情報保護學會論文誌
第17卷 第1號, 2007. 2

SPA에 안전한 Unsigned Left-to-Right 리코딩 방법*

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

SPA-Resistant Unsigned Left-to-Right Recoding Method*

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

요 약

Vuillaume-Okeya는 스칼라 모듈러 지수승 연산에서 SPA 공격에 안전한 리코딩 방법을 제안하였다. 제안한 방법은 역원 연산의 비용이 큰 RSA 또는 DSA 같은 시스템에서 효율적으로 구성 될 수 있게 비밀키의 표현을 0을 포함하지 않는 양의 디짓 셋 $\{1,2,\cdots,2^w-1\}$을 이용해 리코딩 하였다. 제안된 방법은 비밀키의 최하위 비트부터 스캔하면서 리코딩하는 Right-to-Left 기법이다. 따라서 지수승 연산 전에 리코딩이 연산되고 그 결과를 저장하는 추가적인 공간이 필요하게 된다.

본 논문은 Left-to-Right 방향으로 수행하는 새로운 리코딩 방법들을 제안한다. 본 논문에서 제안하는 방법은 (1) 일반적으로 윈도우 크기가 w인 SPA에 안전한 부호가 없는 Left-to-Right 리코딩 방법이고 (2) 윈도우 크기 $w=1$ (즉, $\{1,2\}$로 구성된 부호가 없는 이진 표현)인 경우는 일반적인 윈도우 크기 w에 제안된 기법보다 훨씬 간단하게 변형할 수 있다. 또한 (3) 제안된 리코딩 방법은 부호가 없는 comb 방법에도 적용하여 SPA에 안전하게 수행할 수 있고, (4) 기수가 r인 경우에도 확장하여 SPA에 안전하게 대응할 수 있다. 본 논문에서 제안한 Left-to-Right 리코딩 기법들은 메모리의 제약을 받는 장비인 스마트 카드, 센서 노드에 적합하다.

ABSTRACT

Vuillaume-Okeya presented unsigned recoding methods for protecting modular exponentiations against side channel attacks, which are suitable for tamper-resistant implementations of RSA or DSA which does not benefit from cheap inversions. The proposed method was using a signed representation with digits set $\{1,2,\cdots,2^w-1\}$, where 0 is absent. This recoding method was designed to be computed only from the right-to-left, i.e., it is necessary to finish the recoding and to store the recoded string before starting the left-to-right evaluation stage.

This paper describes new recoding methods for producing SPA-resistant unsigned representations which are scanned from left to right contrary to the previous ones. Our contributions are as follows; (1) SPA-resistant unsigned left-to-right recoding with general width-w, (2) special case when $w=1$, i.e., unsigned binary representation using the digit set $\{1,2\}$, (3) SPA-resistant unsigned left-to-right Comb recoding, (4) extension to unsigned radix-r left-to-right recoding secure against SPA. Hence, these left-to-right methods are suitable for implementing on memory limited devices such as smartcards and sensor nodes

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

I. 서 론

대표적인 공개키 암호 시스템과 응용 프로토콜은 RSA와 전자서명 DSA이다. 이 두 시스템에서 비밀키가 사용되어서 이루어지는 공통적인 연산은 지수승 연산으로 키 k와 기수 g에 대해 g^k를 계산하는 것이다. 계산 능력이나 메모리 같은 자원이 제한된 환경(예를 들면, 스마트 카드, 모바일 폰, 센서 노드)에서 지수승 연산을 구현하기 위해서 효율성을 증대시키기 위한 많은 방법들이 제안되었다. 그러나 이러한 제한된 환경에서 암호 알고리즘을 구동하는 장비에서 발생되는 다양한 부가정보를 활용해 내부에 숨겨진 비밀 정보를 알아내는 부채널 공격(Side Channel Attacks)이 가능하다[13,14]. 부채널 공격 중 전력분석 공격은 크게 단순전력분석 (SPA)과 차분전력분석(DPA)으로 구분되며, SPA는 전력소모량의 통계적인 분석 없이 전력소모량으로부터 직접적으로 비밀 정보를 찾아낼 수 있기 때문에 간단하면서도 쉬운 공격이다. DPA는 여러 개의 전력 소모량의 표본으로부터 비밀 정보와 전력 소모량의 상관관계를 찾기 때문에 SPA에 안전하게 만들어진 암호 시스템의 공격에도 사용할 수 있다. 비밀키가 사용되는 환경에 따라서는, 예를 들어 DSA의 경우, SPA에만 안전하게 대응방법이 고려되어지면 된다. 그리고 DPA까지 고려해야하는 환경에서 만약 SPA에 대한 대응법을 고려하지 않는다면 DPA 대응법은 큰 의미를 갖지 못하게 된다. 따라서 비밀키가 적용되는 어떤 환경에 대해서도 SPA에 대한 대응법은 반드시 고려되어져야 한다.

최근에 SPA의 대응 방법으로 Vuillaume-Okeya이 리코딩 방법을 제안하였다. 이 방법은 역원 연산의 비용이 큰 RSA와 DSA 같은 시스템에서 효율적으로 구성될 수 있게 부호가 없는 스칼라로 변형하는 리코딩 방법이다[28]. 이 방법은 Möller가 제안한 리코딩 방법[19]을 부호가 없는 방법으로 확장하여 윈도우 크기가 w일 경우 부호가 없는 디짓 셋(digit set)인 $\{1, 2, \cdots, 2^w\}$를 이용해 비밀키를 표현하는 것이다. 이 방법은 리코딩 된 값의 비트 길이가 고정적이지 않아서 부가적인 정보가 유출되는 등 여러 가지 단점을 가지고 있지만 가장 중요한 단점은 Right-to-Left 방향으로 리코딩이 된다는 것이다.

1.1 논문의 동기

SPA의 대응 방법 중 스칼라를 새로운 표현 방법으로 리코딩 하는 방법은 스칼라를 최상위 비트(Left-to-Right)부터 스캔하면서 계산하는 방법도 있지만, 최하위 비트(Right-to-Left)부터 스캔하면서 계산하는 방법도 가능하다. 그러나 메모리 같은 자원이 제한된 환경에서는 Right-to-Left 리코딩 방법보다 Left-to-Right 리코딩 방법을 선호하게 된다[20] : 1) 알고리즘의 고속화를 위해 사전계산을 사용하는 윈도우 방법은 Left-to-Right 지수승 계산 방법만이 가능하다. 2) Left-to-Right 지수승 연산이 Right-to-Left 지수승 연산보다 레지스터 개수가 적다는 장점이 있다. (예를 들어 Right-to-Left 방법에서는 g^{2^i}를 저장하기 위한 $Q[0]$가 추가적으로 필요하다.)

그러므로 Right-to-Left 리코딩 방법을 사용 할 경우 Left-to-Right 방향 지수승 연산되는 방법에 적용하기 위해서는 리코딩이 먼저 선행된 후에 스칼라 곱셈이 실행될 수밖에 없다. 바꾸어 말하면, 리코딩 된 값을 저장할 추가 공간이 (즉, O(n)-size RAM) 필요하다는 것이다. 여기서 n은 키의 비트 길이이다. 하지만, 리코딩이 Left-to-Right 방향으로 이루어진다면, 리코딩 된 결과값을 따로 저장하지 않고도 리코딩 알고리즘과 지수승 알고리즘이 하나로 통합될 수 있어서 효율적인 지수승 알고리즘을 얻게 된다.

1.2. 논문의 기여

본 논문에서는 (1) Vuillaume-Okeya의 Right-to-Left 리코딩[28]을 Left-to-Right 방법으로 변환하고, (2) $\{1,2\}$를 이용한 이진수 Left-to-Right 리코딩 방법을 소개한다. 즉, 윈도우 크기가 1인 경우이다. 이 경우에는 일반 윈도우 w에 제안된 기법보다 훨씬 간단하게 리코딩 방법을 수행할 수 있다. 또한 (3) 제안하는 부호가 없는 Left-to-Right 리코딩 방법을 고정 기수 comb 방법[17]에 적용 할 수 있는 리코딩 방법을 제안한다. (4) 마지막으로 이진수 표현 방법에서 F_{r^n}의 유한 체 위에서의 기수 r로 확장한 경우로도 확장하여 SPA에 안전

접수일: 2006년 9월 5일; 채택일: 2006년 12월 6일
* "본 연구의 일부는 정보통신부 및 정보통신연구진흥원의 대학 IT연구센터 지원사업의 연구결과로 수행되었음" (IITA-2006-(C1090-0603-0025))
† 주저자, likesk@cist.korea.ac.kr
‡ 교신저자, christa@etri.re.kr

한 Left-to-Right 리코딩 방법을 제안하였다.

본 논문의 구성은 다음과 같다. 2장에서는 부채널 공격과 그 대응 방법을 소개한다. 3장에서 본 논문에서 제안하는 부호를 사용하지 않는 Left-to-Right 리코딩 방법들을 제안하고, 4장에서 3장에서 제안한 방법을 기수가 r인 경우로 확장한다.

Ⅱ. 부채널 공격 및 대응 방법

암호 시스템은 이론적 안전성과 별도로 전력 소모량과 알고리즘의 수행시간과 같은 부가 정보는 스마트 카드, 모바일 폰, PDA등의 내장된 하드웨어에서 암호 알고리즘의 구현에 대한 실질적인 위협이 되고 있다. 이러한 공격을 부채널 공격(Side channel attack)이라고 한다.

2.1 부채널 공격의 방법

부채널 공격은 암호 시스템이 내장된 장치에서 암호 알고리즘이 실행될 때 발생하는 부가 정보를 이용하여 장치 내에 내장된 비밀 정보(예, 시간, 전력 등)를 알아냄으로써 암호시스템을 공격한다[13,14]. 이 중에서 전력 분석 공격(power analysis)공격은 암호 장치가 연산을 수행하는 동안 소모하는 전력의 추이를 관찰함으로써 비밀키를 찾아낸다. 예를 들어, 지수승 연산 방법 중 Left-to-Right Binary method(또는 Right-to-Left Binary method)는 부채널 공격으로부터 비밀정보를 알아낼 수 있다. 일반적으로 곱셈 연산과 제곱 연산을 계산할 때 소모되는 전력의 차이가 발생하고 이런 정보를 이용하여 지수승에서 사용된 키를 찾는 것이 가능하다.

전력 분석 공격은 단순전력분석 (simple power analysis, SPA) 공격과 차분전력분석 (differential power analysis, DPA) 공격으로 크게 두 가지로 나뉠 수 있다. SPA는 하나의 전력 소모량으로부터 비밀 값의 정보를 얻는 방식이다. 하나의 표본을 관찰함으로써 어느 부분에 어떤 연산이 수행 중인지를 알 수 있게 된다. 즉, 한 번의 전력 소모량을 측정함으로도 비밀 값의 정보 전부 또는 일부를 알아 낼 수가 있다. DPA는 여러 개의 전력 소모량으로부터 비밀 정보를 얻어내는 방법이다. 여러 개의 표본으로부터 비밀 정보와 전력 소모량의 상관관계를 찾기 때문에 SPA에 안전하게 만들어진 암호 시스템의 공격에도 사용할 수 있다.

Left-to-Right binary method

INPUT $k=\sum_{i=0}^{n-1} k_i 2^i$, base g

OUTPUT g^k

1. $Q[0] \leftarrow g$.
2. For $i=n-2$ to 0 by -1 do :
   2.1. $Q[0]=(Q[0])^2$.
   2.2. If $k_i=1$ then $Q[0]=Q[0] \cdot g$
3. Return $Q[0]$.

Right-to-Left binary method

INPUT $k=\sum_{i=0}^{n-1} k_i 2^i$, base g

OUTPUT g^k

1. $Q[0] \leftarrow g$, $Q[1] \leftarrow 1$.
2. For $i=0$ to $n-1$ by 1 do :
   2.1. If $k_i=1$ then $Q[1] \leftarrow Q[0] \cdot Q[1]$.
   2.2. $Q[0] \leftarrow (Q[0])^2$
3. Return $Q[1]$.

2.2 대응 방법

부채널 공격 방법에 대응되는 대응 방법으로 다양한 방법이 제안되었다. 구체적으로 RSA와 ECC 암호에서 비밀 값을 다른 표현으로 바꾸는 방법을 주로 사용한다.

DPA에 안전한 대응 방법은 어렵지 않게 해결 할 수 있다. RSA의 경우 SPA에 안전한 대응 방법을 DPA에 안전한 대응 방법으로 변환 시키는 방법은 비밀 값 k에 대해 $k+r\phi(N)$로 값을 숨긴 후 $g^k=g^{k+r\phi(N)} \bmod N$ 과 같이 지수승 연산을 함으로 각각의 전력 소모량의 표본마다 비밀 값이 달라지기 때문에 DPA 공격을 쉽게 막을 수 있다. 그리고 페어링(pairing) 기반의 암호 시스템이거나 서명 스킴인 DSA나 EC-DSA의 경우는 랜덤 한 값을 이용하기 때문에 SPA의 대응 법만을 요구 할 때도 있다. 그러므로 SPA 공격에 안전한 지수승 연산을 개발하는 것이 먼저 선행되어야 한다.

SPA 공격을 막기 위한 많은 대응책이 제안되어 왔다. SPA 대응방법의 종류는 부가(dummy) 명령어를 삽입하는 방법과 지수승 연산에서 비밀키에 관계없이 부

호를 사용하여 리코딩 하는 방법을 이용하여 항상 고정된 패턴의 연산을 수행하도록 하는 방법들이 있다[19,21,16,26]. 그러나 이와 같이 부호화 표현을 사용하는 방법은 타원곡선의 경우 역원 계산이 아주 손쉽게 계산된다는 장점[1] 때문에 타원곡선에서만 효율적으로 사용될 수 있다. 다시 말해 RSA 또는 DSA와 같이 역원 계산의 비용이 큰 알고리즘의 경우에는 사용하기에 적합하지 않다. 그러므로 RSA 또는 DSA에 적합할 수 있게 부호를 사용하지 않는 리코딩 방법(예, Yen et. al.'s unsigned Ha-Moon 리코딩 방법)을 찾는 것이 필요하다[29].

Vuillaume-Okeya의 대응 방법

최근 Vuillaume-Okeya[28]은 RSA와 DSA에서 적합하도록 SPA에 안전한 부호가 없는 리코딩 방법을 제안하였다. 윈도우 크기가 w일 경우 제안 된 리코딩 방법은 다음과 같은 고정 형식으로 스칼라를 생성한다 : $|\underbrace{0\cdots0}_{w-1}y|\underbrace{0\cdots0}_{w-1}y|\cdots|\underbrace{0\cdots0}_{w-1}y|$

여기서, y는 사전 계산이 되어 있는 값이고 양의 값을 가진다. 이 방법은 Möller가 제안한 타원곡선에서 SPA에 안전하게 구성되는 부호화 리코딩 방법[19]을 부호를 가지지 않는 방법으로 확장한 것이다. 즉, 부호를 사용하여 $\{-2^w, 1, 2, \cdots, 2^{w-1}\}$로 표현되었던 디짓 셋(digit set)을 부호가 없는 디짓 셋인 $\{1,2,\cdots,2^w\}$로 변형한 것이다. 이러한 부호를 없애기 위해서 양의 캐리(carry) 대신에 음의 캐리를 고려한다:

1. 디짓 값이 0일 경우 2^w로 값을 변경한 후, Right-to-Left 방향으로 다음 윈도우 값에 해당되는 캐리에 -1을 더한다.
2. 디짓 값이 -1일 경우 2^w-1로 값을 변경한 후, 캐리에 -1을 더하게 된다.
3. 나머지 경우에는 그대로 값을 유지한다.

Möller가 제안한 리코딩 방법에서 n비트의 값은 최대 $n+1$ 비트의 값이 된다. 그러나 위에 방법처럼 리코딩을 하게 되면 음의 캐리의 전파가 막연히 수행될 수 있다. 이러한 리코딩 된 스칼라의 길이가 비밀 값에 의존하게 되므로 비트 길이 정보를 누출하지 않기 위해 DPA의 대응 방법을 사용하게 된다. RSA의 경우 비밀키 k 대신에 $k+r\phi(N)$을 사용한다. 이 때 ϕ는 Euler Phi 함수이고 r은 랜덤 한 값, 그리고 N은 RSA에서 사용하는 모듈러 값인 두 소수의 곱이다. $\phi(N)$을 k에 더하는 과정을 반복하면 $k<\phi(N)<2^N$이 성립하므로 n비트의 k의 경우 $k_{n+1}=1$, $k_n=0$을 항상 만족한다. 그리고 $g^{k+r\phi(N)}=g^k \bmod N$ 이 성립하기 때문에 스칼라 길이가 확장된 스칼라를 이용하여 계산하더라도 기존의 값의 결과가 같게 된다. 또한, 스칼라의 앞 두 비트가 $(10)_2$이면 리코딩 과정에서 리코딩 된 결과 값이 무한정 늘어나는 것을 막을 수 있을 뿐 아니라 항상 일정한 길이가 나오게 된다. 즉, 기존의 스칼라의 비트 길이가 n이면 리코딩 된 스칼라의 비트 길이는 최대 $n+1$이다. 만약 k와 N이 고정된 값이면 키를 생성하는 단계에서 한 번만 수행하면 된다. Vuillaume-Okeya의 리코딩 방법을 알고리즘으로 나타내면 Recoding 1과 같다.

Recoding 1. Unsigned Right-to-Left recoding

INPUT $(n+2)$-bit exponent $k'=(10k_{n-1}\cdots k_0)_2$, with width w;
OUTPUT Recoded exponent $(u_n\cdots u_0)$;

1. $i\leftarrow0$; $\gamma\leftarrow0$;.
2. while $i\le n-w$ do :
   2.1. $u_i\leftarrow(k_{i+w-1}\cdots k_i)_2-\gamma$;
   2.2. if $u_i\le0$ then $u_i\leftarrow u_i+2^w$; $\gamma\leftarrow1$; else $\gamma\leftarrow0$;
   2.3. $u_i\leftarrow0,\cdots,u_{i+w-1}\leftarrow0$; $i\leftarrow i+w$;
3. if $i<n$ then
   3.1. $u_i\leftarrow(k_{n-1}\cdots k_i)_2-\gamma$;
   3.2. if $u_i\le0$ then $u_i\leftarrow u_i+2^{n-i}$; $\gamma\leftarrow1$;
   else $\gamma\leftarrow0$;
   3.3. $u_{i+1}\leftarrow0,\cdots,u_{n-1}\leftarrow0$;
4. $u_n\leftarrow2-\gamma$;
   return $(u_n\cdots u_0)$;

Algorithm 1은 Recoding 1을 이용하여 부호가 없는 SPA에 안전하게 새로운 표현으로 변환 한 k를 이용하여 g^k을 구하는 것이다. 이 방법은 사전계산 단계, 리코딩 단계, 지수승 계산 단계로 총 3단계로 구성이 된다. 지수승 계산 단계에서는 w의 제곱 연산과 1번의 곱셈 연산으로 이루어져 있다.

1) 표수가 2인 경우 $P=(x,y)$이면 $-P=(x,x+y)$이고 표수가 큰 경우 $-P=(x,-y)$을 만족 한다

Algorithm 1.SPA-resistant Exponentiate based on Recoding 1.

INPUT :
$(n+2)$-bit exponent $k'=(10k_{n-1}\cdots k_0)_2$,with width w, base g;
OUTPUT : $c=g^{k'}$;

1. Pre-computation:
   - $g[1]\leftarrow g$;
   - for i from 2 to 2^w step 2 do:
     $g[i]\leftarrow g[i/2]^2; g[i+1]\leftarrow g[i]*g$;
2. Recoding:
   recoding k with Recoding 1;
   $i\leftarrow n-1; c\leftarrow g[u_n]$;
3. Evaluation:
   3.1. while $i\geq 0$ do:
      3.1.1. $c\leftarrow c^2$;
      3.1.2. if $u_i>0$ then $c\leftarrow c^*g[u_i]$;
      3.1.3. $i\leftarrow i-1$;
4. Return c;

Vuillaume-Okeya의 대응 방법은 역원 계산이 어려운 RSA 또는 DSA와 같은 환경에서 적합하지만 Right-to-Left 방향으로 리코딩이 된다는 단점을 가지고 있다. 즉, 지수승 연산과 리코딩 연산 사이에 추가적인 저장 공간이 필요하게 되므로 메모리 공간이 제한된 환경에서는 적합하지 못하다. 따라서 본 논문에서는 RSA 또는 DSA와 같은 환경에 적합하면서 지수승 연산과 리코딩 연산 사이에 추가적인 저장 공간이 필요하지 않는 리코딩 방법을 제안한다. 다음 장에서 제안하는 방법을 소개한다.

Ⅲ. 제안하는 Unsigned Left-to-Right 리코딩 알고리즘

본 장에서는 SPA에 안전한 Left-to-Right 리코딩 방법을 소개한다. 윈도우 크기가 w인 일반적인 경우와 윈도우 크기가 1인 특별한 경우에 대하여 고정된 형식으로 표현되는 방법을 소개하고, 지수승 알고리즘 중 comb 방법에도 확장하여 적용한다. 여기서 ϕ는 Euler phi 함수이고, RSA에서 선택하는 모듈러의 값 즉, 두 소수의 곱을 $N(=p\cdot q)$이라고 한다.

3.1 Case 1: width- w

본 절에서는 윈도우 크기가 w인 일반적인 경우에 대해서 리코딩 알고리즘을 제안한다. 리코딩 알고리즘을 제안하기에 앞서 사용되는 수식 및 기호들을 정의한다.

Notation :

1) n 비트 정수 k는 $k=\sum_{i=0}^{n-1}k_i2^i=(k_{n-1},\cdots,k_0)_2$와 같이 정의되고 이때 k_i는 0과 1로 되어있다. 그리고 $k'=(10k_{n-1}\cdots k_0)_2$로 k에 $\phi(N)$을 되풀이하여 더한 (n+2) 비트의 정수 값이다.[2)]

2) 윈도우 크기인 w는 2 보다 크고 $d=\lceil n/w\rceil$ 이다.

3) 정수 k가 $wd-1$이 되게 0을 채우고, $k=B^{n-1}\|\cdots\|B^1\|B^0$라고 한다. 이 때, B^j의 길이는 w이다.

4) $[B^j]:=\sum_{i=0}^{w-1}B_i^j2^i$이고 이때 B_i^j는 B^j의 i번째 비트라고 표현한다. 이 때, $[B^j]\in\{0,1,\cdots,2^w-1\}$ 이다.

5) k'를 리코딩 한 값은 $E^{d-1}\|\cdots\|E^1\|E^0$라고 표현하고 이때 E^j의 길이는 w이고 $[E^j]\in\{1,\cdots,2^w\}$이다. $[E^j]:=\sum_{i=0}^{w-1}E_i^j2^i$로 정의된다. 만약 $[E^j]$가 2^w이면 $(\underbrace{11\cdots12}_{w})$로 표현할 수 있다.

Recoding 1은 k'에서 Right-to-Left 방향으로 $\{[E^j]\}$를 생성한다. 즉, 음의 캐리를 전파하면서 $[E^0]$에서 $[E^{d-1}]$로 생성할 수 있게 된다. 본 논문의 목적은 Right-to-Left 방법에서 요구되는 추가적인 저장 공간을 줄이기 위해서 Left-to-Right 방향으로 $\{[E^j]\}$을 생성하는 방법을 제안하는 것이다. 즉, $[E^{d-1}]$에서 $[E^0]$을 생성한다.

본 논문에서 제안하는 방법은 Vuillaume-Okeya가 제안한 Recoding 1을 변형하여 생성할 수 있다. Right-to-Left 방법으로 진행되는 캐리 전파를 없애기 위해서 k를 그룹으로 나누어서 수행한다. 나눈 그룹에 $B_i=0$이 되는 시점과 $B_j\geq 2\ (j>i)$인 시점을 확인하여 $2^{(j-i)w}=\left(\underbrace{(2^w-1)\cdots(2^w-1)}_{j-i-1}2^w\right)$임을 이용하여 리코딩을 수행한다. 이와 같이 나눠진 그룹들 간에는 캐리가 발생하지 않는다. 캐리가 발생하지 않는다는 것은 각 그

2) Vuillaume-Okeya의 대응방법에서 리코딩 연산이 수행하는 동안 스칼라 길이의 정보를 제거하기 위하여 이용한다. 만약 k와 N이 고정되어 있는 값일 경우 더하는 계산은 키 생성 단계에서 한번만 수행된다.

룹을 따로 리코딩 한 후 다시 전체로 묶을 수 있다는 것이다. 제안하는 리코딩 방법에 대한 이해를 돕기 위해 간단한 예를 들어 살펴본다.

예제 1. 우선, k를 여러 그룹으로 나눈다. 그룹 중 $B^9\|B^8\|B^7\|B^6\|B^5$로 표현된 그룹이 있다고 하고, $[B^9] \geq 2$이고 나머지 $B^8 \sim B^5$ 그룹은 0과 1로 구성되어 있고, $[B^4] \geq 2$이다. 이때 아래와 같이 두 가지의 경우를 고려할 수 있다.

1. $B^j = 0 \quad (5 \leq j \leq 8)$인 그룹이 존재하지 않는다. 이 경우 리코딩 값이 바뀌지 않는다.
   즉, $E^j = B^j$.
2. $B^z = 0 \quad (5 \leq z \leq 8)$인 그룹이 존재한다.
   2.1. $[E^9] = [B^9] - 1$;
   2.2. $[E^j] = [B^j] + (2^w - 1) \quad (z < j \leq 8)$.
   만약 $z = 8$인 경우 수행되지 않는다;
   2.3. $[E^z] = 2^w$;
   2.4. $[E^j] = [B^j] \quad (5 \leq j < z)$.
   만약 $z = 5$인 경우 수행되지 않는다.

정리 1. 제안하는 리코딩 방법을 적용하였을 경우 기존의 값과 리코딩 된 결과 값이 동일하다.

증명) Recoding 1 방법의 규칙에 따라서 제안하는 방법이 기존의 값과 변형된 값이 같음을 증명할 수 있고, 식 (1)을 통해서도 확인 할 수 있다.

만약 $\sum_{j=t_1}^{t_2} [B^j] \cdot 2^{jw}$이고, $[B^{t_1}] \geq 2$이고 다른 블록의 값은 0과 1로 표현된다고 가정한다. 그리고 Right-to-Left로 진행될 경우 $[B^z] = 0$ (가정하기를 $t_1 < z < t_2 - 1$)이 되는 첫 인덱스 값이라고 하면 아래와 같은 식이 성립한다.

$$\begin{aligned}
&\sum_{j=t_1}^{t_2} [B^j] \cdot 2^{jw} \\
&= (\underbrace{[B^{t_1}] - 1}_{[E^{t_1}] \neq 0}) \cdot 2^{t_1 w} + \sum_{j=z+1}^{t_2-1} \underbrace{([B^j] + 2^w - 1)}_{[E^j]} \cdot 2^{jw} \\
&+ \underbrace{2^w}_{[E^z]} \cdot 2^{zw} + \sum_{j=t_1}^{z-1} \underbrace{[B_j]}_{[E^j]} \cdot 2^{jw} \qquad \cdots\cdots (1)
\end{aligned}$$

그러므로 리코딩 된 결과 값과 리코딩 전 결과 값이 같음을 쉽게 증명할 수 있다.

Recoding 2.
Unsigned Left-to-Right recoding

INPUT $k' = B^{d-1}\| \cdots \|B^1\|B^0$;
OUTPUT
Recoded exponent $[E^{d-1}], \cdots, [E^1], [E^0]$;

1. **Start**$\leftarrow d-1$;
2. while **Start** ≥ 0 do :
   2.1. **End**$\leftarrow$the first index t(<**Start**) such that $[B^t] \geq 2$ from the left-to-right direction from **Start**;
   If there is not t until index 0, then let **End**$\leftarrow -1$;
   2.2. **Zero**$\leftarrow$ the first index z(>**End**) such that $[B^z] = 0$ from the right-to-left direction from **End**;
   If there is not z between **Start**-1 to **End**$+1$ then **Zero**$\leftarrow$**NULL**;
   2.3. if **Zero**=**NULL**, then
   for j=(**Start**) down to (**End**+1){
   $[E^j] \leftarrow [B^j]$ };
   2.4. else if **Zero** $\neq$ **NULL**, then
   2.4.1. $[E^{Start}] \leftarrow [B^{Start}] - 1$;
   2.4.2. if (**End+1** $\leq$ **Zero**<**Start-1**), then **for** j=(**Start-1**) down to (**Zero+1**) {$[E^j] \leftarrow [B^j] + (2^w - 1)$};
   2.4.3. $[E^{Zero}] \leftarrow 2^w$;
   2.4.4. if (**End+1**<**Zero** $\leq$ **Start-1**) then **for** j=(**Zero-1**) down to (**End+1**){ $[E^j] \leftarrow [B^j]$ };
   2.5. **Start**$\leftarrow$**End**;

위의 리코딩 방법을 알고리즘으로 나타내면 Recoding 2와 같다.

제안하는 Recoding 2를 이용하면 윈도우 크기가 w일 경우 스칼라는 $|0 \cdots 0y|0 \cdots 0y| \cdots |0 \cdots 0y|$와 같이 고정된 형식을 가지고, 이 때 $y \in \{1, 2, \cdots, 2^w\}$이다. 고정된 형식을 갖는 스칼라를 이용한 지수승 연산을 살펴보면 사전계산 단계는 Algorithm 1과 동일하게 수행 된다. 그리고 Recoding 2는 $[E^j]$를 생성하는 순서가 지수승 연산 단계와 동일한 Left-to-Right 방향이기 때문에 리코딩 단계와 지수승 연산 단계를 하나의 단계로 합쳐서

수행 할 수 있다. 각 단계는 한 번의 곱셈과 w의 제곱 연산으로 이루어져있고, 본 리코딩 방법을 이용한 지수승 연산 알고리즘의 안전성은 다음과 같은 가정 1에 의존 한다

가정 1. 제곱 연산 c^2과 곱셈 연산 $c \cdot y$는 하나의 전력 분석 량을 통해서 구별 할 수 있지만 $c \cdot y$와 $c \cdot y+\alpha$는 구별 할 수 없다. 여기서 α는 2개의 $(w+1)$ 비트열 사이의 덧셈 연산 량이다.

Recoding 2에서 단계 2.3과 2.4.4는 $c \cdot y$와 부합되고 나머지는 단계 2.4.2와 같이 $c \cdot y+\alpha$와 부합될 수 있다. c와 w의 비트 길이는 1024 또는 2048이고, 윈도우 크기 w는 위의 가정에 적당하게 10보다 적은 값이 선택된다. α는 큰 수의(>1000 비트) 곱셈 연산의 연산량에 비해 거의 없기 때문에 Recoding 2를 이용한 지수승 연산 알고리즘은 위의 가정을 통해 SPA에 안전하게 수행 할 수 있다. 다시 말해서 각 단계를 하나의 전력 소모량을 통하여 구별 할 수가 없다. 위의 가정은 타원곡선에서도 유사하게 적용된다. 즉, 타원곡선 스칼라 곱셈 연산에서 덧셈과 두 배 연산은 한 번의 전력 소모량을 통해서 구별 될 수 있지만 덧셈과 뺄셈 연산은 구별할 수 없다. 만일 공격자가 이 종류의 차이를 식별 할 수 있으면 Recoding 1에서 u_i를 결정하는 단계에서 단계 2.2와 3.2를 구별할 수 있기 때문에 Algorithm 1의 안전성 또한 논쟁이 될 수 있다.

3.2 Case 1: width-1

윈도우 크기가 1일 경우, 다시 말해 디짓 셋이 {1,2}인 경우 Recoding 2은 더욱 간단하게 Recoding 3처럼 구성될 수 있다. Recoding 1에서와 같이 n 비트를 $(n+2)$ 비트의 값으로 확장하여서 리코딩을 수행하게 된다. 이 때 $k_{n+1}=1, k_n=0$이다. Recoding 3은 식(2)와 같은 형태를 따른다. 이 때 z는 k을 2진수로 표현했을 경우 $k_z=0$이 되는 최하위 비트의 인덱스 값을 의미한다.

$$\sum_{j=0}^{n+1} k_j 2^j = \sum_{j=z+1}^{n} (k_j+1)2^j + 2 \cdot 2^z + \sum_{j=0}^{z-1} k_j 2^j \quad \cdots (2).$$

또한 $(n+2)$ 비트 값이 입력되면 $(n+1)$ 비트 값이 출력이 된다. 식 (2)는 식(3)과 같이 쉽게 증명 될 수 있다.

Recoding 3.
Unsigned Binary Left-to-Right recoding

INPUT
$(n+2)$-bit exponent $k'=(k_{n+1}\cdots k_0)_2$, with $k_{n+1}=1$ and $k_n=0$;
OUTPUT Recoded exponent $(e_n \cdots e_0)$
where $e_j \in \{1,2\}$;

1. Find the first index z from the least significant bit such that $k_z=0$;
2. $j \leftarrow n$;
3. while $j \geq 0$ do :
   3.1. while $j > z$ do:
      3.1.1. if $k_j=0$ then $e_j \leftarrow 1$;
      3.1.2. else if $k_j=1$ then $e_j \leftarrow 2$;
   3.2. while $j \leq z$ do:
      3.2.1. if $j=z$ then $e_j \leftarrow 2$;
      3.2.2. else if $j<z$ then $e_j \leftarrow k_j$;
   3.3. $j \leftarrow j-1$;

$$\begin{aligned}
&\sum_{j=z+1}^{n} (k_j+1)2^j + 2 \cdot 2^z + \sum_{j=0}^{z-1} k_j 2^j \\
&= \sum_{j=z+1}^{n} k_j 2^j + \left(\sum_{j=z+1}^{n} 2^j + 2 \cdot 2^z\right) + \sum_{j=0}^{z-1} k_j 2^j \\
&= \sum_{j=z+1}^{n} k_j 2^j + (2^{n+1}) + \sum_{j=0}^{z-1} k_j 2^j \\
&= \sum_{j=0}^{n+1} k_j 2^j \qquad \cdots\cdots (3)
\end{aligned}$$

예제 2. 윈도우 크기가 1일 경우의 예를 간단히 들어보면 다음과 같다. $k=(1,0,1,0,0,1)$이면 비트 값이 0이 되는 최하위 비트의 위치가 k_1이다. 그러므로 $1 \cdot 2^5 = 1 \cdot 2^4 + \cdots + 1 \cdot 2^2 + 2 \cdot 2^1 + 0 \cdot 2^0$임을 이용하면 $k'=(0+1,1+1,0+1,0+2,1+0)$로 바꿀 수 있다.

위에서 설명한 방법을 알고리즘으로 나타내면 Recoding 3과 같다.

Recoding 3을 이용한 지수승 연산을 알고리즘으로 나타내면 Algorithm 2와 같다. 이 때, Algorithm 2는 일반적인 윈도우 크기인 w인 경우의 지수승 연산과 마찬가지로 Algorithm 1과는 달리 리코딩 단계와 지수승 계산 단계를 하나의 단계로 합쳐서 고려할 수 있다. 즉, 리코딩 단계에서의 결과 값을 따로 저장 하지 않기 때문에 저장 공간이 줄어든다.

Algorithm 2.
SPA-resistant Exponentiate based on Recoding 3.

INPUT
$(n+2)$-bit exponent $k'=(10k_{n-1}\cdots k_0)_2$,
base g;
OUTPUT $c=g^{k'}$;

1. **Pre-computation**:
   - $g[1]\leftarrow g$ and $g[2]\leftarrow g^2$;
   - Find the first index z from the least significant bit such that $k_z=0$;
2. **Recoding+Evaluation**:
   2.1. $j\leftarrow n$;
   2.2. while $j\geq 0$ do:
      2.2.1. $c\leftarrow c^2$;
      2.2.2. if $j>z$ then
         (a) if $k_j=0$ then $c\leftarrow c^*g[1]$;
         (b) else if $k_j=1$ then $c\leftarrow c^*g[2]$;
      2.2.3. if $j\leq z$ then
         (a) if $j=z$ then $c\leftarrow c^*g[2]$;
         (b) else if $j<z$ then $c\leftarrow c^*g[k_j]$;
      2.2.4. $j\leftarrow j-1$;
   2.3. Return c;

3.3 Case 3: Unsigned Left-to-Right Comb 리코딩

본 절에서는 3.2절에서 소개한 Recoding 3 방법을 지수승 연산 방법 중에 사전 계산을 이용하여 계산하는 경우 효율적이라고 알려진 comb 방법에 적용시킨다. comb 방법은 Lim-Lee[17]가 제안하였고 사용되는 용어와 기호를 정의하면 다음과 같다.

- 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 $k^j{}_i$라고 표현한다.
- $\Xi_i\equiv[K_i^{w-1},\cdots,K_i^1,K_i^0]$.
- $(K_{w-1},K_{w-2},\cdots,K_1,K_0)\in Z_2^w$ 일 때,
$[K_{w-1},K_{w-2},\cdots,K_0]\equiv K_{w-1}2^{(w-1)d}+\cdots+K_12^d+K_0$

Recoding 4.
Unsigned Left-to-Right Comb recoding

INPUT $k=(k_{n-1}\cdots k_0)_2$;
OUTPUT $\Xi_{d-1},\cdots,\Xi_1,\Xi_0$;

1. Find the first index z from the least significant bit such that $k_z=0$;
2. for $i=d-1$ down to 0 do:
   2.1. for $j=w-1$ down to 0 do:
      $t=jd+i$;
      2.1.1. if $t\geq n-1$ then $K_i^j\leftarrow 0$;
      2.1.2. if $z<t\leq n-2$
         (a) if $k_t=0$ then $K_i^j\leftarrow 1$;
         (b) else if $k_t=1$ then $K_i^j\leftarrow 2$;
      2.1.3. if $t\leq z$
         (a) if $t=z$ then $K_i^j\leftarrow 2$;
         (b) else if $t<z$ then $K_i^j\leftarrow k_j$;
   2.2. set $\Xi_i=[K_i^{w-1},\cdots,K_i^0]$;

이다.

기존의 comb 방법은 사전 계산 단계, 리코딩 단계, 그리고 지수승 연산 단계[3]로 구성되어 진다. comb 방법에서 리코딩 단계는 스칼라 k를 w열과 d행으로 구성된 배열로 표현하여 w개의 원소로 이루어진 각 열을 사용한다. comb 방법에서도 $\Xi_i\equiv[K_i^{w-1},\cdots,K_i^1,K_i^0]$이 0일 경우와 그렇지 않은 경우 곱셈 연산과 제곱 연산이 다르게 수행되기 때문에 SPA 공격에 취약하다. 따라서 본 절에서는 본 논문에서 제안한 리코딩 방법이 comb 방법에서도 쉽게 확장하여 적용할 수 있음을 보여준다. Recoding 4는 Recoding 3을 확장한 것이다.

Ⅳ. Unsigned Radix-r Left-to-Right 리코딩 알고리즘 확장

본 장에서는 Ⅲ장에서 제안한 부호 없는 이진 Left-to-Right 리코딩 방법(Recoding 3)을 기수가 r로 표현(radix-r)된 경우로 확장하여 설명한다.

3) comb 방법은 리코딩 부분과 지수승 연산 부분을 합쳐서 수행할 수 있다.

4.1 Radix-r을 고려하는 동기

페어링 기반의 암호 시스템(pairing based crypto systems)은 tripartite Diffie-Hellmann scheme[10], ID-based cryptosystems[2], short digital signature[5] 처럼 다양한 암호시스템에서 적용할 수 있고 지금까지도 이러한 연구들이 계속 진행되고 있다. 페어링 기반 암호 시스템의 가장 핵심적인 연산은 Weil 페어링 또는 Tate 페어링 연산과 타원곡선의 스칼라 곱셈 연산으로 이루어져있다. 이 연산들은 대부분 암호시스템의 안전성과 관련된 비밀 값들을 다루고 시간이 많이 소요되기 때문에 페어링 기반의 프로토콜들과 암호시스템들의 안전성과 효율성은 페어링 연산과 스칼라 곱셈 연산에 의존하게 된다. 스칼라 곱셈에 비해 상대적으로 많은 연구가 이루어지지 않았던 페어링 계산의 효율성에 대한 연구가 최근에 활발히 진행되고 있다. 구체적으로 Duursma와 Lee가 표수(characteristic)가 작은 소수 r인 초특이 타원곡선(supersingular elliptic curve)에서 Tate 페어링이 효율적으로 연산이 가능하다는 것을 제안하였고[6], 특히 $r=3$인 경우에 최적화된 알고리즘들이 제안되었다[4,6,8,22,25]. 또한, Harsasawa가 표수가 5인 경우의 타원곡선 페어링을 매우 효율적으로 계산하는 방법을 제안하였다[9]. 이와 같이 페어링 연산의 효율성 때문에 대부분의 페어링 기반 암호시스템은 표수가 작은 소수 r인 타원곡선에서 정의된다. 예를 들어 $GF(3^m)$에서 정의 된 초특이 타원곡선에서는 한 점을 두 번 더하는 연산보다 세 번 더하는 $3P$ 연산이 더 효율적으로 계산된다. 그러므로 페어링 암호시스템에서 스칼라 곱셈 연산은 이진 표현보다 일반적으로 표수 r를 이용하여 스칼라를 r진법으로 표현하는 것이 효율적일 수 있다.

그러나 이러한 연구들은 주로 수학적으로 안전성을 증명할 수 있으면서 실용적인 시스템을 개발하는데 치중해 왔다. 그러나 이러한 암호 시스템 역시 안전성이 수학적으로 증명 가능하더라도 부채널 공격에 여전히 적용할 수 있다. 이와 같은 페어링 기반의 암호시스템은 비밀 값이 매번 다르게 선택되므로 같은 값이 선택 될 확률이 무시할 만하기 때문에 DPA를 적용할 수 없다. 따라서 표수 r에서의 연산에 적합하면서 SPA에 안전한 연산 알고리즘을 제안하는 것 역시 중요한 과제이다. 그러므로 본 장에서는 이전 장에서 제시한 리코딩 알고리즘을 기수가 r인 경우에도 확장하여 수행할 수 있음을 보여준다.

Recoding 5.
Unsigned Radix-r Left-to-Right recoding

INPUT $(a_{n-1}\cdots a_0)_r$ with $a_{n-1}\geq 2$;
OUTPUT Recoded exponent $(u_{n-1}\cdots u_0)_r$ with $u_j\in\{1,2,\cdots,r\}$;

1. **Start**$\leftarrow n-1$;
2. while **Start**≥ 0 do :
   2.1. **End**$\leftarrow$the first index t(<**Start**) such that $a_t\geq 2$ from the left-to-right direction from **Start**;
   If there is not t until index 0, then let **End**$\leftarrow -1$;
   2.2. **Zero**$\leftarrow$ the first index z(>**End**) such that $a_z=0$ from the right-to-left direction from **End**;
   If there is not z between **Start**-1 to **End**$+1$ then **Zero**$\leftarrow$**NULL**;
   2.3. if (**Zero**=**NULL**), then
   for j=(**Start**) down to (**End**+1) { $u_j\leftarrow a_j$ };
   2.4. else if(**Zero**$\neq$**NULL**), then
      2.4.1. $u_{Start}\leftarrow a_{Start}-1$;
      2.4.2. if (**End+1** $\leq$ **Zero**<**Start-1**), **for** j=(**Start**-1) down to (**Zero**+1) { $u_j\leftarrow a_j+(r-1)$ };
      2.4.3. $u_{Zero}\leftarrow r$;
      2.4.4. if (**End+1**<**Zero** $\leq$ **Start-1**), **for** j=(**Zero**-1) down to (**End**+1) { $u_j\leftarrow a_j$ };
   2.5. **Start** $\leftarrow$**End**;

4.2 제안하는 Unsigned Radix-r Left-to-Right 리코딩 알고리즘

본 소절에서는 주어진 기수 r 표현법을 고정된 패턴을 갖는 표현 방법을 설명한다. $\{0,1,2,\cdots,r-1\}$의 원소로 표현된 기수 r의 표현법을 0과 부호수를 사용하지 않는 $\{1,2,\cdots,r-1\}$의 원소들만을 이용하여 최상위 비트부터 리코딩하게 된다. 먼저 기수 r인 k는 $k=$

$\sum_{j=0}^{n-1} a_j r^j := (a_{n-1} \cdots a_0)_r$ 와 같이 표현할 수 있고, 이 때 $a_j \in \{0,1,\cdots,r-1\}$이다. 이 때 k에 (r^n-1)을 더하게 된다면[4], $a_{n-1} \geq 2$라고 가정할 수 있다. 기수 r로 확장하는 방법 역시 3.1장에서 설명한 방법과 동일하게 k를 몇 개의 그룹으로 나누게 되고, 식(4)가 성립함을 알 수 있다. 이 때 $\sum_{j=t_1}^{t_2} a_j r^j$이고, $a_{t_2} \geq 2$이다. 그리고 나머지 값은 0과 1로 이루어져있고, z는 Right-to-Left 방향에서 $a_z = 0$이 되는 처음 인덱스 값이다.

$$\sum_{j=t_1}^{t_2} a_j r^j = a_{t_2} r^{t_2} + \sum_{j=z+1}^{t_2-1} a_j r^j + 0 \cdot r^z + \sum_{j=t_1}^{z-1} a_j r$$

$$= \underbrace{(a_{t_2}-1)r^{t_2}}_{u_{t_2} \neq 0} + \sum_{j=z+1}^{t_2-1} \underbrace{(a_j + r - 1)r^j}_{u_j}$$

$$+ \underbrace{r}_{u_z} \cdot r^z + \sum_{j=t_1}^{z-1} \underbrace{a_j}_{u_j} r^j \quad \cdots\cdots (4)$$

부호를 사용하지 않는 radix-r Left-to-Right 리코딩 방법은 아래 Recoding 5와 같고, $u_j \in \{1,2,\cdots,r\}$는 a_j에 해당되는 리코딩 된 값이다.

V. 결 론

본 논문은 역원 연산이 어려운 RSA, DSA, 그리고 기수가 $r(>2)$에서 효율적으로 연산이 가능한 페어링 기반의 암호시스템에서 효율적으로 SPA를 방어할 수 있는 리코딩 방법을 제안하였다. 제안하는 리코딩 방법은 지수승 연산이 수행되는 방법과 같은 Left-to-Right 방향으로 진행되기 때문에 Right-to-Left로 리코딩 될 때 추가적으로 요구되는 저장 공간이 필요하지 않게 된다. Recoding 5에서 제안하는 기수 r의 표현방법 역시 윈도우 크기가 w인 경우도 확장 가능하고, Comb 방법에도 적용가능 하다. 그러므로 제안하는 대응방법은 기존의 부채널 공격에 안전하면서 적은 저장 공간을 이용하여 효율적으로 연산할 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] D.Boneh and M.Franklin, "Identity Based Encryption from the Weil Pairing," SIAM J. of Computing, Vol. 32, No. 3, pp. 586-615, 2001.

[3] P.Barreto, S.Galbraith, C.hEigeartaigh, and M.Scott, "Efficient Pairing Computation on Supersingular Abelian Varieties," Cryptology ePrint Archive: Report 2004/375, 2005.

[4] G.Bertoni, J.Guajardo, S.Kumar, G.Orlando, C.Paar, and T.Wollinger, "Efficient GF(pm) Arithmetic Architectures for Cryptographic Applications," CT-RSA 2003, LNCS 2612, pp. 158-175, 2003.

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

[6] I.Duursma and H-S.Lee, "Tate Pairing Implementation for Hyperelliptic Curves y2 = xp-x + d," ASIACRYPT 2003, LNCS 2894, pp. 111-123, 2003.

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

[8] K.Harrison, D.Page, and N.Smart, "Software Implementation of Finite Fields of Characteristic Three," LMS Journal of Computation and Mathematics, Vol.5, pp. 181-193, 2002.

[9] R.Harasawa, Y.Sueyoshi, and A.Kudo, "Ate pairing for y2=x5-ax in Characteristic Five," Cryptology ePrint Archive: Report 2006/202, 2006.

[10] A.Joux, "A one round protocol for tripartite Diffie-Hellman," ANTS V, LNCS 1838, pp.385-394, 2000.

[11] M. Joye and S. Yen, "Optimal Left-to-Right

4) (r^n-1)은 $F_{r^n}^*$의 위수이다.
예를 들어 $e(aP,bP)^c = e(aP,bP)^{c+t(r^n-1)}$.

Binary Signed-Digit Recoding," IEEE Trans. Computers, vol. 49, pp. 740-748, July, 2000.

[12] N. Koblitz, "Elliptic curve cryptosystems," In Mathematics of Computation, volume 48, pp. 203-209, 1987.

[13] 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.

[14] P. Kocher, J. Jaffe, B. Jun, "Differential Power Analysis," Advances in Cryptology-CRYPTO'99, LNCS1666, pp. 388-397, 1999.

[15] K. Lauter, "The advantages of elliptic curve cryptography for wireless security," IEEE Wireless Communications, vol. 11, Issue 1, pp. 62-67, Feb., 2004.

[16] C. Lim, "A new method for securing elliptic scalar multiplication against side channel attacks," Information Security and Privacy - ACISP'04, LNCS 3108, pp. 289-300, Springer-Verlag, 2004.

[17] C. Lim and P. Lee, "More Flexible Exponentiation with Precomputation," Advances in Cryptology-CRYPTO'94, LNCS 839, pp. 95-107, Springer-Verlag, 1994.

[18] V.S. Miller, "Use of elliptic curves in cryptography," In Advances in Cryptology-CRYPTO'85, LNCS218, pp. 417-426, 1986.

[19] B. Möller, "Securing Elliptic Curve Point Multiplication against Side-Channel Attacks," Information Security-ISC'01, LNCS2200, pp. 24-334, 2001.

[20] K. Okeya, K. Schmidt-Samoa, C. Spahn, and T. Takagi, "igned Binary Representations Revisited," dvances in Cryptology- CRYP '04, LNCS 3152, pp. 123-139, Springer-Verlag, 2004.

[21] K. Okeya and T. Takagi, "The width-wNAF method provids small memory and fast elliptic scalar multiplications secure against side channel attacks," Topics in Cryptology-CT-RSA'03, LNCS 2612, pp. 328-343, Springer-Verlag, 2003.

[22] D.Page and N.Smart, "Hardware Implementation of Finite Fields of Characteristic Three," CHES 2002, LNCS 2523, pp. 529-539, 2002.

[23] 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.

[24] J.H. Shin, D.J. Park, and P.J. Lee, "DPA Attack on the Improved Ha-Moon Algorithm," Workshop on Information Security Applications-WISA 2005, LNCS 3786, pp. 283-291, Springer-Verlag, 2006.

[25] 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.4 85-497, 2003.

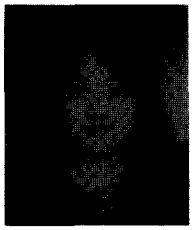
[26] N. Theriault "SPA Resistant Left-to-Right Integer Recodings," Selected Areas in Cryptography-SAC 2005, LNCS 3897, pp. 345-358, Springer-Verlag, 2006.

[27] X. Tian, D. Wong, and R. Zhu, "Analysis and Improvement of an Authenticated Key Exchange Protocol for Sensor Networks," IEEE Communications letters, vol. 9, pp. 970-972, November, 2005.

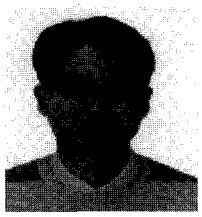
[28] C. Vuillaume and K. Okeya, "Flexible Exponentiation with Resistance to Side Channel Attacks," Applied Cyptography and Network Security, ACNS 2006, LNCS 3989, pp. 268-283, Springer-Verlag, 2006.

[29] S.M. Yen, C.N. Chen, S. Moon, and J. Ha "Improvement on Ha-Moon Randomized Exponentiation Algorithm," International Conference on Information Security and Cryptology-ICISC 2004, LNCS 3506, pp. 154-167, Springer-Verlag, 2005.

〈著者紹介〉



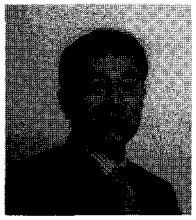
김 성 경 (Sung-Kyoung Kim) 학생회원
2005년 2월: 동의대학교 수학과 학사
2005년 3월 ~ 현재: 고려대학교 정보경영공학전문대학원 석사과정
<관심분야> 부채널 공격, 공개키 암호, 암호칩 설계 기술



한 동 국 (Dong-Guk Han) 일반회원
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

김 호 원 (Ho Won Kim) 정회원
1993년 경북대학교 전자공학과 졸업(학사)
1995년 포항공과대학교 전자전기공학과 석사 (공학석사)
1999년 포항공과대학교 전자전기공학과 박사 (공학박사)
2003년 7월 ~ 2004년 6월 독일 Ruhr University Bochum, Post Doc. 과정
1998년 12월 ~ 현재 : 한국전자통신연구원 정보보호연구단 선임연구원/팀장
<관심분야> RFID 정보보호 기술 및 USN 정보보호 기술, 타원곡선 및 초타원곡선 암호이론, VLSI 설계

정 교 일 (Kyo IL Chung) 정회원
1977. 3. ~ 1981. 2. : 한양대학교 공과대학 전자공학 학사
1981. 3. ~ 1983. 8. : 한양대학교 대학원 전자계산학 석사
1991. 3. ~ 1997. 8. : 한양대학교 대학원 전자공학 박사
1980. 12. ~ 1981. 11. : 엠-시스템즈 사원
1981. 12. ~ 1982. 2. : 한국전기통신연구소 위촉연구원
1982. 3. ~ 현재 : 한국전자통신연구원 책임연구원(정보보호기반그룹 그룹장)
2000. 8. ~ 현재 : International WISA 운영위원장, 조정위원
2001. 1. ~ 현재 : ITU-T SG17 연구위원
2001. 1. ~ 현재 : TTA 국제표준화전문가
2001. 11. ~ 현재 : 한국전자지불산업협회 전자지불포럼위원장
2004. 6. ~ 현재 : 아시아 IC카드 포럼 운영위원



임 종 인 (Jongin Lim) 정회원
1980년 2월 : 고려대학교 수학과 학사
1982년 2월 : 고려대학교 수학과 석사
1986년 2월 : 고려대학교 수학과 박사
1999년 2월 ~ 현재: 고려대학교 정보보호대학원 원장, CIST 센터장
<관심분야> 암호이론, 정보보호정책