2013년 7월 전자공학회 논문지 제 50 권 제 7 호
Journal of The Institute of Electronics Engineers of Korea Vol. 50, NO. 7, July 2013 http://dx.doi.org/10.5573/ieek.2013.50.7.068

논문 2013-50-7-9

카운터 오류 공격에 안전한 Miller 알고리듬

(A Proposal for Enhanced Miller Algorithm Secure Against Counter Fault Attack)

배 기 석*, 박 영 호**

(KiSeok Bae and YoungHo Park©)

요 약

최근 이동 ad hoc 네트워크에 적합한 ID기반 암호시스템의 구현을 위한 Weil, Tate, Ate와 같은 페어링 연산 기법에서는 밀러 알고리듬이 사용된다. 페어링 연산의 활용 영역이 넓어짐에 따라 다양한 오류 공격이 제안되고 있으며, 그중 카운터 오류 공격이 가장 강력한 위협으로 여겨진다. 따라서 본 연구에서는 카운터 오류 공격에 대한 새로운 대응책을 제안한다. 제안 기법은 중간 값을 저장하는 위치를 랜덤하게 함으로써 오류에 의한 변형 가능성을 줄이고, if 구문에 의한 부채널 특성을 제거하여 오류 공격의 시도 자체를 어렵게 한다.

Abstract

Recently, there has been introduced various types of pairing computations to implement ID based cryptosystem for mobile ad hoc network. According to spreading the applications of pairing computations, various fault attacks have been proposed. Among them, a counter fault attack has been considered the strongest threat. Thus this paper proposes a new countermeasure to prevent the counter fault attack on Miller's algorithm. The proposed method is able to reduce the possibility of fault propagation by a random index of intermediate values. Additionally, it is difficult to challenge fault attacks on the proposed method since a simple side channel leakage of 'if' branch is eliminated.

Keywords : 이동 ad hoc 네트워크, 페어링 기법, 카운터 오류 공격, 밀러 알고리듬

Ⅰ. 서 론

이동 ad hoc 네트워크의 활용에 대한 연구가 활발해짐에 따라 이동 ad hoc 네트워크에서의 개인 정보 보안이 중요한 이슈가 되고 있다. 그러나 ad hoc 네트워크를 구성하는 단말장치는 낮은 연산 능력과 저장 공간 등이 부족하기 때문에 장치 자체의 보안성보다 내부에서 동작하는 알고리듬 차원의 개선이 필요하다. 최근에는 이론적으로 높은 안전도를 지닌 페어링 기반의 암호시스템을 사용한 보안 알고리듬의 사용이 각광받고 있다[1~2]. 이 때 실제 구현을 목적으로 페어링 기법을 적용하므로 현재 가장 강력한 물리적 위협으로 알려진 부채널 분석 공격에 대한 취약성의 검증이 필요하다[3~6].

페어링 기법에 대한 오류주입공격은 Page와 Vercauteren에 의해서 처음 소개되었다[7]. 제안된 카운터 오류 공격은 Tate 페어링 연산을 효율적으로 수행하기 위해 제안된 Duursma-Lee 알고리듬과 Eta 페어링 연산에서 사용되는 Kwon-BGOS 알고리듬에 대해서 루프 횟수를 변형하는 오류 모델을 사용한다. 다음으로 Whelan과 Scott에 의해서 Weil 페어링과 Eta 페어링에 대한 오류주입공격이 제안되었다[8]. [8]에서는 알고리듬 연산중에 오류를 주입하여 내부의 중간 값을 변형하는 오류 모델을 사용한다. [7]의 경우 원하는 오류 결과문

* 정회원, 삼성전자
(Samsung Electronics Co., Ltd.)
** 정회원, 경북대학교 산업전자공학과
(Kyungpook National University)
© Corresponding Author(E-mail: parkyh@knu.ac.kr)
※ 이 논문은 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(NRF-2012R1A1A4A01002603)
접수일자: 2013년3월4일, 수정완료일: 2013년6월19일

쌍을 얻기 위한 수차례의 반복적인 오류 주입 시도가 필요하며, [8]의 경우 Weil 페어링에 공격을 적용하기 위해서 연산 도중에 중간 값의 y좌표의 부호를 변경해야하는 높은 정밀도를 요구하고 있다. 따라서 구현이 보다 용이한 카운터 오류 공격이 페어링 기법의 실용화에 있어 가장 강력한 위협으로 여겨지고 있다.

본 논문에서는 중간 값이 저장되는 인덱스를 랜덤하게 설정하여 루프 횟수가 변형될 경우 오류 페어링 결과 값을 획득할 수 없는 밀러 알고리듬을 제안한다. 스칼라 곱셈을 위한 double-and-add always 알고리듬의 형태에 기반하여 밀러 연산을 수행하기 때문에 전력 신호의 분석에 의한 밀러 루프의 라운드 구분이 용이하지 않아 오류 주입의 가능성을 원천적으로 방어할 수 있다.

Ⅱ. 페어링 연산과 밀러 알고리듬

1. 페어링 기법

위수가 l인 두 아벨군 $G_1 \subset E(F_q)$과 $G_2 \subset E(F_{q^k})$와 동일한 위수를 가지는 곱셈에 관한 아벨군 $G_3 \subset F_{q^k}^*$가 있을 때, 페어링 연산은 겹선형성(bilinearity)과 비축퇴성(non-degenerate)을 만족하는 사상(mapping) 함수의 형태로 정의 된다.

$$e : G_1 \times G_2 \rightarrow G_3 \tag{1}$$

일반적으로 페어링 연산은 q개의 원소를 가지는 유한체 F_q상에 존재하는 타원곡선 E 에서 정의된다. 여기서q는 표수(characteristic) p의 멱승 값이다. l은 $l|E(F_q)$를 만족하는 q와 서로소인 양의 정수라고 한다면, embedding degree k는 $l|q^k-1$을 만족하는 가장 작은 정수이다. 제안하는 공격 기법의 대상인 아핀좌표상에서는 Weistrass 타원곡선 방정식이 유한체 F_{q^k}의 표수(characteristic) q가 2나 3이 아닌 경우, 아래의 수식 (2)와 같이 정의 된다.

$$Y^2 = X^3 + aX + b,\ a,b \in F_{q^k} \tag{2}$$

페어링의 결과 값은 연산에서 사용하는 알고리듬마다 차이가 있으나 입력 값 P를 l번 더한 값인 $[l]P \in G_1$의 유리함수(rational function)에 $Q \in G_2$의 divisor를 인가한 값의 형태를 취한다. Divisor에 관해서는 [9]에서 자세히 설명하고 있다. 페어링 결과 $f_{l,P}$는 누적곱셈의 형태이므로 이를 쉽게 구하기 위해서 밀러 알고리듬[10] 등을 사용한다. 보안 영역에서 사용되고 있는 페어링 연산 기법들은 유사한 형태로 구성되어 있으며, 특히 Weil, Tate, Ate 페어링 기법에서는 밀러 알고리듬이 가장 중요한 역할을 하고 있다.

2. 밀러 알고리듬

밀러 알고리듬은 타원곡선 상의 곱셈과 덧셈 특성에 따라 입력 점 P에 대해서 더블링과 덧셈 연산을 수행한다. 위수가 l인 점 P를 입력으로 하여 타원곡선 상의 스칼라 곱셈과 동일하게 $[l]P$를 연산하는 동안의 점과 점 P간의 접선에 대한 함수를 누적 곱하여 계산한다. 아핀 좌표를 사용하는 경우에는 밀러 알고리듬의 매 루프 연산마다 중간 값 점 T의 접선(tangent line)과 수직선(vertical line)의 함수들을 도출하여, 두 함수를 나눈 유리 함수 f_P을 계산한다. 이때 유리 함수 f_P는 그림 1과 같이 더블링인 경우 h_1, 덧셈인 경우 h_2로 정의할 수 있다.

도출된 함수 f_P에 점 Q의 좌표 값을 인가하여 $F_{q^k}^*$상의 한 원소인 밀러 변수(miller variable) $f_P(Q)$를 계산한다. 매 루프마다 연산된 밀러 변수들의 누적 곱 $(f_1(Q)f_2(Q)\cdots f_l(Q))$은 실제 밀러 알고리듬의 출력 값이 되며, 선택한 입력 값 점 P와 연관되기 때문에 $f_{l,P}(Q)$와 같이 표기한다. 실제 알고리듬은 아래 그림과 같다.

입력 : $P \in G_1$, $Q \in G_2$, $l = (l_{n-1}...l_0)$: radix 2 presentation
출력 : $f_{l,P}(Q) \in G_3$

1. $T \leftarrow P$, $f \leftarrow 1$
2. for $i = n-1$ to 0 do
3. $T \leftarrow [2]T$ //doubling
4. $f \leftarrow f^2 \cdot h_1(Q)$, $h_1(Q) = t_{T,T}(Q)/v_{2T}(Q)$
5. if $l_i = 1$ then
6. $T \leftarrow T + P$ //addition
7. $f \leftarrow f \cdot h_2(Q)$, $h_2(Q) = t_{T,P}(Q)/v_{T+P}(Q)$
8. end if
9. end for
10. return $f \in F_{q^k}^*$

그림 1. 밀러 알고리듬
Fig. 1. Miller algorithm.

Ⅲ. 카운터 오류 공격과 대응기법

1. 카운터 오류 공격[7]

카운터 오류 공격은 알고리듬의 라운드를 반복 수행할 때, 루프 횟수(loop counter)에 주입된 오류에 의해 변형되어 알고리듬이 정상적인 반복 횟수를 수행하지 못하게 만든다. 정상적인 루프 횟수가 l의 이진표현 길이인 m이라할 때, 오류로 인해 변형된 루프 횟수는 Δ만큼의 차이를 보이게 된다. 이때의 오류 페어링 결과는 $e_{\Delta}(P,Q)$로 표기한다.

이때 가정한 오류 모델은 루프 횟수를 확인하는 분기 단계를 대상으로 하는 글리치 오류나 루프 횟수가 임시로 저장되는 메모리 또는 레지스터에 대한 오류이다. 메모리나 레지스터에 대한 직접적인 오류주입은 정확한 대상을 찾아야한다는 정밀도에서 구현이 용이하지 않다. 반면 밀러 알고리듬 연산 도중의 분기 단계에서의 오류는 대상 장치의 파형 또는 전자파를 측정하여 단순 전력 분석을 통한 밀러 루프들을 구분할 수 있다는 가정 하에 보다 용이하게 구현할 수 있다.

만약 주입된 오류로 인해 $\Delta = m+1$으로 루프 횟수가 변형된다면 정상 페어링 결과와 오류 페어링 결과의 차이를 통해 비밀 값을 추출할 수 있다.

$$R_1 = e_m(P,Q)$$

$$R_2 = e_{m+1}(P,Q)$$

두 페어링 결과의 비율 또는 나누어진 연산 결과 R_1/R_2는 밀러 알고리듬의 한 루프 동안 연산되는 h_1또는 h_1h_2 함수들이다. 함수의 입력 값 중의 하나인 점 Q는 공개 값이기 때문에 이를 이용하여 Δ루프에 해당되는 점 T의 좌표를 추출할 수 있다. 추출한 점 T는 비밀 값 P의 스칼라 곱셈한 값으로 역연산을 통해 P의 좌표를 찾아낼 수 있다.

정상 루프 횟수 보다 하나의 루프를 더 연산시키는 오류 공격은 사용된 오류 모델이 매우 정교해야한다는 구현상의 단점이 있기 때문에 오류 주입으로 인해 임의의 $\Delta = m \pm r$로 변형하는 방법을 사용한다. 공격자는 몇 차례의 오류 주입 시도를 통해서 공격에 필요한 오류 결과문 쌍을 획득할 수 있다.

$$R_1 = e_{m \pm r}(P,Q)$$

$$R_2 = e_{m \pm r+1}(P,Q)$$

앞서 설명과 마찬가지로 두 오류 결과문 쌍의 비율을 통해 하나의 루프 동안 연산된 밀러 변수를 획득하여 비밀 값을 찾아낸다.

원하는 한 쌍의 오류 결과문을 필요로 하기 때문에 카운터 오류 공격은 많은 오류 주입 시도를 필요로 한다. 실제 환경에서는 루프 횟수가 오류에 의해 임의의 값으로 변형되기 때문에 예측이 쉽지 않다. 따라서 기존의 공격들은 변형된 루프 횟수가 연속적인 결과 문들을 얻기 위한 확률적인 분석을 제시하였고, 이는 루프 횟수를 담당하는 레지스터의 크기에 의존하여 예측된다[11].

2. 대응기법

앞서 설명에서 공격자가 공격을 성공할 수 있는 요인 중 하나는 찾아낸 해당 루프에서 연산된 밀러 변수의 입력 값 Q를 알고 있기 때문이다. 따라서 입력 및 중간 결과 값들을 모두 숨길 수 있는 블라인딩 기법이 Page와 Vercauteren 에 의해서 제안되었다[7].

$$e_m(aP,bQ) = e_m(P,Q)^{ab}$$

이 때 $ab = 1(\mathrm{mod}\, l)$을 만족하는 a와 b를 사용한다.

이외에도 랜덤한 점 R을 사용할 수도 있다.

$$\begin{aligned} & e_m(P,Q+R) \cdot e_m(P,R)^{-1} \\ &= e_m(P,Q) \cdot e_m(P,R) \cdot e_m(P,R)^{-1} \\ &= e_m(P,Q) \end{aligned}$$

랜덤 값 또는 랜덤한 점을 사용하면서 밀러 알고리듬이 연산되는 동안의 변수들은 원래의 P나 Q가 아닌 다른 값을 입력으로 하게 되므로 안전하다고 설명하고 있다[7]. 그러나 정상의 페어링 결과와 루프 횟수가 한번 더 수행된 경우의 비율에서는 랜덤 값이나 랜덤한 점과 상관없이 밀러 변수가 생성되므로 여전히 공격에 취약하다는 결과가 발표되었다[12].

또다른 대응방법으로 Ghosh 등에 의해 오류를 감지하는 밀러 알고리듬이 제안되었다[5]. Ghosh 등의 기법은 루프 횟수 또는 최종 루프 값이 저장되는 메모리 또는 레지스터가 사전에 변형될 수 있음을 가정하여 밀러 루프의 연산 이전 단계에서 l_i의 마지막 비트가 0인지 아닌지를 검증한다.

이러한 방식은 루프 연산 동안에 루프 횟수가 변형되었을 때 취약점을 드러낼 수 있다. l_i의 마지막 비트를 검증하는 단계를 'for' 구문이 끝난 후 수행할 때 연산 동안의 오류 또한 감지할 수 있다.

입력 : $P \in G_1$, $Q \in G_2$, $l=(l_{n-1}...l_0)$: radix 2 presentation
출력 : $f_{l,P}(Q) \in G_3$

1. $T \leftarrow P, f \leftarrow 1$
2. if l[0]=0 then
3. return 0
4. for $i=n-1$ to 0 do
5. $T \leftarrow [2]T$ //doubling
6. $f \leftarrow f^2 \cdot t_{T,T}(Q)/v_{2T}(Q)$
7. if $l_i=1$ then
8. $T \leftarrow T+P$ //addition
9. $f \leftarrow f \cdot t_{T,P}(Q)/v_{T+P}(Q)$
10. end if
11. end for
12. return $f \in F_{q^k}^*$

그림 2. Ghosh 등의 밀러 알고리듬
Fig. 2. Ghosh etc.'s Miller algoritm.

블라인딩 기법과 마찬가지로 오류 감지 기법은 타원곡선이나 RSA 암호 알고리듬에서의 전통적인 오류 공격의 대응책이다. 그러나 'if' 구문에 의한 감지는 이차적인 오류 주입에 의해서 쉽게 생략되거나 무시할 수 있음이 이미 밝혀져 있어[13~14] 여전히 카운터 오류 공격에 효과적인 대응책이 될 수 없다.

Ⅳ. 제안하는 안전한 밀러 알고리듬

카운터 오류 공격에 대응하는 가장 효과적인 방법은 변형된 루프 횟수에 해당하는 오류 페어링 결과를 출력하지 않는 것이다. 그러나 'if' 구문 등에 의한 오류 감지 기법은 이중 오류 기술을 사용할 때 쉽게 무력화되기 때문에 효과적이지 않다. 또한 오류 공격은 획득한 오류 페어링 결과에 의존하기 때문에 공격자가 원하는 오류 페어링 결과가 아닌 경우 수식을 통해 획득한 비밀 값이 올바르지 않게 된다.

따라서 제안하는 대응 기법에서는 밀러 알고리듬의 연산 동안 밀러 변수가 임시 저장되는 레지스터의 위치를 랜덤화하고 정상적인 루프 횟수가 수행되었을 때에만 페어링 결과가 저장된 위치를 지정하여 출력하도록 한다. 이 때 'if' 구문에 의한 오류 감지 구문을 사용하지 않고 현재 l_i의 비트 에 의해 밀러 변수가 저장되는 배열의 위치가 매 루프마다 변경되도록 한다. 만약 주입된 오류로 인해서 비정상적인 루프만 수행하더라도 해당 루프에서 공격자가 의도한 오류 페어링 결과가 아닌 랜덤한 레지스터 위치의 값을 획득하게 되어 공격이 불가능하게 된다. 이때 공격자는 획득한 값이 의도한 오류 페어링 값인지 아닌지 확인할 수 있는 별도의 방법이 없기 때문에 강력한 대응 방법이 된다.

또한 오류 공격을 위한 또다른 변수로 단순 전력 분석을 통한 밀러 루프 횟수 판별이 있다. 카운터 오류 공격은 앞 장에서 설명한 바와 같이 변형된 밀러 루프 횟수가 Δ와 $\Delta+1$인 두 개의 오류 페어링 쌍이 필요하다. 최초의 오류로 인해 Δ만큼의 루프를 수행한 오류 페어링 쌍을 획득했을 때, 새로운 페어링 연산동안 다음의 오류 페어링 값을 얻기 위해서 밀러 루프 횟수를 확인할 필요가 있다. 기존의 공격 기법들은 대부분 단순 전력 분석 공격을 통해 오류 주입 위치를 쉽게 찾아낼 수 있음을 가정하고 있다. 따라서 단순 전력 분석 공격을 방어하는 기법을 함께 적용한다면 보다 안전한 밀러 알고리듬이 될 수 있다.

제안하는 기법에서는 전통적인 단순 전력 분석 공격의 방어책인 double-and-add always 알고리듬의 형태에 따라 l_i의 비트 값이 '0'인 경우에만 동작하는 'if' 구문을 제거하여 비트 값에 상관없이 항상 더블링 연산과 h_1 함수의 연산, 덧셈 연산과 h_2 함수의 연산을 수행하도록 한다. 기존의 단순 전력 분석 공격은 'if' 구문에서 소모되는 클럭량 및 지연 시간에 의존하기 때문에 더 이상 밀러 알고리듬의 루프 횟수를 판별할 수 없게 된다. double-and-add always 알고리듬을 사용할 경우의 부채널 정보의 차단 특성은 [15]에서 검증되어졌다. 따라서 제안하는 밀러 알고리듬의 부채널 정보로는 오류로 인해 변형된 루프 횟수에 대한 정확한 정보를 획득할 수 없기 때문에 카운터 오류 공격의 적용을 미연에 방지할 수 있다.

마지막으로 루프 횟수가 정상적으로 수행되었는지를 확인하는 요소로 밀러 알고리듬 내에 별도의 카운터를 사용한다. 기존에 사용되고 있는 카운터와 정상적인 루프 횟수를 비교하는 방식은 'if' 구문의 형태로 밀러 알고리듬의 연산 동안의 오류 주입 외에 추가적인 오류 주입으로 생략이 가능하여 공격자가 쉽게 오류 페어링 결과값을 획득할 수 있어 위험하다.

제안하는 기법에서는 l의 이진표현 길이 $\lceil \log_2(l) \rceil$를 사전에 레지스터 d에 저장해두고 밀러 루프의 수행 후 수행된 최종 루프 횟수를 카운터와 함께 판별식을 통해 확인한다. 이 때 'if' 구문의 형태가 아닌 산술적인 연

입력 : $P \in G_1$, $Q \in G_2$,
$d = \lceil \log_2(l) \rceil$, $l = (l_{n-1} \cdots l_0)$
출력 : $f_{l,P}(Q) \in G_3$

1. a=b=c=e=0
2. $T \leftarrow P$, $f \leftarrow 1$
3. for $i = n-1$ to 0 do
4. $T \leftarrow [2]T$, $f_a = f_a^2 \times h_1(Q)$
5. $T \leftarrow T+P$, $f_b = f_a^2 \times h_2(Q)$
6. $a = a \oplus l[i]$, $c = c+1$
7. end for
8. $e = b - c + d$
9. return f_e

그림 3. 제안하는 대응 기법이 적용된 밀러 알고리듬
Fig. 3. The Miller algorithm applied to the proposed countermeasure.

산을 사용하여 올바른 페어링 결과가 저장된 레지스터를 지정하도록 하였다. 사용한 판별식은 그림 3의 8번 구문과 같다. 앞서 설명한 것처럼 정상적인 루프 횟수로 밀러 알고리듬이 연산된다면 c와 d의 값이 같으므로 $e = b$를 만족한다. 만약 정상적인 루프 횟수 이전에 오류가 주입되어 변형된다면 b, c, 그리고 d의 값에 의한 판별식 결과가 임의의 값이 되어 공격자는 오류 페어링 결과가 아닌 랜덤한 값을 획득할 수 밖에 없다.

제안하는 기법들이 적용된 안전한 밀러 알고리듬은 그림 3과 같다.

V. 안전성 분석

제안된 대응 기법에서는 double-and-add always 알고리듬 형태와 같이 더블링과 덧셈 연산을 l_i의 비트 값과 상관없이 항상 수행하게 된다. 이때 다음 밀러 루프에서 사용될 밀러 변수의 저장은 현재 루프에 해당하는 l_i의 비트 값과 인덱스 a간의 배타적 논리합(XOR)에 의해 결정된다. 밀러 변수의 저장 위치가 랜덤하게 바뀌므로 만약 8번째 구문이 없다하더라도 공격자는 원하는 밀러 변수가 아니라 잘못된 밀러 변수를 획득하여 공격에 실패할 가능성이 있다.

1. 카운터 오류 공격 분석

제안된 밀러 알고리듬이 실제로 카운터 오류 공격에 안전한지를 확인하기 위해서 임의의 루프 횟수를 가정하여 공격을 적용해본다. 다음의 표 1은 임의의 소수 $l = 179$을 가정하여 안전한 밀러 알고리즘을 수행하였을 때 발생하는 중간 값들을 정리한 것이다. 이때, $l_i = (10110011)$이며, 중간 값이 저장되는 레지스터 배열을 위치를 지정하는 두 변수 a,b는 루프가 진행되는 동안 변하거나 고정된다. 다음 루프 연산에서 사용될 올바른 중간 결과는 a와 b에 따라 고정되지 않고 변함을 확인할 수 있다.

발생할 수 있는 카운터 오류 공격은 공격자의 의도에 따라 l_i, l_{i+1} 쌍이 (0,0), (0,1), (1,0), (1,1)으로 정리된다. 5번째 루프, 6번째 루프까지만 수행하도록 오류가 주입된 경우인 (0,0)을 먼저 분석한다. 5번째 루프의 오류 페어링 결과는 판별식 e가 1-5+8 = 4의 값을 가지므로 중간 결과가 저장되고 있는 f_0나 f_1이 아닌 임의의 값이 된다. 6번째 루프의 오류 페어링 결과 역시 $e = 1-6+8 = 3$의 값을 가지므로 임의의 값이 되어 어떠한 비밀 정보도 찾아낼 수 없다.

두 번째 루프, 세 번째 루프 또는 6번째 루프, 7번째 루프의 경우인 (0,1)을 살펴본다. 이 경우에도 판별식의 값은 $(6 = 0-2+8, 5 = 0-3+8)$ 또는 $(2 = 0-6+8, 1 = 0-7+8)$이 되어 의도한 오류 페어링 결과들을 획득할 수 없다.

표 1. 179(=10110011_2)에서의 안전한 밀러 알고리듬의 중간 연산 결과
Table 1. Intermediate values of secure Miller algorithm in the case of 179(=10110011_2).

l_i	1	0	1	1	0	0	1	1
a	1	1	0	1	1	1	0	1
b	1	0	0	1	0	0	0	1
c	1	2	3	4	5	6	7	8
d	8							
f_0	h_1	$[f_1]^2h_1h_2$	$[f_1]^2h_1h_2$	$[f_0]^2h_1$	$[f_1]^2h_1h_2$	$[f_1]^2h_1h_2$	$[f_1]^2h_1h_2$	$[f_0]^2h_1$
f_1	h_1h_2	$[f_1]^2h_1$	$[f_1]^2h_1$	$[f_0]^2h_1h_2$	$[f_1]^2h_1$	$[f_1]^2h_1$	$[f_1]^2h_1$	$[f_0]^2h_1h_2$
Correct	f_1	f_1	f_0	f_1	f_1	f_1	f_1	f_0

(1,0)인 4번째, 5번째 루프 경우나 (1,1)인 3-4번째 루프 또는 7-8번째 루프의 경우에서도 오류 페어링 결과를 획득할 수 밖에 없다. 특히 7-8번째 루프의 경우에서는 f_0와 f_1의 값을 획득할 가능성이 있다. 그러나 판별식에 의해서 7번째 루프의 정상 결과인 f_1가 아닌 f_0에 저장된 오류 페어링 결과를 얻을 수밖에 없으므로 공격자는 결국 비밀 정보를 획득할 수 없다.

2. 판별식에서의 오류 분석

정상적인 루프 횟수 만큼 수행하였는지를 확인하는 판별식의 경우 'if' 구문에 대한 추가적인 오류 주입 경우와 마찬가지로 산술 연산에 대한 추가적인 오류 주입에 의한 변형 가능성을 분석해야한다. 기존에 소개된 'if' 구문의 생략은 어셈블리어 조합에서 특정 어셈블리어에 오류가 주입되었을 때 발생한다[14]. 따라서 8번째 구문의 어셈블리어 분석을 통해 오류 공격 가능성을 확인한다.

그림 4는 8번 구문의 어셈블리어 조합을 분석한 것이다. 먼저 첫 번째 어셈블리어인 b의 호출에서 오류가 주입된다면 레지스터 eax에는 더미 값만이 남아있게 된다.

따라서 더미 값과 $c-d$ 결과 값 사이의 XOR 연산 결과는 랜덤한 값이 되므로 밀러 알고리듬의 결과 값이 저장되어있는 올바른 위치가 아닌 랜덤한 위치를 지정한다. 그 결과 공격자는 의도한 페어링 결과가 아닌 더미 값을 획득하게 된다. 다음으로 두 번째, 네 번째, 여섯 번째의 carry 제거를 위한 어셈블리어의 생략은 정상적으로 8번째 구문이 수행할 때와 차이가 없다.

두 번째 대상인 c의 호출에서는 앞서와 마찬가지로 ecx 레지스터에 남아있는 더미 값과 d값을 차분하여 b와 XOR 연산을 수행한다. 따라서 그 결과는 랜덤한 위치를 지정하므로 공격자는 더미 값을 획득하게 된다.

세 번째 대상인 d의 경우에도 edx에 남아있는 더미 값에 의해 랜덤 값을 출력하게 된다.

네 번째로 $b-d$를 위한 sub 어셈블리어의 경우 차분이 생략되더라도 앞서 b값이 저장된 eax 레지스터가 다음에 이어지는 덧셈 연산에서 그대로 사용되기 때문에 $e=b$가 아닌 랜덤한 값이 된다. 이 경우에도 공격자는 원하는 페어링 출력 값이 아닌 더미 값만을 획득한다. 이는 마지막인 add 연산에서도 동일하다. 앞 단계에서 수행된 $b-c$ 결과가 eax에 저장되므로 add가 생략된 결과는 정상 값이 아닌 $b-c$가 지정한 값이 된다.

3. 기타 위험성 분석

마지막으로 단순 전력 분석 공격의 방어책인 double-and-add always 알고리듬은 그 성능이 이미 알려져있다[15]. 그러나 더미 연산을 타겟으로 하는 Safe error 공격에는 취약성을 가지고 있다[16~17]. 그러나 Safe error 공격은 더미 연산에 오류를 주입하여 알고리듬 출력의 변형 유무를 체크하고, 변형된 경우 해당 루프의 l_i 값을 찾아내는 방식이다. 따라서 획득한 l_i의 비트 값은 페어링 기법에서는 공개 값으로 공격의 대상이 될 수 없는 무의미한 값이다. 따라서 카운터 오류 공격이외의 위험성은 미진하다.

4. 연산량 분석

부채널 분석에 대한 페어링 기법, 특히 eta 페어링에 대한 다양한 분석 및 대응기법이 소개되었으나 오류 공격에 대해서는 연구가 미흡한 실정이다. 따라서 3장에서 소개한 Page와 Vercauteren 의 블라인딩 기법 2가지와 'if' 구문 기반의 판별식을 쓰는 경우와 비교하여 연산량을 분석하였다. 표 2는 4가지 기법의 추가적인

```
38:          e = b-c+d;
004010BA     mov          eax,dword ptr [ebp-28h]     %b 호출
004010BD     and          eax,0FFh                    %carry 제거
004010C2     mov          ecx,dword ptr [ebp-2Ch]     %c 호출
004010C5     and          ecx,0FFh                    %carry 제거
004010CB     mov          edx,dword ptr [ebp-34h]     %d 호출
004010CE     and          edx,0FFh                    %carry 제거
004010D4     sub          eax,ecx                     %b=b-c
004010D6     add          eax,edx                     %b=b+d
004010D8     mov          byte ptr [ebp-30h],al       %e=eax
```

그림 4. 8번 구문의 어셈블리 프로그램
Fig. 4. Assembly program of 8 number construction.

표 2. 카운터 오류 공격 대응책간의 연산량 비교
Table 2. Additional cost evaluation of countermeasures against counter fault attack.

Method	Additional cost
Blind 1	2 Point Multiplication + 2 field Multiplication
Blind 2	1 pairing computation + 1 field Multiplication + 1 field inversion
if test	1 subtraction
Proposed method	Θ field Multiplication

연산량을 정리한 것이다. Θ는 l_i에서 0의 개수이다.

표에서 확인할 수 있는 바와 같이 블라인딩 2 기법을 제외한 나머지 기법에 비해서 제안하는 밀러 알고리즘은 높은 추가 연산량을 가진다. 그러나 타 기법들은 새로운 카운터 오류 공격 및 추가적인 오류 공격에 취약성을 드러낸 반면 제안한 기법은 이들 공격을 방어할 수 있는 유일한 기법이란 점에서 의의가 있다.

VI. 결 론

본 논문에서는 이동 ad hoc 네트워크의 보안을 위해 사용되는 페어링 연산에서 가장 중요한 밀러 알고리듬을 개선하였다. 가장 강력한 위협인 카운터 오류 공격을 방어하기 위해 연산 중의 중간 값들이 저장되는 경로를 랜덤하게 설정하고 루프 횟수가 정상적인 경우에만 페어링 결과를 확인할 수 있도록 하였다. 카운터 오류 공격을 위해 루프 횟수가 변형된다면 오류 판별식에 의해서 공격자는 페어링 결과가 아닌 더미 값만을 획득하게 된다. 추가적으로 'if' 구문에 의한 덧셈 연산이라는 과정을 제거하여 단순 전력 분석을 통한 루프 횟수의 판별을 어렵게 하였다. 이는 공격자가 원하는 루프 횟수로 변형된 쌍을 쉽게 얻을 수 없게 하여 카운터 오류 공격의 적용을 사전에 차단할 수 있다. 판별식에 대한 오류 주입 가능성을 분석한 결과 기존의 'if' 구문에 의한 오류 감지 기법과 달리 이차적인 오류 주입에 대해서도 안전함을 확인하였다.

REFERENCES

[1] L. Zhou, and Z. J. Haas, "Securing ad hoc networks," *IEEE Network megazine*, vol.13, no.6, pp. 24 - 30, November/December 1999.

[2] A. Khalili, J. Katz, and W.A. Arbaugh, "Toward secure key distribution in truly ad-hoc networks," In IEEE Workshop : Security and Assurance in Ad hoc Networks, pp. 342-346, 2003.

[3] Tae Hyun KIM, Tsuyoshi Takagi, Dong-Guk Han, Ho Won Kim, and Jongin Lim, "Power Analysis Attacks and Countermeasures on ηT Pairing over Binary Fields," *ETRI Journal*, vol.30, no.1, pp. 68-80, Feb. 2009.

[4] N.E. Mrabet, M.L. Flottes, and G. D. Natale, "A practical Differential Power Analysis attack against the Miller algorithm," Research in Microelectronics and Electronics, PRIME 2009. Ph.D., pp.308-311, July 2009.

[5] S. Ghosh, D. Mukhopadhyay, and D. R. Chowdhury, "Fault Attack and Countermeasures on Pairing Based Cryptography," *Internation Journal of Network Security*, vol.12, no.1, pp. 26-33, Jan. 2011.

[6] 배기석, 손교용, 박영호, 문상재, "이동 Ad-Hoc 네트워크 환경에서 페어링 연산의 밀러 알고리듬에 대한 데이터 오류 공격," 전자공학회논문지 제50권 2호, pp. 70-79, 2013년 2월

[7] D. Page and F. Vercauteren, "A Fault Attack on Pairing Based Cryptography," *IEEE Transactions on Computers*, vol.55, no.9, pp. 1075-1080, 2006.

[8] C. Whelan and M. Scott, "The Importance of the Final exponentiation in Pairing when considering Fault Attacks," Proc. of Pairing 2007, pp.225-246, Tokyo, Japan, July 2007.

[9] J. Siverman, The Arithmetic of Elliptic Curves, Springer-Verlag, 1991.

[10] V. Miller. "The Weil Pairing, and its Efficient Calculation," *Journal of Cryptology*, 17, pp. 235-261, 2004.

[11] N. E. Mrabet, "What about Vulnearability to a Fault Attack of the Miller's Algorithm During an Identity Based Protocol?," Advances in Information Security and Assurance, LNCS 5576, pp. 122-134, June 2009.

[12] J.H. Park, G.Y. Sohn and S.J. Moon, "Fault Attack on a Point Blinding Countermeasure of Pairing Algorithms," ETRI Journal, vol. 33, no.6, pp.989-992, 2011.

[13] J. Schmidt and C. Herbst. "A practical fault attack on square and multiply," Proc. of FDTC 2008, pp. 53-58, 2008.

[14] A. Barenghi, G. Bertoni, L. Breveglieri, M. Pellicioli, and G. Pelosi. "Low voltage fault attacks to AES and RSA on general purpose

processors." Cryptology ePrint Archive, Report 2010/130, 2010.
[15] K. Wu, H. Li, T. Chen, F. Yu, "Simple Power Analysis on Elliptic Curve Cryptosystems and Countermeasures: Practical Work," Proc. of ISECS 2009, pp.21-24, May 2009.
[16] S.-M. Yen, S.-J. Kim, S.-G. Lim, and S.J. Moon. "A countermeasure against one physical cryptanalysis may benefit another attack," Proc. of Information Security and Cryptology, LNCS 2288, pp. 414-427, 2002.
[17] S.-M. Yen and C.-S. Laih. "Common-multiplicand multiplication and its application to public-key cryptography," Electronics Letters, vol.29, no.17, pp.1583 - 1584, August 1993.

저 자 소 개

배 기 석(정회원)
2006년 경북대학교 전자전기 컴퓨터학부 학사
2008년 경북대학교 전자전기 컴퓨터학부 석사
2013년 경북대학교 전자전기 컴퓨터학부 박사
2013년～현재 삼성전자
<주관심분야 : 정보보호, 네트워크 보안, 스마트 카드 보안>

박 영 호(정회원)-교신저자
1989년 경북대학교 전자공학과 학사
1991년 경북대학교 전자공학과 석사
1995년 경북대학교 전자공학과 박사
1996년～2008년 상주대학교 전자전기공학부 교수
2003년～2004년 Oregon State Univ. 방문교수
2008년～현재 경북대학교 산업전자공학과 교수
<주관심분야 : 정보보호, 네트워크보안, 모바일 컴퓨팅>