情報保護學會論文誌
第17卷 第1號, 2007. 2

단순전력분석에 안전한 Signed Left-to-Right 리코딩 방법*

한 동 국[1†], 김 성 경[2‡], 김 태 현[2], 김 호 원[1], 임 종 인[2]
[1]한국전자통신연구원, [2]고려대학교 정보경영공학전문대학원

SPA-Resistant Signed Left-to-Right Recoding Method

Dong-Guk Han[1], Sung-kyoung Kim[2], Tae Hyun Kim[2], Ho Won Kim[1], Jongin Lim[2]
[1]Electronics and Telecommunications Research Institute,
[2]Graduate School of Information Management and Security, Korea University

요 약

본 논문에서는 주어진 기수 r 표현법을 SPA에 안전하게 리코딩 하는 방법을 제안한다. 제안된 알고리즘들은 기존의 것들과는 달리, Left-to-Right 리코딩이 가능하도록 구성되어져 있기 때문에 최상위 비트부터 스캔하면서 스칼라 곱셈을 계산하는 알고리즘과 연동이 되어 질 경우, 추가 메모리 없이 쉽게 구현된다는 장점이 있다. 따라서 Left-to-Right 리코딩 기법들은 메모리의 제약을 받는 장비인 스마트 카드, 센서 노드에 적합하다.

ABSTRACT

This paper proposed recoding methods for a radix- r representation of the secret scalar which are resistant to SPA. Unlike existing recoding method, these recoding methods are left-to-right so they can be interleaved with a left-to-right scalar multiplication, removing the need to store both the scalar and its recoding. Hence, these left-to-right methods are suitable for implementing on memory limited devices such as smart cards and sensor nodes

Keywords : *RSA, DSA, 부채널 공격, SPA, left-to-right recoding, fixed pattern*

Ⅰ. 서 론

타원곡선 암호시스템 (elliptic curve crypto -system, ECC) 및 페어링 기반 암호시스템 (pairing based cryptosystems)[3,4]은 작은 키 사이즈를 이용하여 높은 안전성을 이룰 수 있기 때문에 저자원/저전력을 요구하는 환경에 적합하다고 알려져 있다[1]. 타원곡선 암호 시스템과 페어링 기반 암호시스템에서 비밀 값이 사용되는 공통되는 연산은 스칼라 곱셈 알고리즘이다. 그러므로 부채널 공격(Side Chan -nel Attacks)[12,13]에 안전하고 효율적인 스칼라 곱셈 알고리즘을 구성하는 방법과 그것들의 안전성을 분석하는 것은 현재 중요한 연구 주제 중의 하나이다. SPA에 대한 대응 방법 중에서 비밀키를 새로운 표현으로 변환하는 방법이 있다. 계산 능력이나 메모리 같은 자원이 제한된 환경에서 타원곡선 스칼라 곱셈 방법에서는 Right-to-Left 방법보다 Left-to-Right 방법을 이용하여 것이 효율적이다. 하지만 스칼라 곱셈

접수일: 2007년 1월 16일; 채택일: 2007년 2월 6일
* "본 연구의 일부는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" (IITA-2006-(C1090-0603-0025))
† 주저자, christa@etri.re.kr
‡ 교신저자, likesk@cist.korea.ac.kr

은 Left-to-Right 방법으로 수행하고 리코딩은 Right-to-Left 방법으로 수행한다면 리코딩 수행이 끝난 후에 스칼라 곱셈을 수행해야 하므로 리코딩 된 값을 저장하기 위한 추가적인 메모리 공간이 필요하다. 그러므로 Left-to-Right 리코딩 방법의 연구는 현실적으로 중요한 문제이고 현재 활발히 진행되고 있다[11,17,18]. 본 논문에서는 SPA에 안전한 Left-to-Right 리코딩 방법들을 제안한다.

본 논문에서 제안된 리코딩 방법들은 모두 Left-to-Right 기법이므로, 최상위 비트부터 스캔하면서 스칼라 곱셈을 계산하는 알고리즘과 연동이 되어 질 경우, 리코딩 된 표현법을 따로 저장할 추가 n-비트 메모리 ($O(n)$-size RAM) 없이 쉽게 구현된다는 장점이 있다. 여기서 n는 리코딩 되는 정수의 비트 길이이다.

Ⅱ. 스칼라 곱셈 알고리즘과 SPA의 대응 방법들

페어링 기반 암호 시스템의 가장 핵심적인 연산은 Weil 페어링 또는 Tate 페어링 같은 페이링 연산과 타원곡선 스칼라 곱셈 연산이다. [2,7]에서 초특이(supersingular) 타원곡선 위에서 효율성을 높이기 위한 연구가 있었고 Duursma와 Lee가 표수(characteristic)가 작은 소수 r인 초타원곡선에서 Tate페어링을 효율적으로 계산하는 방법을 제안하였다[6]. 이와 같이 페어링 연산의 효율성 때문에 대부분의 페어링 기반 암호시스템은 표수가 작은 소수 r인 타원곡선에서 정의된다. 따라서 페어링 암호시스템에서 스칼라 곱셈 연산은 이진 표현보다 일반적으로 표수 r를 이용하여 스칼라를 r 진법으로 표현하는 것이 보다 효율적일 수 있다. 알고리즘의 고속화를 위하여 사전 계산된 값을 사용하기 위해서는 스칼라 곱셈 알고리즘은 Left-to-Right 방법만이 가능하다.

SPA 공격을 막기 위한 많은 대응책이 제안되어 왔다. 그 중에서도 비밀키를 고정된 패턴으로 만드는 리코딩 방법은 효율성과 안전성 측면에서 가장 많이 사용되고 있는 방법이다. 이와 같은 연구는 대부분 기수가 2인 경우에서 많이 이루어 졌고[16,19], 최근에 [8]에서 기수가 r인 경우에 대한 SPA에 안전하도록 고정된 패턴을 만들어 내는 리코딩 알고리즘이 소개되었다. 이와 같이 부호화 표현을 사용할 수 있는 이유는 타원곡선의 경우 역원 계산이 아주 손쉽게 계산된다는 장점 때문이다.[1] 이와 같은 이유 때문에, SPA에 안전한 고정된 패턴을 가지는 스칼라 곱셈 알고리즘을 설계할 경우 부호화된 표현법이 자주 사용된다.

Radix-r Fixed Pattern Recoding [8] [8] 논문에서 SPA 대응 법으로 제안한 리코딩 방법을 간단하게 소개한다. 주어진 기수 r 표현법 $k=\sum_{j=0}^{n-1} k_j r^j, k_j \in \{0,1,\ldots, r-1\}$를 SPA에 안전한 고정된 패턴을 가지는 새로운 표현 기법이다. 이때 k_0는 0이 아니라고 가정한다. 주어진 기수 r 표현법에서 0이 나타나면 다음과 같은 방법으로 $(\underbrace{0,\ldots,0}_{t},x)_r(1\underbrace{1-r,\ldots,1-r}_{t-1},x-r),\ x\in\{1,2,\ldots,r-1\}$, 0이 없는 부호화된 표현법으로 변환 후, 윈도우-w 슬라이딩 윈도우 방법을 여기에 최하위비트부터 적용하여 다음과 같은 고정된 패턴을 얻어내는 것이다.

$$|\underbrace{0,\ldots,0}_{w-1},y|\underbrace{0,\ldots,0}_{w-1},y|\ldots|\underbrace{0,\ldots,0}_{w-1},y|$$
$$y\in\{\pm1,\ldots,\pm(r^w-1)\}\setminus\{\pm r,\pm 2r,\ldots,\pm(r^w-r)\}$$

이다.

핵심 아이디어는 Okeya-Takagi[17]가 기수가 2인 경우 SPA에 안전한 고정된 패턴을 만들어 내는 리코딩을 기수가 r인 경우로 확장한 것이다. 위의 리코딩 알고리즘들은 Right-to-Left로 새로운 표현의 비밀키를 생성한다. [8]의 리코딩 알고리즘을 윈도우 버전의 알고리즘에 적용하기 위해서는 리코딩이 먼저 선행된 후에 스칼라 곱셈이 실행될 수밖에 없다. 이러한 목적으로 Left-to-Right 리코딩 방법에 대한 연구가 진행되어지고 있다[11,18].

Ⅲ. Radix-r Left-to-Right Recoding with Fixed Pattern

본 절에서는 주어진 기수 r 표현법을 SPA 공격에 안전하도록 고정된 패턴을 갖는 새로운 부호화 표현을 생성하는 알고리즘을 제안하고자 한다. 기존의 알고리즘은 Right-to-Left 리코딩으로 제안되었으나[8], 본 논문에서는 Left-to-Right 스칼라 곱셈 알고리즘과 동시에 계산될 수 있도록 동일한 방향으로 리코딩이 가능한 방법을 제안하고자 한다. 본 논문에서 사용하는 수식 및 기호들을 정의한다.

1) 디짓 셋(digit set)을 $\Lambda_r=\{0,1,\ldots,r-1\}$
$D_r=\{\pm1,\pm2,\ldots,\pm(r-1)\}, \Lambda_{w,r}=\{0,1,\ldots,r^w-1\}$,

1) 예를 들면, 표수가 2인 경우는 $P=(x,y)$면 $-P=(x,x+y)$이고, 표수가 큰 소수인 경우는 $P=(x,y)$면 $-P=(x,-y)$를 만족한다.

$D_{w,r}=\{\pm1,\pm2,...,\pm(r^w-1)\}\setminus\{\pm r,\pm 2r,...,\pm(r^w-r)\}$ 라 정의한다.

2) 기수가 r이고 n 디짓(digit)인 정수 k는 $k=\sum_{i=0}^{n-1}k_i r^i$, $k_i\in\Lambda_r$라고 표현하고, 특별히 기수가 2인 경우를 이진수라고 한다. 이때 k_0는 0이 아니라고 가정한다.

3) 윈도우 크기를 $w(\geq 2)$라고 표기하고 $d=\left\lceil \frac{n}{w} \right\rceil$로 정의한다.

4) k는 d개의 블록 $B^{d-1}\|...\|B^1\|B^0$로 표현 할 수 있고, 이 때 각 블록의 길이는 w이다. k의 비트 길이가 wd보다 작을 경우 나머지는 0으로 채운다. 그리고 B_i^j는 B^j의 i-번째 비트 값으로 정의한다.

Left-to-Right Recoding from Λ_r to D_r

디짓 셋 Λ_r로 표현된 정수 k를 부호화 표현을 이용해 디짓 셋 D_r의 원소로 표현하는 방법을 제안하고자 한다. 물론, 리코딩 하는 방향은 최상위 자리부터 최하위로 (Left-to-Right) 수행되는 것을 목적으로 한다.

기수 r에 의해서 표현된 정수를 0인 비트가 발생하지 않도록 리코딩 하는 방법의 기본 아이디어는 다음과 같다. 여기서 $1<a<r$인 정수에 대하여 $\bar{a}$는 $-a$로 나타낸다.

Conversion 1:

$$\begin{cases}(0,1)_r\Leftrightarrow(1,\overline{r-1})_r & (0,\bar{1})_r\Leftrightarrow(\bar{1},r-1)_r\\ (0,2)_r\Leftrightarrow(1,\overline{r-2})_r & (0,\bar{2})_r\Leftrightarrow(\bar{1},r-2)_r\\ \vdots & \vdots\\ (0,r-1)_r\Leftrightarrow(1,\bar{1})_r & (0,\overline{r-1})_r\Leftrightarrow(\bar{1},1)_r\end{cases}$$

위의 변환 식을 (*Conversion 1*) 이용하면, D_r로 표현되는 Right-to-Left 리코딩 방법은 쉽게 유도됨을 알 수 있다. 제안하는 알고리즘의 설명을 간단하게 하기 위해서 주어진 정수 k로부터 D_r 디짓 셋으로 리코딩된 것을 $k=\sum_{j=0}^{n-1}k'_j r^j$으로 표현하자. 즉, k'_j는 D_r중의 하나의 원소이다. 본 논문에서 제안하는 Left-to-Right 리코딩은 다음의 *Recoding 1*에 의하여 이루어진다. k'의 i-번째 k'_i를 결정하기 위해서는 본 알고리즘에서는 k의 두 자리, 즉 $(i+1)$-번째 k_{i+1}와 i-번째 k_i만 관찰하면 쉽게 결정할 수 있다.

$$\textit{Recoding 1}: k'_i=\begin{cases}k_i & \text{if } k_{i+1}\cdot k_i\neq 0\\ 1 & \text{if } k_{i+1}\neq 0 \text{ and } k_i=0\\ k_i-r & \text{if } k_{i+1}=0 \text{ and } k_i\neq 0\\ 1-r & \text{if } k_{i+1}=0\cdot k_i=0\end{cases}$$

위의 식은 리코딩이 최상위 비트부터 최하위 비트로 될 수 있음을 보여준다. *Recoding 1*에 의해서 리코딩 된 결과 k'가 기존의 값 k와 같다는 것을 (즉, $k'=k$) 증명한다.

proof. 먼저 k가 다음과 같이 m개의 블록으로 표현된다고 가정한다.

$$k=\sum_{i=0}^{n-1}k_i r^i=A^{m-1}\|A^{m-2}\|...\|A^1\|A^0,$$

$$\text{where } A^i=\begin{cases}\textit{block of nonzero digits}, \text{ or}\\ (\underbrace{0,0,...,0}_{t>0},x), x\in D_r\setminus\{0\}\end{cases}$$

앞에서 기술된 *Conversion 1*을 이용하여 Right-to-Left 리코딩을 할 경우, 위에서 나누어진 블록들 $\{A^i\}$ 사이에는 캐리가 발생하지 않음을 알 수 있다. 만약 A^i가 0이 포함되어 있지 않은 첫 번째 경우라면, *Conversion 1*에 의하여 리코딩이 일어나지 않는다. 하지만, $A^i=(\underbrace{0,0,...,0}_{t>0},x)$의 경우이면, *Conversion 1*에 의하여 $(\underbrace{1,\overline{r-1},...,\overline{r-1}}_{t},\overline{r-x})$로 변환된다. $k_n=1$이라는 가정 하에 A^i을 *Recoding 1*로 변환할 경우, A^i가 0이 포함되어 있지 않은 첫 번째 경우에서는 리코딩 결과가 A^i와 같은 결과가 되며, $A^i=(0,0,...,0,x)$의 경우에도 $A_0^{i+1}\neq 0$이기 때문에 $(1,\overline{r-1},...,\overline{r-1},\overline{r-x})$가됨을 알 수 있다. 여기서, A_0^{i+1}는 A^{i+1} 블록의 최하위 디짓(digit)을 의미한다. 따라서, *Recoding 1*에 의하여 k로부터 변환된 k'은 *Conversion 1*에 의하여 변화된 결과와 정확하게 같음을 알 수 있다. 결론적으로, $k=k'$ 임이 증명된 것이다.

따라서 주어진 정수 k를 제안한 *Recoding 1*을 이용하여 변환 할 경우 원래의 값과 다르지 않고 모든 digit 값이 0이 아닌 값으로 표현되기 때문에 SPA 공격에 안전한 표현 방법으로 변환 할 수 있다. 또한, 위의 방법과 같은 부호화 표현을 이용하면 원하는 임의의 비트 길이를 가지는 부호화된 표현법을 생성 할 수 있다.

3.1 Extension to Higher width : from $\Lambda_{w,r}$ to $D_{w,r}$

앞에서 제안한 리코딩 방법을 이용하여 윈도우 크기가 $w(\geq 2)$인 경우로 확장하고자 한다. 앞에서 정의된 k의 w-윈도우 표현법은 $k=B^{d-1}\|...\|B^0$이다. 이때 각 블록의 정수 값은 $\Lambda_{w,r}$의 원소 중 하나이다. 디짓 셋 $\Lambda_{w,r}$로 표현된 k를 0이 포함하지 않는 $D_{w,r}$의 원소들로 표

현하는 방법은 *Recoding 1*을 각 블록 B^i에 적용하면 된다. $k=E^{d-1}\|...\|E^0$를 $D_{w,r}$의 원소들로 표현된 k의 새로운 표현법이라고 하자. 즉, Ξ^i는 비트 길이가 w 이면서, 정수 값은 $D_{w,r}$의 원소이다. $k_{wd}=1$이라 두자.

E^i를 $B_0^{i+1}\|B^i$의 *Recoding 1*에 의한 변환된 결과 값이라고 정의한다면, E_0^i의 정수 값은 $D_{w,r}$의 원소 중에 하나가 됨을 쉽게 알 수 있다. 왜냐하면, Recoding 1에 의하여 E_0^i가 될 수 있는 가능한 디짓은 D_r이기 때문에, E^i의 정수 값은 $D_{w,r}$의 원소이다. E^i를 다음과 같이 정의 한다.

$E^i := Recoding1(B_0^{i+1}\|B^i)$.

좀 더 자세하게 기술하면, $0\le j<w-1$에 대해서, (E_j^i는 E^i의 j-번째 비트 값으로 정의한다.)

$$E_j^i := \begin{cases} Recoding1(B_0^{i+1}\|B_{w-1}^i) & \text{if } j=w-1 \\ Recoding1(B_{j+1}^{i+1}\|B_j^i) & otherwise \end{cases}$$

Ⅳ. Special Case : Radix-2

본 절에서는 Ⅲ절에서 제안한 기수가 r인 경우의 Left-to-Right 리코딩 방법이 기수가 2인 경우에는 좀 더 간단하게 리코딩이 됨을 보인다. 그리고 윈도우-w 방법과 고정 기수 comb 방법에 적용한 고정된 패턴을 가지는 Left-to-Right 리코딩 방법들을 제안한다.

Ⅲ절에서 살펴본 바와 같이, 기수가 임의의 r인 경우에는 i-번째 값을 결정하기 위하여 $(i+1)$-번째와 i-번째의 값을 확인해야만 했지만, 기수가 $r=2$일 경우에서는 $(i+1)$-번째 비트의 결과에만 의존하여 i-번째 비트가 바로 결정됨을 *Recoding 1*로 부터 알 수 있다. 좀 더 자세하게 설명하면, *Recoding 1*에서 첫 번째와 두 번째 경우는 항상 리코딩 된 결과가 1이고, 나머지 2개의 경우는 -1이다. 이때, 처음 두 개의 $(i+1)$-번째 값은 1이고, 나머지 두 개의 $(i+1)$-번째 값은 0임을 알 수 있다. 즉, 기수 2에서의 Left-to-Right 리코딩은 다음과 같이 이루어진다.

$$Recoding\,2: k'_i = \begin{cases} 1 & \text{if } k_{i+1}=1 \\ -1 & \text{if } k_{i+1}=0 \end{cases}$$

또한 *Recoding 2*의 윈도우 버전을 확인하면 다음과 같다. 여기서, $0\le j\le w-1$에 대해서,

$$E_j^i := \begin{cases} Recoding2(B_0^{i+1}) & \text{if } j=w-1 \\ Recoding2(B_{j+1}^i) & otherwise \end{cases}$$

를 만족 한다.

Algorithm1. SPA-resistant Comb algorithm based on Recoding 2

INPUT A point P, an integer $k=(k_{n-1}\cdots k_0)_2$, and a window width $w\ge 2$;
OUTPUT $Q=kP$;

1. Pre-computation Stage
   Compute $[b_{w-1},...,b_0]P$ for all $(b_{w-1},...,b_0)$, where $b\in\{1,-1\}$;
2. Recoding+Evaluation Stage
   2.1. $Q=O$;
   2.2. if $k \bmod 2=0$ then $k\leftarrow k+1$
   else $k\leftarrow k+2$
   2.3. let $k_{wd}=1$ and $\{k_{wd-1},...,k_n\}$be all 0
   2.4. for $i=d-1$ to 0 do:
   2.4.1. for $j=w-1$ to 0 do:
   2.4.1.1. k'_j=Recoding $2(k_{jd+i+1})$;
   2.4.2. set $\Gamma_i\equiv[K_i^{w-1},\cdots,K_i^1,K_i^0]$;
   2.4.3. $Q=2Q$;
   2.4.4. $Q=Q+\Gamma_i P$;
   2.4.5. if k is even return $Q-P$
   else return $Q-2P$

4.1 Extension to Comb Method

본 소절에서는 Lim-Lee[14]가 제안한 고정 기수 comb 방법에 대해서 기술하고, 부채널 분석에 대한 안전성과 기존에 나온 대응 법을 살펴본다. Lim-Lee가 제안한 고정 기수 comb 방법은 사전 계산 단계 , 리코딩 단계 그리고 스칼라 곱셈 단계로 구성되어진다.2)

Notations

- 정수 k는 $k=\sum_{i=0}^{n-1}k_i2^i$와 같이 정의되고 이때 k_i는 0과 1이다. 그리고 윈도우 크기인 w는 2 보다 크고, $d=\lceil n/w\rceil$ 이다.
- 정수 k가 $wd-1$이 되게 0을 채우고, $k=K^{w-1}\|\cdots\|K^1\|K^0$라고 한다. 이 때, K^j의 길이는 d이고 k_i^j의 i번째 비트는 k_i^j라고 표현한다.
- $\Gamma_i\equiv[K_i^{w-1},\cdots,K_i^1,K_i^0]$.
- $(K_{w-1},K_{w-2},\cdots,K_1,K_0)\in Z_2^w$ 일 때,

$[K_{w-1},K_{w-2},\cdots,K_0]\equiv K_{w-1}2^{(w-1)d}+\cdots+K_12^d+K_0$
이다.

2) 고정 기수 comb 방법은 리코딩 단계와 스칼라 곱셈을 같이 수행할 수도 있다.

comb 알고리즘에서도 $\Gamma_i \equiv [K_i^{w-1}, \cdots, K_i^1, K_i^0]$이 0일 경우와 그렇지 않은 경우 타원곡선 덧셈과 두 배 연산이 다르게 수행되기 때문에 SPA 공격에 취약함을 알 수 있다. 최근에, Hedabou등[9](이후, HPB)이 SPA에 안전하도록 고정된 패턴을 가지는 Comb 방법을 제안하였다. HPB가 제안한 방법도 이진수로 표현된 k를 부호화 표현을 이용하여 0이 포함되어 있지 않는 고정된 패턴을 만들어 내는 방법이다. HPB가 제안하는 방법에서 리코딩 과정은 비트가 최하위 비트부터 리코딩 되어져 간다는 것을 확인할 수 있다. 즉, Right-to-Left 리코딩인 것이다. 앞에서 살펴본 것과 같이, Left-to-Right 스칼라 곱 알고리즘을 활용하기 위해서는 리코딩 값을 저장해 놓고 스칼라 곱셈 알고리즘을 계산하는 방법밖에는 없다.

본 논문에서 제안한 *Recoding 2*를 활용한다면, 쉽게 Left-to-Right 리코딩이 가능하다. Recoding 단계와 Evaluation 단계를 하나로 통합된 방식으로 구성할 수 있다는 것이다.

아래 표는 기존의 고정 기수 comb 방법과 제안한 리코딩을 적용한 comb 방법의 연산 량을 비교한 것이다. 타원 곡선 덧셈 연산을 A, 두 배 연산을 D로 표현한다.

표에서 알 수 있듯이 기존의 고정 기수 radix-2 comb 방법에서는 사전 계산량이 2^w-1였지만 제안한 리코딩 알고리즘을 적용하였을 경우 2^{w-1}만 계산하면 된다. 따라서 Pre-computation Stage의 메모리 공간은 기존의 고정 기수 comb 방법의 절반만 필요하게 되나 총 연산량은 윈도우 크기가 4일 경우 0.6% 밖에 늘지 않는다. 그리고 제안하는 리코딩 방법은 Left-to-right 방법이기 때문에 리코딩 연산과 스칼라 곱셈 연산 을 동시에 수행할 수 있으므로 리코딩 결과 값을 저장하기 위한 추가적인 공간이 필요하지 않다.

Method	Step	Memory	Computational Cost
Fixed-base Comb[14]	Pre-computation Stage	2^w-1 points	$(w-1)dD + (2^w-w)A$
	Recoding +Evaluation Stage	w bits for each Γ_i	$(d-1)D + \frac{2^{w-1}}{2^w}(d-1)A$
Algorithm1	Pre-computation Stage	2^{w-1} points	$(w-1)dD + (2^w-w)A$
	Recoding +Evaluation Stage	w bits for each Γ_i	$(d-1)D + (d-1)A$

V. 결론

타원곡선 암호시스템과 페어링 기반 암호시스템에서 스칼라 곱셈 연산은 가장 많은 자원을 소모하고 안전성과 밀접한 관련이 있는 연산이다. 본 논문에서 제안한 Left-to-Right 리코딩 방법을 이용하여

스칼라 곱셈연산을 수행하면 SPA 공격에 안전할 뿐만 아니라 스마트 카드, 모바일 폰, PDA, 센서노드 등과 같은 메모리와 계산 능력이 제한된 환경에서 효율적으로 구현될 수 있다. 타원곡선 암호시스템과 페어링 기반의 암호시스템들은 모두 타원곡선에서 스칼라 곱셈연산을 계산해야 하지만 두 암호시스템에서 사용되는 타원곡선의 종류가 다르기 때문에 두 암호시스템에서 사용되는 스칼라 곱셈 연산 또한 약간 다르게 구현된다. 따라서 본 논문에서는 페어링 기반의 암호시스템에 적합한 기수 r Left-to-Right 리코딩 알고리즘을 제안한다. 또한 윈도우 기반의 리코딩 알고리즘도 제안함으로써 메모리 공간이 여유가 있는 환경에서는 고속 연산을 위하여 사용될 수 있다.

참고문헌

[1] M. Aydos, T. Yank, and C.K. Koc, "High-speed implementation of an ECC-based wireless authentication protocol on an ARM microprocessor," IEE Proceedings Communications, vol. 148, Issue 5, pp. 273-279, Oct., 2001.

[2] P. Barreto, H. Kim, B. Lynn, and M. Scott, "Efficient Algorithms for Pairing-Based Cryptosystems," CRYPTO 2002 ,LNCS 2442, pp. 354-368, 2002.

[3] D.Boneh and M.Franklin, "Identity-Based Encryption from the weil Pairing," Crypto'01 , LNCS 2139, pp. 213-229.

[4] D. Boneh, B. Lynn, and H. Shacham, "Short Signatures from the Weil Pairing," ASIACRYPT 2001, LNCS 2248, pp. 514-532, 2001.

[5] J.S. Coron, "Resistance against Differential Power Analysis for Elliptic Curve Cryptos-

ystems," CHES 1999, LNCS 1717, pp. 292-302, 1999.

[6] I. Duursma and H -S .Lee, "Tate Pairing Implementation for Hyperelliptic Curves y^2= x^p-x+d," ASIACRYPT 2003, LNCS 2894, pp. 111-123, 2003.

[7] S. Galbraith, K. Harrison, and D. Soldera, "Implementing the Tate pairing," ANTS V, LNCS 2369, pp. 324-337, Springer-Verlag, 2002.

[8] D.-G. Han and T. Takagi, "Some Analysis of Radix-r Representations," Cryptography ePrint Archive, Report 2005/402, 2005. http://eprint.iacr.org/2·005/402.

[9] M. Hedabou, P. Pinel, and L. B'eb'eteau, "Countermeasures for Preventing Comb Method Against SCA Attacks," ISPEC'05, LNCS 3439, pp. 85-96, Springer-Verlag, 2005.

[10] M. Joye and C. Tymen, "Protections against differential analysis for elliptic curve cryptography: An algebraic approach," CHES 2001, LNCS 2162, pp. 377-390, 2001.

[11] M. Joye and S. Yen, "Optimal Left-to-Right Binary Signed-Digit Recoding," IEEE Trans. Computers, vol. 49, pp. 740-748, July, 2000.

[12] P. Kocher, "Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems," Advances in Cryptology-CRYPTO '96, LNCS 1109, pp. 104-113, 1996.

[13] P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis," Advances in Cryptology-CRYPTO'99, LNCS 1666, pp.388-397, 1999.

[14] C. Lim and P. Lee, "More Flexible Expo- nentiation with Precomputation," CRYPTO'94, LNCS 839, pp. 95-107, Springer-Verlag, 1994.

[15] V.S. Miller, "Use of elliptic curves in cryptography," In Advances in Cryptology-CRYPTO'85 ,LNCS 218, pp. 417-426, 1986.

[16] B. Moller, "Securing Elliptic Curve Point Multiplication against Side-Channel Attacks," Information Security-ISC"01 , LNCS 2200, pp. 324-334, 2001.

[17] K. Okeya and T. Takagi, "The Width-w NAF Method Provides Small Memory and Fast Elliptic Scalar Multiplications Secure against Side Channel Attacks," CT-RSA 2003, LNCS 2612, pp.328-342, 2003.

[18] X. Ruan and R. Katti, "Left-to-Right Optimal Signed-Binary Representation of a Pair of Integers," IEEE Trans. Computers, vol. 54, pp. 124-131, July, 2005.

[19] N. Smart and J. Westwood, "Point Multiplication on Ordinary Elliptic Curves over Fields of Characteristic Three," Applicable Algebra in Engineering, Communication and Computing, Vol.13, No.6, pp. 485-497, 2003.