情報保護學會論文誌
第19卷 第4號, 2009. 8

일차 차분 전력 분석에 안전한 저면적 AES S-Box 역원기 설계*

김 희 석,[1†] 한 동 국,[2‡] 김 태 현,[1] 홍 석 희[1]
[1]고려대학교 정보경영공학전문대학원, [2]국민대학교 수학과

DPA-Resistant Low-Area Design of AES S-Box Inversion*

Hee Seok Kim,[1†] Dong-Guk Han,[2‡] Tae Hyun Kim,[1] Seokhie Hong[1]
[1]Graduate School of Information Management and Security, Korea University,
[2]Department of Mathematics, Kookmin University

요 약

전력분석 공격이 소개되면서 다양한 대응법들이 제안되었고 그러한 대응법들 중 블록 암호의 경우, 암/복호화 연산, 키 스케줄 연산 도중 중간 값이 전력 측정에 의해 드러나지 않도록 하는 마스킹 기법이 잘 알려져 있다. 블록 암호의 마스킹 기법은 비선형 연산에 대한 비용이 가장 크며, 따라서 AES의 경우 가장 많은 비용이 드는 연산은 S-box의 역원 연산이다. 이로 인해 마스킹 역원 연산에 대한 비용을 단축시키기 위해 다양한 대응법들이 제안되었고, 그 중 Zakeri의 방법은 복합체 위에서 정규 기저를 사용한 가장 효율적인 방법으로 알려져 있다. 본 논문에서는 복합체 위에서의 마스킹 역원 연산 방식을 변형, 중복되는 곱셈을 발견함으로써 기존 Zakeri의 방법보다 총 게이트 수가 10.5% 절감될 수 있는 마스킹 역원 방법을 제안한다.

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 values in the algorithm computations(encryption, decryption, and key-schedule) are well-known among these countermeasures. But the cost of non-linear part is extremely high in the masking method of block cipher, and so the inversion of S-box is the most significant part in the case of AES. This fact make various countermeasures be proposed for reducing the cost of masking inversion and Zakeri's method using normal bases over the composite field is known to be most efficient algorithm among these masking method. We rearrange the masking inversion operation over the composite field and so can find duplicated multiplications. Because of these duplicated multiplications, our method can reduce about 10.5% gates in comparison with Zakeri's method.

Keywords: Side Channel Attack, Masking Method, AES, S-box, Composite field

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[13]. 이러한 부채널 공격이 소개되면서 많은 암호시스템 설계자들은 효율적인 대응법들을 연구하기 시작했고, 부채널 공격 중 하나인 차분 전력 분석(Differential Power Analysis, DPA)[11,12,14]에 대한 대응법으로는 마스킹 대응법(masking method)

접수일(2009년 1월 5일), 게재확정일(2009년 7월 28일)
* 이 연구에 참여한 연구자(의 일부)는 '3단계 BK21사업'의 지원비를 받았음.
† 주저자, heeseokkim@cist.korea.ac.kr
‡ 교신저자, christa@kookmin.ac.kr

이 활발히 연구되어지고 있다[4,6,7,8].

마스킹 기법은 평문 x에 대하여 암호문 y를 얻기 위해 마스킹 난수 m을 이용 $x \oplus m$($\oplus$: xor)의 암호문 $y'(=y \oplus m')$을 구한 후, 최종적으로 y를 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 y를 얻을 수 있다. 하지만 블록암호 알고리즘은 비선형 연산을 수행하므로 수정되지 않은 블록 암호 시스템에서 m'값은 x에 따라 다르며 이 값을 중간 값의 누출 없이 아는 것도 상당한 연산을 필요로 한다. 따라서 마스킹 대응법은 이 비선형 연산에 대한 고려가 불가피하다. 블록 암호 AES의 비선형 연산은 S-box 연산으로 그 값이 $S(x \oplus m) = S(x) \oplus m_x$의 형태이며 m_x의 값이 x의 값마다 다르다. 따라서 일반적인 소프트웨어 마스킹 방법에서는 이 m_x의 값이 모든 x에 대해 같은 값이 되게끔 암호 알고리즘의 최초 수행시마다 새로운 마스킹 S-box(MS)를 만든다. 즉, 모든 x에 대해 $MS(x \oplus m) = S(x) \oplus m'$를 만족하는 MS를 생성한다. 하지만 공간적인 제약이 많이 따르는 하드웨어의 마스킹 방법에서는 이러한 마스킹 테이블을 생성, 저장하는 비용으로 인해 MS를 생성하는 방법이 아닌 직접 연산하는 방법을 일반적으로 사용한다.

AES의 S-box는 $GF(2^8)$ 위에서의 역원 연산과 그에 대한 아핀 변환으로 구성되며 실제 하드웨어 마스킹 대응방법의 대부분의 비용은 역원 연산이 차지한다. 따라서 역원 연산에 대한 다양한 마스킹 방법들이 제안 방법의 효율성에 목적을 두고 제안되었다. 물론 $(xm)^{-1} = x^{-1}m^{-1}$의 형태인 곱셈 마스킹(multiplicative masking)은 쉽게 적용될 수 있지만[8,10], x가 0일 때, $xm = x$의 식이 만족하므로 이는 일차 차분 전력 분석에 완전한 대응법이 되지 못한다. 실제 곱셈 마스킹 방법은 이러한 공격에 취약한 것으로 실험적으로 증명되어져 왔다[9].

본 논문에서는 일차 차분 전력 분석에 안전한 덧셈 마스킹(additional masking)[3,7]을 사용한다. 즉, $x \oplus m$으로부터 중간 값의 누출 없이 $x^{-1} \oplus m'$을 연산하는 AES의 마스킹 역원 계산 방법을 제안한다. 마스킹 역원 계산 방법의 기존 방법들은 그 효율성을 위해 복합체(composite field) 위에서의 연산[5]을 사용한다. 본 논문도 그 효율성을 위해 복합체 위에서의 마스킹 방법을 고려하였고, 새로운 연산 식을 세워 기존 방법들에 비해 보다 효율적인 연산 방법을 제시하였다. 제안하는 방법은 기존의 가장 효율적으로 알려진 정규 기저(Normal bases)를 이용한 복합체 위에서의 마스킹 역원 연산 방법[3]에 비해 약 10.5%의 비용 절감을 할 수 있다.

본 논문의 구성은 다음과 같다. 2절은 AES S-box에 대해 설명하고, 3절에서는 기존의 마스킹 역원 계산 방법들을, 4절에서는 제안하는 마스킹 역원 계산 방법에 대해 소개한다. 본 논문에서 제안하는 마스킹 방법의 효율성과 안전성은 5절에서 소개한다.

II. AES S-Box

블록 암호 알고리즘 AES(Advanced Encryption Standard)는 2001년 NIST(National Institute of Standards and Technology. 미국 국립 표준 기술원)에 의해 차세대 표준 암호 알고리즘으로 채택되었다[1]. AES의 한 라운드는 AddRoundKey, Subbytes, ShiftRows, MixColumn의 네 단계로 구성되어 있으며, 이 네 단계 중 Subbyte 연산, 즉 S-Box 연산 부분은 블록 암호 알고리즘의 비선형 연산을 수행한다. AES의 S-Box는 x^{-1}의 아핀 변환(affine transform) 형태로 다음의 연산을 수행한다.

$$S: GF(2^8) \rightarrow GF(2^8)$$
$$S(x) = Bx^{-1} \oplus b$$

위의 연산에서 x^{-1}의 연산은 유한체 $GF(2^8)$ (기약다항식 : $x^8+x^4+x^3+x+1$) 상에서 이루어지며 이 연산을 위한 행렬과 벡터의 값은 다음과 같다.

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

2.1 복합체(Composite Field) 위에서의 역원 연산

일반적으로 AES의 하드웨어 설계에 있어서 S-Box를 저장하고 호출하는 방식은 공간적인 제약이 많이 따르는 편이며, 이로 인해 S-Box를 연산하는 방법이 주로 사용된다. 하지만 x^{-1}의 연산과 아핀 변환

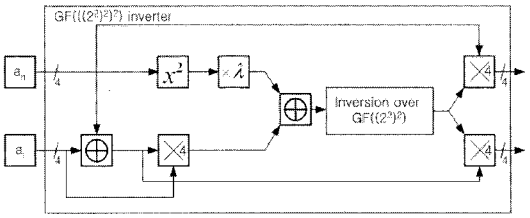
으로 이루어지는 S-Box 연산은 x^{-1}, 즉 $GF(2^8)$에서의 역원계산에서 상당한 비용이 요구되며, 실제 역원계산에 들어가는 비용은 AES 라운드 연산에서 상당한 부분을 차지한다. 따라서 역원 계산의 효율성은 전체 암호 알고리즘의 성능에 크게 영향을 미치며, 이러한 이유로 역원 계산의 비용을 감소시키기 위한 다양한 방법들이 연구되어졌으며 복합체의 개념도 소개되어졌다[2,5]. 즉, $GF(2^8)$ 위에서의 일반적인 역원계산이 아닌 역원 계산의 비용이 작은 복합체로의 변환을 수행한 후 역원 계산을 수행, 결과 값을 다시 $GF(2^8)$ 위의 원소로 역변환 하는 방법이 소개되어졌다.

복합체에서의 역원연산은 부분체(subfield)에서의 연산을 통해 이루어진다. S-box 연산에서 사용되어지는 $GF(2^8)$ 위에서의 연산은 복합체인 $GF(((2^2)^2)^2)$ 위에서의 연산으로 변형되어질 수 있으며, 각 부분체에서 사용되어지는 기약다항식의 형태는 다음과 같다.

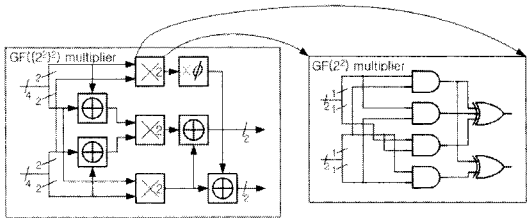
$$\begin{aligned} &GF(2^2)\ over\ GF(2): && P_0(x)=x^2+x+1\\ &GF((2^2)^2)\ over\ GF(2^2): && P_1(x)=x^2+x+\phi\\ &GF(((2^2)^2)^2)\ over\ GF((2^2)^2): && P_2(x)=x^2+x+\lambda \end{aligned}$$

$P_0(x)$의 근을 α, $P_1(x)$의 근을 β, $P_2(x)$의 근을 γ라 한다면($\alpha\in GF(2^2)$, $\beta\in GF((2^2)^2)$,$\gamma\in GF(((2^2)^2)^2)$), $GF(2^2)$의 모든 원소는 $a_1\alpha+a_0$, $GF((2^2)^2)$의 모든 원소는 $(a_3\alpha+a_2)\beta+(a_1\alpha+a_0)$, $GF(((2^2)^2)^2)$의 모든 원소는 $\{(a_7\alpha+a_6)\beta+(a_5\alpha+a_4)\}\gamma+\{(a_3\alpha+a_2)\beta+(a_1\alpha+a_0)\}$의 형태로 표현된다. 연산의 효율성을 위해 $P_1(x)$, $P_2(x)$가 기약인 성질을 만족하는 원소 중에 $GF(2^2)$의 원소 ϕ는 $\alpha((10)_2)$로, $GF((2^2)^2)$의 원소 λ는 $(\alpha+1)\beta((1100)_2)$로 선택하였다.

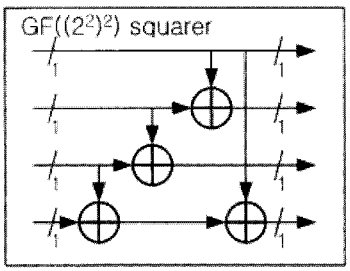
복합체 위에서의 역원 연산은 다음의 연산을 통해 이루어진다. $A\in GF(((2^2)^2)^2)$의 역원 A^{-1} 연산을 위해 $C^{-1}A^{16}(C=A^{17}\in GF((2^2)^2))$을 연산한다. 또한 $C\in GF((2^2)^2)$의 역원 C^{-1}는 $D^{-1}C^4(D=C^5\in GF(2^2))$ 연산을 통해 이루어진다. 즉 $GF(((2^2)^2)^2)$위에서의 역원 연산이 $GF(2^2)$ 위에서의 역원 연산을 통해 이루어진다. 물론, 역원 연산을 위해 추가적인 A^{16}, A^{17}과 같은 연산이 필요하지만 복합체 연산의 특성상 $A=a_h\gamma+a_l$의 16승 연산 결과는 $a_h\gamma+(a_h+a_l)$의 4비트 xor 연산, $(a_h\gamma+a_l)^{17}$의 연산 결과는 $a_h^2\lambda+(a_h+a_l)a_l$로 $GF((2^2)^2)$ 위에서의 한 번의 제곱 연산과 곱셈 연산만을 요구한다. 다음 그림은 $GF(((2^2)^2)^2)$위에서의 역원 연산과 $GF((2^2)^2)$ 위에서의 곱셈, 제곱 연산의 그림이다[2].



[그림 1] $GF(((2^2)^2)^2)$ 역원 연산[2]



[그림 2] $GF((2^2)^2)$ 곱셈 연산[2]



[그림 3] $GF((2^2)^2)$ 제곱 연산[2]

[그림 1]에 따른 $A=a_h\gamma+a_l$의 역원 $A^{-1}=a_h'\gamma+a_l'$를 계산하는 수식은 식 (1)과 같다.

$$\begin{aligned} d&=\lambda a_h^2+a_l(a_h+a_l)\\ d'&=d^{-1}\\ a_h'&=d'a_h\\ a_l'&=d'(a_h+a_l) \end{aligned} \tag{1}$$

III. 기존의 inversion 마스킹

AES의 마스킹 방법의 대부분의 추가 비용은 비선형 연산인 S-box의 역원 연산에 의해 소비된다. 따라서 역원 연산의 마스킹에 대한 효율성을 증대시키고자 다양한 대응 방법들이 제안되어졌다.

3.1 Akkar의 방법[10]

Akkar에 의해 제안된 방법은 곱셈 마스킹 방법을 사용한다. 이 방법은 $x \oplus m$ 값으로부터 $x^{-1} \oplus m$ 값을 얻기 위해 난수 y를 이용, 다음의 연산을 수행한다.

$$x^{-1} \oplus m = (((\underbrace{(x \oplus m)y \oplus my}_{xy})^{-1} \oplus my^{-1})y$$

Akkar가 제안한 방법은 곱셈 마스킹에 대한 취약함 이외에도 상당한 추가 연산을 요구한다. 추가 연산들은 모두 $GF(2^8)$ 위에서의 연산으로 곱셈 연산 4개, 역원 연산 1개, 8비트 XOR 연산 2개이다. 다음 장에서 제안할 우리의 방법과 동등한 비교를 위해 Akkar의 방법이 역원 연산을 복합체 위에서 수행한다는 것과, 최적화된 $GF(2^8)$ 위에서의 곱셈 연산이 복합체인 $GF((2^2)^2)$ 위에서의 곱셈 연산 3개, 제곱 연산 1개와 대략적으로 비슷하다는 것을 가정한다. 즉, $GF((2^2)^2)$ 위에서의 상위 연산인 곱셈 연산은 Akkar의 방법에서 총 4*3+2*3=18개가 필요하다.

3.2 Blomer 의 방법[8]

Blomer은 곱셈에 대한 덧셈 마스킹 방법을 제안한다. 즉, 두 입력 $a'(=a \oplus r)$, $b'(=b \oplus s)$에 대해 $ab \oplus t$를 출력할 수 있는 곱셈 마스킹 방법을 제안한다. 이 연산은 $t \oplus a'b' \oplus a's \oplus b'r \oplus rs$의 연산을 통해 이루어지며, 따라서 4개의 일반 곱셈기를 요구한다. 즉, Blomer의 곱셈 마스킹 연산을 적용하여 복합체 위에서의 AES 마스킹 역원 연산기를 설계할 경우, $GF((2^2)^2)$위에서의 곱셈은 총 4*3=12개가 필요하다.

3.3 Oswald의 방법 & Zakery 의 방법[3,7]

Oswald와 Zakeri에 의해 제안된 방법은 복합체 위에서의 전체적인 구조에 대한 마스킹 방법이다. 즉, 연산의 효율성을 위해 전체 구조에서 $GF((2^2)^2)$ 곱셈에 대해 중복되는 수식을 찾아 그 비용을 감소 시켰다. Oswald의 방법[7]과 Zakery의 방법[3]에서 가장 큰 차이는 기저의 선택에 있으며 Oswald의 방법은 다항식 기저(Polynomial bases)를, Zakeri의 방법은 정규 기저(Normal bases)를 선택하였다. 두 방법 모두 $GF((2^2)^2)$위에서의 곱셈기는 8개가 필요하지만, 그 하위 연산들인 $GF((2^2)^2)$에서의 제곱 연산 및 덧셈 연산에 대해서 Zakeri의 비용이 더 작게 요구된다. 이 두 방법은 역원 연산의 전체 구조에 대해 마스킹을 설계하는 방법으로 하위 연산의 마스킹 방법들을 기초로 설계된 기존 방법들의 불필요한 연산들을 제거하여 현재까지 알려진 방법들 중 최적화된 것으로 알려져 있다.

IV. 제안하는 마스킹 역원기

본 절에서는 기존의 방법보다 공간적인 비용을 감소할 수 있는 마스킹 역원기를 소개한다. 제안하는 마스킹 역원기의 소개에 앞서 쉬운 기술을 위해 다음과 같은 기호를 정의한다.

- $a \| b = a^* 2^{length(b)} + b$
- $A : GF(((2^2)^2)^2)$역원기의 실제 입력 값. $(A = (a_h \| a_l),\ a_h, a_l \in GF((2^2)^2))$
- $A^{-1} : GF(((2^2)^2)^2)$역원기의 실제 출력 값. $(A^{-1} = (a_h' \| a_l'),\ a_h', a_l' \in GF((2^2)^2))$
- $m : GF(((2^2)^2)^2)$의 원소로 A의 마스킹 값. $(m = (m_h \| m_l),\ m_h, m_l \in GF((2^2)^2))$
- $m' : GF(((2^2)^2)^2)$의 원소로 역원기의 출력 마스킹 값. $(m' = (m_h' \| m_l'),\ m_h', m_l' \in GF((2^2)^2))$
- $d : GF((2^2)^2)$역원기의 실제 입력 값.
- $d' : GF((2^2)^2)$역원기의 실제 출력 값.
- $m_d : GF((2^2)^2)$의 원소로 d의 마스킹 값.
- $m_d' : GF((2^2)^2)$의 원소로 $GF((2^2)^2)$ 역원기의 출력 마스킹 값.
- $A + m = \widetilde{a_h} \| \widetilde{a_l} = (a_h + m_h) \| (a_l + m_l)$
- $d' + m_h = \widetilde{d'}$
- $A^{-1} + m' = \widetilde{a_h'} \| \widetilde{a_l'} = (a_h' + m_h') \| (a_l' + m_l')$

[그림 1]의 복합체 위에서의 역원 연산을 안전하게 설계하기 위해 제안하는 마스킹 역원 계산은 다음의 네 단계로 이루어진다.

1 단계. $A + m = (a_h + m_h)\gamma + (a_l + m_l)$로부터 $d + m_d$의 계산.
2 단계. $d + m_d$로부터 $d' + m_d'$의 계산
3 단계. $d' + m_d'$과 $a_h + m_h$로부터 $d'a_h + m_h'$의 계산
4 단계. $d' + m_d'$, $a_h + m_h$, $a_l + m_l$로부터 $d'(a_h + a_l) + m_l'$의 계산

1 단계는 [그림 1]에서 $GF(((2^2)^2)^2)$ 역원기의 8 비트 입력 값 A로부터 $GF(((2^2)^2))$ 역원기의 4 비트 입력 값 d를 안전하게 연산하기 위한 단계이다. 따라서 A는 마스킹 값 m에 의해 마스킹 되어 입력되어지며, d는 마스킹 값 m_d에 의해 마스킹 되어 연산된다. 물론 이 단계에서 연산되는 모든 중간 값은 편중되지 않은 난수에 의해 마스킹 되어야만 한다. 2 단계는 [그림 1]에서 $GF(((2^2)^2))$ 역원 연산을 안전하게 연산하기 위한 단계로 마스킹 된 입력 값 $d+m_d$로부터 마스킹 된 출력 값 $d'+m_d'$를 안전하게 연산한다. 3 단계에서는 [그림 1]에서 $GF(((2^2)^2))$ 역원기의 출력 값 d'과 $GF(((2^2)^2)^2)$의 입력 값의 상위 4비트 a_h로부터 $GF(((2^2)^2)^2)$ 역원기의 출력 값의 상위 4비트 $a_h'(d'a_h)$를 안전하게 연산하기 위한 단계로 제안하는 역원기에서 a_h'은 m_h'에 의해 마스킹 되어 출력된다. 마지막 4 단계는 [그림 1]에서 $GF(((2^2)^2)^2)$ 역원기의 출력 값의 하위 4비트 $a_l'(d'(a_h+a_l))$에 해당하는 부분을 안전하게 연산하기 위한 단계이다. 이 부분은 4비트 난수 m_l'에 의해 마스킹 되어 출력된다.

각 단계의 연산에서 나타날 수 있는 모든 중간 값은 A의 8비트 값과 특정 상수로부터 생성되어질 수 있는 모든 중간 값에 독립적이어야만 한다. m_d, m_d'은 $GF((2^2)^2)$의 마스킹 역원 계산에 대한 입출력 마스킹 값으로 본 논문에서는 효율적인 연산을 위해 $m_d'=m_h$로 선택한다.

본 논문에서는 2 단계의 $GF((2^2)^2)$ 위에서의 마스킹 역원 계산은 언급하지 않는다. 이는 $GF(((2^2)^2)^2)$ 위에서의 마스킹 역원 방법이 $GF((2^2)^2)$에서도 동일하게 적용될 수 있기 때문이다. $GF(((2^2)^2)^2)$에서의 역원 연산은 [그림 2], [그림 3]에서 보이는 것처럼 $GF((2^2)^2)$ 곱셈기에 대부분의 비용이 소요된다. 따라서 본 논문에서는 마스킹 역원 계산에서 사용되어지는 $GF((2^2)^2)$ 곱셈기의 개수를 최소화하는 방법을 고려하였다.

1 단계. $d+m_d$의 연산

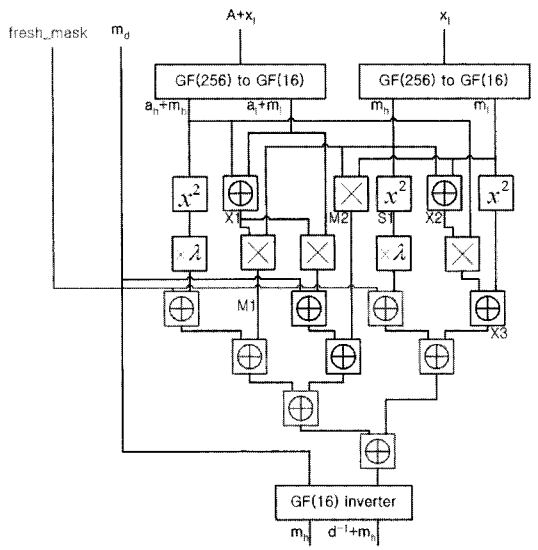
$d+m_d$는 다음의 수식 연산을 통해 이루어진다.

$$\begin{aligned} d+m_d = {} & \underbrace{\lambda(\widetilde{a_h})^2}_{a}+\underbrace{m_h(\widetilde{a_h}+\widetilde{a_l})}_{M1}+m_d \\ & +\underbrace{\widetilde{a_l}(\widetilde{a_h}+\widetilde{a_l})}_{b}+\underbrace{m_h m_l}_{M2}+ \\ & \underbrace{\widetilde{a_h}(m_h+m_l)+m_l^2}_{X3}+\lambda\underbrace{m_h^2}_{S1} \end{aligned} \tag{2}$$

각 곱셈, 제곱, 상수곱 연산에 대한 안전성은 다음 절에 논의한다. 식 (2)의 각 곱셈, 제곱, 상수곱에 대한 결과 값들을 덧셈 연산(XOR 연산)할 때에는 의미 있는 중간 값과 연관성이 없도록 순서를 잘 고려하여 XOR 연산하여야 한다. 예를 들어, (2)의 식을 연산하는 과정에서 $a+S1$의 연산이 먼저 수행된다면 이는 λa_h^2 값으로 전력 분석 공격에 취약점이 드러난다. 이러한 단순한 예에서 뿐만 아니라 마스킹 값의 각 비트 값이 0.5가 아닌 다른 확률로 '0' 또는 '1'에 편중된다면 이 부분도 역시 취약점이 될 수 있다. 이러한 편중을 막기 위해 연관성이 생기는 경우는 fresh_mask를 사용하여 이를 제거해야 한다. fresh_mask란 최종 결과의 마스킹 값과는 연관이 없는 값으로 중간 연산 결과의 데이터 편중을 막기 위해 사용되어지는 값이다. fresh_mask값(fm)을 사용해서 식 (2)의 순서를 고려하여 변형하면 식 (3)과 같다.

$$\begin{aligned} d+m_d = {} & ((fm+a)+M1)+((m_d+b) \\ & +M2))+((fm+\lambda S1)+X3) \end{aligned} \tag{3}$$

[그림 4]는 식 1 단계에 대한 식 (3)을 도식화한 것이다.



[그림 4] $d+m_d$의 연산

3 단계. $d'a_h+m_h'$의 연산

$d'a_h+m_h'$는 다음의 수식 연산을 통해 이루어진다.

$$d'a_h+m_h' = m_h'+X3+\underbrace{(\tilde{a_h}+(m_l+m_h))(\tilde{d'}+m_l)+m_l'}_{M4} + \underbrace{\tilde{d'}m_l+m_l'+S1+M2}_{X4} \quad (4)$$

식 (4)에서 m_l'을 두 번 더하는 이유는 fresh_mask를 사용하는 것과 같은 이유이다. 즉, 각 덧셈 연산의 결과가 m_l'으로 인해 중간 값과 완벽하게 연관성이 없는 데이터가 되기 위함이다. 식 (4)를 우선 순위를 고려하여 변형하면 식 (5)와 같다.

$$d'a_h+m_h' = (((m_h'+X3)+M4)+m_l') + \underbrace{((\tilde{d'}m_l+m_l')+(S1+M2))}_{X4} \quad (5)$$

4 단계. $d'(a_h+a_l)+m_l'$의 연산

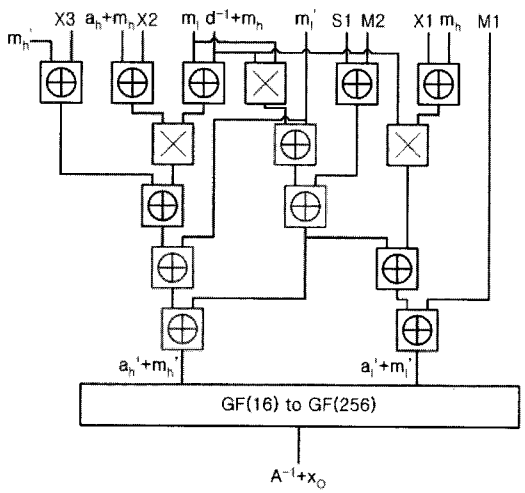
$d'(a_h+a_l)+m_l'$는 다음의 수식 연산을 통해 이루어진다.

$$d'(a_h+a_l)+m_l' = (\tilde{a_h}+\tilde{a_l}+m_h)\tilde{d'}+M1+X4 \quad (6)$$

식 (3)과 마찬가지로 안전성을 고려하여 순서를 재배치 하면 식 (6)은 식 (7)과 같이 변형된다.

$$d'(a_h+a_l)+m_l' = ((\tilde{a_h}+\tilde{a_l}+m_h)\tilde{d'}+X4)+M1 \quad (7)$$

3 단계와 4 단계의 식 (5), 식 (7)을 도식화하면 [그림 5]와 같다.



[그림 5] $d'a_h+m_h'$, $d'(a_h+a_l)+m_l'$의 연산

[그림 4]와 [그림 5]에서 안전성을 위해 추가한 fresh_mask 혹은 m_l' 값은 최종적인 결과 값에는 영향을 미치지 않음을 확인할 수 있다.

본 논문에서 제안하는 방법에서 fresh_mask와 $GF(16)$ inverter의 입력 마스킹인 m_d는 새로 생성한 4비트 난수일 수도 있지만, 난수 생성에 대한 비용을 감소하기 위해 m_h' 또는 m_l'을 선택해서 사용할 수 있다. 이 값은 (3)의 식을 연산할 때 다른 값들과는 연관성이 없는 난수로 안전성에는 영향을 미치지 않는다. 하지만 $fm=m_d=(m_h'$ or $m_l')$의 경우 $((fm+a)+M1)+((m_d+b)+M2)$의 연산은 $(a+M1+b+M2)=\lambda a_h^2+a_l(a_h+a_l)+(a_h+\lambda m_h+m_h)m_h+(a_h+m_h+m_l)m_l$값이 며 이 값의 마스킹 값은 $(a_h+\lambda m_h+m_h)m_h+(a_h+m_h+m_l)m_l$이 된다. 마스킹 값에서 $(a_h+m_h+m_l)m_l$은 다음 절에서 소개하는 안전성 증명에 의해 안전성이 보장된다. 하지만 $(a_h+\lambda m_h+m_h)m_h$의 값은 a_h에 독립이 아니며, 두 값을 더한 값의 안전성을 더욱 보장하기 위해 본 논문에서는 $fm\neq m_d$의 값을 사용하도록 한다.

V. 제안하는 inversion 마스킹의 효율성 및 안전성

5.1 안전성

제안하는 마스킹 방법은 다음의 Lemma에 의해 그 안전성을 검증받을 수 있다. 다음 Lemma의 증명은 [7]의 논문을 참고한다.

Lemma 1) 원소 a가 $GF(2^n)$의 원소이고, m_a가 a와 무관한 $GF(2^n)$ 위에서 균등한 분포에 의해 선택되어진 값일 때, $a+m_a$는 a에 독립이다.

Lemma 2) 원소 a, b가 $GF(2^n)$의 원소이고, m_a, m_b가 a, b와 무관한 $GF(2^n)$ 위에서 균등한 분포에 의해 선택되어진 값일 때, $(a+m_a)(b+m_b)$는 a, b에 독립이다.

Lemma 3) 원소 a가 $GF(2^n)$의 원소이고, m_a, m_b가 a와 무관한 $GF(2^n)$ 위에서 균등한 분포에 의해 선택되어진 값일 때, $(a+m_a)m_b$는 a에 독립이다.

Lemma 4) 원소 a가 $GF(2^n)$의 원소이고 원소 λ가 $GF(2^n)$에서의 고정된 상수 값일 때, m_a가 a, λ와 무관한 $GF(2^n)$ 위에서 균등한 분포에 의해 선택되어진 값이면 $(a+m_a)^2\lambda$는 a에 독립이다.

*Lemma 1*은 임의의 편중되지 않은 난수 값이 실제 값을 완벽하게 랜덤화할 수 있음을 의미하며 이는 마스킹 구조가 일차 차분전력분석으로부터 안전할 수

〔표 1〕 기존 결과와 제안 방법의 비교

	$GF((2^2)^2)$ 곱셈	$GF((2^2)^2)$ 제곱	$GF((2^2)^2)$ 상수곱
Akkar〔10〕	18	6	4
Blomer〔8〕	12	1	2
Oswald〔7〕	8	4	2
Zakeri〔3〕	8	2	2
제안방법	7	3	2

있음을 의미한다. *Lemma 2*와 *Lemma 3*은 안전한 역원기의 설계를 위해 곱셈기의 입력 값으로 선택되어질 수 있는 값들의 구조를 의미한다. 즉, 곱셈기의 두 입력 값은 서로 연관성이 없는 두 마스킹 값을 사용해야만 한다. 마지막으로 *Lemma 4*는 제곱, 상수 곱 연산의 입력 값이 편중되지 않은 난수에 의해 마스킹 될 경우, 출력 값 또한 일차 차분 전력분석으로부터 안전하게 보호될 수 있음을 의미한다.

제안하는 역원기는 안전한 것으로 증명된 위의 *Lemma*만을 사용하여 설계되었으며, 따라서 그 안전성이 증명된다.

5.2 효율성

〔표 1〕은 제안하는 마스킹 역원 계산 방법과 기존의 결과들을 비교한 것이다. 기존 결과와의 비교는 상위 연산($GF((2^2)^2)$ 곱셈, $GF((2^2)^2)$ 제곱 연산, $GF((2^2)^2)$ 상수곱 연산)을 중심으로 이루어졌다. 하위 연산인 $GF((2^2)^2)$ 덧셈 연산은 다섯 가지 방법에서 큰 차이가 없으며, 그 비용이 상위 연산에 비해 상당히 작기 때문에 고려하지 않았다. 또한 $GF((2^2)^2)$에서의 역원 연산도 상위 연산이 유사하게 적용되므로 전체 역원 연산에 대한 비용이 상위 연산의 비용에 비례한다고 간주되어질 수 있다. $GF((2^2)^2)$의 곱셈과 제곱은 〔그림2,〕 〔그림 3〕에서 보이는 것처럼 비용 차이가 크다. 실제 $GF((2^2)^2)$의 곱셈의 비용은 제곱의 비용보다 약 5.5배의 게이트 수가 요구되며, 상수곱 연산보다 약 7.5배가 요구된다. 즉, 제안하는 방법은 Zakeri의 방법에 비해 약 10.5%의 게이트 절감 효과를 볼 수 있다.

VI. 결 론

본 논문에서는 AES S-box에 사용되어지는 역원 계산에 대한 효율적인 마스킹 기법을 제안하였다. 제안하는 방법은 기존의 복합체 위에서의 역원 연산 방법을 이용하여 설계되었으며, 마스킹 방법을 적용 시 나타날 수 있는 곱셈 연산에서 중복될 수 있는 수식을 찾아 연산량을 단축시켰다. 제안하는 방법은 기존 역원 마스킹 기법 중 가장 비용이 적은 것으로 알려진 Zakeri의 방법보다 총 게이트 수를 약 10.5% 절감할 수 있다.

참 고 문 헌

〔1〕 Advanced Encryption Standard(AES), FIPS PUB 197, Nov. 2001. http://csrc.nist.gov/encryption/aes

〔2〕 A. Satoh, S. Morioka, K. Takano, and S. Munetoh, "A Compact Rijndael Hardware Architecture with S-Box Optimization," ASIACRYPT 2001, LNCS 2248, pp. 239-254, 2001.

〔3〕 B. Zakeri, M. Salmasizadeh, A. Moradi, M. Tabandeh, and M. Shalmani, "Compact and Secure Design of Masked AES S-Box," ICICS 2007, LNCS 4861, pp. 216 - 229, 2007.

〔4〕 C. Herbst, E. Oswald, and S. Mangard, "An AES Smart Card Implementation Resistant to Power Analysis Attacks," ACNS 2006, LNCS 3989, pp. 239-252, 2006.

〔5〕 D. Canright, "A Very Compact Rijndael S-box.Technical Report," NPS-MA-04-001, Naval Postgraduate School, 2004.

〔6〕 E. Oswald and K. Schramm, "An Efficient Masking Scheme for AES Software Implementations," WISA 2005, LNCS 3786, pp. 292 - 305, 2006.

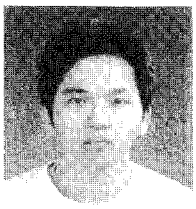
〔7〕 E. Oswald, S. Mangard, N. Pramstaller, and V. Rijmen, "A Side-Channel Analysis Resistant Description of the AES S-box," FSE 2005, LNCS 3557, pp. 413 - 423, 2005.

〔8〕 J. Bl¨omer, J. Guajardo, and V. Krummel, "Provably Secure Masking of AES," SAC 2004, LNCS 3357, pp. 69 - 83, 2005.

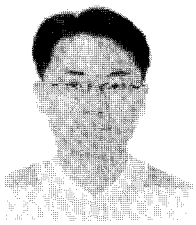
〔9〕 J.D. Golic and C. Tymen, "Multiplicative Masking and Power Analysis of AES,"

CHES 2002, LNCS 2523, pp. 198 - 212, 2003.
[10] M.L. Akkar and C. Giraud, "An Implementation of DES and AES, Secure against Some Attacks," CHES 2001, LNCS 2162, pp. 309-318, 2001.
[11] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," CRYPTO 1999, LNCS 1666, pp. 388-397, 1999.
[12] P. Kocher, J. Jaffe, and B. Jun, "Introduction to differential power analysis and related attacks," http://www. cryptography. com/dpa/technical, 1998.
[13] P. Kocher, J. Jaffe, and B. Jun, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Others Systems," CRYPTO 1996, LNCS 1109, pp. 104-113, 1996.
[14] T.S. Messerges, E.A. Dabbish, and R.H. Sloan, "Power analysis attacks on modular exponentiation in Smart cards," CHES 1999, LNCS 1717, pp. 144-157, 1999.

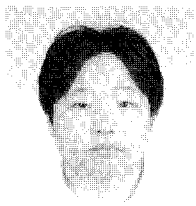
〈著者紹介〉



김 희 석 (Hee Seok Kim) 학생회원
2006년 2월: 연세대학교 수학과 졸업(학사)
2008년 2월: 고려대학교 정보경영공학전문대학원 공학석사
2008년 3월 ~ 현재: 고려대학교 정보경영공학전문대학원 박사과정
〈관심분야〉 부채널 공격, 암호시스템 안전성 분석 및 고속구현, 암호칩 설계 기술



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 ~ 2009년 2월: 한국전자통신연구원 정보보호연구단 선임연구원
2009년 3월 ~ 현재: 국민대학교 수학과 조교수
〈관심분야〉 암호시스템 안전성 분석 및 고속 구현, 부채널 분석, RFID/USN 정보보호 기술



김 태 현 (Tae Hyun Kim) 정회원
2002년 2월: 서울 시립대학교 수학과 이학사
2004년 8월: 고려대학교 정보보호 대학원 공학석사
2009년 2월: 고려대학교 정보경영공학전문대학원 공학박사
〈관심분야〉 부채널 공격, 공개키 암호 알고리즘, 암호칩 설계 기술

홍 석 희 (Seok Hie 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 ~ 2008년 8월: 고려대학교 정보경영공학전문대학원 조교수
2008년 9월 ~ 현재: 고려대학교 정보경영공학전문대학원 부교수
〈관심분야〉 대칭키 암호 알고리즘, 공개키 암호 알고리즘, 포렌식