타원 곡선위에서의 ElGamal 암호 기법과 Schnorr 디지탈 서명 기법의 구현*

이은정, 최영주
포항공과대학교 수학과

Implementation of ElGamal Cryptosystem and Schnorr Digital Signature Scheme on Elliptic Curves

EunJeong Lee, YoungJu Choie
Department of Mathematics, POSTECH, Pohang.

요 약

Diffie-Hellman 의 공개 키 암호 프로토콜이 제안된 이후 이산 대수 문제의 어려움이 프로토콜의 안전도와 깊이 연관되었다. 유한체를 이용한 암호 기법을 ElGamal 이 세웠으나, Index-Calculus 알고리듬에 의해 유한체위에서 이산 대수 문제가 subexponential 알고리듬이되어 ElGamal 기법의 안전도가 약해졌다. Nonsupersingular 타원 곡선을 선택하여 유한체대신 ElGamal 암호 기법에 적용하면 안전한 암호 시스템을 설계할 수 있다. 이 논문에서는 콤퓨터 구현시 용이한 nonsupersingular 타원 곡선을 선택하는 방법, 유한체위에서의 연산, 평문을 타원 곡선의 원소로 임베드(Imbedding) 하는 방법 등 타원 곡선을 암호 시스템에 적용하기 어려운 점들에 대한 해결 방법을 소개하고, 실제로 콤퓨터로 구현하여 그 실행 결과와 ElGamal 기법을 개선한 Schnorr 기법을 실행한 결과를 밝혔다.

1 서론

1976년 Diffie와 Hellman은 공개된 채널을 이용하여 공용 키를 생성하는 공개 키 암호 프로토콜을 제안하였다 [3]. 이 프로토콜의 안전도는 유한체에서 이산 대수 문제를 풀기가 어렵다는 것에 기초한다. 1985년 ElGamal은 이산 대수 문제에 바탕을 둔 공개 키 암호 알고리듬과 디지탈 서명 방법을 세웠다 [4]. 이 프로 토콜들은 모든 유한 cyclic 군에 적용될 수 있다.

q 개의 원소를 갖는 유한체 F_q위에서 정의된 타원곡선 $E(F_q)$ 는 가환군을 이룬다. 이 가환군의 연산은 유한체위에서의 연산들로 이루어졌고 하드웨어나 소프트웨어로 구현하기가 용이하다. 따라서, 가환 군 $E(F_q)$ 는 Diffie 와 Hellman 의 공개 키 암호 프로토콜이나 ElGamal 의 프로토콜을 구현하는데 사용될 수 있다. Koblitz [6] 와 Miller [11] 가 타원 곡선을 이용한 암호 시스템을 처음 제안하였다.

타원 곡선에서 이산 대수 문제를 풀기위해 적용될 수 있는 알고리듬은 Pohlig-Hellman 과 Shank 의 알고리듬이다 [7]. 두 알고리듬은 큰 솟수가 군의 order 를 나눌때 풀기 어려워지며, 실질적으로 군의 order 가 약 30자리 수 이상인 소인수를 가질 경우 두 알고리듬에 대해 안전하다고 알려져 있다 [8].

한편, 유한체에서 이산 대수 문제는 Index-Calculus 알고리듬을 사용하면 subexponential time 하에 풀린다. Menezes, Okamoto, Vanstone 은 Weil Pairing 을 이용하여 타원 곡선위의 이산 대

*이 논문은 1994년도 산업 과학 기술 연구소(RIST)의 순수 기초 연구비(R94004), 기초 과학 연구소(N94123)의 지원으로 연구되었음.

수 문제를 유한체에서 이산 대수 문제로 전환 시키는 알고리듬을 제안하였다 [12]. 타원 곡선의 한 점 $P \in E(F_q)$ 로 generate 된 cyclic 군 $< P >$ 의 order 를 m 이라 할 때

$$E[m] = \{R \in E \mid mR = O\} \subset E(F_{q^k})$$

이면 $E(F_q)$ 에서 이산 대수 문제는 유한체 F_{q^k} 위에서의 이산 대수 문제로 옮겨질 수 있다. Supersingular 타원 곡선은 $k \leq 6$ 이므로 F_q 가 큰 유한체이어야 F_{q^k} 에서 이산 대수 문제를 풀기 어렵다 [12]. 또한, $q = 2^n$ 일 때, Coppersmith 알고리듬 [2]에 의해 n 은 이론상으로는 1280이상, 실제로는 600 이상이 되어야만 유한체위에서 Diffie-Hellman 형의 암호 시스템이 안전하다. 따라서, supersingular 타원 곡선 $E(F_{2^n})$을 이용하여 Diffie-Hellman형의 암호 시스템 설계시 $n > 200$ 이상이어야 안전하다고 할 수있다.

그러나, nonsupersingular 타원 곡선 $E(F_q)$ 일때 $k > (\log q)^2$ 일 확률이 높아 타원 곡선 $E(F_q)$ 위에서의 이산 대수 문제를 유한체 F_{q^k} 위에서의 문제로 옮긴 후 Index-Calculus 알고리듬을 사용한다 할지라도 subexponential time 하에 이산 대수 문제를 풀기 어렵다. 그 이유는 Index-Calculus 알고리듬은

$$O(exp(c(\log q^k)^{\frac{1}{3}}(\log\log q^k)^{\frac{2}{3}})$$

의 running time [9] 을 갖는데 $k > (\log q)^2$ 이면 이 알고리듬의 running time은 exponential time 이 되기 때문이다. 따라서, nonsupersingular 타원 곡선 $E(F_q)$의 사용시 q의 크기는 작으면서 안전한 Diffie-Hellman형의 암호 시스템 설계가 가능하다.

이 논문에서는 먼저 타원 곡선에 대해 간단히 설명하고 유한체 F_{2^n} 위에서의 연산을 소개하였다. 이를 이용하여 nonsupersingular 타원 곡선을 사용한 ElGamal 암호 시스템을 구현하는 과정을 설명하였고 실제 예로 프로그램 실행 결과를 보였다. 또한, [1] 에서 사용한 다항식 기저와 변형된 다항식 기저를 이용하여 타원 곡선을 이용한 ElGamal 암호 기법과 Schnorr 기법을 구현하였다.

2 유한체위에서의 타원 곡선

$\mathbb{K} = F_q, q = p^r$ (p 는 소수, r 은 양의 정수)를 q 개의 원소를 갖는 유한체라 하자. $\mathbb{K}$위에서 정의된 타원 곡선은

$$\begin{aligned} E(\mathbb{K}) &= \{(x,y) \in \mathbb{K} \times \mathbb{K} \mid y^2 + a_1xy + a_3y = x^3 + a_2x^2 + a_4x + a_6, &(1)\\ &\qquad a_1, a_3, a_2, a_4, a_6 \in \mathbb{K}\} \cup \{O\} &(2)\end{aligned}$$

이다. 여기서 O 는 무한대 점으로 가환 군 $E(\mathbb{K})$ 의 항등원(identity)에 해당한다 [10, 15].

특히, $q = 2^r$ 일때, 즉 $\mathbb{K} = F_q$ 의 characteristic 이 2 일때 (1) 은 다음과 같은 두 식으로 간단히 표현된다 [10, 15].

$$y^2 + a_3y = x^3 + a_4x + a_6, \; a_3, \; a_4, \; a_6 \in F_q, \quad a_3 \neq 0 \qquad (3)$$

$$y^2 + xy = x^3 + a_2x^2 + a_6, \; a_4, \; a_6 \in F_q, \quad a_6 \neq 0 \qquad (4)$$

(3) 은 supersingular 타원 곡선이고 (4) 는 nonsupersingular 타원 곡선이다. 각각의 경우 자세한 성질은 [15] 를 참고 하기 바란다. $P = (x_1, y_1)$ 와 $Q = (x_2, y_2)$ 를 $E(F_q)$ 위의 점이라 할때 곡선위의 더하기 $P + Q = (x_3, y_3)$ 는 다음과 같이 정의된다[10].

더하기 공식 (1) - supersingular case

$$x_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + x_1 + x_2, & P \neq Q\text{일때}, \\ \left(\frac{x_1^2+a_4}{a_3}\right)^2, & P = Q\text{일때}, \end{cases}$$

$$y_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1 + x_3) + y_1 + a_3, & P \neq Q\text{일때}, \\ \left(\frac{x_1^2+a_4}{a_3}\right)(x_1 + x_3) + y_1 + a_3, & P = Q\text{일때}, \end{cases}$$

더하기 공식 (2) - nonsupersingular case

$$x_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)^2 + \frac{y_1+y_2}{x_1+x_2} + x_1 + x_2 + a_2, & P \neq Q\text{일때}, \\ x_1^2 + \frac{a_6}{x_1^2}, & P = Q\text{일때}, \end{cases}$$

$$y_3 = \begin{cases} \left(\frac{y_1+y_2}{x_1+x_2}\right)(x_1 + x_3) + x_3 + y_1, & P \neq Q\text{일때}, \\ x_1^2 + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3, & P = Q\text{일때}, \end{cases}$$

[표 1] 에는 타원 곡선에서 더하기 연산을 할때 유한체 위에서의 연산이 몇 번 필요한가를 적어 놓았다.

위 덧셈 연산하에 타원 곡선 E(𝕂)이 가환 군이 된다는 것은 잘 알려진 사실이다. Hasse에 의해 이 군의 order는

$$|\, E(F_q) \,| = q + 1 - t, \quad |\, t \,| \leq 2\sqrt{q} \tag{5}$$

를 만족함이 증명되었다[10, 15].

또한, $E(F_{q^n})$ 의 order $|\, E(F_{q^n}) \,|$ 는 Weil conjecture (1934년에 Hasse 에 의해 증명되었음)을 이용하여

$$|\, E(F_{q^n}) \,| = q^n + 1 - \alpha^n - \beta^n, \tag{6}$$

$$1 - tT + qT^2 = (1 - \alpha T)(1 - \beta T) \tag{7}$$

임을 알 수 있다 [10, 15].

그러므로,

$$|\, E(F_{q^n}) \,| = q^n + 1 - a_n, \tag{8}$$

$$a_n = ta_{n-1} - qa_{n-2}, \; a_0 = 1, \; a_1 = t \tag{9}$$

처럼 Fibonacci 형태의 수열로 나타낼 수 있다.

[표 1] $P+Q$ 를 위한 유한체에서 연산 횟수

		곱셈	제곱	역수
supersingular	$P \neq Q$	2	1	1
	$P = Q$	2	2	1
nonsupersingular	$P \neq Q$	2	1	1
	$P = Q$	3	2	1

3 타원 곡선에서의 이산 대수 문제

타원 곡선 $E(F_q)$ 위에서 이산 대수 문제는 다음과 같이 정의된다.

주어진 두 점 $P, Q \in E(F_q)$ 에 대해 $Q = mP$ 를 만족하는 자연수 m 이 존재한다면 m 을 찾으라.

이 m 을 base P 로 하는 Q 의 이산 대수라 부르며, m 은 modular $Ord(P)$ 에 의해 유일하게 결정된다.

타원 곡선을 이용한 암호 시스템은 타원 곡선에서의 이산 대수 문제의 어려움으로 안전성을 얻게된다.

유한체에서 이산 대수 문제를 풀기위하여 여러 가지 알고리듬이 개발되었는데, 그 중 타원 곡선에 적용할 수 있는 것은 Shank 의 baby-step giant-step 알고리듬과 Pohlig-Hellman 알고리듬이 있다[7]. 이 두 알고리듬은 군의 order 를 나누는 가장 큰 소인수의 이중근에 비례하는 running time 을 갖는다. 유한체에서 이산 대수 문제를 subexponential time 에 풀 수 있는 Index-calculus 알고리듬은 타원 곡선에 적용할 수 없음을 Miller 가 보였으며[11], 일반적인 타원 곡선에서의 이산 대수 문제를 푸는 subexponential 알고리듬으로 알려진 것이 없다.

최근에 Weil pairing 을 이용하여 $E(F_q)$ 에서의 이산 대수 문제를 조금 더 커진 유한체 F_{q^k} 에서의 이산 대수 문제로 전환시키는 MOV 공격이 발견되었으며, 결론적으로 이 공격으로 인하여 $E(F_q)$ 에서 이산 대수 문제를 $k \leq (\log_2 q)^2$ 일때 subexponential time 에 풀 수 있다[12].

타원 곡선 $E(F_q)$ 가 supersingular일 때, 즉 (3)-type 일때, $k \leq 6$ 임이 알려졌다. 따라서, 타원 곡선 $E(F_q)$ 선택시 MOV 공격에 안전하기 위하여 F_{q^k} 에서 이산 대수 문제를 풀기 어려울 정도로(Index-calculus 알고리듬을 사용하여도) q 를 충분히 크게 선택해야 한다. $q = 2^n$ 일 때, 즉 F_q 의 characteristic 이 2 인 경우에 n 이 200 이상이어야 암호 시스템이 안전하다[14].

또한, $E(F_q)$ 를 임의의 nonsupersingular 타원 곡선으로 선택 할 때 $k > (\log_2 q)^2$ 일 확률은 매우 높다고 알려져 있다. 임의로 택한 타원 곡선을 이용한 암호 시스템이 MOV 공격에 안전한가를, 즉 $k > (\log_2 q)^2$ 인가를 다음과 같이 확인할 수 있다[10].

(4) 에서 a_2, a_6 를 F_q 에서 임의로 선택하여 타원 곡선 $E(F_q)$ 를 정했을 때, $E(F_q)$ 의 order 가 가장 큰 소인수 v 를 갖는다고 하자. Weil Pairing 을 사용하여 F_{q^k} 위에서의 이산 대수 문제로 전환될 수 있는 필요 조건은 v 가 $q^k - 1$ 을 나누는 것이다. 따라서, $1 \leq l \leq (\log_2 q)^2$ 인 모든 l 에 대하여 v 가 $q^l - 1$을 나누는지를 테스트한다. 만약, $1 \leq l \leq (\log_2 q)^2$ 인 모든 l 에 대하여 나누지 않

는다면 $k > (\log_2 q)^2$이므로 $E(F_q)$ 에서 이산 대수 문제를 풀기 위한 running time은 exponential time이 된다.

4 유한체 F_{2^n} 에서 연산

[표 1] 에서 타원 곡선위에서 한 번의 연산을 하기위해 유한체에서 연산을 여러 번 사용해야 함을 알았다. 따라서, 타원 곡선을 사용한 암호 시스템은 유한체를 사용한 암호 시스템보다 속도가 느리게 된다. 실용적인 암호 시스템을 위해서는 타원 곡선의 연산을 빠르게 할 수 있는 방법을 찾아야하며, 결국 유한체에서 연산의 속도에 달려있다.

유한체 F_{2^n} 는 F_2 위에서 차원이 n 인 벡터 공간이다. 유한체 F_{2^n} 의 원소의 표현 방법은 다항식 기저, 최적 정규 기저 그리고 변형된 다항식 기저등을 사용할 수 있으나[5], [1] 에서 컴퓨터로 구현한 결과 변형된 다항식 기저가 정규 기저에 비해 훨씬 효율적임을 밝혔다. 따라서, 이 논문에서는 Elgamal 암호 시스템 구현시 변형된 다항식 기저를 사용하였다.

변형된 다항식 기저를 간단히 소개한다.

변형된 다항식 기저 [1, 5]

F_{2^n} 은 차원(dimension) n 인 F_2 위에서의 벡터 공간(vector space)이다. $n = st$ 라고 할때, F_{2^n} 은 차원 t 인 F_{2^s} 위에서의 벡터 공간이고, F_{2^s} 는 차원 s 인 F_2 위에서의 벡터 공간이다. s 와 t 가 서로 소일때, F_2 위에서 차수가 t 인 기약 다항식(irreducible polynomial) $f(x)$은 F_{2^s} 위에서도 차수가 t 인 기약 다항식이다.

α 를 $f(x) = 0$ 의 근(root)이라 할 때, F_{2^n} 에 있는 원소 z 는

$$z = c_0 + c_1\alpha + \ldots + c_{t-1}\alpha^{t-1}, \quad c_i \in F_{2^s}$$

로 표현된다.

F_2 위에서 차수 s 를 갖는 원시적 다항식(primitive polynomial)을 $g(x)$ 라 하고, β 를 $g(x) = 0$ 의 근이라 하면 F_{2^s}의 한 원소 z 는

$$\begin{aligned} z &= \beta^i, \quad i = 0\ldots, 2^s - 1 && (10) \\ &= d_0 + d_1\beta + \ldots + d_{s-1}\beta^{s-1}, d_j \in \{0,1\} && (11) \end{aligned}$$

로 표현된다.

실제 계산을 위해서 다음과 같이 F_{2^s}의 원소를 표현하는 것이 좋다.

$$\begin{aligned} z &= x^i \ (mod\ g(x)), \quad i = 0\ldots, 2^s - 1 && (12) \\ &= d_0 + d_1x + \ldots + d_{s-1}x^{s-1}, \quad d_j \in 0, 1 && (13) \end{aligned}$$

즉, z 는 F_2 위에서 정의된 차수가 s 보다 작은 다항식이다.

F_{2^n} 에서 연산을 위해 s 를 작은 수로 선택하여 $1 \le i \le 2^{s-1}$ 에 대하여 두 테이블

$$\log[z] = i, \quad \text{antilog}[i] = z$$

을 먼저 만들어 놓는다. 이 계산은 (12) 에서 0 에서 $2^s - 1$ 까지의 i 에 대하여 x^i $(mod\ g(x))$ 를 계산하면 된다. antilog[i] 는 $x^i(modg(x))$ 를 계산한 것으로 차수가 s 보다 작은 다항식을 저장하게 된다. 여기서 계산된 다항식, 즉 z 는 컴퓨터 구현에서는 $z = d_{s-1}d_{s-2}\cdots d_1d_0$ 의 이진수로 표현할 수 있다. 따라서, z 는 2^s 보다 작은 정수가 되므로 log[z] = i 로 배열의 위치를 찾아 각 i 를 저장할 수 있다.

위의 테이블을 이용하여 F_{2^s} 에서 연산을 다음과 같이 한다.

$v, w \in F_{2^s}$ 에 대하여

$$
\begin{aligned}
v \cdot w &= x^i \cdot x^j\ (mod\ g(x)) \\
&= x^{i+j\ (mod\ 2^s-1)} \\
&= \text{antilog}[\log[v] + \log[w]\ (mod\ 2^s - 1)] \\
v^{-1} &= \text{antilog}[2^s - 1 - \log[v]]
\end{aligned}
$$

이다.

F_{2^n} 는 다항식 기저를 F_{2^s} 위에서 가지므로 F_{2^n}에서의 연산은 다항식 기저를 사용한 유한체에서의 연산 (곱셈, 더하기, 역수)을 한다[1].

5 타원 곡선위에서 ElGamal 과 Schnorr 기법

원래의 ElGamal 암호 시스템은 유한체 $\mathbb{Z}_p$ 를 이용하였고 여기에서는 $\mathbb{Z}_p$ 대신 타원 곡선 $E(F_q)$ 를 이용한다.

$\boxed{\text{타원 곡선을 이용한 ElGamal 암호 기법}}$ [13]

- (setup) 한 유한체 F_q 위에서 타원 곡선 $E(F_q)$ 와 $E(F_q)$ 위의 한 점 P 를 선택하여 공개한다. 각 사용자는 랜덤수 $s \in \{1, ..., N = Ord(P)\}$(비밀 키) 를 선택하고 $Q = sP$ 를 계산하여 Q 를 공개한다.

  사용자 A 가 메세지 M 을 사용자 B 에게 암호화 하여 보내기를 원한다고 가정하자.

- A 는 랜덤 수 $k \in \{1, ..., N\}$ 를 선택하여 $R = kP$ 를 계산한다. 메세지 M 을 $E(F_q)$ 위의 한 점 P_M 으로 끼워넣기(imbedding)한다.
- A 는 B 의 공개 키 $Q_B = s_B P$ 를 찾아 $(R, S = P_M + kQ_B)$ 를 B 에게 보낸다.
- B 는 $S - s_B R = P_M + kQ_B - s_B(kP) = P_M$ 으로 메세지 M 을 얻는다.

$\boxed{\text{타원 곡선을 이용한 ElGamal 디지탈 서명 기법}}$ [13]

- (setup) 한 유한체 F_q 위에서 타원 곡선 $E(F_q)$ 와 $E(F_q)$ 위의 한 점 P 를 선택하여 공개한다. $N = Ord(P)$ 도 공개한다.

각 사용자는 랜덤수 $s \in \{0, 1, ..., N-1\}$(비밀 키) 를 선택하고 $Q = sP$ 를 계산하여 Q 를 공개한다. f 와 g 를 각각 메세지와 타원 곡선에서 $\{0, 1, 2, 3, ..., N-1\}$ 으로 보내는 일대일 대응이라 놓자.

- **서명 생성** : 사용자 A 는 비밀 키 s_A 를 가지고 메세지 M 에 대한 서명을 생성하고자 한다.
  1. A 는 랜덤 수 $k \in \{0, 1, ..., N-1\}$ 를 $gcd(k, N) = 1$ 이도록 선택하고 $R = kP$ 를 계산한다.
  2. 아래의 방정식을 풀어 x 를 구한다.

$$f(M) = s_A g(R) + kx \ (mod\ N)$$

  3. M 에 대한 서명은 (R, x) 이다.

(메세지도 암호화하여 보낼 때엔 위의 타원 곡선을 이용한 ElGamal 암호 시스템에서 얻은 R 과 S 도 같이 전송한다.)

- **서명 확인** : 주어진 메세지 M 에 대한 서명 (R, x) 을 확인 하고자한다.
  1. xR 와 $g(R)Q_A$ 를 계산한다. ($Q_A = s_A P$ 는 A 의 공개 키이다.)
  2. $xR + g(R)Q_A$ 와 $f(M)P$ 를 계산하고 두 결과가 일치하는지 확인한다. 일치하면 서명이 옳다는 것이 확인된 것이다.

(암호화된 메세지를 받은 경우 먼저 메세지 M 을 위의 ElGamal 암호 시스템 에서 사용자 B 가 하는 작업을 한 후 복호화 된 M 으로 서명 확인을 한다.)

타원 곡선을 이용한 Schnorr 디지탈 서명 기법 [13]

- (setup) 한 유한체 F_q 위에서 타원 곡선 $E(F_q)$ 와 $E(F_q)$ 위의 한 점 P 를 선택하여 공개한다. 여기서, P 의 order $N = Ord(P)$ 도 계산한 후 공개한다. 각 사용자는 랜덤수 s(비밀 키) 를 선택하고 $Q = sP$ 를 계산하여 Q 를 공개한다. 메세지 M 과 $E(F_q)$ 위의 한 점을 input 으로 하고 t 보다 작은 정수를 output 으로 하는 해쉬 함수

$$h : \mathrm{M} \times E(F_q) \longrightarrow \{0, 1, ..., t\}$$

를 택하여 공개한다.

- **서명 생성** : 사용자 A 는 비밀 키 s_A 를 가지고 메세지 M 에 대한 서명을 하고자 한다.
  1. A 는 랜덤 수 $k \in \{0, 1, ..., N-1\}$ 를 선택하고 $R = kP$ 를 계산한다.
  2. $e = h(M, R)$ 을 계산한다.
  3. $x \equiv s_A e + k(mod\ \ N)$ 을 계산한다.
  4. M 에 대한 서명은 (e, x) 이다.

- **서명 확인** : 주어진 메세지 M 에 대한 서명 (e, x) 을 A가 서명하였는지 확인 하고자한다.
  1. xP 와 eQ_A 를 계산한다. ($Q_A = s_A P$ 는 A 의 공개 키이다.)
  2. $\bar{\mathrm{R}} = xP - eQ_A = xP - e(s_A P)$ 를 계산한다.
  3. $h(M, \bar{\mathrm{R}})$ 과 e 가 같은가 확인 한다. 두 결과가 같다면 바른 서명임이 확인 된 것이다.

다음은 실제로 ElGamal 과 Schnorr 기법을 구현하기 위하여 필요한 작업들이다.

5.1 타원 곡선의 선택

Pohlig-Hellman 알고리듬이나 Shank 의 baby-step giant-step 알고리듬에 안전하기 위하여 타원 곡선의 order 는 큰 소인수를 가져야 한다. 그 소인수의 크기가 30 자리수 이상이면 안전하다고 알려져 있다[8].

MOV 공격에 안전하기 위하여 nonsupersingular 타원 곡선을 택한다. 식 (4) 에서 a_2 와 $a_6 \neq 0$ 를 F_q 에서 임의로 선택한다. 먼저, 선택된 타원 곡선의 order를 구해야 한다. Order를 구하는 Schoof 의 알고리듬은 $O((\log q)^8)$ 비트 연산을 하므로 polynomial time 알고리듬이지만 실용성은 적다 [10]. 효율적으로 order를 구하기 위하여 작은 유한체에서 정의된 타원 곡선 $E(F_{2^s})$, 즉 s 를 작은 수로 선택하여 F_{2^s} 의 원소들을 직접 대입함으로써

$$| E(F_{2^s}) | = 2^s + 1 - k$$

의 원소의 갯수를 구한다. 다음에 Weil 의 정리를 이용하여

$$| E(F_{2^{st}}) | = 2^{st} + 1 - \alpha^t - \beta^t$$

을 구한다. 여기서 α 와 β 는 $1 - kT + 2^s T^2 = 0$ 의 두 근이다. 식 (8) 에 의해 $t = 1, 2, \ldots$ 에 대하여 $| E(F_{2^{st}}) |$ 를 구하고 이 order 가 큰 소인수를 갖는지 검사한다. 모든 소인수가 30 자리수 이하이면 a_2, a_6 을 다시 선택한다.

5.2 임의의 점 $P \in E(F_q)(q = 2^n, n$은 홀수$)$ 찾기 [13]

step 1. x 를 F_q 에서 랜덤하게 선택한다.
step 2. $z = x + a_2 + a_6 x^{-2}$ 을 계산한다.
step 3. $Tr(z) = 0$ 이면 $P = (x, xz) \in E(F_q)$ 이다.
step 4. $Tr(z) \neq 0$ 이면 step 1 로 간다.

여기서, $Tr(z) = z + z^2 + z^{2^2} + \ldots + z^{2^{n-1}}$ 이다.

5.3 메세지를 타원 곡선의 원소로 끼워넣기(imbedding)

다음에 위의 알고리듬을 이용하여 메세지를 타원 곡선의 원소로 끼워넣는 알고리듬을 설명하였다.

메세지 M을 타원 곡선 $E(F_q)$ ($q = 2^n, n$은 홀수) 위의 점 P_M 로 끼워넣기(imbedding)하여 P_M 에 암호 알고리듬을 적용시켜야 한다. $E(F_q)$ 위의 점 $P_M = (x(M), y(M))$ 에 메세지 M 을 imbedding 하고자한다.

메세지 M 의 비트 크기가 n 보다 작으면 M 에 0 을 덧붙여 (이것은 메세지에 블랭크(blank) 를 더한것이다.) M 이 n 비트 수가 되게한다.

step 1. $x(M) = M$ 이라 놓는다.
step 2. 6.2 에서 스텝 2, 스텝 3 을 $x(M)$ 에 적용한다.
step 3. $Tr(z) \neq 0$ 이면 $x(M)$ 의 한 비트를 바꾸어 step 2 로 간다.

위의 세번째 스텝에서 비트를 바꿀때 메세지 내용과 직접 연관이 없는 비트를 바꾸어준다. 예를 들어 $s = 9, t = 17$ 로 유한체 $F_{2^{st}}$ 가 선택 되었을 때, 한 문자(character)는 8 비트를 사용하므로 16 비트 크기 17 개로 구성된 배열에 M 을 저장한다. 즉, 암호화 하는 한 단위는 17 개의 문자이고 각 배열 원소에 한 문자씩 저장된다. 만약 위의 세번째 스텝에서 한 비트를 바꾸어주어야 한다면 어느 한 배열 원소를 택하여 16 비트중 최하위에서 9번째 비트를 0 또는 1로 바꾼다. 선택할 수 있는 배열 원소는 17개 이므로 2^{17} 개의 다른 M 을 얻을 수 있다. 각 M 에 대한 $Tr(z)$ 는 0 또는 1 이므로 개략적으로 말해서 2^{17} 중 반은 $Tr(z) = 0$ 일 수 있다. 따라서, 메세지 M 에 대하여 $P_M = (x(M), y(M))$ 을 쉽게 얻을 수 있다.

$x(M)$ 에서 각 배열 원소의 최하위 8 비트만을 택하면 메세지를 복구할 수 있다.

6 컴퓨터 구현

타원 곡선에서의 연산 속도를 빠르게 하기위하여 유한체에서는 연산을 변형된 다항식 기저를 사용하였다. 따라서, 유한체 F_q, $q = 2^n$ 선택시 작은 수 s 와 서로 소인 t 를 먼저 선택하여야 한다.

$s = 9$ 로 선택하였는데, 이것은 한 문자가 8비트임을 감안한 것이다. 즉 각 문자 c 는 2^8 보다 작은 이진수

$$c = b_7 b_6 b_5 ... b_1 b_0$$

로 표현되고

$$c = b_7 z^7 + b_6 z^6 + ... + b_1 z + b_0$$

는 F_{2^9} 의 한 원소이기 때문이다. 한 비트가 남는 것은 6절에서 설명한 끼워 넣기(imbedding) 를 위한 것이다

차수가 9 인 원시적 다항식(primitive polynomial) 은 $g(z) = z^9 + z^4 + 1$ 이므로 유한체 F_{2^9} 은

$$F_{2^9} = \{z^i \ (mod\ g(z)) \mid i = 0, ..., 2^9 - 1\}$$

이다.

식 (4) 에서 $a_2 = 0, a_6 = z^{19} \ (mod\ g(z)) = 1 + z + z^4$ 로 선택하여

$$y^2 + xy = x^3 + (1 + z + z^4) \tag{14}$$

을 얻었다. 이 식에 2^{18} 개의 쌍 $(x, y) \in F_{2^9} \times F_{2^9}$ 를 대입하여

$$\mid E(F_{2^9}) \mid = 2^9 + 1 - k = 499,$$

$$k = 14$$

를 얻었다.

$t = 0, 1, \ldots$ 에 대하여 (8), (9) 를 이용하여 $|\ E(F_{2^{9t}})\ |$ 를 구한다. t_1 이 t 를 나누면 $|\ E(F_{2^{9t_1}})\ |$ 은 $|\ E(F_{2^{9t}})\ |$ 를 나누므로 [8], $|\ E(F_{2^{9t}})\ |$ 가 30 자리수 이상의 큰 소인수를 갖기 위하여 t 를 솟수로 선택한다.

이 논문에서는 t 를 17로 택하였고, 따라서 $q = 2^{9\times 17} = 2^{153}$ 개의 원소를 갖는 유한체 F_q 위에서 정의된 타원 곡선을 ElGamal 과 Schnorr 디지탈 서명 방법에 사용하였다.
$E(F_q)$ 의 order $|\ E(F_q)\ |$ 는

$$\begin{aligned} |\ E(F_{2^{153}})\ | &= 2^2 \times 5^3 \\ &\times\ 22835963083295358096932990593328038755822577 \end{aligned}$$

이고, 가장 큰 소인수는 44 자리수이다.
$E(F_{2^{153}})$ 의 한 원소를 임의로 선택한 후 그 order를 구했더니, $|\ E(F_{2^{153}})\ |$ 이었고, 따라서 선택된 타원 곡선 $E(F_{2^{153}})$ 은 cyclic 군 임을 알 수 있다.
군 $E(F_{2^{153}})$의 생성자(generator)는 Appendix A 에 밝혔다.

구현 결과

메세지 "ChristDiedForUs" 의 암호문과 이 메세지에 대한 서명을 생성하였고 그 결과는 다음과 같다.
구현하는 예는 Appendix B 에 있다.

- **ElGamal 암호 기법과 서명 기법의 구현 결과**

암호문 생성 (Ciphertext)	R	62565427712107372740512734560244204463443317443672 6 55153514376641721464745056430066602412753673645454 6
	S	530001045073001346156376514402067721231405422253755 331374036645056631573532455716604244476403757242442
	시간	2.04 sec
서 명 생 성	R	606056637352535243760261125740517265332117017325611 212345755620201435562537223052364731743522632672516
	x	98991876074714226829810451044127011260562727 8
	시간	1.72 sec
복호화	시간	0.49 sec
서명 확인	시간	4.02 sec

- **Schnorr 서명 기법의 구현 결과**

서 명 생 성	e	33213851884762208539230 4
	x	570899389716816674061301274943530944355610411 2
	시간	1.45 sec
서명확인	시간	1.94 sec

한 페이지 분량의 1874개 문자(약 15000 비트)를 ElGamal의 기법으로 암호화하고, Schnorr 기법으로 서명하는데 5분 42초가 소요되었다. 평균 한 블럭(17개의 문자)당 3.1초가 소요된 셈이다.

복호화와 서명 확인하는데는 전체 4분 39초, 평균 2.5 초가 소요되었다. 현재 구현된 알고리듬은 메세지의 각 블럭마다 서명을 하도록 하였으나, 실제 사용할 때는 타원 곡선을 사용하는 암호 시스템이 느리므로 메세지 전체에 대해 한 번 서명하도록 한다.

사용한 컴퓨터는 SUN 4 이다.

7 결론

유한체를 사용한 ElGamal 암호 시스템보다 타원 곡선을 사용하는 암호 시스템이 더 느리다는 단점이 있지만, 정보 보호를 위한 비용보다 안전도가 더 요구될 때 타원 곡선을 사용할 수 있으므로 그 실용화를 위해 많이 연구 되어져야 한다.

ElGamal 시스템을 개선한 Schnorr 시스템은 해쉬 함수(hash function)를 사용하여 작은 키 크기(key size)로 서명을 한다. 키 크기가 작으므로 타원 곡선위에서의 연산 수도 줄일 수 있고 복호화 과정에서 반복 더하기 계산이 두 번 뿐이다. (ElGamal 시스템은 세 번이다.)

타원 곡선을 이용하여 ElGamal 암호 기법을 구현한 결과 17개의 문자를 2.04초로 암호화할 수 있었다. ElGamal 기법으로 서명 생성은 1.72초, 확인은 4.02초로 할 수 있었고, Schnorr 기법으로 서명 생성은 1.45초, 확인은 1.94초로 할 수 있었다.

또한, anomalous 곡선과 같은 성질을 가진 특정한 타원 곡선에서 정규 기저를 사용하여 Schnorr 시스템을 구현하는 것도 타원 곡선을 이용한 서명 기법의 실용화에 큰 도움이 될 것이다.

Appendix

A 구현시 사용된 군과 generator

ElGamal 과 Schnorr 기법 구현시 사용된 군은

$$\begin{aligned} E(F_{2^{153}}) &= \{(x,y) \in F_{2^{153}} \times F_{2^{153}} \mid y^2 + xy = x^3 + (1 + z + z^4), \\ &\qquad 1 + z + z^4 \in F_{2^9}\} \end{aligned}$$

이고, generator $P = (x, y)$ 는 다음과 같다.

$$\begin{aligned} x &= z^{152} + z^{150} + z^{149} + z^{148} + z^{144} + z^{143} + z^{139} \\ &+ z^{138} + z^{133} + z^{132} + z^{131} + z^{130} + z^{129} + z^{127} \\ &+ z^{126} + z^{124} + z^{120} + z^{118} + z^{115} + z^{114} + z^{113} \\ &+ z^{112} + z^{110} + z^{108} + z^{104} + z^{102} + z^{101} + z^{98} \\ &+ z^{97} + z^{96} + z^{94} + z^{93} + z^{92} + z^{91} + z^{90} + z^{87} \\ &+ z^{86} + z^{85} + z^{84} + z^{83} + z^{82} + z^{77} + z^{76} + z^{75} \\ &+ z^{72} + z^{61} + z^{55} + z^{54} + z^{53} + z^{50} + z^{49} + z^{46} \\ &+ z^{44} + z^{41} + z^{40} + z^{39} + z^{38} + z^{36} + z^{34} + z^{31} \\ &+ z^{29} + z^{25} + z^{24} + z^{23} + z^{20} + z^{19} + z^{18} + z^{16} \end{aligned}$$

$$
\begin{aligned}
 & + \quad z^{14} + z^{11} + z^{10} + 1 \\
y & = \quad z^{151} + z^{150} + z^{149} + z^{145} + z^{138} + z^{137} + z^{135} \\
 & + \quad z^{134} + z^{133} + z^{132} + z^{130} + z^{128} + z^{127} + z^{125} \\
 & + \quad z^{124} + z^{122} + z^{117} + z^{115} + z^{113} + z^{112} + z^{110} \\
 & + \quad z^{108} + z^{107} + z^{105} + z^{101} + z^{100} + z^{98} + z^{97} \\
 & + \quad z^{96} + z^{94} + z^{91} + z^{90} + z^{89} + z^{88} + z^{87} + z^{82} \\
 & + \quad z^{79} + z^{74} + z^{71} + z^{70} + z^{68} + z^{67} + z^{66} + z^{65} \\
 & + \quad z^{64} + z^{62} + z^{52} + z^{51} + z^{50} + z^{47} + z^{41} + z^{40} \\
 & + \quad z^{37} + z^{35} + z^{32} + z^{31} + z^{27} + z^{25} + z^{23} + z^{22} \\
 & + \quad z^{20} + z^{18} + z^{16} + z^{15} + z^{9} + z^{8} + z^{7}
\end{aligned}
$$

B Appendix 구현 예

사용자 ejlee 는 msg.dat 에 자신의 메세지를 저장한 후, 다음을 실행한다.
(굵은 글씨는 입력 정보 이다.)

- Enter ID / New(to register) : **ejlee**
- Secret Key (16 characters) :
- Create/Verify (1/0) : **1**
- Enter Reciever's ID : **hsh**
- Message file name : **msg.dat**
- Ouput file name : **out.dat**
- Encrypting time : 2.03 sec
- Signing time : 1.45 sec

out.dat 에 디지탈 서명의 전송 내용인 암호문과 서명 (R, S, R_s, x) 이 저장된다. 다른 사용자 hsh 에게 out.dat 를 전송한다.

서명 확인

사용자 hsh 는 암호화 된 메세지와 서명이 담긴 파일 out.dat을 받은 후, 다음을 실행한다.

- Enter ID / New(to register) : **hsh**
- Secret Key (16 characters) :
- Create/Verify (1/0) : **0**

- Enter Sender's ID : **ejlee**
- Ciphertext file name : **out.dat**
- Output file name : **vef.dat**
- Decrypting time : 0.48 sec
- Verifying time : 1.94 sec

vef.dat 에는 복호화된 메세지와 그 메세지에 대한 서명 확인 여부의 내용이 쓰여진다.

vef.dat :
ChristDiedForUs (메세지가 복구되었음.)
Verifying success !! (서명 확인)

참고 문헌

[1] YoungJu Choie and Hyo Sun Hwoang, On the cryptosystem using elliptic curves, *Proceedings of Korea-Japan JW-ISC '93*, pp.105-113, 1993

[2] D. Coppersmith, Fast evaluation of logarithms in fields of characteristic two, *IEEE Transaction on Information Theory,* **IT-30**, pp.587-594, 1984

[3] W. Diffie and M. Hellman, New directions in cryptography, *IEEE Transaction on Information Theory*, **22**, pp.644-654, 1976

[4] T. ElGamal, A Public Key Cryptosystem and a Signiture Scheme based on Discrete Logarithms, *IEEE Transaction on Information Theory*, **31**, pp.469-472, 1985

[5] G. Harper, A. Menezes and S. A. Vanstone, Public-Key Cryptosystems with Very Small Key Lengths, *Eurocrypt 92*, 1992.

[6] N. Koblitz, Elliptic curve cryptosystems, *Math. Comp.*, **48**, pp.203-209, 1987.

[7] N. Koblitz, *A Course in Number Theory and Cryptography*, Springer-Verlag, 1987.

[8] N. Koblitz, CM-curves with good cryptographic properties, *Advances in Cryptology*, **3**, pp.187-199, 1991.

[9] A. K. Lenstra and H. W. Lenstra, Jr.Algorithms in number theory, in:*Handbook of Theoretical Science*, Vol. **A**, *Algorithms and Complexity*, ed. by J. Van Leeuwen, Amsterdam: Elsevier, pp.673-715, 1990.

[10] A. Menezes, *Elliptic Curve Public key Cryptosystems*, Kluwer Academic Publishers, 1993.

[11] V. Miller, Uses of elliptic curves in cryptography, *Advances in cryptology - Crypto '85*, Lecture notes in computer science, **218**, pp.417-426, 1986.

[12] A. Menezes, T. Okamoto, and S. A. Vanstone, Reducing elliptic curve logarithms to logarithms in a finite field, *Proceedings of the 23rd ACM Symp. Theory of Computing*, 1991.

[13] A. Menezes and S. A. Vanstone, Elliptic Curve Cryptosystems and their Implementation, *Advances in cryptology*, **6**, pp.209-224, 1993.

[14] A. Odlyzko, Discrete Logarithms in Finite Fields and their Cryptographic Significance, *Advances in cryptology*, Lecture Note in Computer Science, **Vol.209**, Spriger-Verlag, NY, pp.224-314, 1984.

[15] J. H. Silverman, *The Arithmetic of Elliptic curves*, Spriger-Verlag, 1986.