Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.25, NO.2, Apr. 2015 ISSN 2288-2715(Online)
http://dx.doi.org/10.13089/JKIISC.2015.25.2.253

축소 마스킹이 적용된 경량 블록 암호 LEA-128에 대한 부채널 공격*

박 명 서,[1†] 김 종 성[2,3‡]
[1]ETRI 부설 연구소, [2]국민대학교 수학과, [3]국민대학교 금융정보보안학과

Side-Channel Attacks on LEA with reduced masked rounds*

Park, Myungseo,[1†] Kim, Jongsung[2,3‡]
[1]The Affiliated Institute of ETRI, [2]Dept. of Mathematics, Kookmin University,
[3]Dept. of Financial Information Security, Kookmin University

요 약

부채널 공격(Side Channel Attack)은 전력신호, 전자파, 소리 등과 같은 부가적인 채널의 정보를 이용하여 암호 알고리즘을 분석하는 방법이다. 이러한 공격에 대한 블록 암호의 대응 기법으로 마스킹 덧붙이기가 널리 사용된다. 하지만 마스킹의 적용은 암호 알고리즘의 부하가 크기 때문에 처음 또는 마지막 몇 라운드에만 마스킹을 덧붙이는 축소 마스킹을 사용한다. 본 논문에서는 처음 1~6라운드 축소 마스킹이 적용된 경량 블록 암호 LEA에 대한 부채널 공격을 처음으로 제안한다. 제안하는 공격은 암호화 수행 과정에서 획득할 수 있는 중간 값에 대한 해밍 웨이트와 차분 특성을 이용하여 공격을 수행한다. 실험 결과에 의하면, 128 비트 마스터 키를 사용하는 LEA의 첫 번째 라운드 키 192 비트 중에 25 비트를 복구할 수 있다.

ABSTRACT

The side-channel attack is widely known as an attack on implementations of cryptographic algorithms using additional side-channel information such as power traces, electromagnetic waves and sounds. As a countermeasure of side channel attack, the masking method is usually used, however full-round masking makes the efficiency of ciphers dramatically decreased. In order to avoid such a loss of efficiency, one can use reduced-round masking. In this paper, we describe a side channel attack on the lightweight block cipher LEA with the first one~six rounds masked. Our attack is based on differentials and power traces which provide knowledge of Hamming weight for the intermediate data computed during the enciphering of plaintexts. According to our experimental result, it is possible to recover 25 bits of the first round key in LEA-128.

Keywords: lightweight block cipher LEA, side-channel attack, masking method

I. 서 론

부채널 공격은 블록 암호, 공개키 암호, 디지털 서명 등과 같은 암호학적 알고리즘에 대한 공격 방법 중 하나이다. 암호 알고리즘의 구조적 특성이 아닌 전력 신호, 전자파, 소리 등과 같이 부가적인 채널의 정보를 획득하여 암호 알고리즘을 분석하는 기법으로 전력분석(Power Analysis), 시차 분석(Timing Attack), 오류 주입 공격(Fault Attack)이 대표적이다. 이러한 부채널 공격에 안전한 암호 알고리즘을 구현하기

접수일(2014년 12월 14일), 게재확정일(2015년 2월 16일)
* 이 논문은 2013년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(No. 2013R1A1A2059864).
† 주저자, pms91@nsr.re.kr
‡ 교신저자, jskim@kookmin.ac.kr(Corresponding author)

위해 각 라운드 별로 마스킹 기법을 적용하여 널리 사용한다[1,2]. 하지만 마스킹 적용 방식에 따라 2~100배까지 암호 알고리즘의 부하로 연결될 수 있기 때문에 일반적으로 처음과 마지막 몇 라운드에만 마스킹 기법을 적용시킨 축소 마스킹을 사용한다[4]. 특히, 금융 IC 카드의 경우 서비스 품질 및 효율성을 위해 처음과 마지막 몇 라운드에만 마스킹 기법을 적용하고 있다. 하지만 축소 마스킹이 적용된 블록 암호 DES, AES 등은 부채널 공격에 취약하다는 많은 연구 결과가 발표되었다[5,6,7].

본 논문에서는 처음 1~6 라운드 축소 마스킹이 적용된 블록 암호 LEA에 대한 부채널 공격을 제안한다. 공격을 위해 암호 과정 중 수집된 전력 파형을 통한 해밍 웨이트를 이용하여 필요한 평문 쌍을 수집한다. 그 다음 수집된 해밍 웨이트 필터링 평문 쌍을 통해 192 비트의 첫 번째 라운드 키 중에 25 비트의 키 정보를 복구한다. 결과적으로 블록 암호 LEA에 6 라운드 이하의 축소 마스킹 적용은 부채널 공격에 안전하지 않다.

본 논문의 구성은 다음과 같다. 2장에서 공격 대상 경량 블록 암호 LEA에 대해 소개한다. 3장에서 해밍 웨이트 필터링에 대해 설명하고, 4장에서는 본 논문에서 제안하는 공격 방법에 대해 설명한다. 마지막으로 5장에서 결론을 맺는다.

II. 경량 블록 암호 LEA 알고리즘

LEA[3]는 128 비트 블록 단위로 암·복호화를 수행하는 경량 블록 암호이다. 마스터 키는 128, 192, 256 비트를 사용하며 길이에 따라 LEA-128, LEA-192, LEA-256으로 구분한다. 데이터의 암·복호화를 위해 사용되는 LEA의 라운드 함수는 S-box를 사용하지 않는 ARX 구조로 법덧셈(⊞) 및 법뺄셈(⊟), 비트 순환 이동(ROL, ROR), XOR(⊕) 연산만으로 구성되어 있다.

암호화 라운드 함수는 Fig. 1. 과 같이 4개의 워드 $X^i[j](i=1,\cdots,24,\ j=0,\cdots,3)$를 입력으로 하여 라운드 키 덧셈, 법덧셈, 비트 순환이동의 과정을 거쳐 $X^{i+1}[j]$를 출력으로 생성한다. 암호화 과정은 평문 128 비트를 입력하여 키 길이에 따라 24, 28, 32 라운드를 거친 후 나온 128비트의 암호문을 출력으로 얻는다. 복호화 과정은 암호화 과정을 역으로 수행하며, 128 비트 암호문을 입력하여 128 비트의

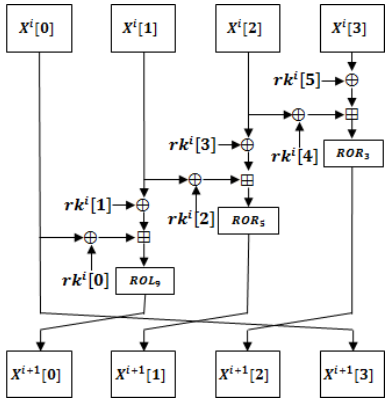

Fig. 1. An encryption round function of LEA algorithm

평문을 출력한다. 이때 라운드 키는 암호화 과정에서 사용된 라운드 키를 역으로 주입하고, 법덧셈은 법뺄셈으로 적용한다.

LEA의 키 스케줄은 마스터 키를 입력하여 라운드 키를 생성한다. 이 때 각 라운드 키는 192 비트로 구성되어 있으며 $rk^i[0]$, $rk^i[1]$, $rk^i[2]$, $rk^i[3]$, $rk^i[4]$, $rk^i[5]$(i=해당 라운드)로 표기한다.

LEA-128은 LEA-192와 LEA-256과는 달리 $rk^i[1]=rk^i[3]=rk^i[5]$라는 특성을 만족하기 때문에 실제 라운드 키 사용 비트 수는 128 비트이다. 또한 특정 한 라운드 키를 복구한다면, 키 스케줄의 역 연산을 통해 마스터 키까지 복구할 수 있다. 이러한 특성들은 본 논문에서 제시한 공격을 위해 중요한 요소로 작용한다. 블록 암호 LEA의 키 스케줄은 본 논문의 분석 과정에서 사용하지 않기 때문에 세부 알고리즘 설명은 생략한다.

III. 해밍 웨이트 필터링

본 논문에서 제안하는 공격은 축소 마스킹이 적용된 LEA에서 마스킹이 풀어지는 라운드의 출력 값에 대한 차분을 이용하여 수행한다. 하지만 암호 수행 과정에서는 중간 값을 확인할 수 없기 때문에 원하는 라운드의 출력 차분을 갖는 평문 쌍을 획득하기 위한 방법이 필요하다. 이를 위해 해밍 웨이트 필터링을 이용한다. 해밍 웨이트 필터링은 암호 과정 수행 중 부채널 분석을 통해 특정 중간 값에서 획득한 해밍 웨이트 값을 이용하여 불필요한 평문 쌍을 필터링 하는 기법이다.* 이때 해밍 웨이트는 비트별 해밍 웨

이트를 의미하며, 하나의 워드 x에 대한 해밍 웨이트 $hwt(x)$는 0~32의 값을 가질 수 있다. 제안하는 공격에 필요한 해밍 웨이트 필터링을 적용시키기 위해 property 1. ~ property 3.을 이용하여 해밍 웨이트 필터링 확률을 계산한다. property 1. 은 랜덤한 두 워드에 대한 해밍 웨이트 차이 d를 만족할 확률을 나타낸다.

property 1. 32 비트 두 워드 X, X'에 대한 해밍 웨이트의 값은 $hwt(X)$과 $hwt(X')$이다. 이때 X와 X'의 값이 랜덤하다면, $hwt(X) = hwt(X') \pm d$ $(d = 0,1,\cdots,32)$, 즉 해밍 웨이트의 차이가 d일 확률 $\Pr(hwt(X) = hwt(X') \pm d)$는 식 (1), (2)와 같다.

if $(d = 0)$

$$\Pr(hwt(X) = hwt(X')) = \sum_{n=0}^{32} \left(\frac{\binom{32}{n}}{2^{32}} \right)^2 \quad (1)$$

else

$$\Pr(hwt(X) = hwt(X') \pm d) = 2 \times \sum_{n=d}^{32-d} \left(\frac{\binom{32}{n}}{2^{32}} \times \frac{\binom{32}{n+d}}{2^{32}} \right) \quad (2)$$

식 (1), (2)을 이용하여 두 워드의 해밍 웨이트의 차이가 d일 확률을 계산하면 Table 1. 과 같다.

제안하는 공격에서는 Table 1. 의 $d = 0 \sim 2$ 확률을 이용하여 공격에 필요한 차분 특성을 만족하는 중간 평문 쌍을 필터링한다.

property 2. 는 두 워드 X, X'에 대한 해밍 웨이트 차이 d와 차분 $\triangle X$와 관계를 나타낸다. 해밍 웨이트 차이 d는 차분 $\triangle X$의 값이 홀수 또는 짝수일 경우로 구분할 수 있다.

property 2. 32 비트 두 워드 X, X'에 대한 차분이 $\triangle X (= X \oplus X')$을 만족한다고 하자. 그러면 차분 $\triangle X$에 대해 해밍 웨이트는 다음과 같은 성질을 갖는다.

if $(hwt(\triangle X) = odd)$

$hwt(X) = hwt(X') \pm d$, $(d = 1,3,\cdots,hwt(\triangle X))$

else if $(hwt(\triangle X) = even)$

$hwt(X) = hwt(X') \pm d$, $(d = 0,2,\cdots,hwt(\triangle X))$

예를 들어, $hwt(\triangle X)$가 짝수인 경우로 $\triangle X = 0x00001200$ $(hwt(\triangle X) = 2)$이라면, 0 차분이 아닌 비트 위치인 하위 10번째 비트, 13번째 비트에 따라 해밍 웨이트 차이 d는 0, 2인 경우가 생긴다. 반면에 $hwt(\triangle X)$가 홀수인 경우인 $\triangle X = 0x28000200$ $(hwt(\triangle X) = 3)$이면 0 차분이 아닌 비트 위치인 하위 10번째 비트, 28번째 비트, 30번째 비트에 따라 해밍 웨이트 차이 d는 1, 3인 경우가 생긴다.

property 3. 은 property 1. 과 property 2. 를 이용하여 랜덤한 128 비트 평문 쌍에 대한 i 라운드 차분의 해밍 웨이트 필터링 통과 확률을 계산한 것이다.

property 3. 128 비트의 랜덤한 두 평문 P, P'에 대한 i 라운드 함수 출력 $I(= I[0] \| I[1] \| I[2] \| I[3])$, $I'(= I'[0] \| I'[1] \| I'[2] \| I'[3])$가 차분 $\beta(= \beta[0] \| \beta[1] \| \beta[2] \| \beta[3])$를 만족한다고 하자. 이때 i 라운드 해밍 웨이트 필터링을 통과하기 위해서는 차분 β에 대해서 4개의 랜덤한 워드 차분을 동시에 만족해야한다. 즉, 확률 1인 워드 차분은 해밍 웨이트 필터링 확률 계산에서 제외한다. 이를

Table 3. The probability of hamming weight difference d

difference d	probability (Left side of eq. (1),(2))
0	$2^{-3.331}$
1	$2^{-2.376}$
2	$2^{-2.509}$
3	$2^{-2.731}$
4	$2^{-3.043}$
5	$2^{-3.445}$
6	$2^{-3.939}$
7	$2^{-4.523}$
⋮	⋮

1) 실제 암호 수행 과정 중에 수집한 전력 파형을 이용하여 필요한 중간 값의 해밍 웨이트를 구할 수 있다. 이는 수집된 전력 파형을 통한 템플릿 구성으로 가능하다[9].

Table 4. The probability of differential characteristic and hamming weight filtering

Round	Differential[8]	p_{diff_i}	p_{hwt_i}(eq. (3))	$\widehat{p_{hwt_i}}$(eq. (4))
1	$0x80000000$ $0x80000000$ $0x80000010$ $0x80000014$	2^{-3}	$2^{-4.752}$	$2^{-2.667}$
2	$0x00000000$ $0x80000000$ $0x80000000$ $0x80000000$	2^{-3}	$2^{-9.038}$	$2^{-2.981}$
3	$0x00000100$ $0x00000000$ $0x00000000$ $0x00000000$	2^{-4}	$2^{-11.414}$	$2^{-3.993}$
4	$0x00020000$ $0x00000000$ $0x00000000$ $0x00000100$	2^{-6}	$2^{-10.459}$	$2^{-5.940}$
5	$0x04000000$ $0x00000000$ $0x00000020$ $0x00020000$	2^{-10}	$2^{-8.990}$	$2^{-8.409}$
6	$0x00000008$ $0x00000001$ $0x00004004$ $0x40000000$	2^{-18}	$2^{-7.323}$	$2^{-7.322}$
7	$0x00001200$ $0x28000200$ $0x80800800$ $0x00000008$	·	·	·

고려하여 해밍 웨이트 필터링을 통과할 확률 p_{hwt_i}을 구하면 다음 식 (3)과 같다.

$$p_{hwt_i} = \prod_{j=0, j\neq s}^{3}\left(\sum_{d=0}^{t}\Pr(hwt(I[j]) = hwt(I[j]') \pm d^*)\right) \quad (3)$$

$$\text{where } d^* = \begin{pmatrix} 2d, & hwt(\beta[j]) = even \\ 2d+1, & hwt(\beta[j]) = odd \end{pmatrix},$$
$$s = 0,\ 1,\ 2 \text{ or } 3,$$
$$t = \left\lfloor \frac{hwt(\beta[j])}{2} \right\rfloor.$$

여기서 확률 p_{hwt_i}은 랜덤한 평문 쌍에 대한 해밍 웨이트 필터링 통과 확률이다. 이렇게 계산된 p_{hwt_i}를 통해 4.1절에서 정해진 차분 평문 쌍에 대한 해밍 웨이트 필터링 통과 확률을 구하여 실제 공격에 필요한 필터링 평문 쌍을 획득하는데 이용한다.

IV. 축소 마스킹이 적용된 LEA-128에 대한 부채널 공격

4.1 해밍 웨이트 필터링을 통한 차분 평문 쌍 획득

축소 마스킹 기법이 적용된 LEA-128에 대한 공격을 수행하기 위해서는 공격에 필요한 평문 쌍을 획득하는 과정이 필요하다. 이를 위해서 해밍 웨이트 필터링을 사용한다. i 라운드 축소 마스킹이 적용된 경우 i 라운드 출력 이후로는 마스킹 기법이 적용되지 않기 때문에 i 라운드 출력 차분은 그대로 유지된다. 따라서 마스킹 기법이 적용되지 않는 i 라운드 출력 차분은 축소 마스킹이 적용된 i 라운드 출력 차분과 동일하다. 하지만 암호화 과정이 수행되는 동안에는 중간 값을 확인할 수 없기 때문에 부채널 분석을 통해 해밍 웨이트를 획득하고, i 라운드 출력 차분을 만족하는 해밍 웨이트에 대한 경우의 수를 고려하여 이를 만족하는 평문 쌍을 획득해야 한다.

본 논문에서 제안한 공격에서는 입력 차분 α ($\alpha_0 = 0x80000000$, $\alpha_1 = 0x80000000$, $\alpha_2 = 0x80000010$, $\alpha_3 = 0x80000014$)로 정해진 차분 평문 쌍을 이용하며, 각 라운드에 대한 차분 특성의 만족 여부에 따라 확률을 계산한다. 만약 i 라운드 차분 특성이 만족한다면, 누적차분특성 확률 p_{diff_i}로 차분 경로를 따르며, 해밍 웨이트 필터링을 통과한다. 반면에 i 라운드 차분 특성을 만족하지 않는다면, $(1-p_{diff_i})\times p_{hwt_i}$의 확률로 필터링을 통과한다. 따라서 정해진 입력 차분 특성에 대한 해밍 웨이트 필터링 확률 $\widehat{p_{hwt_i}}$은 식 (4)와 같이 정리할 수 있다.

$$\widehat{p_{hwt_i}} = p_{diff_i} + (1-p_{diff_i}) \times p_{hwt_i} \quad (4)$$

식 (4)를 통해 구한 해밍 웨이트 필터링 확률 $\widehat{p_{hwt_i}}$는 Table 2. 의 5번째 열의 값과 같다.

특히, 1, 2 라운드의 경우 출력 차분이 확률 1을 만족하는 워드가 존재한다. Fig. 2. 의 음영 부분은 확률 1을 만족하는 워드를 나타낸 것이고, 점선 부분은 차분 확률에 영향을 받는 워드를 나타낸 것이다. 확률 1을 만족하는 워드의 경우 해밍 웨이트 필터링에 영향을 받지 않기 때문에 그 이외의 워드와 식 (4)를 이용하여 해밍 웨이트 필터링 확률을 구해야 한다. 1 라운드 출력 차분의 경우 $X^2[1]$, $X^2[2]$ 워드를 이용하고(**property** 3.에서 $s = 0, 3$), 2 라운드 출력 차분의 경우 $X^3[0]$, $X^3[1]$, $X^3[2]$ 워드를 이용하여(**property** 3.에서 $s = 3$) 해밍 웨이트 필

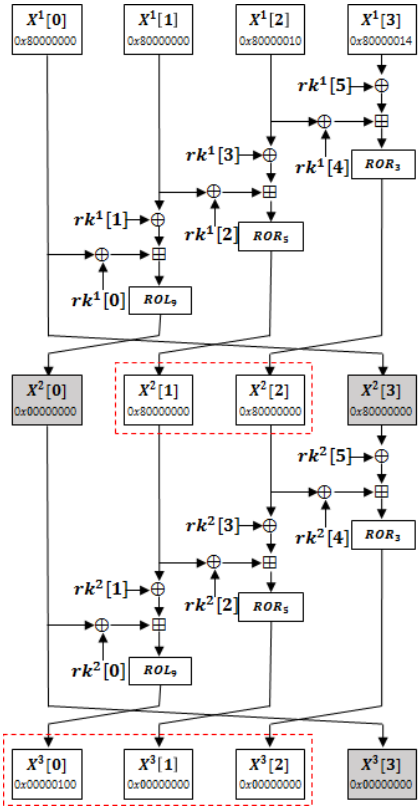

Fig. 2. Differential path of 1,2 round

터링 확률 p_{hwt_1}, p_{hwt_2}를 구한다. 3 라운드 이후부터는 전체 워드를 이용하여 해밍 웨이트 필터링 확률을 구한다.

본 논문에서 제안하는 공격에 필요한 평문 쌍을 획득하기 위해 사용하는 해밍 웨이트 필터링은 확률 $\widehat{p_{hwt_i}}$을 만족한다. 시뮬레이션 결과, 프로그래밍 실험치가 이론치 $\widehat{p_{hwt_i}}$(Table 2.)를 따라가는 것을 확인할 수 있었다.

4.2 키 복구 방법

키 복구는 4.1에서 설명한 해밍 웨이트 필터링을 통과한 평문 쌍을 이용한다. 이 때 필터링을 통과한 평문 쌍이 첫 번째 라운드 차분 경로를 따랐다면, 출력 차분은 $\omega(\omega_0 = 0x00000000,\ \omega_1 = 0x80000000,\ \omega_2 = 0x80000000,\ \omega_3 = 0x80000000)$를 만족한다. 첫 번째 라운드의 키 복구는 차분 ω를 이용하여 이루어지며, 필터링을 통과한 평문임에도 불구하고 첫

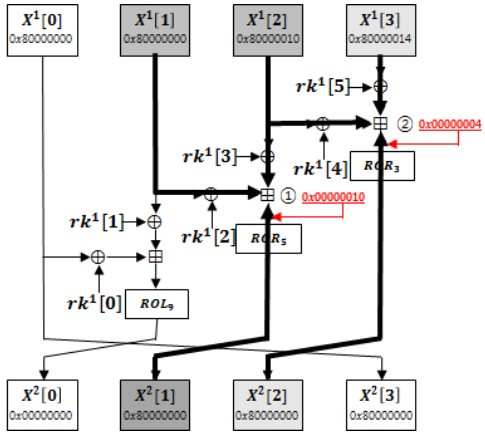

Fig. 3. 1st. round differential path for the key recovery

번째 라운드 차분 특성을 따르지 않는 평문 쌍은 키 추측 시 노이즈 평문 쌍으로 작용한다.

첫 번째 라운드 출력의 2 번째 워드 차분인 $\omega_1 = 0x80000000$는 ROR_5를 역으로 수행하면 $ROL_5(\omega_1) = 0x00000010$을 얻을 수 있다. 이를 통해 $rk^1[2]$, $rk^1[3]$의 하위 5 비트 키를 추측하여 가장 많이 카운팅 되는 값을 옳은 키로 복구한다(Fig. 3. 의 ① 경로). 또한 LEA-128 키 스케줄 특성 상 $rk^1[1] = rk^1[3] = rk^1[5]$를 만족하기 때문에 이러한 특성을 이용하여 알 수 있는 $rk^1[5]$의 값을 통해 $rk^1[4]$를 복구한다(Fig. 3. 의 ② 경로). 따라서 LEA-128의 첫 번째 라운드 키 192 비트 중에서 25 비트 키 정보를 복구할 수 있다. 실제 복구 되는 위치는 $rk^1[1]$, $rk^1[2]$, $rk^1[3]$, $rk^1[4]$, $rk^1[5]$의 하위 5비트이다.

본 논문에서 제안하는 첫 번째 라운드 키 25 비트를 복구하는 방법에 대한 구체적인 과정은 Step 1~ Step 3과 같으며, i 라운드 축소 마스킹이 적용된 LEA-128을 가정한다.

Step 1. 필터링 평문 쌍(N 개) 획득

차분 $\alpha(\alpha_0 = 0x80000000,\ \alpha_1 = 0x80000000,\ \alpha_2 = 0x80000010,\ \alpha_3 = 0x80000014)$를 만족하는 평문 쌍 P, P'에 대해 i 라운드 출력에 대한 해밍 웨이트 필터링을 통과한 평문 쌍을 획득한다. 이 때 공격에 필요한 해밍 웨이트 필터링을 통과한 평문 쌍이 N개라면, 총 평문의 개수는 $2 \times N \times (\widehat{p_{hwt_n}})^{-1}$이

Table 5. Our attack complexity

Round	# of required filtered plaintext pairs by hamming weights (N)	# of plaintext pairs following the 1^{st} round differential among N pairs	Required entire plaintext pairs	Data complexity	Time complexity
1	200	30	2^8	2^9	2^{17}
2	65	50	2^9	2^{10}	2^{18}
3	45	40	2^9	2^{10}	2^{18}
4	50	40	2^{11}	2^{12}	2^{20}
5	80	50	2^{13}	2^{14}	2^{22}
6	130	55	2^{14}	2^{15}	2^{23}

다. 여기서 i 라운드 출력 차분에 대한 해밍 웨이트 필터링 확률은 Table 2. 의 5열의 값을 이용한다.

Step 2. $rk^1[2]$, $rk^1[3]$의 부분 정보 복구

획득한 필터링 평문 쌍을 통해 식 (5)을 만족하는 $rk^1[2], rk^1[3]$의 하위 5 비트를 추측한다. 즉, $2^{10}(=2^5\times 2^5)$의 복잡도로 키를 추측한다. 실제 해밍 웨이트 필터링 평문 쌍에 포함되어 있는 노이즈로 인하여 랜덤 키가 카운팅 되는 경우가 발생하기 때문에 키 랭킹을 통해 옳은 키를 복구한다. 여기서 식 (5)은 Fig. 3. 의 ① 경로를 따른다.

$$\triangle ROL_5(X^2[1]) = ((X^1[1]\oplus rk^1[2])\boxplus (X^1[2]\oplus rk^1[3]))\oplus((X^{1'}[1]\oplus rk^1[2])\boxplus (X^{1'}[2]\oplus rk^1[3])) \quad (5)$$

Step 3. $rk^1[4]$의 부분 정보 복구

$rk^1[5](=rk^1[3])$을 이용하여 식 (6)을 만족하는 $rk^1[4]$의 하위 5비트를 추측한다.(키 랭킹 사용) 즉, 2^5의 복잡도로 키를 추측한다. 여기서 식 (6)은 Fig. 3. 의 ② 경로를 따른다.

$$\triangle ROL_3(X^2[2]) = ((X^1[2]\oplus rk^1[4])\boxplus (X^1[3]\oplus rk^1[5]))\oplus((X^{1'}[2]\oplus rk^1[4])\boxplus (X^{1'}[3]\oplus rk^1[5])) \quad (6)$$

이론상으로 올바른 1 라운드 차분을 만족하는 평문 쌍만을 이용하여 공격을 수행하면 16개 $((2^{-3})^{-1}\times 2)$의 평문 쌍으로 올바른 키를 복구할 수 있다. 하지만 해밍 웨이트 필터링을 통해 획득한 평문 쌍은 1 라운드 차분을 만족하는 쌍과 만족하지 않는 노이즈 쌍이 혼합되어 있다. 이러한 이유로 키 복구를 위해 더 많은 필터링 평문 쌍을 필요로 한다. 특히 1 라운드의 경우 차분 확률인 p_{diff_1}과 해밍 웨이트 필터링 확률 $\widehat{p_{hwt_1}}$의 차이가 크고, 이에 따라 노이즈 쌍의 비율이 더 크기 때문에 많은 필터링 평문 쌍을 이용해야 키 복구가 가능했다.

Table 3. 의 2열은 공격을 위한 해밍 웨이트 필터링 평문 쌍의 개수 N을 나타내고, 3열은 그 중 1 라운드 차분 특성을 만족하는 평문 쌍의 개수를 나타낸다. 이는 시뮬레이션을 통한 결과이다. 1~6 라운드의 키 복구를 위해 필요한 전체 평문 쌍 수는 Table 3. 의 4열, 데이터 복잡도는 5열, 시간 복잡도는 6열이다. 여기서 시간 복잡도는 (데이터 복잡도 $\times$ $2^{10.044}$ $\times$ $\frac{1}{3}$(branch rate) $\times$ $\frac{1}{24}$(round rate)를 계산한 결과이다($2^{10.044}(=2^{10}\times 2^5)$는 키 추측 양을 뜻함). 또한 획득한 필터링 평문 쌍을 이용하여 첫 번째 라운드 키 25비트를 복구하는 프로그램을 구현한 결과, Intel Core i5 CPU, 8GB 메모리의 일반 PC 환경에서 수 초 이내에 복구되는 것을 확인하였다. 본 연구결과, 7 라운드 이상 마스킹이 적용된 경우 p_{diff_i}와 $\widehat{p_{hwt_i}}$ 값의 차가 기하급수적으로 커져 필터링 이후 노이즈 평문쌍이 너무 많아져 키 랭킹 방법으로 실제 키를 찾는 것이 어렵다.

4.3 24 라운드 키 후보 결정 방법

4.1절의 각 단계에서 24 라운드의 키 후보를 구하기 위해서 오류가 주입된 암호문과 오류가 주입되

지 않은 암호문에 대한 차분을 이용한다. 효율적인 공격을 위해 오류 주입 수와 위치를 알맞게 결정하는 것이 중요하다. 제안하는 공격에서는 $X^{24}[1], X^{24}[2], X^{24}[3]$의 위치에 오류 주입을 하며, 각 32 비트의 1, 8, 16, 24번째 비트에 오류가 주입된 암호문을 사용한다.

먼저 24번째 비트에 오류가 주입된 암호문에 대한 차분 형태를 통해 하위 8 비트에 대한 라운드 키 후보를 결정할 수 있다. 이 때 주입된 오류를 통해 법덧셈에서 캐리가 발생할 수 있기 때문에 식 (3), (7), (11)의 차분 확인 과정에서 23번째 비트 부분까지의 확인이 필요하다. 24번째 비트에 오류 주입된 암호문 25개를 이용하면 하위 8 비트의 키 후보를 유일하게 결정할 수 있다. 그 다음 16번째 비트에 오류가 주입된 암호문 25개와 유일하게 결정된 하위 8 비트 정보를 이용하여, 16~23번째 비트의 키 후보를 유일하게 결정한다. 8~15번째 비트의 키 후보도 8번째 비트에 오류 주입된 암호문을 이용하여 동일한 방법으로 유일하게 결정한다. 마지막 0~7번째 비트의 키 후보를 결정하기 위해 1번째 비트에 오류 주입된 암호문을 사용한다. 0번째 비트에 오류 주입을 할 경우 법덧셈에 대한 캐리가 일어난 차분을 확인할 수 없기 때문에 많은 후보가 발생하나, 1번째 비트에 오류 주입된 암호문 25개를 사용할 경우 0~7번째 비트의 키 후보가 2개 남는다. 이런 방식으로 각 단계마다 약 $100(=25\times4)$개, 총 300개의 오류 주입 암호문으로 24 라운드 키 정보에 대한 후보를 2^3개($rk^{24}[0]$의 후보 개수 $\alpha=2$, $rk^{24}[1]\oplus rk^{24}[2]$의 후보 개수 $\beta=2$, $rk^{24}[3]\oplus rk^{24}[4]$의 후보 개수 $\gamma=2$), 24 라운드 키 후보 2^{35}개를 얻을 수 있다.

V. 결 론

본 논문에서는 처음 1~6 라운드 축소 마스킹 기법이 적용된 LEA-128에 대한 부채널 공격을 제안하였다. 공격을 통해 첫 번째 라운드 키 25비트를 복구하였다. 실제 공격에서의 데이터 복잡도와 시간 복잡도는 축소 마스킹이 적용된 라운드에 따라 차이가 있으며, 각 라운드에 대한 복잡도는 Table 3. 의 5, 6열의 값과 같다. 실제 키 복구 시뮬레이션을 통해 수 초 이내에 키 정보를 복구가 가능하다. 이를 통해 6 라운드 이하의 축소 마스킹이 적용된 LEA는 부채널 공격에 취약하다는 사실을 알 수 있다. 따라서 부채널 공격에 내성을 갖기 위해서는 7 라운드 이상의 마스킹 기법을 적용해야 한다.

References

[1] P.C. Kocher, J. Jaffe, and B. Jun, "Differential Power Analysis," CRYPTO 1999, LNCS 1666, pp. 388-397, Dec. 1999

[2] C.K. Koc and C. Paar, "On Boolean and Arithmetic Masking against Differential Power Analysis," CHES 2000, LNCS 1965, pp. 231-237, Aug. 2000

[3] J. Park, D. Hong, D. Kim, D. Kwon, and H. Park, "128-Bit Block Cipher LEA," TTAK.KO-12.0223, Dec. 2013

[4] HeeSeok Kim, Seokhie Hong, and Jongin Lim, "A Fast and Provably Secure Higher-Order Masking of AES S-Box," CHES 2011, LNCS 6917, pp. 95-107, Oct. 2011

[5] Jongsung Kim, Seokhie Hong, Dong-Guk Han, and Sangjin Lee, "Improved Side-Channel Attack on DES with the First Four Rounds Masked," ETRI Journal, 31(5), pp. 625-627, Oct. 2009

[6] Jongsung Kim, Yuseop Lee, Sangjin Lee, "DES with any reduced masked rounds is not secure against side-channel attacks," Computers & Mathematics with Application, vol. 60, no. 2, pp. 347-354, July 2010

[7] J. Kim, S. Hong, "Side Channel Attack using Meet in the Middle Technique," The Computer Journal, vol. 53, no. 7, pp. 934-938, June 2009

[8] Deukjo Hong, Jung-Keun Lee, Dong-Chan Kim, Daesung Kwon, Kwon Ho Ryu, and Dong-Geon Lee, "LEA: A 128-Bit Block Cipher for Fast Encryption on Common Processors," WISA 2013, LNCS 8267, pp. 3-27, Nov. 2013

[9] Stefan Mangard, Elisabeth Oswald, and Thomas Popp : Power Analysis

Attacks(Revealing the Secrets of Smart Cards)," pp. 105-107, 2007

〈저자소개〉

박 명 서 (Myungseo Park) 학생회원
2013년 2월: 국민대학교 수학과 학사
2013년 3월~2015년 2월: 국민대학교 금융정보보안학과 석사
2014년 12월~현재: ETRI 부설연구소 연구원
〈관심분야〉 정보보호, 암호 알고리즘, 이동통신보안

김 종 성 (Jongsung Kim) 종신회원
2000년 8월/2002년 8월: 고려대학교 수학 학사/이학석사
2006년 11월: K.U.Leuven, ESAT/SCD-COSIC 정보보호 공학박사
2007년 2월: 고려대학교 정보보호대학원 공학박사
2007년 3월~2009년 8월: 고려대학교 정보보호기술연구센터 연구교수
2009년 9월~2013년 2월: 경남대학교 e-비즈니스학과 조교수
2013년 3월~현재: 국민대학교 수학과 조교수
2014년 3월~현재: 국민대학교 일반대학원 금융정보보안학과 조교수
〈관심분야〉 정보보호, 암호 알고리즘, 디지털 포렌식