타원곡선을 이용한 암호화와 전자서명

양승해*, 조인석*, 이병관**
관동대학교 전자계산공학과

Encryption and Digital Signature Using Elliptic Curve

Seung-Hae Yang*, In-Seock Cho*, Byung-Kwan Lee**
*Dept. of Computer Science, Kwandong University
*Dept. of Computer Science, Kwandong University
**Dept. of Computer Engineering, Kwandong University
E-mail : yang7177@mail.kwandong.ac.kr
cho9088@hanmail.net
bklee@mail.kwandong.ac.kr

요 약

보안프로토콜로 기존의 SSL에서 인증 기능을 각각의 사용자, 상점, 금융기관에 강화시킨 ECSET프로토콜이 있다. ECSET프로토콜의 구성성분으로 비밀키 알고리즘의 DES, 공개키 알고리즘의 RSA, 메시지 서명 알고리즘인 SHA중 공개키 알고리즘의 RSA를 ECC로 대체함으로 암, 복호화 속도를 분석하였고, 서명 알고리즘으로 타원곡선을 이용한 ECDSA을 보였다.

1. 서론

웹 보안 프로토콜은 초기의 SSL(secure socket layer)의 방법에서 인증기능을 강화한 ECSET(elliptic curve secure electronic transaction) 메커니즘으로 발전하였다. ECSET는 통신 응용간의 메시지의 내용을 허가받지 않고 도청하거나 복사하는 등 노출되는 것을 방지하는 기능인 기밀성(confidentiality), 송신자는 수신자의 전송 정보가 도중에 불법적으로 변경이 되지 않고 제대로 도착했는가를 확인할 수 있어야 하는 메시지 무결성(data integrity), 송신자나 수신자가 전송 메시지를 부인하지 못하도록 막는 즉, 메시지가 송신되었을 때 수신자는 그 메시지가 실제로 송신자에 의해서 송신되었음을 확인하는 부인방지(non-repudiation), 송신자 이외의 사람이 송신자의 신분을 위장하여 메시지를 작성하는 행위를 방지하기 위한 인증(authentication)서비스를 만족시켜줄 수 있다.

본 연구의 ECSET 에서는 기존의 비밀키 알고리즘인 DES를 가변키를 이용한 VDES(virtual-key data encryption standard)를 차후의 과제로 정하고 곱셈연산을 통하여 암·복호화를 수행하는 RSA(rivest, shamir, adleman)와 성능이 보다 향상된 덧셈연산을 주체로 하는 ECC(elliptic curve cryptographic)를 구현함으로써 암호화의 기본 성과인 계산량, 키 크기, 대역폭 에서의 조건을 만족하는 알고리즘을 다룰 것이다. 2장에서는 관련연구로 ECSET프로토콜을 구성하고 있는 객체들간에 처리되는 암호의 종류와 과정을 보인다. 3장에서는 ECSET프로토콜을 구성하는 암호화 방법 중 공개키 암·복호화 알고리즘 즉, RSA 알고리즘과 ECC알고리즘과 서명 알고리즘으로 타원곡선을 이용한 ECDSA(elliptic curve digital signature algorithm)에 대하여 설명하고, 4장에서는 시뮬레이션 및 결과를 보인다. 그리고 5장에서는 결론과 향후연구 과제를 설명한다.

*준회원 : 관동대학교 전자계산공학과 대학원
* 관동대학교 전자계산공학과 대학원
** 관동대학교 컴퓨터공학과 교수

2. 관련연구

2.1 ECSET 암호 메커니즘

ECSET은 공개키 암호, 대칭키 암호, 서명, 해쉬 등을 사용하여 안전성을 보장해 주게 된다. 공개키 암호 방식은 비 대칭키 암호화 방식으로 키를 두 개로 나누어 하나는 암호화 키로 또 하나는 복호화 키로 사용한다. 암호화 키는 공개 목록에 등록하는 형식의 공개키이고, 복호화 키는 개인이 비밀리에 보관하는 비밀키이다. 비밀통신 가입 수신자 A에게 비밀 통신을 하려는 송신자 B는 수신자 A가 공개한 공개키로 전달하려는 평문을 암호화하여 수신자 A에게 암호문을 전송하면 가입자 A는 자신이 비밀리에 보관하던 비밀키로 암호문을 복호화 한다. 지금까지의 대표적인 예로 1024bit의 RSA 알고리즘이있다. 대칭키 암호화 방식은 관용 암호 방식 즉, 비밀 암호화 방식이라고도 하며 암호화 키와 복호화 키가 동일하다. 이러한 관용 암호화 방식으로 대표적인 것으로 DES(data encryption standard)가 있다. 디지털 서명은 송신자 B가 평문에 송신측의 서명을 함으로서 수신자 A는 메시지가 송신자 B에 의해 전송된다는 것을 알 수 있다. 그리고 해쉬 알고리즘은 디지털 서명에서 서명문의 길이가 제한적으로 전송하여야 하는데, 이러한 경우 디지털 서명에서 서명문의 무결성 보장이 가능하고 한 번에 서명을 실행할 수 있는 양으로 서명문을 압축할 수 있는 방법이 필요하다. 해쉬함수가 서명문 압축과 디지털 서명을 효율적으로 계산할 수 있는 방법을 제공하고 있다. 그림 2-1은 ECSET의 암호 처리 과정을 표현한 것이다.

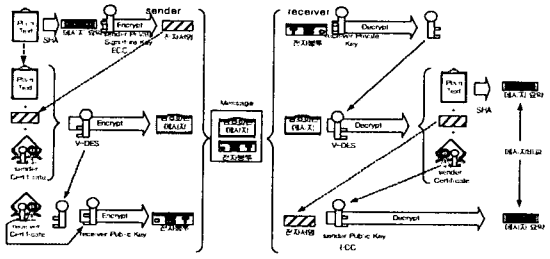

그림 2-1 ECSET의 암호 처리 과정

2.3 ECSET 보안 메커니즘

그림 2-2는 ECSET 메커니즘의 시나리오를 보여주고 있다.

SET 메커니즘은 카드 소지자, 상점, 또 금융기관 사이에 인증 기능을 두어 안전한 채널을 제공해 줄 수 있는 장점을 가지고 있다.

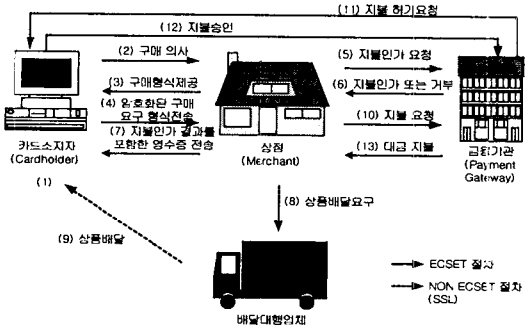

그림 2-2 SET 쇼핑 시나리오

반면에 속도가 상대적으로 SSL에 비하여 느리다는 단점과 별도의 소프트웨어의 설치가 상점, 사용자, 그리고 금융기관간에 요구됨으로 인하여 비용 면에서 부담이 증가된다. 게이트웨이는 사용자의 신용카드 번호를 보호해주며, 판매자들은 실시간으로 신용카드의 유효 여부를 확인할 수 있게 해주며, 또한 판매자와 인터넷 사이에서 인터페이스 역할을 해주며, 인터넷에서 받은 신용카드 주문정보를 ISO-8583 표준으로 번역하여 기존의 은행 처리 시스템으로 전달해 준다. 이것이 인터넷상의 온라인 구매 행위를 위한 게이트웨이 시스템의 흐름이 된다. 이러한 ECSET 메커니즘의 가장 중요한 기능 중의 하나는 CA(certificate authority)의 역할을 해주는데 있다. ECSET 프로토콜은 구매자 자신이 정당한 사용자임을 증명해주고 상인 자신의 신분도 보장받게 해주는 특성을 갖춘다.

3. 공개키 암/복호화 알고리즘

3.1 암호방식

암호방식이란 정보 내용과 정보 운송자 사이에 존재하는 다양성을 이용해서 정보 내용과 정보 운송자 사이의 대응 관계를 제삼자에게 비밀로하여 정보를 교환하는 방법을 말한다. 정보를 도청자(공격자)로부터 보호하기 위한 암호 방식의 구성은 그림 3-1과 같다.

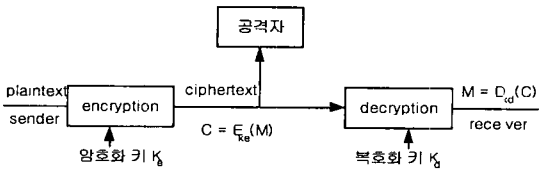

그림 3-1 암호 방식의 구성

평문(plaintext)은 송신자가 수신자에게 전달하려는 정보 내용으로 누구나 그 의미를 알 수 있는 정보이다. 송신자는 평문 M을 암호화키 K_e와 암호(encryption)알고리즘을 적용시켜 암호문(ciphertext) C를 생성하여 수신자에게 전달한다.

$$C = E_{ke}(M)$$

암호문 C는 전송 상태에 있으므로 그 내용을 알 수 없는 데이터이다. 이때 공격자가 전송중인 data를 침략하더라도 복호화 키가 없으면 평문을 읽을 수 없다. 수신자는 송신자가 전송한 암호문 C를 수신하여 복호화(decryption)알고리즘을 적용시켜 송신자가 전송하려는 평문 M을 복원한다.

$$M = D_{kd}(C)$$

관용 암호 방식 즉, 대칭키 암호화 방식은 암호화키 K_e와 복호화키 K_d가 동일하다. 그러므로 송신자와 수신자가 비밀 통신을 시작하기 전에 비밀리에 키를 공유하고있어야 한다. 따라서 사전에 비밀리에 키를 공유하기 위한 키 분배 과정이 필요하다.

그러나 공개키 암호 방식은 암호화키 K_e와 복호화키 K_d를 분리하여 암호화키 K_e를 공개하고 복호화키 K_d를 비밀리에 보관한다. 물론 공개암호화키 K_e로부터 비밀 복호화키 K_d를 계산할 수 없어야 한다. 따라서 공개키 암호방식은 암호화키를 공개함으로써 관용 암호 방식에서와 같이 키를 분배할 필요가 없다.

3.2 RSA와 ECSET 알고리즘

(1) RSA 알고리즘

RSA는 합성수의 소인수 분해의 어려움을 이용한 암호화 방법이다. 가입자는 백 자리 크기 이상의 두 개의 소수 p, q를 선택하여 $n = p \cdot q$를 계산한다. 소수 p, q의 크기가 클수록 암호화 시키는 비중은 커지게 되며 보안의 강도는 비례적으로 강하게 된다. 이때 p와 q를 알고 있는 사람은 n을 계산하기 쉽지만 n만 알고 있는 사람은 n으로부터 p와 q를 찾는 소인수 분해는 어렵다.

RSA 암호 방식의 구성 절차와 암·복호화 과정은 그림 3-2와 같다.

수신자 B에게 평문 M을 비밀리에 전달하려는 송신자 A는 공개목록에서 수신자 B의 공개 암호화 키 K_{eB}를 찾아, 암호문 $C \equiv M^{K_{eB}} \bmod n_B$를 계산하여 수신자 B에게 전송한다. 수신자 B는 송신자 A로부터 수신된 암호문 C를 자신이 비밀리에 보관하고 있는 복호화 키 K_{dB}로 평문 $M \equiv C^{K_{dB}} \bmod n_B$를 복원한다.

역으로 송신자 B가 수신자 A에게 평문을 비밀리에 전송하려면 수신자 A의 공개 암호화 키 K_{eA}를 공개목록에서 찾아 암호문 $C \equiv M^{K_{eA}} \bmod n_A$를 계산하여 수신자 A에게 전송한다. 수신자 A는 송신자 B로부터 수신한 암호문 C를 자신이 비밀리에 보관하고 있던 비밀 복호화 키 K_{dA}로 평문 $M \equiv C^{K_{dA}} \bmod n_A$을 복원한다.

송신자 A		수신자 B
p_A, q_A $n_A = p_A \cdot q_A$ $\Phi(n_A) = (p_A - 1)(q_A - 1)$ $\gcd(K_{eA}, \Phi(n_A)) = 1$ $K_{eA} \cdot K_{dA} \equiv 1 \bmod \Phi(n_A)$ $C \equiv M^{K_{eB}} \bmod n_B$	C →	p_B, q_B $n_B = p_B \cdot q_B$ $\Phi(n_B) = (p_B - 1)(q_B - 1)$ $\gcd(K_{eB}, \Phi(n_B)) = 1$ $K_{eB} \cdot K_{dB} \equiv 1 \bmod \Phi(n_B)$ $M \equiv C^{K_{dB}} \bmod n_B$

그림 3-2 RSA 공개키 암호 방식

(2) ECSET 알고리즘

타원곡선은 유한체 상에 정의된 타원곡선에 대하여 타원곡선군은 3차 방정식을 만족하는 순서쌍들과 무한점을 포함한 집합을 말한다. 점의 가환군을 계산하는 암호화 방법으로 두 가지 계산 방법이 있다. 동일한 점 즉, $P = Q$의 경우와 점 $P \neq Q$인 경우이다. 그림 3-3은 $P \neq Q$인 경우이고, 그림 3-4는 $P = Q$인 경우이다.

그림 3-3은 타원곡선 위의 두 점 P와 Q의 합 $P + Q$를 구하기 위해 먼저 P와 Q를 지나는 직선 L을 긋는다.

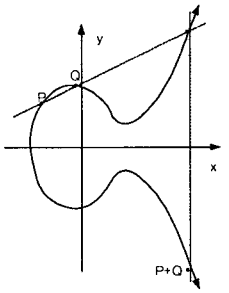

그림 3-3 P≠Q의 ECC 그래프

그러면 L과 원래의 타원곡선과 만나는 점이 반드시 존재하게 되고 그 점을 바로 $-(P+Q)$로 정의한

다. 그리고 x축과 대칭 이동되는 점이 $P+Q$가 된다.

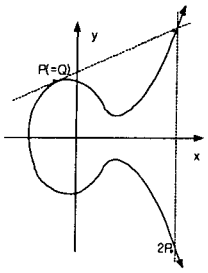

그림 3-4 P=Q의 ECC 그래프

그림 3-4는 $P=Q$인 경우로 같은 점 즉, $P+P=2P=Q$인 점을 구하는 방식이다. $P\neq Q$와 같이 P에서 Addition $2P$를 한 점 $-(2P)$를 찾고, x축을 기준으로 대칭 이동하여 $2P$의 값을 갖는다.

타원곡선 암호화시스템은 타원곡선상에서 점 P를 a번 더하는 계산이 주를 이룬다. 즉, $Q=aP$를 구하는 더하기 연산은 modular 곱셈을 통해 이루어지며 주요 공개키 암호 시스템은 효율적인 modular 곱셈에 의존하고 있다. 또 타원곡선 암호화가 RSA 암호화 시스템보다 능률적인 이유는 소수 p의 값이 작아도 안전성이 뛰어나다는 것이다. 즉, 타원곡선 암호시스템의 안전도는 타원곡선 이산대수문제에 의존하고 있으며, 효율성은 aP를 얼마나 빠르게 계산해 낼 수 있는가에 달려있다.

이러한 ECC 암호화 방법은 유한체 k위에서 정의된 타원곡선 E위의 점들이 가환군의 형태를 이룬다. 이 가환군의 더하기 연산은 기초 체(field) k에서의 산술 연산 몇 개를 포함하며, 하드웨어와 소프트웨어로 구현하기가 쉽다.

3.3 RSA와 ECC 구현

(1) RSA 구현모드

그림 3-5는 RSA를 구현하기 위한 기본 모드로서 암호화시 평문 내용을 포함하고 있는 plaintext라는 특정파일을 읽어 임의의 두 개의 키 p와 q를 선택하여 이를 이용하여 e,n인 공개키와 d,n인 비밀키를 생성하고 $C=M^e \bmod(n)$으로 암호화 하여 ciphertext를 출력하였다.

복호화는 암호화를 $M=C^d \bmod(n)$을 수행하여 ciphertext파일을 입력받아 복호화 루프를 수행한 후 palintext를 확인할 수 있다.

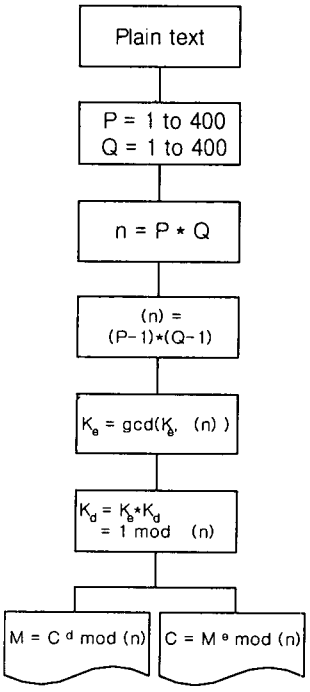

그림 3-5 RSA 구현모드

단계별로 수행하는 RSA알고리즘 순서는 다음과 같다.

Step 1] 임의의 소수 p와 q를 선택한다.

Step 2] $n=p\times q$를 계산한다.

Step 3] $\Phi(n)=(p-1)(q-1)$을 구한다.

Step 4] $\Phi(n)$과 서로소인 공개키 K_e를 구한다.

$\gcd(k_e, \Phi(n)) = 1$

gcd(Φ (n), k_e)

```
int compute_gcd(int p1, int q1)
{
  int temp;
  while (q1)
  {
  temp = p1%q1;
  p1=q1;
  q1=temp;
  }
  return p1;
}
```

Step 5] 비밀키 K_d값을 구한다.

$K_e \cdot K_d = 1 \bmod \Phi(n)$

(2) ECC 구현모듈

본 장에서는 송신자(sender)가 암호화 메시지를 수신자(receiver)에게 보내는 과정을 보여준다. 간단하게 메시지는 소수 p보다 작은 정수의 쌍이다. 랜덤한 정수 r과 메시지를 송신코드에 의해 수신자에게 보낸다. 수신자의 복호는 수신자의 비밀키 k로 메시지를 복호 한다.

타원곡선 암·복호화 과정은 다음과 같다.

Step 1] Receiver : Choose p

Step 2] Receiver : Choose an elliptic curve

$y^2 = x^3 + bx + c$ (Choose b and c)

Step 3] Receiver : Find a point P on the curve

```
Addition (P=Q)

PointEC(x1, y1, b, c) + PointEC(x1, y1, b, c)
        := Module(L, IdentityECQ, x3, y3),
          IdentityECQ = True;
           L = (3x1^2 +b)/ 2y1;
              x3 = L^2 -2x1;
              y3 = L(x1-x3)-y1;
    Return(PointEC(x3, y3, b, c))
```

```
Addition (P+Q)

  PointEC(x1, y1, b, c) + PointEC(x2, y2, b, c)
          := Module(L, IdentityECQ, x3, y3),
            IdentityECQ = False;
             L = (y2 -y1)/ (x2 -x1);
                    x3 = L^2 -x1 - x2;
                    y3 = -y1 + L(x1-x3);
    Return(PointEC(x3, y3, b, c))
```

Step 4] Receiver : Choose an integer k

Step 5] Receiver : Calculate kP

Step 6] Receiver : Tell Sender the values b, c, p, P, kP

Step 7] Sender : Choose a random integer r

Step 8] Sender : Calculate rP and $r(kP)$

Step 9] Sender : Let $rP=(i,j)$, $r(kP)=(m,n)$, $d=mx, e=ny$ Tell i, j, d, e to Receiver

Step 10] Receiver : Let $rP=(i,j)$. Calculate $r(kP)=(m,n)$

Step 11] Receiver : Solve the equations $mx = d(\bmod p)$ and $ny = e(\bmod p)$

이로서 타원곡선 암·복호화의 과정을 보았다.

3.4 ECDSA

전자 서명은 문서나 메시지를 보낸 사람의 신원이 진짜임을 증명하기 위해 사용되는 서명이다. 이것은 또한 전달된 메시지나 문서의 원문의 내용이 변조되지 않았다는 사실을 보증하기 위해 사용된다.

ECDSA의 키 생성방법에서 각 객체(entity) 즉, 송신자의 키 생성방법은 다음과 같다.

Step 1] Zp위에 정의된 타원 곡선 E를 선택하고, $E(Zp)$에 점의 수는 큰 소수 n으로 나누어져야 한다.

Step 2] 차수 n의 점 $P \in E(Zp)$를 선택한다.

Step 3] [1, n-1]사이에 있는 통계적으로 유일하고 예측 불가능한 정수 d를 선택하여 비밀키로 한다.

Step 4] $Q = dP$를 계산한다.

Step 5] 송신자의 공개키는 E, P, n, Q이고, 비밀키는 d이다.

ECDSA의 서명 생성(signature generation)방법에서 메시지 m을 서명하기 위하여 송신자는 다음과 같은 과정을 수행한다.

Step 1] 구간[1, n-1]사이에 있는 통계적으로 유일하고 예측 불가능한 정수 k를 선택한다.

Step 2] $kP=(x_1, y_1)$과 $r = x_1 \bmod n$을 계산한다.

Step 3] $k^{-1} \bmod n$을 연산한다.

Step 4] $s = k^{-1}\{h(m) + dr\} \bmod n$을 연산한다. (h는 SHA-1의 알고리즘이다.)

Step 5] $s=0$ 이면, Step 1]로 간다.

Step 6] 메시지 m에 대한 서명은 한 쌍의 정수 (r, s)이다.

ECDSA의 서명 생성(signature generation)방법에서 메시지 m을 검증하기 위하여 수신자는 다음과 같은

과정을 수행한다.

Step 1] 송신자의 공개키 E, P, n, Q의 인증을 얻는다.

Step 2] r과 s가 [1, n-1]에 정수들인 것을 검증한다.

Step 3] $w = s^{-1} \bmod n$과 $h(m)$을 연산한다.

Step 4] $u_1 = h(m)w \bmod n$과 $u_2 = rw \bmod n$을 연산한다.

Step 5] $u_1 P + u_2 Q = (x_0, y_0)$ 와 $v = x_0 \bmod n$을 연산한다.

Step 6] $v = r$일 경우에만 서명을 인정한다.

이로서 ECDSA의 서명 알고리즘을 보았다. Step 6]에서 $v = r$를 통하여 메시지가 변조되지 않는 데이터 무결성을 수신자는 알 수 있고, 이로 인한 디지털 서명을 통하여 발신자의 부인 방지를 확인할 수 있다.

4. 시뮬레이션 및 결과

RSA와 ECC를 비교분석 하였는데, 그림 4-1과 그림 4-2는 O/S는 Win 2000, CPU가 Pen-Ⅲ 450Mhz, RAM은 128MB에서의 비교이다. 암호화 수행 시간 비교 시 key size가 5byte일 때에는 6:1, 10일 때에는 18.3:1, 15일 때에는 44.1:1, 20일 때에는 23.9:1, 25일 때에는 76.6:1, 30일 때에는 136.9:1로 key size가 커질수록 ECC가 상당히 빠른 비율을 보였고, 복호화 수행 시간 비교는 key size가 5일 때에는 1.5:1, 10일 때에는 6.3:1, 15일 때에는 23.3:1, 20일 때에는 47.5:1, 25일 때에는 166.3:1, 30일 때에는 215.6:1의 큰 비율을 나타냈다.

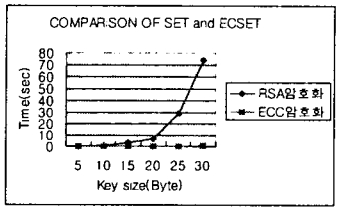

그림 4-1 RSA와 ECC의 암호화 시간 비교

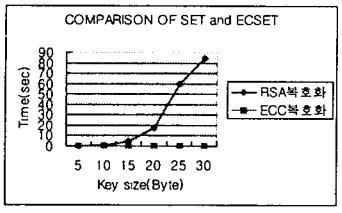

그림 4-2 RSA와 ECC의 복호화 시간 비교

이로서 컴퓨터 환경이 발달될수록 RSA와 ECC의 암·복호화 시간의 차이는 월등히 커지는 것을 보았다.

5. 결론

지금까지의 SET프로토콜은 RSA방식을 사용하여 합성수의 소인수분해에 관한 곱셈의 연산으로 암·복호화의 시간이 길다. 일반적으로 RSA를 깨는 것은 정수 n(공개키)에 대한 정수 분해 문제를 푸는 것과 같다. 정수 분해 문제에 대한 효과적인 알고리즘이 없기 때문에 n을 충분히 크게 설정함으로써 시스템을 보다 안전하게 만들 수 있다. 이러한 대책으로 타원곡선을 이용한 ECSET를 개발함으로서 RSA보다 짧은 구현시간을 얻을 수 있다. ECSET는 짧은 Key크기를 사용함으로 타원곡선 암호시스템은 스마트카드나 무선환경에서처럼 상대적으로 작은 Key 크기와 제한적 대역폭과 메모리가 요구되는 분야에서 각광받고있는 차세대 암호시스템이다. 향후과제로는 지금의 RSA보다 ECC에서의 시간 향상이 ECSET를 개발한 것 즉, DES를 VDES로 RSA를 ECC알고리즘으로 대치시켰을 때 얼마만큼의 효율성을 갖을 수 있는지로 한다.

참고문헌

[1] Dabid Pointchebal and Jacques Stern "Security Proofs for Signature Schemes" Advances in Cryptology-Proceedings of EUROCRYPT' 96 page387-398

[2] Toshio HASEGAWA, Junko NAKAJIMA, Mitsuru MATSUI "A Small and Fast Software Implementation of Elliptic Curve Cryptosystems over GF(p) on a 16-Bit Microcomputer" IEICE TRANS.FUNDAMENTALS,VOL.E82-A, NO.1 JANUARY 1999

[3] Michael J. Wiener, Robert J. Zuccherato "Faster Attacks on Elliptic Curve Cryptosystems" Canada K1V 1A7 April 8, 1998

[4] Gadiel Seroussi "Compact representation of elliptic curve points over F_{2^n}" Hewlett-Packard Laboratories April 6,1998