타원곡선암호 및 구현 방법에 대한 연구[1)]

김정식°, 강부중 , 노인우, 임을규
한양대학교
bisa1004@hanmail.net°, deviri@hanyang.ac.kr, inwoo13@hotmail.com, imeg@hanyang.ac.kr

The study of Elliptic Curve Cryptography and Implementation

Jung Sik Kim°, Boo Joong Kang, In Woo Ro, Eul Gyu Im
Hanyang University

요 약

암호는 컴퓨터의 보안을 위해서 사용되는 중요한 기법중 하나이다. 암호 시스템은 많은 연구를 통해 연구되고 발전되었는데, 키를 사용하는 암호 알고리즘은 대칭키 기법과 비대칭키 기법으로 나눌 수 있다. 대칭키 기법은 비대칭키 기법에 비해 좋은 성능을 보여주지만 키 교환의 문제로 인해, 비대칭키 기법과 병행하여 사용하는 경우가 많다. 하지만 비대칭키 기법은 성능 면에서 부담이 되기 때문에 센서 네트워크와 같이 적은 자원을 가지는 네트워크 환경에서 사용하기 힘들다. 하지만 1985년 타원곡선암호의 발표로 비대칭키 기법의 성능은 매우 향상되게 되었다. 본 논문에서는 타원곡선암호에 대해 알아보고, 타원곡선암호의 구현에 관련된 연구에 대해 알아본다.

1. 서론

보안은 정보의 중요성이 인식된 이후 꾸준히 연구되고 있는 분야이다. 최근에는 컴퓨터에 대한 의존도가 높아지면서 데이터 보안의 중요성이 날로 늘어나고 있고, 이에 따라 요구되는 안전도도 늘어나고 있다. 이런 상황에서 요구되는 안전도를 만족시켜주기 위한 방법 중 하나가 암호이다. 암호는 고대 때부터 사용해오던 방법이지만, 현재에는 수학의 발전과 함께 강력한 보안 방법의 하나로 사용되고 있다.

현대에서 사용되는 컴퓨터 암호는 고전 암호에서 사용하던 기법들을 수학적 문제를 통해 적용되는 경우가 많은데, 일반적으로 암호문을 만들거나 해석하는데 '키' 가 필요한 경우가 많다. 키를 사용하는 암호화 방법은 키의 종류에 따라 대칭키(Symmetric key) 알고리즘과 비대칭키(Asymmetric key) 알고리즘으로 구분할 수 있다. 대칭키 알고리즘은 암호를 주고받는 양측이 "공유키" 라 불리 우는 키를 미리 알고 있어야 한다는 조건이 존재한다. 이에 반해 비대칭키 알고리즘은 "공개키" 와 "비밀키" 라 불리 우는 두 개의 키를 가지고 암호화를 수행하게 된다.

대칭키 알고리즘은 비대칭키 알고리즘에 비해 연산의 속도가 빠르고, 키 길이 당 안전도가 높다는 것을 장점으로 하고 있다. 예로 brute-force 공격에 대해 80bits의 대칭키 길이는 768 bits의 비대칭키 길이와 비슷한 수준의 안전성을 보여주게 된다.[1] 이러한 이유로 일반적인 데이터의 암호는 대칭키 알고리즘을 사용하여 이루어지게 된다. 하지만 대칭키 알고리즘의 가장 큰 문제는 공유키를 미리 교환해야 한다는 점인데 현재까지 이 문제를 완벽히 해결하지는 못한 상황이다.

비대칭키 알고리즘은 연산 속도와 키 길이 당 안전도가 대칭키 알고리즘에 비해 떨어지지만 수학적 방법을 통해 안전하게 키 교환을 할 수 있다는 장점이 있다. 그래서 비대칭키 알고리즘은 대칭키에서 사용되는 공유키를 생성하기 위해 종종 사용되고 있다.

하지만 유비쿼터스 컴퓨팅에 대한 관심이 증가하면서 센서 네트워크, 애드-혹 네트워크 등 저성능 기기를 사용하는 네트워크의 보안이 중요해지면서, 기존에 사용하는 알고리즘을 사용하기 힘든 경우가 발생하게 되었다. 비대칭키 알고리즘도 이 경우에 속하는데, 많은 연산 능력을 요구하기 때문에 자원적인 면에서 문제가 발생하게 되었다.

이에 따라 더 효율적인 비대칭키 알고리즘이 필요하게 되었고, 1985년 Kiblitz와 Miller가 발표한 타원곡선을 이용한 타원곡선암호에 대해 관심을 가지게 되었다. 타원곡선암호는 기존의 비대칭키 알고리즘에 비해 수배 이상의 속도를 보여주는데, 실제 서명 알고리즘을 구현한 결과 3~5배정도의 속도차이를 보여주었다.[2]

본 논문에서는 이 타원곡선암호가 무엇인지에 대해 알아보고, 그 구현 방법에 대한 조사를 하였다. 그리고 효율성을 측정하기 위한 접근방법에 대해 XXXX(고민? 언급?...) 해보았다. 2장에서는 타원곡선의 수학적 배경과 타원곡선 암호에 대해 언급하였고, 3장에서는 ...

1) 본 연구는 한국과학재단 특정기초연구 (R01-2006-000-11196-0) 지원으로 수행되었음.

2. 타원곡선암호

2.1 개요

타원곡선암호(ECC, Elliptic Curve Cryptography)는 타원곡선의 이산로그문제의 어려움에 기초한 암호 알고리즘이다. 1985년 Koblitz와 Miller에 의해 각각 발표된 이후, 타원곡선은 암호분야에서 점차 관심을 받고 있다.

타원곡선암호가 관심을 받는 이유 중 하나는 타원곡선 암호의 효율성과 안전성에 때문이다. 일반적으로 많이 사용되는 공개키 암호 알고리즘인 RSA, DSA등은 유한체의 이산로그문제나 합성수의 인수분해 문제의 어려움에 기반을 하는 알고리즘이다. 그런데 유한체의 이산로그 문제와 합성수의 인수분해 문제는 서로 다른 문제임에도 불구하고, 이들의 해결방식이 매우 비슷하기 때문에 어느 한쪽을 풀 수 있는 알고리즘은 다른 쪽에 적용 가능한 경우가 많다. 이에 비해 타원곡선 암호의 타원곡선의 이산로그 문제는 아직 효과적인 공격방법이 발견되지 않았으며, 다른 문제들의 해법을 적용하기 어렵다고 알려져 있다.

타원곡선암호에 대한 효과적인 공격방법이 없기 때문에 타원곡선의 이산로그 문제는 다른 문제에 비해 어렵다고 생각된다. 따라서 타원곡선암호는 다른 공개키 암호 알고리즘에 비해 훨씬 짧은 길이의 키를 사용하여 같은 안전도를 보장할 수 있게 된다. [표 1]은 [3]에서 제시된 자료로 알고리즘에 따라 같은 안전도를 가지는 키 길이를 비교한 것이다.

Time to break (MIPS years)	RSA key size (bits)	ECC key size (bits)	RSA/ECC key size ratio
10^4	512	106	5 : 1
10^8	768	132	6 : 1
10^{11}	1024	160	7 : 1
10^{20}	2048	210	10 : 1
10^{78}	21000	600	35 : 1

[표 1] 안전도에 따른 키 길이의 비교

[표 1]를 보면 키 길이 당 안전도가 타원곡선암호가 크게 높은 것을 볼 수 있다. 그렇기 때문에 타원곡선암호는 상대적으로 작은 키 길이를 사용하여도 같은 안전도를 보장해 줄 수 있기 때문에 RSA, DSA 알고리즘에 비해 빠른 속도로 동작할 수 있는 것이다.

2.2 타원곡선의 수학적 배경

타원곡선이란 무한원점(point at infinity)이라고 불리는 점 O와 체 K위에서 정의되고 discriminant가 0이 아닌 Weierstrass 방정식을 만족하는 점들로 이루어진 집합이다. 여기서 체 K는 다음과 같이 정의할 수 있다.

$$E(K)= \left\{(x,y)\in K^2 \mid y^2+a_1xy+a_3y=x^3+a_2x^2+a_4x+a6\right\} \cup \{O\} \quad (1)$$

여기서 K는 실수, 유리수, 복소수, 유한체 등이 될 수 있다.

타원곡선은 체 위에서 정의되기 때문에, 사칙연산이 가능하며 항등원, 역원이 존재한다. 또한 G를 타원곡선 E의 원소라고 할 때, 타원곡선에서 kG와 같은 상수배의 곱셈은 유한체 또는 실수에서의 지수함수의 역할을 하게 된다. 즉, G는 밑(base)의 역할을 하게 되고, 상수 k배는 멱승의 역할을 한다. 따라서 타원곡선의 로그함수란 것은 kG와 G로부터 k를 구하는 것이다.

2.3 타원곡선을 이용한 암호시스템

타원곡선의 이산로그문제는 유한체의 이산로그문제와 비교하면 정의되어있는 유한군은 다르지만 본질적인 성질은 동일하다. 따라서 유한체의 이산로그문제를 이용한 암호시스템은 그대로 타원곡선의 이산로그문제를 이용한 암호시스템으로 변환할 수 있다.

유한체의 이산로그문제를 이용한 암호시스템인 Diffei-Hellman 키 교환 알고리즘과 DSA 알고리즘은 타원곡선을 이용한 ECDH, ECDSA로 변환된다. 다음은 두 암호시스템이 어떻게 변환되는지를 알아본다.

2.3.1 Diffie-Hellman 키 교환 알고리즘

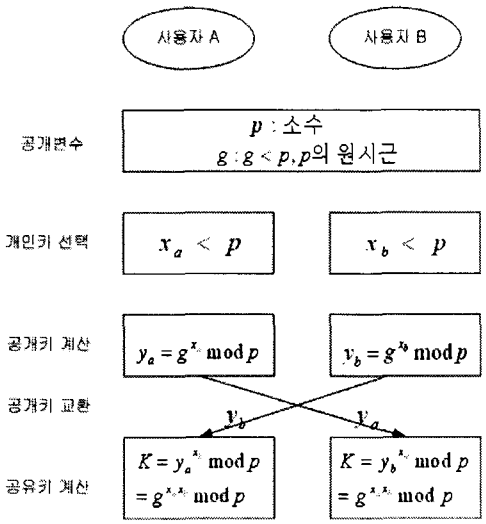
Diffie-Hellman 키 교환 알고리즘은 1976년 스탠퍼드 대학의 diffie와 hellman이 공동 개발한 방법으로, 공개키 기법에 기반을 한 키 교환 알고리즘이다.

알고리즘은 다음과 같이 동작한다. 사용자 A와 B가 존재하고 둘은 공개변수 p와 g를 알고 있다. 여기서 p는 소수이고, g는 p보다 작고 p의 원시근(primitive root 이다. 다음 A, B는 각각 자신들의 개인키 x_a, $x_b \in \{2, \ldots, p-1\}$를 선택하고, 공개키 $y_a = g^{x_a} \bmod p$, $y_b = g^{x_b} \bmod p$를 계산한다. 그리고 공개키를 서로 교환하면 A와 B는 상대방의 공개키와 자신의 개인키를 이용하여 공유키 $K = y_a^{x_b} \bmod p = y_b^{x_a} \bmod p = g^{x_a x_b} \bmod p$를 얻을 수 있다. [그림 1]은 이 과정을 그림으로 나타낸 것이다.

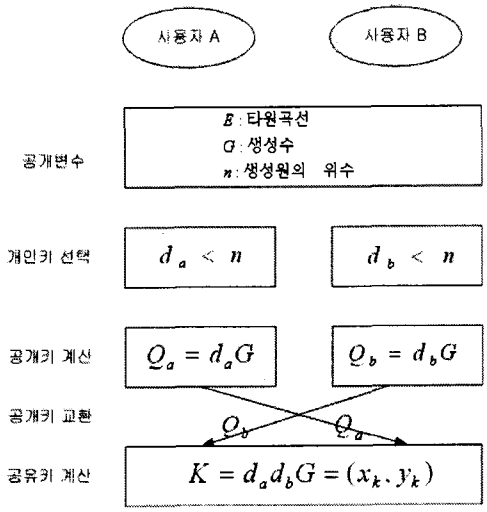
Diffie-Hellman 키 교환 알고리즘은 공개키는 공개되지만 공격자가 이를 알아도 유한체의 이산로그 문제를 해결하기 전까지는 안전하기 때문에 p가 충분히 크다면 개인키가 노출될 가능성은 거의 없다.

이 방법은 공개키가 공개되어 공격자가 이를 알 수 있지만 공유키로부터 개인키를 알아내기 위해서는 유한체의 이산로그 문제를 해결해야 하므로 p가 충분히 크다면 개인키가 노출될 가능성은 거의 없다. 하지만 알고리즘이 안전하기 위해서는 p가 커야하기 때문에 많은 계산능력을 필요하게 된다.

Diffie-Hellman 키 교환 알고리즘을 타원곡선의 이산로그문제를 사용한 것이 ECDH(Elliptic Curve Diffie-Hellman) 키 교환 알고리즘이다. 알고리즘의 동작과정은 기존 Diffie-Hellman 방식과 동일하지만 사용되는 변수와 계산방법이 타원곡선의 것을 사용하게 된



[그림 1] Diffie-Hellman 키 교환 알고리즘



[그림 2] Elliptic Curve Diffie-Hellman 키 교환 알고리즘

다.

ECDH에서는 두 사용자 A, B가 알고 있는 공개변수는 타원곡선 E와 생성원 G, 생성원의 위수 n이다. 공개변수를 통해서 A, B는 자신들의 개인키 d_a, $d_b \in \{2, ..., n-2\}$를 선택하여 공개키 $Q_a = d_a G$, $Q_b = d_b G$를 계산한다. 그리고 공개키를 서로 교환하여 상대방의 공개키와 자신의 개인키를 이용하여 공유키 $K = d_a d_b G = (x_k, y_k)$를 공유하게 된다. [그림 2]는 이 과정을 그림으로 나타낸 것이다.

2.3.2 DSA (Digital Signature Algorithm)

DSA(Digital Signature Algorithm)는 미국 연방이 표준으로 지정한 전자 서명을 위한 알고리즘이다. DSA도 Diffie-Hellman 키 교환 알고리즘과 같이 유한체의 이산로그문제에 기반을 한 알고리즘으로 서명을 위해서 공개변수 p와 g가 필요하고, p-1을 나누는 160비트의 소수 q가 필요하다. 사용자는 서명을 위한 키 x를 $1<x<q$의 정수사이에서 고르고, 검증을 위한 키 $y=g^x \bmod p$를 계산하여 공개키로 공개한다.

이렇게 선택된 서명키와 공개키를 이용하여 메시지를 서명할 경우에는 $0<k<q$의 범위에서 난수 k를 고르고, 다음의 계산 과정을 수행한다.

$$r = (g^k \bmod p) \bmod q$$
$$s = (k^{-1}(Hash(M) + xr)) \bmod q \quad (2)$$

서명된 메시지의 검증은 다음과 같은 계산 과정을 통해 이루어진다.

$$u_1 = (Hash(M)s^{-1}) \bmod q$$
$$u_2 = rs^{-1} \bmod q \quad (3)$$
$$r = (g^{u1} y^{u2} \bmod p) \bmod q$$

타원곡선을 사용한 DSA인 ECDSA(Elliptic Curve Digital Signature Algorithm)의 공개 변수는 타원곡선 E와 생성원 G 그리고 q이다. 공개 변수로부터 서명키 d, $0<d<q$를 고르고, 검증키 Q=dG를 계산하여 다음의 계산과정을 통해 서명을 한다.

$$(x_1, y_1) = kG, where\ k \in [1, n)$$
$$r = x_1 \bmod n \quad (4)$$
$$s = (k^{-1}(Hash(M) + dr)) \bmod n$$

서명의 검증은 다음의 계산 과정을 통해 이루어진다.

$$u_1 = (Hash(M)s^{-1}) \bmod n$$
$$u_2 = rs^{-1} \bmod n$$
$$(x_1, y_1) = u_1 G + u_2 Q \quad (5)$$
$$r = x_1 \bmod n$$

3. 타원곡선암호의 구현

타원곡선암호가 발표되었을 때, 안전성에 대한 신뢰도의 문제로 당시에는 널리 사용되지 못하였다. 하지만 이후 안전성에 대한 분석이 충분히 이루어진 이후 사용범위가 확대되었고, 타원곡선을 이용한 서명 알고리즘의 경우 국내 및 해외에서 표준화 작업까지 이루어 졌다.[5][6]

본 장에서는 센서 네트워크와 하드웨어 상에서 타원곡선암호를 구현한 연구에 대해 내용을 알아보도록 한다.

3.1 센서 네트워크에서의 구현

현재 구현되어 사용되고 있는 타원곡선암호는 자원적인 여유가 있는 환경에서 사용되고 있다. 하지만 아직도 타원곡선암호를 적용할 수 없는 환경도 존재하는데 센서 네트워크는 그 중 하나에 속한다.

센서 네트워크는 여러 종류의 네트워크 중에서도 가장

	DH 1024bits (Modular exponentiation)	ECC-DH 163bits (Public key)
Time(secs)	54.1144	34.161
Clock(cycles)	$3.9897X10^8$	$2.512X10^8$
Energy(joules)	1.185	0.816

[표 2] DH vs. ECC-DH

작은 자원을 가지고 있는 환경의 네트워크이다. 하지만 대부분의 센서 네트워크는 무선통신을 기반으로 하기 때문에 보안의 중요성이 높은데, 현실적으로 자원의 한계로 인하여 공개키 기반의 암호 알고리즘을 사용하지 못한다. 그래서 센서 네트워크에서는 대안적인 키 교환 방법이 연구되고 있는데, 공개키 기반 알고리즘 수준의 안전성을 보장해주지 못한다. 이에 Malan et al.은 센서 네트워크에서 타원곡선암호를 사용하여 Diffie-Hellman 키 교환 알고리즘은 구현하고 분석하였다.[4]

구현에는 버클리 대학에서 개발된 센서 노드인 MICA2 Mote를 사용하였고, 운영체제는 TinyOS를 선택하였다. 실험은 구현된 성능을 비교하기 위해서 기존의 Diffie-Hellman 키 교환 알고리즘의 성능을 측정한 뒤, 타원곡선암호를 적용하여 성능을 측정하였다. [표 2]는 측정 결과이다.

[표 2]를 보면 타원곡선을 이용하지 않은 Diffie-Hellman 알고리즘은 전체 연산중 단 1회의 모듈러 연산을 계산하는데 소비된 값이고, 타원곡선은 하나의 공개키를 완전하게 계산하는데 소비된 값이다. 두 값을 비교해보면 타원곡선을 이용하지 않았을 경우에는 센서 네트워크 환경에서 키 생성 자체가 불가능한 결과를 나타낸다는 것을 알 수 있다. 하지만 타원곡선암호를 적용하여도 센서 노드의 대부분의 전력을 소비하여 키를 생성한 것이기 때문에 이 실험에서는 163 bits 보다 큰 키를 생성하는 것은 실패하였다.

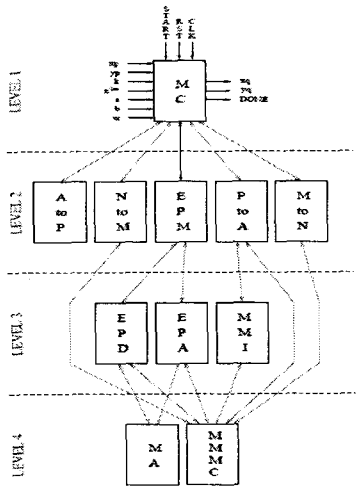
3.2 하드웨어 상에서 타원곡선암호의 구현

타원곡선암호는 소프트웨어뿐만 아니라 하드웨어로도 구현되는 시도가 많이 이루어졌다. 이는 하드웨어로 구현할 경우, 소프트웨어로 구현된 경우보다 매우 빠른 속도를 보여주기 때문이다.

Mentens et al.도 GF(2^m)상에서 타원곡선암호를 구현하였는데, 구현을 위해 FPGA를 이용하였다.[7] 하드웨어 구현은 point addition과 doubling을 위한 공식은 Lopez와 Dahab의 방법을 사용하였으며, kG를 구하는 공식은 빠른 속도를 보여주는 Montgomery multiplication 방법을 사용하였다.[8][9]

[그림 3]은 구현된 하드웨어의 구조이며, 블록의 역할은 다음과 같다.

- Level 1 :
  MC : Main Controller
- Level 2 :



[그림 3] 블록별 회로 구조도

w	1	4	8
Clock Period(ns)	19.956	19.977	20.923
Latency of ECP(ms)	9.652	7.249	3.801

[표 3] 160 비트 키의 생성 결과

AtoP : affine to projective converter
NtoM : normal to Montgomery converter
EPM : EC point multiplier
PtoA : projective to affine converter
MtoN : Montgomery to normal converter

- Level 3 :
  EPD : EC point doubling circuit
  EPA : EC point addition circuit
  MMI : modular multiplicative inverter
- Level 4 :
  MA : modular addition
  MMMC : Montgomery modular multiplication circuit

[표 3]은 구현된 하드웨어를 이용하여 160비트의 키를 생성할 때의 결과이다.

w는 워드의 길이를 말하는데, 워드의 길이가 8일 때 47MHz상에서 3.801초가 걸린다고 한다.

3.3 하드웨어의 효율성에 관한 토의

타원곡선암호의 구현방법에 대해서는 많은 연구가 진행되어 있는 상황인데, 아직 센서 네트워크와 같이 한정된 자원을 가진 환경에서는 사용하기 힘들다는 것을 알 수 있다.

하드웨어로의 구현을 통해 더 빠른 속도의 타원곡선모듈을 사용하게 되면 이를 해결할 수 있겠지만, 실용화를

위해서는 노드의 가격적인 문제가 발생하게 된다. 그렇기 때문에 최소한의 하드웨어를 추가함으로 향상되는 효율성을 측정하게 된다면, 가격과 속도향상 두 가지 모두 좋은 결과를 기대할 수 있을 것이다. 이를 위해 소프트웨어 모듈과 하드웨어 모듈의 성능차이를 세부적으로 분석하는 과정이 요구된다.

4. 결론

타원곡선암호는 타원곡선의 이산로그문제에 기반을 하는 공개키 기반의 암호시스템으로, 기존 공개키 기반의 암호 알고리즘에 비해 빠른 속도를 보여주고 있다. 본 논문에서는 타원곡선암호가 무엇인지에 대해 조사해 보았고, 구현에 관한 몇 가지 연구결과를 조사해 보았다.

현재 타원곡선암호는 그 효용성으로 여러 분야에서 사용되고 있고 그 사용범위가 점차 증가하고 있다. 하지만 센서 네트워크 등 제한된 자원을 사용하는 네트워크의 경우에는 적용하기 어렵기 때문에, 이에 대한 연구는 꾸준히 진행되어야 한다.

5. 참고문헌

[1] B. Schneier, "Applied Cryptography", John Wiley & Sons, 1996.

[2] 이동훈, 황효선, 임채훈, "타원곡선 암호의 기초와 응용", Technical Report, (주)퓨처시스템, Aug. 2001.

[3] Randall K. Nichols, "ICSA Guide to Cryptography", McGraw-Hill, Dec. 1999.

[4] David J. Malan, Matt Welsh, Michael D. Smith, "A Public-key Infrastructure for Key Distribution in TinyOS Based on Elliptic Curve Cryptography", First IEEE International Conference on Sensor and Ad-hoc Communications and Network, Oct. 2004

[5] ANSI X.962, "Public key cryptograpgy for the financial services industry : The elliptic curve digital signature algorithm(ECDSA)", X.962-1998, Jan. 1999.

[6] 한국정보통신 기술협회, "부가형 전자서명 방식 표준 - 제 3부: 타원곡선을 이용한 인증서 기반 전자서명 알고리즘", TTAS.KO-12.0015, Dec. 2001.

[7] Nele Mentens, Siddika Berna Ors, Bart Prenell, "An FPGA Implementation of an Elliptic Curve Processor over GF(2^m)", ACM GLSVLSI'04, Apr. 2004.

[8] J. L´opez and R. Dahab, "Improved algorithms for elliptic curve arithmetic in GF(2n)", Technical Report IC-98-39, Institute of Computing, Oct. 1998.

[9] P. Montgomery, "Modular multiplication without trial division", Mathematics of Computation, Vol.44:519-521, 1985.