m \oplus m_1) \wedge m_1 \wedge m_2) \tag{10}$$

결과적으로 RM_{OR} 마스크값을 계산할 때 중간 결괏값 a와 b의 노출이 없었으며 그림에서 보는 바와 같이 OR 연산에 대한 마스크값 M_{OR}을 계산하는데 4번의 NOT 연산, 2번의 OR 연산, 6번의 AND 연산, 2번의 XOR 연산이 필요하다.

4.3 1차 마스킹 전력 부채널 공격 실험 결과

본 논문에서는 상기한 비선형 마스킹 기법을 사용하면 S-layer와 P-layer에서 마스크값 갱신 연산만 추가하는 방식으로 부채널 공격에 대응할 수 있음을 보이고자 한다. 이를 위해 이전 공격 실험에 사용한 환경과 동일하게 CPA 공격을 시도하였다. 그러나 PIPO에 대한 1차 마스킹 대응 기법을 적용해도 구현상 발생하는 커플링 효과에 의해 일부 키가 노출될 수 있다[18]. 따라서 본 논문에서는 이 커플링 효과를 감쇄시키기 위해 셔플링 기법을 사용하였는데, 셔플링에는 Fisher–Yates 알고리즘을 채택하였다[19].

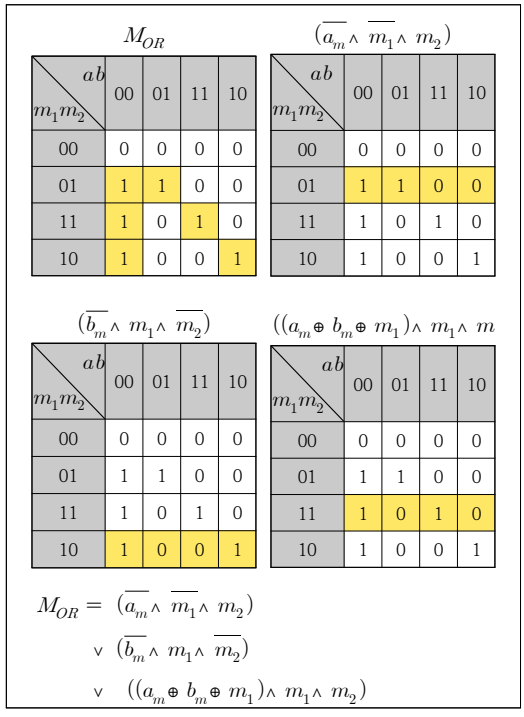

Fig. 6. Karnaugh map of random mask M_{OR} for the OR operation.

셔플링은 2가지 연산에 적용하였는데 하나는 마스크 값을 구할 때 8비트를 한번에 구하는 것이 아니라 비트 단위로 나누어 연산 순서를 임의로 결정하는 데 사용하였다. 나머지 하나는 AND나 OR 마스크값 M_{AND}나 M_{OR}를 계산할 때 3개의 중간 값을 구한 후 논리적 합을 계산하는 과정이다. 이때 각각 3개의 항을 연산하게 되는데 이 연산 순서를 임의로 결정하도록 셔플링을 적용하였다. 상기한 두 가지의 셔플링 연산 적용은 비밀키 한 비트를 찾아내는데 목표로 하는 전력 소비 위치 즉, POI(Point of Interest)를 무작위로 변경시키며 레지스터의 중복 사용을 막아 커플링 효과에 의한 정보 누출을 막을 수 있다.

제안하는 마스킹 기법만을 적용할 경우에는 추가로 필요한 연산이 매우 적지만, 셔플링을 적용하는 경우에는 많은 추가 연산이 필요하다. 실제로, 암호화 연산 과정에서 추가되는 연산의 사이클을 비교한 것이 Table 3이다. PIPO를 제안하면서 권고한 마스킹 기법인 ISW 기법은 AND 마스킹을 사용할 경우 4번의 AND, 5번의 XOR 연산이 추가되어야 하며 OR 마스킹을 사용할 경우 4번의 AND, 9번의 XOR 연산이 추가로 필요하다. 그러나 ISW 기법은 마스킹 처리를 위해 2중 반복문을 사용하게 되어 처리 시간의 오버헤드가 많이 발생하게 될 뿐만 아니라 커플링 효과로 인해 1차 CPA 공격에 취약한 특성을 보이는 경우가 많다.

제안하는 AND 마스킹 기법에서는 마스크값을 계산하는데 2번의 NOT 연산, 2번의 OR 연산, 6번의 AND 연산, 3번의 XOR 연산이 필요하며 실제 중간 값 $RAND_M = (a_m \wedge b_m) \oplus r$을 계산하는데 XOR 연산이 추가된다. 또한 제안하는 OR 마스킹 기법에서는 마스킹 값을 계산하는데 4번의 NOT 연산, 2번의 OR 연산, 6번의 AND 연산, 3번의 XOR 연산이 필요하며 실제 중간 값 $ROR_M = (a_m \vee b_m) \oplus r$을 계산하는데 XOR 연산이 추가된다.

셔플링 연산의 추가로 인한 제안한 방법의 오버헤드를 줄이기 위해서 마스킹을 적용하는 라운드를 처음 1, 2, 3라운드와 마지막 11, 12, 13라운드에만 적용하는 축소 라운드 마스킹 기법을 사용하였다. 블록 암호에 대한 공격 특성상 공격을 위해서는 앞선 모든 라운드 키를 알아야 해서 마스킹이 적용되지 않은 4~10라운드를 대상으로는 공격을 수행할 수 없다. 따라서 일부 라운드에만 마스킹을 적용하는 축소 마스킹을 사용하더라도 부채널 분석으로부터 암호 알고리즘을 보호할 수 있다.

Table 3. Performance comparison of masking countermeasures.

	PIPO-BS	ISW masking [16]		Proposed masking	
Security	Not secure	Not secure		Secure	
Additive operations for AND masking	-	NOT OR AND XOR	0 0 4 8	NOT OR AND XOR	2 2 6 3
Additive operations for OR masking	-	NOT OR AND XOR	0 0 4 12	NOT OR AND XOR	4 2 6 3
Encryption Cycles (Optimization -O0)	22,373 (No masking)	179,962 (Full round masking)		478,516 (Full round masking)	
				232,567 (Reduced round masking	

다음 Fig. 7은 기존 ISW 기법을 사용한 경우와 제하는 마스킹 기법을 적용한 경우 전력 분석 공격을 시도한 것이다. 이때, 마스킹이 적용된 PIPO에 대한 공격 실험은 앞선 대응책이 없는 PIPO를 대상으로하는 공격 실험과 동일한 환경에서 수행한다.

각 그림의 위쪽의 파형은 전력 소비 파형을 나타낸 것이고 아래쪽 파형은 비밀키와의 상관도를 나타낸 것이다. Fig. 7(a)는 기존 ISW 기법을 적용해도 CPA 공격에 의해 비밀키를 찾을 수 있음을 보인 것이다. 그러나 Fig. 7(b)를 보면 제안하는 비선형 연산에 대해 1차 마스킹과 셔플링 기법을 사용하면 비밀키를 찾을 수 없음을 확인할 수 있다.

전체적으로 암호화 연산을 수행하는데 필요한 사이클 수를 비교해 본 결과, 기존 ISW 기법을 이용한 1차 마스킹이 8배 이상 연산 사이클이 증가하지만, 이차 공격에 의해 비밀키가 노출되어 안전하지 않음을 확인할 수 있었다. 제안한 1차 마스킹은 부채널 대응책이 없는 기존 PIPO의 비트 슬라이스 구현체에 비해 축소 라운드 마스킹을 적용할 경우 약 10배 연산 사이클만 증가한다. 이는 부채널 공격이 적용된 기존 ISW 기법보다는 1.3배 정도 증가하여 부채널 공격 대응책으로서의 실용성은 갖추고 있다고 할 수 있다.

암호 알고리즘 PIPO는 부채널 공격에 효과적으로 대응하기 위해 설계되었으나 구현 실험을 한 결과 1차 마스킹 대응 기법만으로는 정보 누출을 막기 힘들고 셔플링과 같은 커플링 효과를 감쇄시킬 수 있는 대응 기법을 혼용해야 한다는 것을 확인하였다.

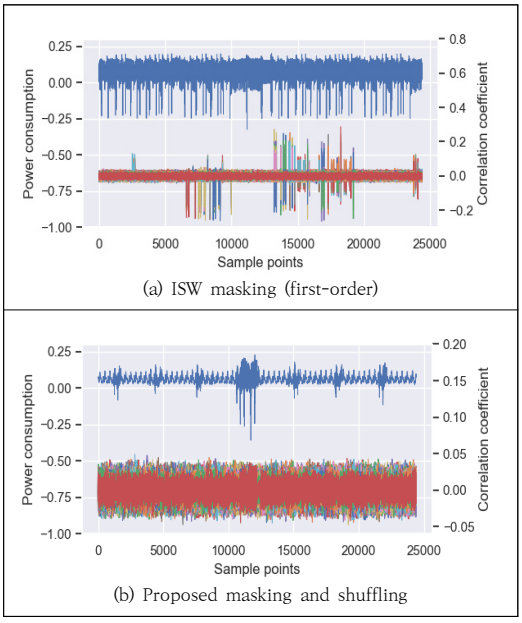

Fig. 7. Experimental results of CPA on first-order masked implementation.

5. 결론

최근 제안된 블록 암호 알고리즘 PIPO는 향후 스마트 홈이나 스마트 제어 설비와 같은 IoT 환경 등에 많이 사용될 수 있는 초경량 암호 알고리즘이다. PIPO 암호 알고리즘의 특징은 부채널 공격에 효과적으로 대응하기 위한 비선형 연산자를 최소화하였고 고속 구현이 가능한 효율적인 구조로 되어 있다는 점이다.

본 논문에서는 PIPO 암호 알고리즘이 부채널 공격 대응책을 사용하지 않을 경우 전력 분석 공격으로 쉽게 비밀키가 복구될 수 있음을 실험을 통해 확인하였다. 그리고 PIPO에 대한 1차 전력 분석에 대응하기 위해 마스킹 기법을 제안하고 연산 과정에 셔플링 기법을 적용하여 비밀키 누출을 방어할 수 있음을 실험을 통해 확인하였다. 제안한 마스킹 및 셔플링 방법을 축소된 라운드에 적용하면 오버헤드가 크게 증가하지 않으며, 1차 전력 분석 공격의 대응 기법으로 활용할 수 있다.

References

[1] D. Hong, J. Lee, D. Kim, D. Kwon, K. Ryu, and D. Lee, "LEA, A 128-bit block cipher for fast encryption on common processors," WISA'13, LNCS 8267, pp. 3-27, 2014.
DOI: https://doi.org/10.1007/978-3-319-05149-9_1

[2] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK block ciphers on AVR 8-bit microcontrollers," LightSec'14, LNCS 8898, pp. 3-20, 2014.
DOI: https://doi.org/10.1007/978-3-319-16363-5_1

[3] H. Kim, Y. Jeon, G. Kim, J. Kim, B. Sim, D. Han, H. Seo, S. Kim, S. Hong, J. Sung, and D. Hong, "PIPO :A Lightweight Block Cipher with Efficient Higher-Order Masking Software Implementations," ICISC'20, LNCS 12593, pp. 99-122, 2020.
DOI: https://doi.org/10.1007/978-3-030-68890-5_6

[4] P. Kocher, "Timing Attacks on Implementation of Diffie-Hellman, RSA, DSS, and Other Systems," CRYPTO'96, LNCS 1109, pp. 104-113, 1996.
DOI: https://doi.org/10.1007/3-540-68697-5_9

[5] T. Messerges, "Securing the AES finalists against power analysis attacks," FSE'00, LNCS 1978, pp. 150-164, 2001
DOI: https://doi.org/10.1007/3-540-44706-7_11

[6] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," CRYPTO'99, LNCS 1666, pp. 388-397, 1999
DOI: https://doi.org/10.1007/3-540-48405-1_25

[7] J. Coron, "Resistance against differential power analysis for elliptic curve cryptosystems", CHES'99, LNCS 1717, pp. 292-302, Springer-Verlag, 1999.
DOI: https://doi.org/10.1007/3-540-48059-5_25

[8] E. Brier, C. Clavier, and F. Olivier, "Correlation Power Analysis with a Leakage Model," CHES'04, LNCS 3156. pp. 16-29, 2004
DOI: https://doi.org/10.1007/978-3-540-28632-5_2

[9] L. Goubin, "A sound method for switching between Boolean and arithmetic masking," CHES'01, LNCS 2162, pp. 3-15, 2001.
DOI: https://doi.org/10.1007/3-540-44709-1_2

[10] B. Debraize, "Efficient and provably secure methods for switching from arithmetic to Boolean masking," CHES'12, LNCS 7428, pp. 107-121, 2012.
DOI: https://doi.org/10.1007/978-3-642-33027-8_7

[11] T. Messerges, "Securing the AES finalists against power analysis attacks," FSE'00, LNCS 1978, pp.150-164, 2001.
DOI: https://doi.org/10.1007/3-540-44706-7_11

[12] J. Coron and A. Tchulkine, "A new algorithm for switching from arithmetic to Boolean Masking," CHES'03, LNCS 2779, pp. 89-97, 2003.
DOI: https://doi.org/10.1007/978-3-540-45238-6_8

[13] C. O'Flynn and Z. Chen, "ChipWhisperer: An Open-Source Platform for Hardware Embedded Security Research, COSADE'14, LNCS 8622, pp. 243-260, 2014.
DOI: https://doi.org/10.1007/978-3-319-10175-0_17

[14] E. Park, S. Oh, and J. Ha, "Masking-based block cipher LEA resistant to side channel attacks," Journal of KIISC, Vol. 27 No. 5, pp. 1023-1032, 2014.
DOI: https://doi.org/10.13089/JKIISC.2017.27.5.1023

[15] N. Veyrat-Charvillon, M. Medwed and F. Standaert, "Shuffling against side-channel attacks: a comprehensive study with cautionary note," ASIACRYPT'14, LNCS, 7658, pp. 740-757, 2014.
DOI: https://doi.org/10.1007/978-3-642-34961-4_44

[16] Y. Ishai, A. Sahai, and D. Wagner, "Private Circuits: Securing Hardware against Probing Attacks," CRYPTO'03, LNCS 2729, pp. 463-481, 2003.
DOI: https://doi.org/10.1007/978-3-540-45146-4_27

[17] D. Goudarzi, A. Journault, M. Rivain, and F. Standaert, "Secure Multiplication for Bitslice Higher-Order Masking: Optimisation and Comparison," COSADE'18, LNCS 10815, pp. 3-22, 2018.
DOI: https://doi.org/10.1007/978-3-319-89641-0_1

[18] K. Papagiannopoulos and N. Veshchikov, "Mind the Gap: Towards Secure 1st-order Masking in Software," COSADE'17, LNCS 10348, pp. 282-297, 2017.
DOI: https://doi.org/10.1007/978-3-319-64647-3_17

[19] M. Ahmad, P. Khan and M. Ansari "A Simple and Efficient Key-Dependent S-Box Design Using Fisher-Yates Shuffle Technique," SNDS'14, Communications in Computer and Information Science, Vol. 420, pp. 540-550, Springer-Verlag, 2014.
DOI: https://doi.org/10.1007/978-3-642-54525-2_48

장 세 창(Sechang Jang) [준회원]

2016년 3월 ~ 현재 : 호서대학교 컴퓨터공학부 학부과정
- 2021년 3월 ~ 현재 : 호서대학교 정보보호학과 학·석사연계과정

〈관심분야〉
부채널 공격, 양자내성암호, 머신러닝

장 재 원(Jaewon Jang) [준회원]

- 2016년 3월 ~ 현재 : 호서대학교 컴퓨터공학부 학부과정
- 2021년 3월 ~ 현재 : 호서대학교 정보보호학과 학·석사연계과정

〈관심분야〉
양자내성암호, 머신러닝

배 대 현(Daehyeon Bae) [준회원]

- 2021년 2월 : 호서대학교 컴퓨터정보공학부 (학사)
- 2022년 2월 : 호서대학교 정보보호학과 (석사)

〈관심분야〉
부채널 공격, 암호학, 정보보호

하 재 철(Jaecheol Ha) [종신회원]

- 1989년 2월 : 경북대학교 전자공학과 (학사)
- 1993년 8월 : 경북대학교 전자공학과 (석사)
- 1998년 2월 : 경북대학교 전자공학과 (박사)
- 1998년 3월 ~ 2007년 2월 : 나사렛대학교 정보통신학과 교수
- 2007년 3월 ~ 현재 : 호서대학교 컴퓨터공학부 교수
- 2013년 1월 ~ 현재 : 한국정보보호학회 상임부회장
- 2009년 1월 ~ 현재 : 한국산학기술학회 이사

〈관심분야〉
암호학, 네트워크 보안, 부채널 공격