균일한 해밍웨이트를 제공하는 소프트웨어 AES에 대한 부채널 분석*)

원유승,[1] 한동국,[1] 최두호[2]
[1]국민대학교 수학과, [2]한국전자통신연구원
e-mail:[1]{mathwys87, christa}@kookmin.ac.kr, [2]dhchoi@etri.re.kr

Side Channel Analysis of the S/W AES with Uniform Hamming Weight Representation

Yoo-Seung Won,[1] Dong-Guk Han,[1] Dooho Choi[2]
[1]Dept. of Mathematics, Kookmin University, [2]ETRI

요 약

암호 알고리즘이 탑재된 환경에서 암호 알고리즘의 이론적 안전성이 고려되어도 환경에 의존한 부가적 정보를 활용하는 부채널 분석에 대한 안전성이 검토되어야 한다. 최근까지 부채널 분석에 대한 안전성을 고려한 대응기법으로 마스킹 기법이 적용되었으나, 이와는 상반된 개념인 하드웨어 DPL(Dual-rail with Precharge Logic) 기법을 응용한 균일한 해밍웨이트를 제공하는 소프트웨어 AES(Advanced Encryption Standard)가 제안되었다. 최근, 소프트웨어 기반 블록암호에 대해 고차 마스킹 부채널 대응법의 비효율성으로 새로운 방법에 대한 다양한 시도가 되고 있으며, 그 중 균일한 해밍웨이트를 제공하는 표현 방법이 효율적이고 안전한 새로운 대응법으로 검토되어지고 있다. 하지만, 논문에서는 균일한 해밍웨이트 데이터 표현방법 기반 부채널 대응법을 해독하는 차분전력분석 방법을 소개한다. 실험을 통해, AES 128비트 키 중 일부분이 분석됨을 확인하였다. 이는 공격자가 테이블 변환 정보를 활용할 수 있다는 다소 강력한 가정하에 실험하였기 때문이다. 앞선 가정 하에 안전성을 제공하기 위해서는 차후 추가적 대응기법이 고려되어야 한다.

1. 서론

암호 알고리즘의 안전성을 보장하는 통계적 분석, 대수적 분석 등과는 달리 암호 알고리즘이 탑재된 환경을 이용하여 부가적인 정보를 분석하는 부채널 분석이 1996년 Paul Kocher에 의해 도입되었다.[1] 그 이후 전력, 전자파 측정과 같은 부가적인 정보를 공격자가 활용하지 못하도록 부채널 분석에 대한 대응기법이 제안되었다. 대응기법 중 최근 많이 사용되어지는 n차 마스킹 기법[2]은 암호 연산이 수행되어지는 하나의 데이터 중간 값을 n개로 랜덤 분할하여 암호 연산을 수행한다. 하지만 이론적으로 추측되어지는 중간 값이 측정된 파형에 전력소모 패턴이 남아있기 때문에 n차 이하 상관전력분석 공격에 대하여 증명가능한 안전성(Provably Security)을 보증하지만 $(n+1)$차 상관전력분석공격에 대하여 취약하다. 하지만 실제로 암호 알고리즘의 속도를 고려하지 않는다면 2차 이상 대응기법이 적용될 경우 분석을 위한 많은 파형이 필요하여 계산적 안전성(Computational Security)을 보증할 수 있다. 그러나 [2]에서 마스킹 대응 기법을 블록 암호 알고리즘 AES(Advanced Encryption Standard)[3]에 적용하면 2차 마스킹 경우 약 23배, 3차 마스킹은 약 41배가 느려진다. 따라서 부채널 분석에 대한 안전성과 연산속도와 같은 효율성을 고려한 대응기법이 필요하다.

전통적인 대응기법인 마스킹 기법과는 상반된 개념인 하드웨어 대응 기법으로 DPL(Dual-rail with Precharge Logic)이 제안되었다.[4] 이는 전력소모가 임의의 값에 대하여 일정하게 나타나게 하는 부채널분석에 대한 대응기법이다. 그러나 이 또한 하드웨어로 구현되어져야 한다는 한계점을 벗어나지 못하였지만 최근 하드웨어로 구현된 DPL과 같은 효과를 주는 소프트웨어 구현법이 제안되었다.[5][6] 즉, 소프트웨어로 구현된 암호 알고리즘의 중간 값이 해밍웨이트에 비례하다는 정보를 활용하지 못하게 한다. 연산 속도 측면에서도 효율성이 떨어지지 않고 마스킹 차수에 의존하지 않는다는 장점을 가진다. 그러나 부채널분석 공격에 대한 충분한 실험적 분석 결과가 연구되지 않아, 본 논문에서 실제 공격자가 적용된 대응 기법을 알고 있다고 가정한 후, 균일한 해밍웨이트를 제공하는 소프트웨어 AES에 대해 부채널 분석 대응법의 취약성을 검토하였다. 적용된 대응 기법에 대해 차분 전력 분석을 수행한 결과, 공격자가 대응 기법에 대한 테이블 변환 정보를 안다는 가정 하에 실험을 하여 균일한 해밍웨이트를 제공

*) 본 연구는 ETRI의 연구개발과제인 K-SCARF 프로젝트로 수행하였음(암호키 누출 검증 및 방지 원천 기술 연구)

하는 소프트웨어 AES 키의 일부분이 분석되었다. 또한, 분석된 일부분의 키와 분석되지 않는 키에 대해서도 해석한다.

2. 선행 연구

2.1 소비전력 모델

암호 알고리즘이 탑재된 장치에서 연산 처리가 수행되어질 때, 암호 알고리즘이 소프트웨어로 구현된 경우 데이터 값에 비례하여 전력소모가 일어난다. 즉, 데이터를 비트 표현으로 나타내었을 때, 비트 표현이 '1'인 경우가 '0'인 경우보다 더 많은 전력 소모가 일어난다는 사실에 기반을 둔다.[7]

2.2 차분전력분석공격

차분전력분석 공격은 각각의 가능한 키 후보군을 추측하여 암호 연산의 소비전력모델에 따라 파형 상에 나타날 것으로 예상되는 중간 값을 연산한 후, 분류함수를 적용한다. 1비트 추측일 경우 각각 가능한 키 후보군에 대해서 해밍웨이트 0과 1로 기준을 적용하여 수집된 전력파형을 두 개의 집합으로 분류한다. 각각 분류된 집합의 전력파형에 대해 평균을 내어 차분을 구함으로써 추측한 키가 옳은지에 대하여 판단할 수 있다. 추측한 키가 옳은 키라면 평균의 차는 0이 아닌 수가 나타나고, 틀린 키라면 0 값으로 나타난다. 또한 분류함수를 다수 비트로 확장하면 1비트씩 평균 차분 값을 합하여 옳은 키와 틀린 키를 구분짓는다. 다수 비트를 수행하기 위한 분류함수는 다음과 같은 식으로 나타낼 수 있다.

$$A_0^m[j] = \frac{1}{|S_0|^m} \sum_{S_i^m[j] \in S_0^m} S_i^m[j]$$

$$A_1^m[j] = \frac{1}{|S_1|^m} \sum_{S_i^m[j] \in S_1^m} S_i^m[j] \text{ where } |S_0^m| + |S_1^m| = N$$

N: 파형 수

l: 공격하고자 하는 총 비트 수 ($1 \leq m \leq l$)

j: 추측 키 (n비트 키인 경우 $0 \leq j \leq 2^n - 1$)

S_0^m: m번째 비트 해밍웨이트 값이 0

S_1^m: m번째 비트 해밍웨이트 값이 1

A_0^m와 A_1^m으로 분류 후 차분 값인 $T^m[j] (= A_1^m[j] - A_0^m[j])$를 구하여 $T[j] = \sum_{m=1}^{l} T^m[j]$을 계산한다. 만약 옳은 키라면 $T[j]$가 0이 아닌 수가 나타나고 틀린 키라면 0인 수가 나타난다.

2.3 균일한 해밍웨이트를 제공하는 소프트웨어 AES 구현

[5]에서 소개된 균일한 해밍웨이트를 제공하는 소프트웨어 AES를 구현하기 위해서는 치환 표현 및 데이터 연산에 대한 선행 연구가 필요하다.

2.3.1 n비트 데이터 치환 표현

n비트 데이터를 균일한 해밍웨이트 값 w를 갖는 t비트로 치환하는 방법은 다음과 같은 식을 만족하게 구성해야 한다.

$$2^n \leq {}_tC_w$$

n: 치환 하기 전 데이터 비트 수,

t: 치환 후 데이터 비트 수,

w: 치환 후 균일하게 구성할 해밍웨이트 수

즉, $n=4$, $t=6, w=3$ 을 택하면 4비트 데이터를 해밍웨이트 3을 갖는 6비트 데이터로 나타낸다. 간단히 표로 나타내면 다음과 같다.

<표 1> 해밍웨이트 3을 갖는 6비트 치환 값
(이진수 표현)

No.	치환 값	No.	치환 값
1	000111	11	100011
2	001011	12	100101
3	001101	13	100110
4	001110	14	101001
5	010011	15	101010
6	010101	16	101100
7	010110	17	110001
8	011001	18	110010
9	011010	19	110100
10	011100	20	111000

4비트는 총 16가지(0~F)의 경우의 수를 가지므로 위의 <표 1>에서의 20가지 중 치환 값 16개를 임의로 선택하여 대체한다.

2.3.2 치환에 의한 데이터 연산

균일한 해밍웨이트를 제공하는 소프트웨어 AES를 구현을 하기 위해서는 XOR, S-Box, $GF(2^8)$에서의 2배·3배 연산이 필요하다. 하지만, 기본 데이터 값을 <표 1>에서와 같은 치환을 적용하면 모든 일반 연산이 적용되지 않기 때문에 치환이 적용되어도 옳은 연산 값이 나오게 구성해야 한다. 치환 전의 두 데이터가 x, y 이고, 치환 후 값을 $Sub(x)$, $Sub(y)$라 하면 다음과 같이 필요한 연산에 대하여 테이블 참조를 이용하여 구성한다.

$$Table(Sub(x) \| Sub(y)) = Sub(x \circledcirc y)$$

$\circledcirc$: 연산 (XOR, S-Box, $GF(2^8)$에서 2배·3배 연산)

$Table(\cdot)$: $\circledcirc$ 연산 출력이 구현된 테이블

<표 1>에서 No. 6, 8, 10, 15를 제외하고 4비트 데이터 값 0~F까지 차례로 치환하여 적용한 예를 들면 테이블은 XOR, S-Box, $X2(GF(2^8)$에서의 2배 연산), $X3(GF(2^8)$에서의 3배 연산)에 대해 필요하지만 이를 4비트 출력(치환 적용 6비트)으로 구현할 경우 XOR를 제외하고 각 연산에 대해서 테이블이 2개씩 필요하여, 총 7개의 테이블이 필요하다. 즉, 다음과 같이 테이블을 구현한다.

$$XOR(sub(x) \| sub(y)) = sub(x \oplus y)$$
$$S-Box_H(sub(x_H) \| sub(x_L)) = sub(S-Box(x)_H)$$
$$S-Box_L(sub(x_H) \| sub(x_L)) = sub(S-Box(x)_L)$$
$$X2_H(sub(x_H) \| sub(x_L)) = sub(\{02\} \bullet x)_H)$$
$$X2_L(sub(x_H) \| sub(x_L)) = sub(\{02\} \bullet x)_L)$$
$$X3_H(sub(x_H) \| sub(x_L)) = sub(\{03\} \bullet x)_H)$$
$$X3_L(sub(x_H) \| sub(x_L)) = sub(\{03\} \bullet x)_L)$$

$\cdot_H$: 실제 연산 값 $2n$비트 중 상위 n비트 값

$\cdot_L$: 실제 연산 값 $2n$비트 중 하위 n비트 값

$\bullet$: AES에서의 MixColumns 수행 시 유한체 곱

이는 1개의 테이블 당 (입력의 최대 값+1)=(111000111000_2+1)이므로 3641 바이트가 필요하다. 따라서 총 3641×7＝25487바이트가 필요하다.

3. 균일한 해밍웨이트를 제공하는 소프트웨어 AES 구현에 대한 차분전력분석공격

3.1 차분전력분석공격을 위한 실험환경

아래 <표 2>와 같은 실험조건을 적용하여 차분전력분석공격을 적용하였다.

<표 2> 차분전력분석공격 실험환경

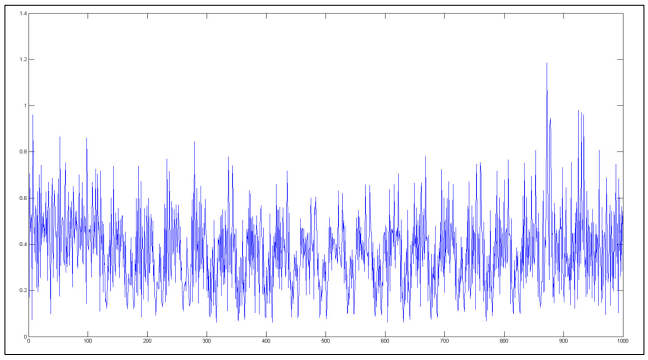
실험조건	
분석공격	6비트 차분전력분석공격
공격타겟	AES S-Box 출력
알고리즘 탑재된 보드	ARM 보드
수집파형개수	50,000개
치환 표현	4비트 데이터를 6비트 표현 치환 (해밍웨이트 3)

치환 표현은 <표 1>에서 No. 6, 8, 10, 15를 제외하고 4비트 데이터 값 0~F까지 차례로 치환하여 적용하였다.

3.2 차분전력분석공격 실험결과

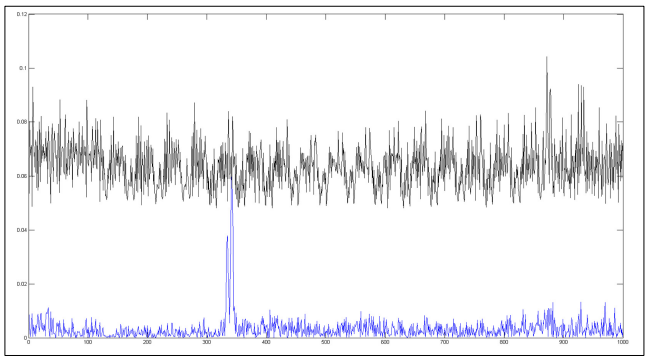
구현된 AES의 키는 000102030405060708090A0B0C0D0E0F 값으로 적용하였다.

(그림 1)은 분석에 쓰인 전력파형을 나타내고 (그림 2, 3)은 6비트 차분전력분석공격에 대한 5번째 AES S-Box 분석 결과이다.



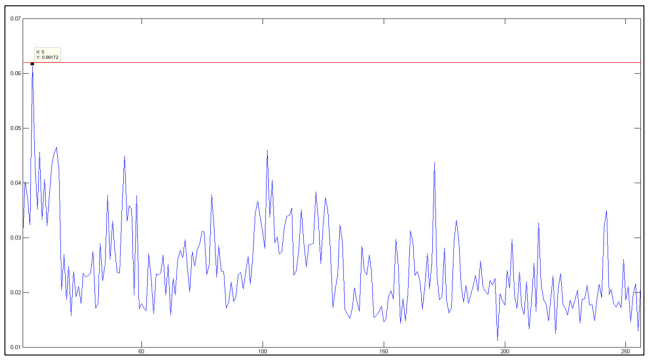
(그림 1) 균일한 해밍웨이트를 제공하는 소프트웨어 AES S-Box 전력파형

(그림 1)은 차분전력분석공격을 위해 균일한 해밍웨이트를 제공하는 소프트웨어 AES에서 실제 공격에 쓰인 S-Box 부분을 나타낸다. x축은 시간, y축은 소비된 전력을 나타낸다.



(그림 2) 5번째 바이트 분석 위치 (키 : 04)

(그림 2)에서 상위 위치한 파형은 (그림 1)에서의 파형이고, 아래의 파형은 5번째 바이트가 전력파형 상에서 분석된 위치를 나타낸다. x축은 시간, y축은 6비트 차분 평균의 합을 나타낸다.



(그림 3) 5번째 바이트에 대한 분석결과 (키 : 04)

(그림 3)에서 x축은 256개의 키 후보를 나타내고 y축은 6비트 차분 평균의 합을 나태난다. 즉, 256개의 키 후보 중 옳은 키에 대한 5번째 바이트 키(붉은 선)가 틀린 키보다 6비트 차분 평균 값의 합이 상위에 있다는 걸 나타낸다. 또한, 실제 분석이 이루어진 바이트는 3, 5, 9, 10, 13, 14, 15, 16번째 바이트(총 8바이트)가 분석이 이루어지고 1, 2, 4, 6, 7, 8, 11, 12번째 바이트(총 8바이트)가 옳은 키가 나타나지 않았다.

3.3 차분전력분석공격 실험결과 분석

하드웨어 DPL기법을 소프트웨어 AES에 적용하면 이에 따른 데이터 처리는 해밍웨이트 특성을 따른다. 따라서 4비트 표현 방식을 해밍웨이트가 균일하게 6비트로 제공한다 하여도 단일 비트 해밍웨이트는 0과 1로 분류된다. 따라서 전체 6비트 기준으로 해밍웨이트가 일정하여도 단일 비트 기준으로는 균일한 해밍웨이트를 제공하지 못한다. 이러한 이유 때문에 균일한 해밍웨이트를 제공하는 소프트웨어 AES에 대해 실험결과 AES의 키 8바이트가 분석되었다.

키 바이트 분석 기준은 Ratio 1.0 (Ratio: $\frac{MAX(\text{틀린 키 평균차분 값})}{\text{옳은 키 평균차분 값}}$)을 적용하였다. 아래 <표 3>은 키 바이트가 분석된 경우인 3, 5, 9, 10, 13, 14, 15, 16번째 바이트에 대해서는 분석결과를 해석하기 위해서는 Ratio 값을 나타내고, 그렇지 못한 경우 옳은 키가 분석된 틀린 키 범주에 속하는지 알아보기 위하여 Reverse Ratio(Ratio의 역수) 값을 나타낸다.

<표 3> 차분전력분석공격 실험환경

분석 바이트	Ratio	Reverse Ratio
1	-	1.0929
2	-	1.1724
3	1.0755	-
4	-	1.0325
5	1.3276	-
6	-	1.0683
7	-	1.0693
8	-	1.1116
9	1.2339	-
10	1.0777	-
11	-	1.0636
12	-	1.2611
13	1.1616	-
14	1.2237	-
15	1.0726	-
16	1.0702	-

Reverse Ratio는 Reverse Ratio 내에 존재하는 틀린 키의 수가 적을 때 키 노출 위험성을 나타낼 수 있다. 12번째 바이트를 제외하고는 Reverse Ratio 내에 모두 6~7개만의 틀린 키가 존재한다. 따라서 실제로 공격자가 모든 키 바이트가 분석되지 않아도 각 바이트에 대한 부분 키 후보군 몇 개를 선정하여 옳은 평문, 암호문 쌍에 적용하여 키를 찾아낼 수 있다. 즉, 공격자가 앞선 기준으로 키를 분석한다면 12번째 키를 제외하고 공격자에 의해 모든 바이트 키가 분석되었다고 할 수 있다.

4. 결과

스마트 디바이스에 탑재된 암호 알고리즘의 부채널 분석에 대한 대응기법으로 하드웨어 DPL 기법을 소프트웨어로 구현하여 균일한 해밍웨이트를 제공하는 소프트웨어 AES가 소개되었다. 하지만 이는 공격자가 치환 표현을 모두 안다는 가정 하에 차분전력분석공격을 적용한 결과, 키의 일부분이 분석되었다. 따라서 앞선 가정 하에 안전성을 제공하기 위해서는 균일한 해밍웨이트를 제공하는 AES를 차분전력분석 공격에 대해 현실성을 고려한 추가적 대응기법이 필요하다. 그러나 공격자가 치환 표현을 안다는 가정을 제외한다면 차수에 의존하는 일반적인 대응기법인 n차 마스킹 기법보다 효율적일 것으로 기대된다.

참고문헌

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

[2] 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, 2011.

[3] NIST, FIPS 197: Advanced Encryption Standard , 2001

[4] S. Guilley, L. Sauvage, F. Flament, P. Hoogvorst, and R. Pacalet. “Evaluation of Power-Constant Dual-Rail Logics Counter-Measures against DPA with Design-Time Security Metrics.” IEEE Transactions on Computers 2010, Vol. 59, No. 9, pp. 1250-1263, 2010

[5] P. Hoogvorst, G. Duc, and J.-L Danger. “Software Implementation of Dual-Rail Representation.” COSADE 2011, pp. 73-81, 2011.

[6] Akira MAEKAWA, Noritaka YAMASHITA, and Toshihiko OKAMURA, “Tamper-Resistance Techniques Based on Symbolic Implementation Against Power Analysis”, SCIS 2013.

[7] E. Brier, C. Clavier, and F. Olivier. “Correlation power analysis with a leakage model.” CHES 2004, pp. 16-29, 2004.