마스킹 기반 대응방안에 대한 1차 DPA 취약성 분석

김창균*, 유형소**, 박일환*, 문상재**

*국가보안기술연구소

**경북대학교 전자공학과

The Vulnerability of a Masking based Countermeasures against 1st-order Differential Power Analysis

ChangKyun Kim*, HyungSo Yoo**, IlHwan Park*, SangJae Moon**

*National Security Research Institute.

**Department of Elec. Eng., Kyungpook National University.

요 약

P. Kocher에 의해 DPA 공격이 소개된 이후 이를 방어하기 위한 연구가 활발하게 진행되고 있다. 그에 대한 일환으로 블록암호알고리즘 구현 시 소프트웨어적인 대응방안으로 마스킹 기반의 대응기법이 많이 사용되고 있으며 이는 1차 DPA 공격에 안전한 것으로 인지되어 왔다. 본 논문에서는 부주의하게 구현된 마스킹 기반의 대응기법이 2차 DPA 공격이 아닌 1차 DPA 공격에도 취약한 사실을 증명하였으며 이를 실험을 통해서 검증하였다.

I. 서론

P. Kocher에 의해 하드웨어 암호모듈에 대한 전력분석공격(power analysis attack)이 처음으로 소개된 이후 다양하고 깊이 있는 연구가 진행되고 있다 [1]. 특히 스마트카드를 대상으로 하는 전력분석공격은 암호시스템 분석 측면에서 볼 때 매우 실질적인 공격으로 간주되고 있으며 다수의 실험결과가 보고되고 있다. 이에 따라 학계는 물론 암호모듈을 제작하는 업계에서 전력분석공격에 안전한 암호모듈을 구현하기 위해 다양한 각도로 대응방법을 연구하고 있다.

대응방법에는 크게 하드웨어적인 접근법과 소프트웨어적인 접근법이 있다. 하드웨어적인 접근법은 구현하고자 하는 암호 알고리즘에 독립적인 방법이므로 별도의 대응방법이 고려된 암호 알고리즘을 설계하지 않아도 되는 장점이 있다 [2, 3]. 하지만 기존의 하드웨어 플랫폼을 사용할 수 없을 뿐만 아니라 별도의 셀 라이브러리를 만들어야 하는 등 호환성 문제 및 비용적인 측면에서 다소 어려움을 가지고 있다. 반면 소프트웨어적인 접근법은 별도의 대응방법이 고려된 암호 알고리즘을 설계하면 되기 때문에 기존 하드웨어 플랫폼과 호환성이 좋고 구현 비용이 매우 저렴한 장점을 지니고 있어 하드웨어 접근법 보다 더욱 활발하게 연구되고 있다 [4-6]. 특히 비밀키 알고리즘에서 가장 활발히 연구되고 있는 소프트웨적인 접근법이 마스킹을 이용한 대응기법이다. 비록 2차 DPA(differential power analysis) 공격에 취약한 점이 있지만 구현하기가 간단하고 1차 DPA에 매우 효과적이기 때문에 널리 사용되고 있다.

본 논문에서는 마스킹 기반의 대응기법을 부주의하게 구현할 경우 2차 DPA 공격이 아닌 1차 DPA 공격에도 여전히 취약한 사실을 증명하였다. 또한 마스킹 기반의 대응기법이 적용된 8비트 스마트카드를 대상으로 실험을 통해 이

를 검증하였다.

II. ARIA 알고리즘에 대한 1차 DPA

대부분의 블록암호 알고리즘은 테이블 참조 연산, 대수 연산, 산술 연산 등으로 이루어져 있다. ARIA는 국내표준 블록암호알고리즘으로서 128비트 블록크기와 가변키(128, 192, 256비트)크기를 가지는 Involutional SPN 구조를 가진다 [7]. 최근 ARIA에 대한 DPA 공격 취약성 및 안전성 연구가 진행되고 있다 [5,6,8,9].

ARIA 알고리즘에 대한 1차 DPA 공격 실험에서 사용된 소비전력모델은 비트의 상태변화(0→1 또는 1→0)에 기반을 둔 Hamming distance 모델을 적용하였다. 또한 실험에 사용된 스마트카드는 ATmega163 프로세서가 탑재된 8비트 스마트카드이다.

1. AddRoundKey에 대한 1차 DPA 공격

AddRoundKey는 128비트 입력평문과 128비트 라운드 키가 XOR하는 연산이다. 이를 C로 구현하면 다음과 같다.

```
AddRoundKey(){
  output_data[0] ^= round_key[0];
  output_data[1] ^= round_key[1];
  ...
  output_data[15] ^= round_key[15];
}
```

그림 1. AddRoundKey 연산에 대한 C 코드

(그림 1)에서 Hamming distance 모델을 적용하기 위해서 i번째 8비트 XOR 출력과 $i+1$번째 8비트 입력평문 사이의 비트 상태변이에 초점을 맞추었다. 따라서 분류함수는 수식 (1)과 같이 표현될 수 있다.

$$HW(x[i] \oplus k[i] \oplus x[i+1]) \qquad (1)$$

여기서 x와 k는 각각 output_data와 round_key를 의미한다.

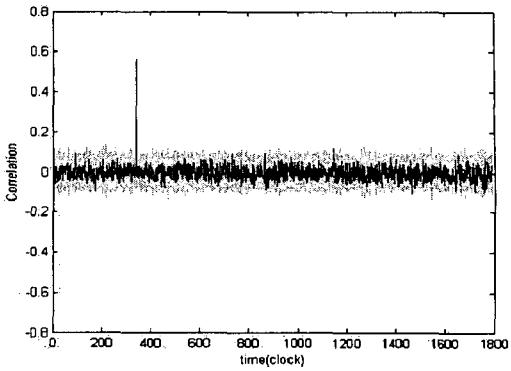

그림 2. AddRoundKey에 대한 DPA 결과

(그림 2)에서 검정색 파형은 올바른 키 8비트 추측 시 상관계수이며 회색 파형은 나머지 255가지에 대한 상관계수이다. 회색 파형을 살펴보면 틀린 키 추측 시에도 피크가 발생함을 알 수 있는데 이는 실제 키의 보수에 해당되는 값이다. 따라서 공격자는 상관계수의 부호 등을 보고 실제 키 값을 구별해야 한다.

2. SubBytes에 대한 1차 DPA 공격

통상 SubBytes는 알고리즘의 효율성을 고려하여 미리 계산하여 저장된 테이블을 참조도록 구현된다. 즉, 입력 x에 대해 출력 $y = Table[x]$를 결과로 얻는 연산이다. 이를 C로 구현하면 다음과 같다.

```
SBOX_Table_Lookup(){
  t[0] = S[0][output_data[0];
  t[1] = S[1][output_data[1];
  ...
  t[15] = S[3][output_data[15];
}
```

그림 3. SubBytes 연산에 대한 C 코드

우리는 AddRoundKey에 대한 1차 DPA 공격과 유사하게 공격에 사용될 분류함수를 수식 (2)와 같이 가정하였다.

$$HW(S[x[i] \oplus k[i]] \oplus (x[i] \oplus k[i])) \qquad (2)$$

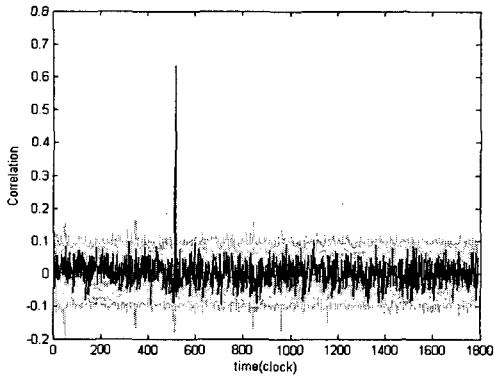

그림 4. SubBytes에 대한 DPA 결과

(그림 4)에서 검정색 파형은 올바른 키 8비트 추측 시 상관계수이며 회색 파형은 나머지 255가지에 대한 상관계수이다. 결국 AddRoundKey와 마찬가지로 1차 DPA 공격을 통해 128비트 라운드 키를 찾을 수 있다.

3. 기존 마스킹 기반의 대응기법

앞에서 설명한 바와 DPA 공격에 취약한 문제점을 해결하기 위하여 여러 가지 대응방안을 사용한다. 블록암호알고리즘 경우 마스킹 기반의 대응기법을 널리 사용하는데, 각 연산에 따라 크게 테이블 참조, 비트별 연산(XOR, AND, OR), 쉬프트 및 로테이트, 모듈러 연산, 선형 변환 등으로 나눌 수 있다 [4,6]. 특히 가장 많이 사용되는 XOR 연산과 테이블 참조 연산은 DPA 공격에 안전한 블록암호알고리즘 구현에 매우 중요한 요소로 작용한다.

마스킹된 데이터를 피연산자로 하는 XOR 연산은 단순히 두 개의 피연산자에 대한 XOR를 그대로 수행하면 된다. 즉 $x' = x \oplus r_x$, $y' = y \oplus r_y$라고 했을 때 $z' = x' \oplus y'$가 되고 z'의 마스크는 $r_z = r_x \oplus r_y$가 된다.

테이블 참조 연산에 대한 마스킹을 수행하기 위해서는 테이블 자체를 새로운 테이블로 변경해야 한다. 테이블을 마스킹하기 위한 가장 쉬운 방법은 수식 (3)과 같이 입력 마스크 m과 출력 마스크 m'을 사용한, 가산(additive) 마스킹 방법이다.

$$T'[x \oplus m] = T[x] \oplus m' \quad (3)$$

수식 (3)에서 입력과 출력 마스크 값을 동일한 값으로 사용할 경우, 구현방법에 따라 1차 DPA 공격에 취약할 수 있다고 [6]에서 언급하고 있지만 그에 대한 자세한 이론적 분석은 제시하지 않고 있다. 다음 장에서는 동일한 입·출력 마스크 값을 사용할 경우 1차 DPA 공격에 취약함을 이론적 근거와 실험을 통하여 설명할 것이다.

III. 마스킹 기반 대응기법의 취약성

마스킹 기반의 대응기법은 비록 2차 DPA 공격에 취약한 단점이 있지만 1차 DPA 공격을 방어하는데 효과적이며 비교적 단순하게 구현할 수 있다. 따라서 2차 또는 고차 DPA 공격을 방어하기 위해 마스킹 기반의 대응방법과 함께 다양한 방법의 랜덤화 방법을 동시에 사용된다. 하지만 마스킹 기반의 대응기법을 부주의하게 구현할 경우 2차 DPA 공격이 아닌 1차 DPA 공격에도 매우 취약할 수 있다. 다음 절에서는 마스킹 기반의 대응기법을 적용했음에도 불구하고 1차 DPA 공격에 취약한 2가지 경우에 대해서 설명할 것이다.

1. 마스킹된 AddRoundKey에 대한 1차 DPA 취약성

AddRoundKey에는 입력평문과 라운드 키간 XOR 연산이 이루어진다. 이 부분에 마스킹을 적용하기 위해 (그림 5)와 같이 8비트 랜덤 마스크 m을 이용하여 입력평문과 XOR한다.

```
Input_data_Masking(){
  output_data[0] ^= rand_mask;
  output_data[1] ^= rand_mask;
  ...
  output_data[15] ^= rand_mask;
}
```

그림 5. 입력평문에 대한 마스킹

마스킹된 입력평문은 (그림 1)와 같이 라운

드 키와 XOR 연산을 거친다. 이 경우 동일한 마스크 값으로 마스킹된 값이 연속적으로 데이터 버스를 이동함으로써 마스킹이 벗겨지는 효과가 발생하게 된다. (그림 1)을 어셈블리로 변환한 코드를 보면 그 이유에 대해 알 수 있다.

```
AddRoundKey(){
01: MOVW   R30,  R18;
02: LD     R27,  Z  ;  //Load output_data[0]
03: MOVW   R30,  R16;  //Assign address of round
                          _key
04: LD     R26,  Z  ;  //Load round_key
05: EOR    R27,  R26;  //XOR output_data[0] &
                          round_key[0]
06: MOVW   R30,  R18;  //Assign address of
                          output_data
07: ST     Z,    R27;  //Store the result of
                          XOR
08: MOVW   R30,  R18;  //Assign address of
                          output_data
09: LDD    R27,  Z+1;  //Load output_data[1]
10: MOVW   R30,  R16;  //Assign address of round
                          _key
...
11: STD    Z+15, R27;  //Store the result of XOR
}
```

그림 6. 입력평문에 대한 마스킹

공격자는 수식 (1)에서 정의한 것처럼 $x[i]\oplus k[i]$와 $x[i+1]$ 사이의 비트 변화가 발생하는 부분을 찾아야한다. (그림 6)에서 XOR 결과인 $x[i]\oplus k[i]$이 저장되는 ST 명령어(라인 7) 다음에 실행되는 명령어중 버스라인에 영향을 줄 수 있는 명령어는 $x[i+1]$이 로드되는 라인 9의 LDD 명령어이다. 이 경우 다음 수식에 의해 마스킹이 벗겨지는 효과가 발생함을 알 수 있다.

$$(x'[i]\oplus k[i])\oplus x'[i+1] = (x[i]\oplus m\oplus k[i])\oplus x[i+1]\oplus m = x[i]\oplus k[i]\oplus x[i+1]$$

따라서 공격자는 마스크에 상관없이 대응방법이 없는 AddRoundKey에 대한 1차 DPA 공격과 동일하게 공격할 수 있다. (그림 7)은 마스킹된 AddRoundKey에 대한 1차 DPA 공격 결과이다.

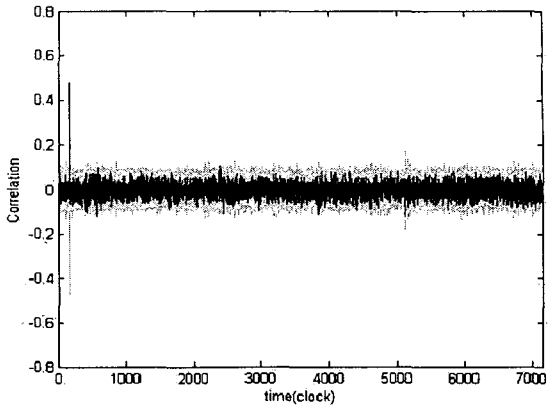

그림 7. 마스킹된 AddRoundKey에 대한 1차 DPA 결과

(그림 7)에서 올바른 키를 추측했을 때의 검정색 파형과 틀린 키를 추측했을 때의 회색 파형이 구별됨을 알 수 있다.

2. 마스킹된 SubBytes에 대한 1차 DPA 취약성

먼저 구현의 효율성 및 구현환경의 제약 등을 고려하여 수식 (3)에서 Sbox의 입력 마스크 m과 출력 마스크 m'을 동일한 값으로 사용했다고 가정하자. 이 경우 공격자는 수식 (2)와 같이 Sbox 입력과 출력사이의 Hamming distance에 의한 분류함수를 이용하여 다음과 수식과 같이 마스크 값이 상쇄되는 효과를 얻을 수 있다.

$$S'_{output}\oplus S'_{input} = (S(x\oplus m)\oplus m)\oplus(x\oplus m) = S(x\oplus m)\oplus x = S(x)\oplus x$$

즉, 동일한 마스크 값에 의해 마스킹된 Sbox 입·출력 값이 버스라인에 순차적으로 전송이 될 경우 AddRoundKey와 마찬가지로 마스크 값이 벗겨지는 효과가 일어난다. (그림 8)은 마스킹된 Sbox에 대한 1차 DPA 공격 결과파형이다. (그림 8)에서 검은색으로 표시된 파형은 올바른 키 추측 시 발생하는 상관계수이며 반면 회색으로 표시된 파형은 올바른 키를 제외한 틀린 키 추측 시 발생하는 상관계수를 나타낸 것이다. 따라서 부주의하게 구현된 마스킹 기반의

대응기법이 1차 DPA 공격에 의해 공격됨을 확인할 수 있었다.

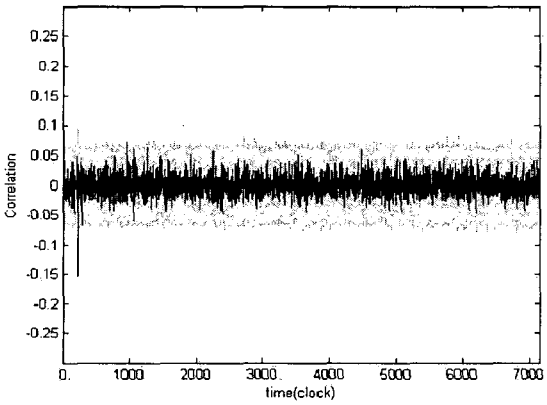

그림 8 마스킹된 Sbox에 대한 1차 DPA 결과

서로 다른 여러 개의 Sbox를 동일한 출력 마스크로 마스킹할 경우에도 마스크가 벗겨지는 현상이 발생한다. 만약 서로 이웃하는 Sbox 1과 Sbox 2가 동일한 마스크 값을 사용했다고 가정하자. 공격자는 두 개의 Sbox 출력사이의 Hamming distance에 의한 분류함수를 이용할 경우 아래 수식과 같이 마스크 값이 상쇄되는 효과를 얻을 수 있다.

$$S1'_{output} \oplus S2'_{output} = (S1(x_1) \oplus m) \oplus (S2(x_2) \oplus m) = S1(x_1) \oplus S2(x_2)$$

단 입력 x_1, x_2에 해당되는 라운드 키 16비트를 동시에 추측해야하는 어려움이 있지만 이 역시도 1차 DPA 공격에 취약함을 알 수 있다.

3. 대응방안

1차 DPA 공격에 안전한 마스킹 기반 대응기법을 구현하기 위해서 다음과 같이 몇 가지 방법을 고려해 볼 수 있다.

- 서로 다른 마스크 값 사용: 마스킹된 값을 처리하는 어셈블리 명령어가 순차적으로 발생하더라도 서로 다른 마스크 값을 사용하면 Hamming distance 기반 분류함수에 따른 마스크 값 상쇄효과를 방지할 수 있다. 하지만 비록 서로 다른 마스크 값을 사용할지라도 그 값들 사이에 어떠한 관계가 있을 경우 취약성이 발생할 수 있으므로 주의하여 구현해야 한다.
- LD 명령어 삽입: 버스라인에 순차적으로 전송되어 마스크 값이 제거되는 현상을 방지하기 위해여 임의의 값을 가지는 LD 명령어를 삽입함으로써 버스라인을 랜덤화하거나 초기화 하는 방법이다.
- 랜덤 순서화 및 가상 연산: 연산 순서의 랜덤화하는 랜덤 순서화 기법과 임의의 값으로 동일한 연산을 하는 가상 연산 기법을 함께 사용함으로써 안전한 대응기법을 구현할 수 있다. 이들 방법은 1차 DPA 뿐만 아니라 고차 DPA 공격을 방어하는데 있어서도 매우 효과적인 방법이다.

IV. 결론

통상 마스킹 기반 대응기법을 구현할 경우, 구현환경의 제약성과 구현상의 용의성 때문에 동일한 마스크 값을 사용한다. 본 논문에서는 1차 DPA 공격에 안전하고 효율적이라고 알려진 마스킹 기반 대응기법을 구현할 경우, 동일한 마스크 값을 중복하여 사용할 경우 1차 DPA 공격에도 취약할 수 있음을 이론적 근거와 실험을 통하여 증명하였다. 이를 방어하기 위해서는 서로 다른 마스크 값을 사용하는 방법, LD 명령어를 삽입하는 방법, 랜덤 순서화 및 가상 연산 등의 방법을 제시하였다.

[참고문헌]

[1] Paul Kocher, Joshua Jaffe, and Benjamin Jun, "Differential Power Analysis," in proceedings of Advances in Cryptology -CRYPTO '99, Springer-Verlag, 1999, pp.388-397

[2] K.Tiri, M.Akmal, and I.Verbauwhede, "A Dynamic and Differential CMOS Logic with Signal Independent Power Consumption to Withsatand Differential Power Analysis on Smart Cards," in proceedings of ESSCIRC2002, 2002

[3] K.Tiri and I.Verbauwhede, "Securing Encryption Algorithms against DPA at the Logic Level:Next Generation Smart Card Technology," in proceedings of CHES2003, LNCS 2779, pp.125~136, Springer, 2003

[4] Thomas S. Messerges, "Power Analysis Attacks and Countermeasures for Cryptographic Algorithms," Ph.D Thesis 2000.

[5] 유형소, 하재철, 김창균, 박일환, 문상재 "저메모리 환경에 적합한 마스킹 기반의 ARIA 구현" 정보보호학회 논문지, 2006.

[6] 유형소, 하재철, 김창균, 박일환, 문상재 "랜덤 마스킹 기법을 이용한 DPA 공격에 안전한 ARIA 구현" 정보보호학회 논문지, 2006.

[7] Daesung Kwon et al., "New Block Cipher ARIA," in proceedings of ICISC 2002, LNCS 2971 ,Springer-Verlag, pp.541~548, 2002.

[8] 서정갑, 김창균, 하재철, 문상재, 박일환 "블럭 암호 ARIA에 대한 차분전력분석공격" 정보보호학회 논문지, 제15권 제1호, pp. 99-107, 2005.

[9] JaeCheol Ha, ChangKyun Kim, SangJae Moon, IlHwan Park, and HyungSo Yoo, "Differential Power Analysis on Block Cipher ARIA," in proceedings of HPCC 2005, LNCS 3726, pp.541~548, Springer-Verlag, 2005.