情報保護學會論文誌
第18卷 第3號, 2008. 6

# 부채널 분석에 안전한 고속 ARIA 마스킹 기법*

김 희 석[1*†], 김 태 현[1*], 류 정 춘[1], 한 동 국[2**], 홍 석 희[1*‡]
[1]고려대학교 정보경영공학전문대학원, [2]한국전자통신연구원

# A High-speed Masking Method to protect ARIA against Side Channel Analysis*

HeeSeok Kim[1*†], Tae Hyun Kim[1*], Jeong-Choon Ryoo[1], Dong-Guk Han[2**], SeokHie Hong[1*‡]
[1]Graduate School of Information Management and Security, Korea University,
[2]Electronics and Telecommunications Research Institute

## 요 약

전력분석 공격이 소개되면서 다양한 대응법들이 제안되었고 그러한 대응법들 중 블록 암호의 경우, 암/복호화, 키 스케쥴링의 연산 도중 중간 값이 전력 측정에 의해 드러나지 않도록 하는 마스킹 기법이 잘 알려져 있다. 마스킹 기법은 블록 암호의 구성에 따라 적용 방법이 달라질 수 있으며, 각각의 블록암호에 적합한 마스킹 기법에 대한 연구가 진행되고 있다. ARIA의 경우, 기존 마스킹 방법들은 마스킹 보정작업으로 인해 암호 연산시간이 상당히 길며 키스케쥴링 공격이 다른 블록 암호들보다 ARIA에 더 위협적임에도 불구하고 키스케쥴링 과정에 마스킹 방법을 고려하지 않는다. 본 논문에서는 키 스케쥴링 과정을 포함한 ARIA에 적합한 효율적인 마스킹 기법을 제안한다. 제안하는 방법은 기존 방법들보다 암호 연산 시간을 단축시키고 일반적인 마스킹 기법의 (256*8 byte)에 대한 테이블 크기 문제도 (256*6 byte)로 단축시킨다..

## ABSTRACT

In the recent years, power attacks were widely investigated, and so various countermeasures have been proposed. In the case of block ciphers, masking methods that blind the intermediate results in the algorithm computations(encryption, decryption, and key-schedule) are well-known. Applications of masking methods are able to vary in different block ciphers, therefore suitable masking methods about each ciphers have been researched. Existed methods of ARIA have many revisions of mask value. And because existed masking methods pay no regard for key schedule, secret information can be exposed. In the case of ARIA, this problem is more serious than different block ciphers. Therefore we proposes an efficient masking scheme of ARIA including the key-schedule. Our method reduces time-complexity of ARIA encryption, and solve table-size problem of the general ARIA masking scheme from 256*8 byte to 256*6 byte.

**Keywords** : *Side Channel Attacks, Power Analysis, The Masking Method, ARIA*

## Ⅰ. 서 론

수학적으로 안전한 것으로 알려진 알고리즘조차도 구현 단계에서 고려되지 못한 부가적인 정보의 누출이 있다는 것이 알려졌고, 이로부터 비밀 키의 값을 알아낼 수 있는 부채널 공격(Side Channel Attack)이 소개되었다[1]. 이러한 부채널 공격이 소개되면서 많은 암호시스템 설계자들은 효율적인 대응법들을 연구하기 시작했고, 부채널 공격 중 하나인 차분 전력 분석(Differential Power Analysis, DPA)[2,3,4]에 대한 대응법으로는 마스킹 대응법(masking method)이 활발히 연구되어지고 있다[3,6,7,8].

KS 인증을 획득한 ARIA의 기존 마스킹 방법들에는 네 개의 S-box에서 4개의 마스킹 테이블을 생성하는 방법과 $INV$에 대한 테이블만을 저장하여 이에 대한 마스킹 테이블을 생성한 후, S-box 연산을 수행하는 방법이 있다. 이 방법들은 연산시간 측면과 공간적인 측면에서 각각 장점이 있다. 하지만 이러한 방법들은 ARIA에 사용되어지는 S-box의 특징과 키스케쥴링과 암복호화 연산이 병렬화되지 않는 ARIA 알고리즘 상의 특징을 이용하거나 고려하지 않았다.

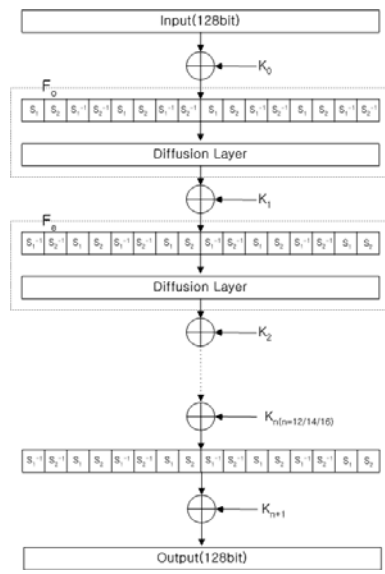
본 논문은 이러한 알고리즘의 특징을 이용하여 ARIA에 적합한 효율적이고 안전한 마스킹 기법을 제안한다. 제안하는 마스킹 기법은 ARIA가 네 개의 S-box를 가지고 있지만, 두 개의 S-box만을 저장한 후 새로운 마스킹 S-box 네 개를 암호연산시마다 생성해 사용하는 방법으로 본래의 ARIA 알고리즘보다 두 개의 S-box만을 더 필요로 하는 방법이다. 이 방법은 일반적인 마스킹 기법을 사용했을 경우 총 8개의 S-box를 저장해야 하는 단점을 극복한 것이며, ARIA에 적합한 마스킹 기법을 사용하여 기존 마스킹 기법보다 연산 시간도 단축시키는 방법이다. 또한 ARIA의 경우 S. Mangard가 제안한 키스케쥴링 공격[9]에 심각한 위험성이 있음을 논하고, 이에 대한 대응법도 고려하여 기존의 방법들보다 더 안전하게 구성하였다.

본 논문의 구성은 다음과 같다. 2절은 ARIA 알고리즘에 대해 설명하고, 3절은 기존의 마스킹 기법과 기존 방법의 문제점을 언급한다. 본 논문에서 제안하는 마스킹 방법은 4절에서 소개하며, 마지막으로 5절에서는 기존 방법과 제안하는 방법의 효율성과 안전성을 비교, 분석한다.

## Ⅱ. ARIA

ARIA는 2004년에 산업자원부의 KS인증(KS 규격번호 : KS X 1213)을 획득한 블록알고리즘으로 Academy(학계), Research Institute(연구소), Agency(정부기관)로 구성된 개발팀에 의해 개발되었기 때문에 그 앞 글자를 이용하여 명명되었다. ARIA는 [그림 1]과 같이 키 에디션(Key addition)과 치환계층(Substitution Layer), 확산계층(Diffusion Layer)으로 라운드 함수를 구성하며 세부 사양은 다음과 같다.

- 기본구조 : ISPN구조(Involutional SPN구조)
- 입, 출력크기 : 128 비트
- 키 크기 : 128 / 192 / 256 비트
- 라운드 키 크기 : 128 비트
- 라운드 수 : 12 / 14 / 16 라운드




[그림 1] ARIA 전체구조

접수일 : 2007년 10월 25일; 수정일 : 2008년 1월 21일
채택일 : 2008년 3월 7일
* “본 연구는 지식경제부 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업의 연구결과로 수행되었음.” (IITA-2008-(C1090-0801-0025))
** “본 연구의 일부는 지식경제부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음 (IITA-2008-(C1090-0801-0025)), 그리고 정보통신부 및 정보통신연구진흥원의 IT신성장동력핵심기술개발사업[2005-S-088-04,안전한 RFID/USN 을 위한 정보보호 기술] 사업의 일환으로 수행하였음.”
† 주저자, heeseokkim@cist.korea.ac.kr
‡ 교신저자, hsh@cist.korea.ac.kr

### 2.1 치환 계층

각 라운드 함수의 치환계층은 8 비트 입출력을 가지는 두 개의 S-Box와 그에 대한 두 개의 역 치환 S-Box를 포함, 총 네 개의 S-box를 이용한다. S-box($S_1$, $S_2$)는 두 $x^{-1}$, $x^{247}(=x^{-8}=Dx^{-1}, D: x^8$을 수행하는 행렬)의 아핀 변환(affine transform) 형태로 다음의 연산을 수행한다.

$$S_1,\ S_2:\ GF(2^8) \rightarrow GF(2^8)$$
$$S_1(x) = Bx^{-1} \oplus b$$
$$S_2(x) = CDx^{-1} \oplus c$$

위의 연산을 위한 행렬과 벡터의 값은 다음과 같다.

$$B=\begin{pmatrix}1&0&0&0&1&1&1&1\\1&1&0&0&0&1&1&1\\1&1&1&0&0&0&1&1\\1&1&1&1&0&0&0&1\\1&1&1&1&1&0&0&0\\0&1&1&1&1&1&0&0\\0&0&1&1&1&1&1&0\\0&0&0&1&1&1&1&1\end{pmatrix} \quad b=\begin{pmatrix}1\\1\\0\\0\\0\\1\\1\\0\end{pmatrix}$$

$$CD=\begin{pmatrix}0&1&0&1&0&1&1&1\\0&0&1&1&1&1&1&1\\1&1&1&0&1&1&0&1\\1&1&0&0&0&0&1&1\\0&1&0&0&0&0&1&1\\1&1&0&0&1&1&1&0\\0&1&1&0&0&0&1&1\\1&1&1&1&0&1&1&0\end{pmatrix} \quad c=\begin{pmatrix}0\\1\\0\\0\\0\\1\\1\\1\end{pmatrix}$$

이 두 개의 S-Box와 그 역치환 $S_1^{-1}$, $S_2^{-1}$를 이용하여 ARIA의 라운드 함수 중 치환계층을 구성한다. 이 때, 홀수 라운드 함수($F_o$)와 짝수 라운드 함수($F_e$)의 치환계층은 [그림 1]처럼 S-Box의 순서를 반대로 구성한다.

### 2.2 확산 계층

ARIA 라운드 함수의 확산계층은 입력 16 바이트 $(x_0, x_1, \ldots, x_{15})$에 대한 출력 16바이트$(y_0, y_1, \ldots, y_{15})$의 값을 다음의 행렬 곱셈을 통하여 연산한다.

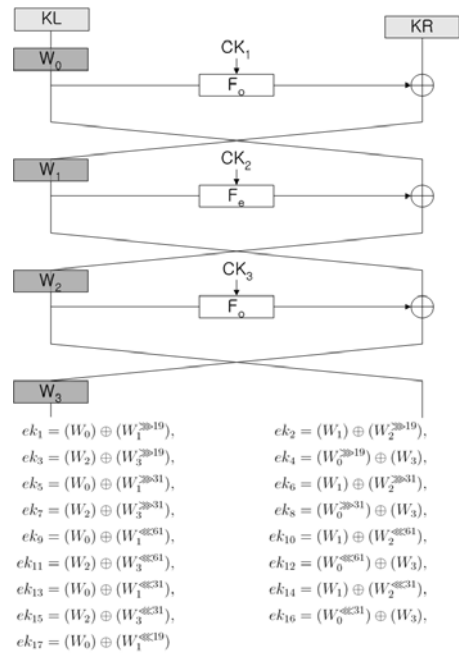
$$\begin{pmatrix}y_0\\y_1\\y_2\\y_3\\y_4\\y_5\\y_6\\y_7\\y_8\\y_9\\y_{10}\\y_{11}\\y_{12}\\y_{13}\\y_{14}\\y_{15}\end{pmatrix}=\begin{pmatrix}0&0&0&1&1&0&1&0&1&1&0&0&0&1&1&0\\0&0&1&0&0&1&0&1&1&1&0&0&1&0&0&1\\0&1&0&0&1&0&1&0&0&0&1&1&1&0&0&1\\1&0&0&0&0&1&0&1&0&0&1&1&0&1&1&0\\1&0&1&0&0&1&0&0&1&0&0&1&0&0&1&1\\0&1&0&1&1&0&0&0&0&1&1&0&0&0&1&1\\1&0&1&0&0&0&0&1&0&1&1&0&1&1&0&0\\0&1&0&1&0&0&1&0&1&0&0&1&1&1&0&0\\1&1&0&0&1&0&0&1&0&0&1&0&0&1&0&1\\1&1&0&0&0&1&1&0&0&0&0&1&1&0&1&0\\0&0&1&1&0&1&1&0&1&0&0&0&0&1&0&1\\0&0&1&1&1&0&0&1&0&1&0&0&1&0&1&0\\0&1&1&0&0&0&1&1&0&1&0&1&1&0&0&0\\1&0&0&1&0&0&1&1&1&0&1&0&0&1&0&0\\1&0&0&1&1&1&0&0&0&1&0&1&0&0&1&0\\0&1&1&0&1&1&0&0&1&0&1&0&0&0&0&1\end{pmatrix}\cdot\begin{pmatrix}x_0\\x_1\\x_2\\x_3\\x_4\\x_5\\x_6\\x_7\\x_8\\x_9\\x_{10}\\x_{11}\\x_{12}\\x_{13}\\x_{14}\\x_{15}\end{pmatrix}$$

즉, 각 행마다 치환계층을 거친 16 바이트 중 7 바이트의 xor형태로 연산되어진다.

### 2.3 키 스케줄링

ARIA의 키 스케줄링은 [그림 2]와 같다. [그림 2]에서 KL은 키의 상위 128비트 값이며, KR은 키의 나머지 부분에 0으로 padding한 128비트 값이다. $CK_1$, $CK_2$, $CK_3$의 세 개의 고정된 상수 값은 각 라운드함수($F_o$, $F_e$, $F_o$)의 키 역할을 수행하고 이로 인해 얻은 $W_0$, $W_1$, $W_2$, $W_3$을 통해 [그림 2]의 수식처럼 라운드 키를 얻는다.

ARIA의 키 스케줄링은 알고리즘의 특성상 암복호화 연산과 병렬화되어 수행되어질 수 없다는 특징이 있다.




[그림 2] ARIA 키 스케줄

## Ⅲ. 관련 연구

전력 분석 공격이 소개되면서 여러 가지 대응법들이 소개되었다. 그 중 암호 알고리즘의 연산이 수행되는 도중 중간 값의 정보를 숨기는 마스킹 기법이 일반적으로

잘 알려져 있다. 마스킹 기법은 평문 $x$에 대하여 암호문 $y$를 얻기 위해 마스킹 난수 $m$을 이용 $x \oplus m$($\oplus$ : xor)의 암호문 $y'(=y \oplus m')$을 구한 후, 최종적으로 $y$를 얻기 위해 $y' \oplus m'$의 연산을 수행한다.(경우에 따라 마스킹 기법은 다르게 구성한다.) 따라서 암호화 중 중간 값을 알 수 없기 때문에 일반적인 전력 분석 공격은 성공할 수 없다.

이러한 마스킹 기법을 사용한 경우, $x \oplus m$의 암호문 $y'(=y \oplus m')$에서 $m'$을 알아야 실제 원하는 암호문 $y$를 얻을 수 있다. 하지만 블록암호 알고리즘은 비선형 연산을 수행하므로 수정되지 않은 블록 암호 시스템에서 $m'$ 값은 $x$에 따라 다르며 이 값을 중간 값의 누출 없이 아는 것도 상당한 연산을 필요로 한다. 따라서 마스킹 대응법은 이 비선형 연산에 대한 고려가 불가피하다. 블록암호의 대표적인 비선형 연산은 S-box 연산으로 그 값이 $S(x \oplus m) = S(x) \oplus m_x$의 형태이며 $m_x$의 값이 $x$의 값마다 다르다. 따라서 이 $m_x$의 값이 모든 $x$에 대해 같은 값이 되게끔 암호 알고리즘의 최초 수행시마다 새로운 마스킹 S-box($MS$)를 만든다. 즉, 모든 $x$에 대해 $MS(x \oplus m) = S(x) \oplus m'$를 만족하는 $MS$를 생성한다. 이 때, $m$, $m'$의 두 개의 다른 마스킹 값을 사용하는 이유는 마스킹 값이 같을 경우, 구현된 장비에 따라 취약점이 발생할 수 있기 때문이다. 본 절에서는 ARIA에 대한 기존 마스킹 기법들과 이 방법들의 안전성에 대해 논한다.

### 3.1 일반적인 마스킹 기법

유형소 등에 의해 제안된 마스킹 기법은 일반적인 마스킹 기법을 ARIA에 적용시킨 것이다[10]. 즉, ARIA의 네 개의 S-box에 대하여 네 개의 새로운 마스킹 S-box를 만든 후, 이 네 개의 새로운 마스킹 S-box를 이용하여 암호 연산을 수행하는 것이다. 물론 각 라운드의 확산 계층에 대한 연산을 마칠 때마다 원하는 마스킹 값(마스킹 S-box 생성 시 이용한 마스킹 값)을 얻기 위해 반드시 마스킹 보정 작업을 수행해야만 한다.

이 방법은 일반적인 마스킹 방법으로 마스킹 보정을 위한 연산이 비교적 많은 편이다. 뿐만 아니라 총 4개의 S-box를 저장해야하며 새로운 4개의 S-box를 암호연산 시마다 생성해야 하므로 총 8개의 S-box(2048 바이트)에 대한 저장 공간이 필요하다.

### 3.2 저 메모리 환경의 마스킹 기법

필요한 저장 공간이 많은 위의 방법을 보완하기 위해 저 메모리 환경에서의 마스킹 기법이 제안되었다[11]. 이 기법은 $x^{-1}(x \in GF(2^8))$에 대한 256 바이트의 테이블($INV$)만을 저장한다. 그리고 암호연산마다 $INV$에 대한 마스킹 테이블($MINV$)을 구성한다. 즉, S-box의 연산 $S_1(x) = Bx^{-1} \oplus b$, $S_2(x) = CDx^{-1} \oplus c$에서 $x^{-1}$의 연산은 테이블을 이용하며, 아핀 변환은 직접 계산하는 방법이다. 이러한 계산을 이용한 두 S-box에 대한 출력 값의 마스킹 값은 어렵지 않게 알 수 있지만, 마스킹 값이 일정하지 않기 때문에 확산z계층 연산을 수행하기 전 마스킹 보정 작업을 수행하도록 이 방법은 구성되었으며, $S_1^{-1}$, $S_2^{-1}$에 대한 계산에서도 $MINV$의 테이블 참조 전 불가피하게 마스킹 보정작업을 수행하도록 되어 있다. 즉, 총 필요한 테이블의 개수는 두 개인 반면 치환 계층의 연산과 마스킹 보정 작업으로 인한 연산 시간이 비교적 큰 편이다.

### 3.3 키 스케줄링을 고려하지 않은 기존 ARIA 마스킹 방법의 문제점

Mangard가 제안한 AES의 키 스케줄링 공격방법은 키 스케줄링에 대한 마스킹 방법을 고려하지 않았을 시 키가 노출될 수 있는 위험성에 대해 언급하고 있다[9]. 즉, 대응법을 고려하지 않은 경우 키 스케줄링 과정 중 키에 대한 헤밍웨이트가 노출되어 후보 키의 범위를 줄일 수 있으며, 전수조사를 통해 AES의 전체 키를 찾을 수 있다는 것이다. ARIA의 경우 알고리즘의 특성상 AES와는 달리 키스케줄링과 암호연산이 병렬화되어 수행되어질 수 없다. 즉, 키 스케줄링 후에 암호 연산을 수행하여야만 한다. 이러한 경우 키 스케줄링에 대한 부분의 전력이 암호 연산과 상관없이 그대로 노출되어 이러한 공격방법은 AES의 경우보다 훨씬 위협적이다.

이러한 일차적인 문제외에도 키가 마스킹되어 있지 않으면, 헤밍 디스턴스 모델(hamming distance model)의 경우 키 에디션 부분의 연산에서 키의 헤밍웨이트를 알아내는 것이 가능하다. 그 이유는 키 에디션 전후 헤밍 디스턴스가 다음 식을 만족하기 때문이다.

$$(x \oplus m) \oplus (x \oplus k \oplus m) = x \oplus (x \oplus k)$$

즉, 헤밍 디스턴스 모델에서 키 에디션 부분의 전력이 마스킹 값과 상관없이 키에만 의존하기 때문에 키의 헤밍 웨이트가 누출될 수 있다. 그러므로 본 논문에서는 키 스케쥴링에서의 효율적인 마스킹 방법도 고려하도록 한다.

### 3.4 Template attack의 가정 및 Template attack에 대한 마스킹 기법의 취약성

Template attack[12,13]은 다음의 가정이 성립할 때 적용할 수 있는 공격 방법이다.

***Remark(Template attack의 가정)***

- **a.** 공격자는 실제 공격 대상 장비와 같은 모델의 장비를 소유하고 있다.
- **b.** 공격자는 공격 대상 장비의 상세한 구현 내용을 알고 있어, 암호 연산에 해당하는 전력 파형에서 공격자가 원하는 시점을 정확히 구분해낼 수 있어야 한다.

마스킹 방법에 대한 Template attack은 다양하게 적용될 수 있으며 하나의 예로 다음과 같이 적용 가능하다.

- **1)** 256개의 마스킹 값을 로딩시키는 전력에 대한 template(reference power trace)을 생성한다.
- **2)** 실제 공격 대상 장비에서 얻은 각 전력 파형마다 마스킹이 로딩되는 시점의 마스킹 값을 **1)**의 과정에서 생성한 256개의 template을 이용하여 찾아낸다.
- **3)** 찾아낸 마스킹 값을 이용하여 차분 전력 분석을 시도한다.

Template attack은 모든 마스킹 기법에 적용할 수 있는 방법이며 이는 ARIA에 대한 기존의 마스킹 방법에도 적용이 가능하다. 하지만 공격 대상 장비와 같은 실험 장비가 공격자에게 만약 주어지지 않는다면, 혹은 마스킹 값이 로딩되는 시점을 공격자가 정확히 알 수 없다면 Template attack은 적용이 불가능하며 마스킹 기법은 여전히 안전하다.

## Ⅳ. 제안하는 마스킹 기법

본 절에서는 Template attack의 가정이 성립하지 않는 환경에서 안전한 마스킹 기법을 제안한다. 우선 제안하는 마스킹 기법은 ARIA의 S-box들의 특징을 고려하여 두 개의 S-box($S_1$, $S_2$)만을 저장하고, 이 두 개의 S-box를 이용하여 네 개의 마스킹 S-box를 구성한다. 그리고 키 스케쥴링과 암호 연산 시 이 네 개의 마스킹 S-box를 이용한다.

### 4.1 마스킹 S-box의 구성

우선, 두 개의 마스킹 값 $m, m'$에 대하여 다음 네 개의 마스킹 S-box를 만든다.

$$\begin{aligned} &MS_1(x\oplus m) = S_1(x)\oplus m' \\ &MS_2(x\oplus m) = S_2(x)\oplus m' \\ &MS_3(x\oplus m') = S_1^{-1}(x)\oplus m \\ &MS_4(x\oplus m') = S_2^{-1}(x)\oplus m \end{aligned} \quad (1)$$

네 개의 마스킹 S-box를 만들 때는 $S_1$, $S_2$만을 이용하며, 네 개의 마스킹 S-box를 구성하는 방법은 알고리즘 1과 같다.

**알고리즘 1. 마스킹 S-box 구성 알고리즘**
**입력** $S_1$, $S_2$, $m$, $m'$
**출력** $MS_1$, $MS_2$, $MS_3$, $MS_4$
1. For $x$ from 0x00 to 0xff
   a. $u = x\oplus m$, $v = S_1(x)\oplus m'$, $w = S_2(x)\oplus m'$
   b. $MS_1(u) = v$, $MS_2(u) = w$,
      $MS_3(v) = u$, $MS_4(w) = u$.
2. Return $MS_1$, $MS_2$, $MS_3$, $MS_4$

알고리즘 1에 따라 마스킹 S-box를 구성하면 식 (1)의 앞의 두 식 $MS_1(x\oplus m) = S_1(x)\oplus m'$, $MS_2(x\oplus m) = S_2(x)\oplus m'$은 성립하지만, 뒤의 두 식이 성립하는지는 다음과 같이 확인해 보아야 한다.

***Proposition 1)*** $MS_3(S_1(x)\oplus m') = x\oplus m$이면, $MS_3(x\oplus m') = S_1^{-1}(x)\oplus m$를 만족한다.

**Proof.** $S_1$은 치환이므로 모든 $x$에 대해 $x = S_1^{-1}(t)$를 만족하는 $t$가 존재한다. 따라서 $x$에 $x = S_1^{-1}(t)$를 대입하면 $MS_3(t\oplus m') = S_1^{-1}(t)\oplus m$을 만족한다.

따라서 위의 Proposition 1에 따라 (1)의 식을 만족하는 네 개의 마스킹 S-box를 알고리즘 1에 의해 구성할 수 있다.

### 4.2 암/복호화 과정에서의 대응법

본 소절에서는 앞에서 구성한 마스킹 S-box를 이용하여 암/복호화 과정에서의 대응법을 제안한다.

우선 제안하는 암/복호화 과정에서의 마스킹 기법에 대한 전체 윤곽은 다음과 같다.

**1)** 두 개의 1 byte 마스킹 난수 $m, m'$을 생성 후, $m'' = m \oplus m'$을 계산한다.

**2)** $S_1$, $S_2$에 대하여 식 (1)을 만족하는 네 개의 마스킹 S-box를 알고리즘 1에 따라 생성한다.

**3)** 키 스케쥴링 과정에서 16 바이트의 모든 라운드 키가 $(m'')^{16}$(16 바이트)로 마스킹되도록 조정한다. (다음 소절에서 상세히 설명한다.)

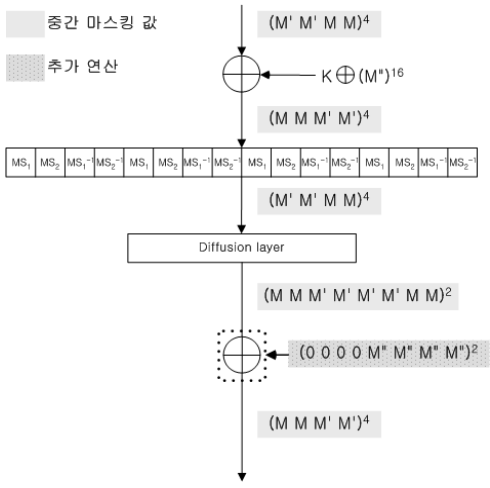
**4)** 입력 평문 16 바이트를 $m'm'mm/m'm'mm/m'm'mm/m'm'mm$으로 마스킹한다.

**5)** 각 라운드에서 확산 계층 연산 후 16 바이트의 중간 값에 $0000/m''m''m''m''/0000/m''m''m''m''$값을 xor한다. (8 바이트 xor 연산)

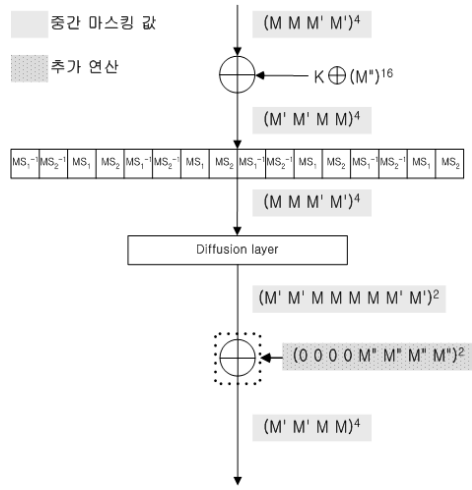
**6)** 출력 평문의 16 바이트 마스킹 값 $m'm'mm/m'm'mm/m'm'mm/m'm'mm$을 벗겨낸다.

위의 순서에 따른 전체 구조에 대한 대응법은 [그림 3, 4]를 참고한다.

홀수라운드 함수에서는 키 에디션 후 치환 계층에 알맞은 마스킹 값으로 입력되기 위해 마스킹 값이 $(mmm'm')^4$이 되도록 조정되어야 한다. 이 조정된 값에 대하여 치환 계층을 거친 후의 중간 값은 $(m'm'mm)^4$으로 마스킹 된다. 확산 계층에서 7개의 값이 xor되면 $(mmm'm'/m'm'mm)^2$으로 중간 값은 마스킹 된다. 이 때, xor되는 7개 중 네 개는 $m'(\text{or } m)$, 나머지 세 개는 $m(\text{or } m')$으로 마스킹된 값이다. 이 7 바이트의 값을 xor할 때는 중간 값이 드러나지 않게끔 마스킹된 값이 번갈아가면서 xor되도록 7 바이트의 xor 순서를 조정하거나 또는 7 바이트의 xor 연산 도중 중간 값이 데이터 버스를 지나지 않도록 구현하여야 한다.




[그림 3] 홀수 라운드 마스킹 기법




[그림 4] 짝수 라운드 마스킹 기법

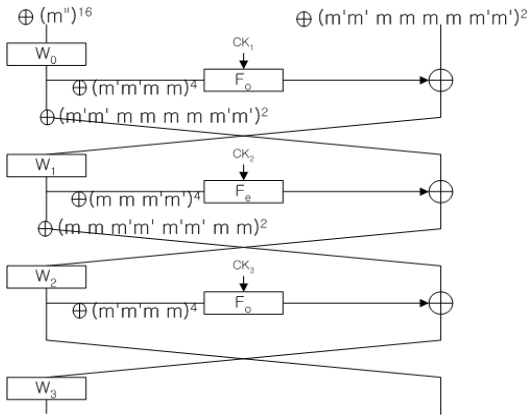
$(mmm'm'/m'm'mm)^2$으로 마스킹 된 값을 다음의 $(m'')^{16}$으로 마스킹된 키와 xor 했을 때, 짝수라운드 함수의 치환 계층에 입력 가능한 마스킹 값으로 바꾸기 위해 확산 계층을 거친 후에 $(0\,0\,0\,0/m''m''m''m'')^2$을 xor하는 추가적인 연산이 필요하다. 하지만 이 연산은 일반 ARIA 알고리즘에 비해 필요한 유일한 추가적인 연산으로 연산량이 상당히 작다.

짝수 라운드 함수에 대한 마스킹 기법도 유사한 방법으로 구성된다.

### 4.3 키 스케쥴링에 대한 대응법

키 스케쥴링에 대한 공격은 DPA를 고려하지 않는다. 그 이유는 고정된 키에 대해 항상 같은 연산을 수행하기 때문에 랜덤한 평문을 다수 입력해 파형을 얻는 DPA방법은 키 스케쥴링 과정에서 의미가 없기 때문이다. 따라서 키 스케쥴링에 대한 대응법은 키의 헤밍웨이트를 숨기는 마스킹 기법으로 충분한 대응법이 된다.

키 스케쥴링에 대한 마스킹 기법은 [그림 5]를 참고한다.




[그림 5] 키 스케쥴링 대응법

[그림 5]에 표시된 xor 연산은 128 비트의 $W_0$, $W_1$, $W_2$, $W_3$을 모두 $(m'')^{16}$으로 마스킹 된 값으로 얻기 위해 추가적으로 필요한 연산이다.

이 네 개의 값을 이용해 [그림 2]에서의 첫 번째 라운드 키는 다음 식처럼 얻어낼 수 있다.

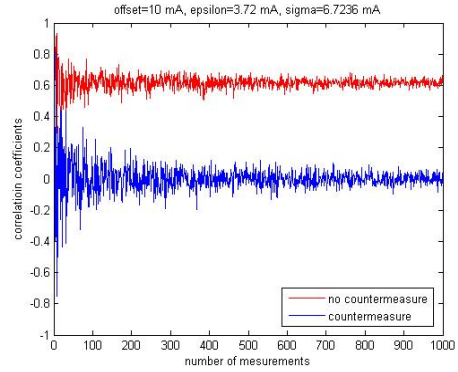
$$ek_1 = W_0 \oplus (m'' \gg 3)^{16} \oplus (W_1 \gg 19)$$

위의 식에서 $m'' \gg 19 = m'' \gg 3$이기 때문에 첫 번째 라운드 키는 $(m'')^{16}$으로 마스킹 된 값으로 얻어낼 수 있다. 다른 모든 라운드 키도 유사한 방법에 의해 $(m'')^{16}$으로 마스킹 된 값으로 얻어낼 수 있다.

## Ⅴ. 비교 및 결론

본 논문에서는 ARIA의 S-box가 $S_1$, $S_2$와 그들의 역치환으로 구성되었다는 점에 착안하여 새로운 마스킹 테이블 생성법을 소개하고, 이를 이용하여 공간적인 측면과 연산시간 측면에서 효율적인 마스킹 방법을 제안한다. 또한 키스케쥴링과 암복호화 연산이 병렬화되지 않는 ARIA의 특징을 고려하여 키스케쥴링에 대한 전력 노출을 막을 수 있는 키 스케쥴링의 마스킹 방법도 제안하였다. 제안하는 마스킹 기법의 안전성을 증명하기 위해 Messarges가 제안한 전력 소비 모델($C(t) = offset + \epsilon Hw(data) + N$ ($offset, \epsilon$ : 상수, $N$ : 노이즈, $Hw(a)$ : $a$의 헤밍웨이트)[14]에 따라 일차 차분 전력 분석에 대한 결과를 조사 했을 때, 그 결과는 [그림 6]과 같았다. 그림 상에서 sigma의 값은 노이즈의 표준 편차에 해당하는 값이다.

[그림 6]에서 보는 것처럼 마스킹 방법이 적용되지




[그림 6] 옳은 키에 대한 상관계수

[표 1] ARIA의 500000번 수행 시 연산시간 비교

| | ARIA | 방법1[10] | 방법2[11] | 제안방법 |
|---|---|---|---|---|
| 테이블크기 | 1024byte | 2048byte | 512byte | 1536byte |
| 키스케쥴 | 3.484s | 고려X | 고려X | 4.047s |
| 마스킹테이블 | X | 2.121s | 0.637s | 2.127s |
| 암호연산 | 0.814s | 1.125s | 7.325s | 0.851s |
| 키스케쥴링 공격 | X | X | X | 안전 |
| Hw분석공격 | X | X | X | 안전 |

않은 ARIA는 상관계수가 0.6에 수렴하는 반면, 마스킹 방법이 적용된 ARIA는 상관계수가 0에 수렴함을 확인할 수 있었다. 즉, 옳은 키에 대해 예측한 데이터의 중간 값과 전력사이의 상관성이 존재하지 않으며, 이는 제안하는 마스킹 방법이 일차 차분 전력 분석에 안전함을 의미한다.

제안하는 마스킹 기법과 기존의 방법1[10], 방법2[11]를 500000번 수행 후 연산 시간을 비교했을 때, 연산시간의 차이는 [표 1]과 같았다. [표 1]에서 HW 분석 공격이란 3절에서 언급한 키가 마스킹 되지 않은 경우 키 에디션 연산 시 키에 대한 헤밍웨이트를 알아내는 공격을 뜻한다.

[표 1]에서 보여지는 것처럼 제안하는 마스킹 기법은 암호 연산 중 추가적인 시간이 거의 들지 않는다. 따라서 기존의 방법1에 비해 저장 공간 뿐 아니라 연산시간, 안전성 측면에서도 더 좋다. 또한 방법2는 저장 공간은 크지 않지만 치환 계층에 대한 연산과 마스킹 보정을 위한 추가적인 연산을 수행해야 하며, $S_1, S_2, S_1^{-1}, S_2^{-1}$가 존재하지 않기 때문에 키 스케쥴링의 과정에서 마스킹 방법을 고려 시 상당한 연산시간이 필요할 것으로

예상된다. 따라서 제안하는 마스킹 방법은 방법2에 비해 연산 시간을 상당히 단축시킬 수 있다.

일반적인 마스킹 기법은 고차 DPA에 취약한 것으로 알려져 있다. 하지만 마스킹 기법은 전력분석공격이 성공하기 위해 공격자가 취해야 할 전력 파형의 개수를 증가시키고, 추가적인 대응법들의 고려시 충분히 고차 DPA의 대응법이 될 수 있다. 고차 DPA에 대한 대응법으로 알려진 방법들에는 첫 번째 라운드와 마지막 라운드의 S-box 입출력 바이트마다 마스킹 값을 다르게 하는 방법과 S. Mangard에 의해 제안된 동일한 마스킹 값을 사용하면서 1라운드와 마지막 라운드의 키 에디션, 치환 계층, 확산 계층의 순서(각각의 16 byte 연산에 대한 순서)를 랜덤하게 바꾸는 방법이 있다[5]. 하지만 최근 들어 입출력바이트마다 다른 마스킹을 씌우는 방법을 3차 DPA로 분석하는 방법이 소개됨[15]에 따라 제안하는 방법에는 추가적으로 두 번째 방법을 적용하는 것이 고차 DPA에 보다 더 안전하게 구성할 수 있다.

## 참고문헌


[1] P. Kocher, J. Jaffe, and B. Jun, “Timing Attacks on Implementations of Diffie- Hellman, RSA, DSS, and Others Systems.” CRYPTO’96, LNCS 1109, pp.104-113, Springer- Verlag, 1996.

[2] P. Kocher, J. Jaffe, and B. Jun, “Differential power analysis,” CRYPTO’99, pp.388-397, Springer- Verlag, 1999.

[3] P. Kocher, J. Jaffe, and B. Jun, “Introduction to differential power analysis and related attacks,” http://www.cryptography.com/dpa/technical, 1998.

[4] T. S. Messerges, E. A. Dabbish, and R. H. Sloan, “Power analysis attacks on modular exponentiation in Smart cards,” Proc. of Workshop on Cryptographic Hardware and Embedded Systems, pp. 144-157, Springer- Verlag, 1999.

[5] C. Herbst, E. Oswald, S. Mangard, “An AES Smart Card Implementation Resistant to Power Analysis Attacks,” ACNS 2006, LNCS 3989, pp. 239-252, Springer, 2006.

[6] E. Oswald and K. Schramm. “An Efficient Masking Scheme for AES Software Implementations,” WISA 2005, LNCS 3786, pp. 292–305, Springer, 2006.

[7] E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen., “A Side-Channel Analysis Resistant Description of the AES S-box,” FSE 2005, LNCS 3557, pp. 3–423, Springer, 2005.

[8] J. Bl¨omer, J. Guajardo, and V. Krummel. “Provably Secure Masking of AES,” SAC 2004, LNCS 3357, pp. 69–83, Springer, 2005.

[9] S. Mangard, “A Simple Power-Analysis (SPA) Attack on Implementations of the AES Key Expansion”, ICISC 2002, LNCS 2587, pp. 343-358, Springer, 2003.

[10] 유형소, 하재철, 김창균, 박일환, 문상재, “랜덤 마스킹 기법을 이용한 DPA 공격에 안전한 ARIA 구현”, 한국정보보호학회논문지 16(2), April 2006

[11] 유형소, 하재철, 김창균, 박일환, 문상재, “저메모리 환경에 적합한 마스킹 기반의 ARIA 구현”, 한국정보보호학회논문지 16(3), June 2006

[12] C. Rechberger and E. Oswald. “Practical Template Attacks,” WISA 2004, LNCS 3325, pp. 440-456, Springer- Verlag, 2004.

[13] Dakshi Agrawal, Josyula R. Rao, Pankaj Rohatgi, and Kai Schramm, “Template as Master Keys,” CHES 2005, LNCS 3659, pp. 15-29, Springer, 2005.

[14] Messerges, T.S., “Using Second-Order Power Analysis to Attack DPA resistant Software,” CHES 2000, LNCS 1965, pp. 238-251, Springer- Verlag, 2000.

[15] J. S. Coron, E. Prouff, and M. Rivain, “Side Channel Cryptanalysis of a Higher Order Masking Scheme”, CHES 2007, LNCS 4727, pp. 28-44, Springer-Verlag, 2007.

## <著 者 紹 介>

김 희 석 (HeeSeok Kim) 학생회원
2006년 2월 : 연세대학교 수학과 졸업(학사)
2008년 2월 : 고려대학교 정보경영공학전문대학원 공학석사
2008년 3월~현재 : 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 부채널 공격, 공개키 암호시스템 안전성 분석 및 고속구현, 타원곡선

김 태 현 (Tae Hyun KIM) 학생회원
2002년 2월 : 서울 시립대학교 수학과 이학사
2004년 8월 : 고려대학교 정보보호 대학원 공학석사
2005년 2월~현재 : 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 부채널 공격, 공개키 암호 알고리즘, 암호칩 설계 기술

류 정 춘 (Jeong-Choon Ryoo) 학생회원
1988년 2월 : 경북대학교 전자공학과 졸업(학사)
1990년 2월 : 경북대학교 전자공학과 석사(공학석사)
1990년 1월~1995년 4월 : LG정보통신 연구원 근무
1996년 1월~1999년 11월 : 대우그룹 해외통신사업본부 근무
2005년 3월~현재 : 고려대학교 정보경영공학전문대학원 박사과정
<관심분야> 부채널 공격, 대칭키 암호의 분석 및 설계, 이동통신 암호 프로토콜

한 동 국 (Dong-Guk Han) 정회원
1999년 : 고려대학교 수학과 졸업(학사)
2002년 : 고려대학교 수학과 석사 (이학석사)
2005년 : 고려대학교 정보보호대학원 박사 (공학박사)
2004년 4월～2005년 4월 : 일본 Kyushu Univ., 방문연구원
2005년 4월～2006년 4월 : 일본 Future Univ.-Hakodate, Post.Doc.
2006년 6월～현재 : 한국전자통신연구원 정보보호연구단 선임연구원
<관심분야> 공개키 암호시스템 안전성 분석 및 고속 구현, 부채널 분석, RFID/USN 정보보호 기술

홍 석 희 (SeokHie Hong) 종신회원
1995년 : 고려대학교 수학과 학사
1997년 : 고려대학교 수학과 석사
2001년 : 고려대학교 수학과 박사
1999년 8월~2004년 2월 : (주)시큐리티 테크놀로지스 선임연구원
2003년 3월~2004년 2월 : 고려대학교 시간강사
2004년 4월~2005년 2월 : K.U. Leuven 박사후연구원
2005년 3월~현재 : 고려대학교 정보경영전문대학원 조교수
<관심분야> 대칭키 암호 알고리즘, 공개키 암호 알고리즘, 포렌식