타원곡선 암호의 최적화 구현, 부채널 대응기법 동향분석 및 벤치마킹*

송진교[0] 서석충
국민대학교 금융정보보안학과
sjk9304@kookmin.ac.kr, scseo@kookmin.ac.kr

Survey and Benchmarking for Optimized Implementation and Side Channel Countermeasures of Elliptic Curve Cryptography

JinGyo Song[0] SeogChung Seo
Department of Financial Information Security, Kookmin University

요 약

5G 이동통신이 본격적으로 상용화되면서, 다양한 모바일 기기가 이동통신 망(E-UTRN)에 접속하여 방대한 데이터를 통신하고 있다. 이러한 데이터 통신은 무선통신으로 이루어지기 때문에 다양한 보안 위협이 존재한다. 따라서 전송 시 암호화 및 인증이 요구된다. 특히 5G부터는 보안을 위해 공개키 암호(타원곡선 암호)를 지원한다. 하지만 공개키 암호는 수학적인 난제에 어려움을 기반을 두어, 수학적인 연산이 많아서 대칭키 암호보다 성능 부하가 많이 발생하는 단점이 존재한다. 게다가 공개키 암호가 수학적으로 안전하더라도, 구현 시 부채널 대응방안을 고려하지 않으면, 부채널 취약점이 존재한다. 따라서 공개키 암호구현 시 최적화뿐만 아니라 부채널 대응방안이 고려되어야 한다. 본 논문에서는 타원곡선 암호의 최적화 및 부채널 대응방안의 기존 연구 동향에 대해 분석한다. 또한 벤치마킹을 통해 각 최적화 및 부채널 대응방안에 대한 성능 측정 결과를 제시한다.

1. 서 론

4차 산업혁명 기술의 발전으로, 사회는 스마트 홈, 스마트 시티, 스마트 팩토리 같은 서비스를 사용자에게 실현 가능한 스마트 사회로 발전하고 있다. 이러한 서비스는 사용자에게 편리함을 줄 수 있지만, 대부분의 통신은 무선으로 이루어지기 때문에 해킹, 스푸핑, 중간자 공격과 같은 보안사고에 매우 취약하다. 따라서 전송되는 데이터는 암호 알고리즘을 통해 데이터의 보호가 이루어져야 한다. 특히 그 중 공개키 암호는 데이터의 기밀성, 인증, 키 교환과 같은 보안서비스를 제공하는 데 있어 핵심적인 역할을 수행한다.

현재 널리 사용되고 있는 공개키 암호는 타원곡선 암호 기반으로 타원곡선 이산대수 문제에 안전성을 기반을 두고 있다. 타원곡선 암호는 이전 공개키 암호 알고리즘인 RSA에 비해 짧은 키 길이로써 같은 보안 강도를 제공하는 장점이 존재한다. 예를 들어 타원곡선 암호의 256-bit의 키는 RSA의 3072-bit 키와 같은 보안 강도를 제공한다. 키 길이가 짧아짐으로써, 빠른 연산속도와 더 적은 메모리만으로 보안서비스를 제공할 수 있는 장점이 존재한다. 하지만 공개키 암호는 대칭키 암호 및 해시함수와 달리 복잡한 수학적인 연산을 다수 수행하기 때문에 연산 부하가 걸리는 단점이 존재한다. 따라서 성능 향상을 위해서는 각 플랫폼에 맞는 최적화 연구가 필요하다.

타원곡선 암호의 핵심적인 연산은 스칼라 곱셈 연산으로 스칼라 비트에 따라 스칼라 비트가 1일 경우에는 ECADD, ECDBL 연산을 수행하며, 스칼라 비트가 0일 경우에는 ECDBL 연산만을 수행한다. 따라서 스칼라 비트에 따라 연산 소모량이 다르며, 이를 단순 구현하게 되면 부채널 취약점이 존재한다. 다수의 연구가 전력 분석(SPA, DPA), TA, Fault Attack 등 다양한 관점으로 타원곡선 암호의 부채널 공격이 가능함을 보여주고 있다[1-3]. 이에 대응하기 위해서는 구현 시 부채널 대응방안이 적용되어야 하며, 현재도 타원곡선 암호의 다양한 부채널 대응방안이 연구되고 있다.

본 논문에서는 타원곡선 암호의 최적화 및 부채널 대응방안의 연구 동향을 살펴본다. [4-6] 최적화 관점에서는 Jacobian Projective, Window NAF, Comb 방법을 분석하며, 부채널 대응 관점으로는 전력분석인 SPA의 대응방안 연구 동향에 대해 분석한다. [5-6] 또한 벤치마킹을 통해 최적화 및 부채널 대응방안에 대한 성능 측정 결과를 제시한다. 본 논문의 요약은 다음과 같다. 2장에서는 타원곡선 암호의 최적화 구현 연구 동향을 분석하며, 3장에서는 타원곡선 암호의 부채널 대응방안의 연구 동향을 분석한다. 4장에서는 벤치마킹을 통해 최적화 및 부채널 대응방안의 성능 측정 결과를 제시하며, 5장에서는 본 논문의 결론 및 향후 계획을 제시하고 마무리한다.

* 본 논문은 2021년도 과학기술정보통신부의 재원으로 정보통신기획평가원의 지원(No.2021-0-00540,GPU/ASIC 기반 암호알고리즘 고속화 설계 및 구현 기술개발, 50%)과 2021년도 정부의 재원으로 한국연구재단의 지원(No.2019R1F1A1068494, 50%)을 받아 수행된 연구임

2. 타원곡선 암호의 최적화 구현 동향

본 절에서는 타원곡선 암호의 최적화 구현에 대한 동향을 분석한다. [4]에서 제시한 다양한 최적화 기법 중 널리 사용되는 Jacobian Projective, Window NAF, Comb 방법을 분석한다.

2.1. Jacobian Projective

(x,y) 좌표는 아핀 좌표계라 하며, 타원곡선은 타원곡선만의 장점인 2차원 좌표를 3차원으로 확장이 가능하다. 즉 아핀 점을 (x/z, y/z)로 표현하면 이를 3차원 좌표계인 (x, y, z)로 확장할 수 있다. 이를 Projective 좌표계라 한다. 이를 통해 가장 성능 부하가 큰 역원 연산을 효율적으로 줄 일 수 있다. 즉 매 스칼라 비트에서 발생하는 역원 연산을 제거하여 성능을 극대화 시킬 수 있다. Projective 좌표계 중 나눗셈을 효율적으로 제거하기 위한 것이 Jacobian 좌표계이다. 기존 Standard Projective 좌표계에서 z의 지수 승이 2, 3으로 Jacobian 좌표계는 $(x,y,z)=(x/z^2,y/z^3,1)\in E\colon y^2=x^3+axz^4+bz^6$ 로 표현된다. Jacobian ECADD, ECDBL은 각각 8M+3S+6A+1CM, 4M+4S+5A+3CM+1SH의 연산량을 가진다. 이는 아핀 기반 ECADD, ECDBL의 연산량과 비교해보았을 때, 곱셈 연산이 증가하였지만, 가장 성능 부하가 큰 역원 연산이 제거되어 성능을 극대화할 수 있다.

2.1. Window NAF

Window NAF의 기본적인 아이디어는 1의 개수를 줄여 ECADD의 연산을 효율적으로 감소시키는 것이다. 기존의 타원곡선 암호에서의 스칼라 비트는 1, 0만을 사용하였지만, Window NAF에서는 -1, 0, 1을 사용함으로써 스칼라 비트 1의 개수를 최소화한다. 예를 들어 $(11111111)_2(=255)$는 스칼라 비트에 -1의 원소를 추가하게 되면 $(10000000\bar{1})_2(=255)$로 나타낼 수 있다. 이는 8번의 ECADD 연산을 2번만으로 효율적으로 감소시켰다. Window NAF를 적용하기 위해서는 먼저 스칼라 비트를 -1로 확장시키기 위한 NAF 변환을 수행해야한다. Window NAF는 NAF 변환을 통해 효율적으로 스칼라 비트의 1의 개수를 줄인 후, Window size를 정하여 Window size만큼 비트를 스캔하여 스칼라 곱셈을 수행한다. 기존 단순 비트 스캔 방법을 Window 방법으로 확장하여 효율적으로 성능을 높이는 방법이다. Window 방법으로 확장하기 위해서는 Window size에 해당하는 만큼의 사전연산이 필요하며 일반적으로 4~7-bit의 Window size를 많이 사용하므로, 사전연산량이 많지 않아 가변 점 스칼라 곱셈에서 효율적으로 적용이 가능하다. Window NAF 스칼라 곱셈 알고리즘은 [그림 1]과 같다.

Algorithm 3.36 Window NAF method for point multiplication

INPUT: Window width w, positive integer k, $P\in E(\mathbb{F}_q)$.
OUTPUT: kP.

1. Use Algorithm 3.35 to compute $\mathrm{NAF}_w(k)=\sum_{i=0}^{l-1}k_i2^i$.
2. Compute $P_i=iP$ for $i\in\{1,3,5,\ldots,2^{w-1}-1\}$.
3. $Q\leftarrow\infty$.
4. For i from $l-1$ downto 0 do
   4.1 $Q\leftarrow 2Q$.
   4.2 If $k_i\neq 0$ then:
   If $k_i>0$ then $Q\leftarrow Q+P_{k_i}$;
   Else $Q\leftarrow Q-P_{-k_i}$.
5. Return(Q).

[그림 1] Window NAF 스칼라 곱셈 알고리즘 [4]

- $(31132)_4=862=(0011||0101||1110)_2$
- $w=4,\ d=3\rightarrow(001||101||011||110)$
- Precompute: $[512P,64P,8P,P]$의 모든 덧셈 조합

2	1	0
1	1	0
0	1	1
1	0	1
0	0	1

✓ **d = 2**
Q = 65P
✓ **d = 1**
Q = 130P+9P=139P
✓ **d = 0**
Q = 278P+584P=862P

[그림 2] Comb 스칼라 곱셈의 예제

2.3. Comb

Comb 스칼라 곱셈 방법은 스칼라 비트 열을 w 비트가 되도록 균등하게 분할 후, 열 단위로 스칼라 비트를 스캔하여 스칼라 곱셈을 수행하는 최적화 기법이다. [그림 2]는 Comb 스칼라 곱셈 방법의 예제이다. 예를 들어 스칼라 비트를 862하고, 비트 열(w)를 4, 비트 행(d)=3으로 설정하면 아래의 그림 표와 같이 862에 대한 스칼라 비트가 행과 열로 나타내진다. 스칼라 곱셈 시에는 이를 열 단위로 스캔하며 스칼라 곱셈이 수행되며, 위의 예제는 4개의 행으로 나누어졌고 행의 원소 개수가 3이므로, P, 8P, 64P, 512P에 대한 사전연산이 필요하다. Comb 방법은 Window NAF 방법과 마찬가지로 사전연산이 필요하며, Comb 방법은 Window NAF 방법보다 더 많은 사전연산을 요구하는 단점이 존재하여, 고정된 점에 대한 스칼라 곱셈에서 매우 효율적이다. Comb 알고리즘에 대한 전체 과정은 [그림 3]과 같다.

Algorithm Fixed-base comb method for point multiplication

INPUT: Window width w, $d=\lceil t/w\rceil$, $k=(k_{t-1},\ldots,k_1,k_0)_2$, $P\in E(\mathbb{F}_q)$.
OUTPUT: kP.

1. *Precomputation.* Compute $[a_{w-1},\ldots,a_1,a_0]P$ for all bit strings $(a_{w-1},\ldots,a_1,a_0)$ of length w.
2. By padding k on the left with 0s if necessary, write $k=K^{w-1}\|\cdots\|K^1\|K^0$, where each K^j is a bit string of length d. Let K_i^j denote the ith bit of K^j.
3. $Q\leftarrow\infty$.
4. For i from $d-1$ downto 0 do
   4.1 $Q\leftarrow 2Q$.
   4.2 $Q\leftarrow Q+[K_i^{w-1},\ldots,K_i^1,K_i^0]P$.
5. Return(Q).

[그림 3] Comb 스칼라 곱셈 알고리즘 [4]

3. 타원곡선 암호의 부채널 대응 동향

본 절에서는 타원곡선 암호의 부채널 대응 동향에 대해 분석한다. 대표적인 부채널 공격인 전력 분석 관점(SPA)에서 대응방안 연구 동향을 분석한다. [5-6]

3.1. Atomic Block

단순 구현은 스칼라 비트에 따라 ECADD와 ECDBL 연산 수행량이 다르기 때문에 부채널 분석에 취약하다. 이를 대응하기 위한 간단한 예로 스칼라 비트에 의존하지 않고 모든 비트마다 ECADD, ECDBL 연산을 수행하는 것이다. 하지만 이는 성능 부하를 발생시키는 단점이 존재한다. Atomic Block은 ECADD와 ECDBL 연산의 내부를 부채널 동치관계인 Atomic Block들로 구성하여, 공격자의 전력분석 공격을 어렵게 하는 것이다. [5]에서는 ECADD와 ECDBL을 구성하기 위해 +,-,+,*를 하나의 Atomic Block으로 만들었다. 이를 통해 10개의 Atomic Block으로 ECDBL 연산, 16개의 Atomic Block으로 ECADD 연산을 구성하였다. ECADD와 ECDBL을 모두 부채널 동치관계인 Atomic Block으로 구성하기 위해 일부 Fake 연산도 연산 구성 시 포함된다. Atomic Block의 전체 과정은 [그림 4]와 같다.

Input: $\boldsymbol{P_1} = (X_1, Y_1, Z_1), d = (1, d_{m-2}, \ldots, d_0)_2$, and matrix $(u^*_{k,l})$ as above
Output: $\boldsymbol{P_d} = d\boldsymbol{P_1}$

$R_0 \leftarrow a\,;\, R_1 \leftarrow X_1\,;\, R_2 \leftarrow Y_1\,;\, R_3 \leftarrow Z_1\,;\, R_7 \leftarrow X_1\,;\, R_8 \leftarrow Y_1\,;\, R_9 \leftarrow Z_1$
$i \leftarrow m-2\,;\, s \leftarrow 1$
while $(i \geq 0)$ **do**
  $k \leftarrow (\neg s) \cdot (k+1)$
  $s \leftarrow d_i \cdot (k \text{ div } 25) + (\neg d_i) \cdot (k \text{ div } 9)$
  $R_{u^*_{k,0}} \leftarrow R_{u^*_{k,1}} \cdot R_{u^*_{k,2}}\,;\, R_{u^*_{k,3}} \leftarrow R_{u^*_{k,4}} + R_{u^*_{k,5}}\,;\, R_{u^*_{k,6}} \leftarrow -R_{u^*_{k,6}}\,;\, R_{u^*_{k,7}} \leftarrow R_{u^*_{k,8}} + R_{u^*_{k,9}}$
  $i \leftarrow i - s$
endwhile
return (R_1, R_2, R_3)

[그림 4] Atomic Block 스칼라 곱셈 알고리즘[5]

3.2. Improved Montgomery

Improved Montgomery 방법은 기존 SPA 대응방법인 Montgomery Ladder 보다 향상된 대응방법이다. [6]에서는 Projective 기반 Montgomery Ladder 방법에서 X, Z 좌표만을 Montgomery Ladder 방법을 수행하고 Y 좌표는 마지막에 한 번에 복구하는 방법이다. 이를 통해 Y 좌표에 대한 계산을 효율적으로 제거하였다. X, Z 좌표만으로 Montgomery Ladder를 계산 시 xECADDDBL 함수를 통해 ECADD와 ECDBL을 동시에 계산하며, Montgomery Ladder와 마찬가지로 부채널 대응을 위해 Constant-time 구현으로 이루어져 있다. xECADDDBL 함수를 통해 스칼라 비트를 스캔하며, X, Z 좌표에 대해서만 계산 후, Y 좌표는 마지막 YRecovery 함수를 통해 구할 수 있다.

4. 벤치마킹

본 절에서는 타원곡선 P-256에 대해 최적화 및 부채널 대응방안 연구의 성능을 비교한다. 환경은 Intel Core i7-9700K이며, Visual Studio Release x64에서 측정하였다. 성능은 CPU cycles로 측정되었으며, 총 10,000번의 평균 결과이다. 성능 측정 결과는 [표 1]과 같다. Binary 구현 대비 상승치는 Jacobian 좌표계와 비교한 결과이다. 최적화 구현 기법에서는 Comb 스칼라 곱셈 방법이 가장 빠른 성능을 달성하였으며, 부채널 대응방안에서는 Atomic Block 방법이 가장 빠른 성능을 달성하였다.

구현 방법	성능 (Cycles)	Binary 구현 대비 상승치
Jacobian 좌표계[4]	2,415,602	-
wNAF(w=4) [4]	2,068,947	16.8 ↑
Comb(w=4) [4]	529,259	365.4 ↑
Atomic Block [5]	2,852,734	18.1 ↓
Improved Montgomery [6]	3,279,388	35.8 ↓

[표 1] 성능 측정 결과

4. 결론 및 향후 계획

본 논문은 타원곡선 암호에 대한 최적화 및 부채널 대응 기법에 대한 동향을 분석하고, 이를 벤치마킹하여 실제 성능 측정 결과를 제시하였다. 최적화에서는 Jacobian Projective, Window NAF, Comb 방법을 분석하였으며, 부채널 대응기법으로는 Atomic Block, Improved Montgomery 방법을 분석하였다. 향후 계획으로는 차세대 곡선인 Curve25519, Curve448에 대해서도 최적화 및 부채널 대응에 대한 연구 동향을 분석할 계획이다.

참 고 문 헌

[1] Huiyun Li, Keke Wu, Guoqing Xu, Hai Yuan, Peng Luo, “Simple power analysis attacks using chosen message against ECC hardware implementations”, World Congress on Internet Security (WorldCIS), 2011

[2] Toru Akishita, Tsuyoshi Takagi, “Power Analysis to ECC Using Differential Power Between Multiplication and Squaring”, CARDIS, 2006

[3] Jörn-Marc Schmidt, Marcel Medwed, “A Fault Attack on ECDSA”, Workshop on Fault Diagnosis and Tolerance in Cryptography (FDTC), 2009

[4] Darrel Hankerson, Alfred Menezes, Scott Vanstone, “Guide to Elliptic Curve Cryptography”

[5] B. Chevallier-Mames, M. Ciet, and M. Joye, “Low-Cost Solutions for Preventing Simple Side-Channel Analysis: Side-Channel Atomicity”, IEEE Transactions on Computers, Volume 53, 2004

[6] T. Izu, B.Möller, and T.Takagi, “Improved Elliptic Curve Multiplication Methods Resistant against Side Channel Attacks”, Indocrypto 2002, India, December, 16-18