Aplikasi Elliptic Curve Cryptography (ECC) untuk Smart Card

Dian Intania Savitri (13503081)

Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung
2006

Abstraksi

Elliptic curve cryptography (ECC) merupakan kriptografi kunci publik yang berdasarkan pada struktur aljabar kurva ellips pada daerah finite. Penggunaan kurva ellips dalam kriptografi dianjurkan oleh Neal Koblitz dan Victor S. Miller pada tahun 1985. Kurva ellips juga digunakan dalam beberapa algoritma pemfaktoran integer yang memiliki aplikasinya dalam kriptografi, seperti *Lenstra elliptic curve factorization*, dan lain-lain.

Elliptic curve cryptosystems (ECCS) menjadi semakin populer karena pengurangan jumlah bit kunci yang dibutuhkan dalam perbandingan dengan *cryptosystem* lainnya (contohnya, 160 bit ECC memiliki tingkat keamanan yang sama dengan 1024 bit RSA). ECCS cocok digunakan untuk *smart card*, karena *smart card* memiliki keterbatasan memory dan kecepatan komputasi pada devicenya. Makalah ini akan membahas optimisasi dari implementasi algoritma *digital signature* kurva ellips pada *smart card* Motorola. Tingkat keamanan yang tinggi pada ECC (karena tingkat keamanan ECC tidak tergantung pada panjang kuncinya), dan juga ECC dapat menghasilkan implementasi yang lebih cepat dibandingkan RSA atau algoritma kriptografi lainnya, serta penggunaan *bandwith* dan power yang sedikit inilah yang membuat ECC cocok digunakan pada aplikasi yang krusial seperti *smart card*.

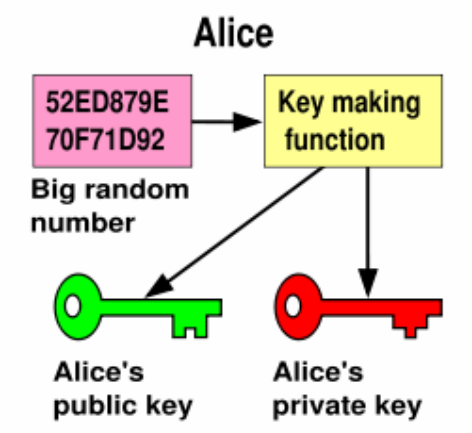
Kata kunci: Elliptic Curve Cryptography (ECC), Smart Card

I. Public Key Cryptography

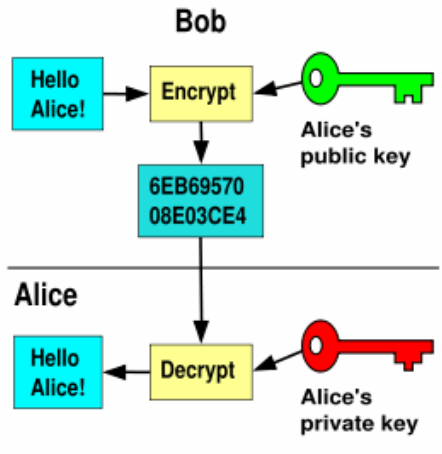
Public key cryptography atau disebut juga algoritma kunci publik, adalah salah satu bentuk kriptografi yang secara umum dapat membuat pengguna saling berkomunikasi dengan aman tanpa memiliki akses tertentu terhadap kunci rahasia yang sama dan dishare. Hal ini dapat dilakukan dengan menggunaka dua buah tipe kunci, yaitu kunci publik dan kunci privat, yang saling berhubungan secara matematis.

Pada algoritma kunci publik, kunci privat bersifat rahasia, sedangkan kunci publik dapat didistribusikan secara luas. Dalam arti lain, satu kunci digunakan untuk 'mengunci' sebuah kunci, sedangkan yang lainnya digunakan untuk membuka kunci tersebut.

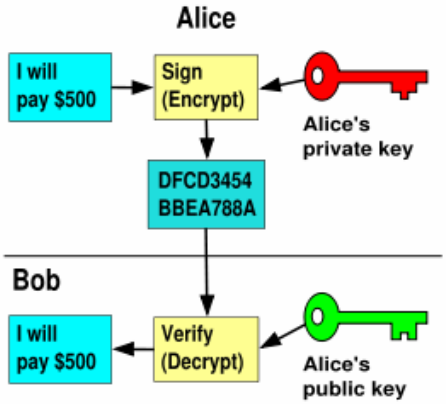
Berikut adalah beberapa hal yang dapat dilakukan menggunakan algoritma kunci publik:



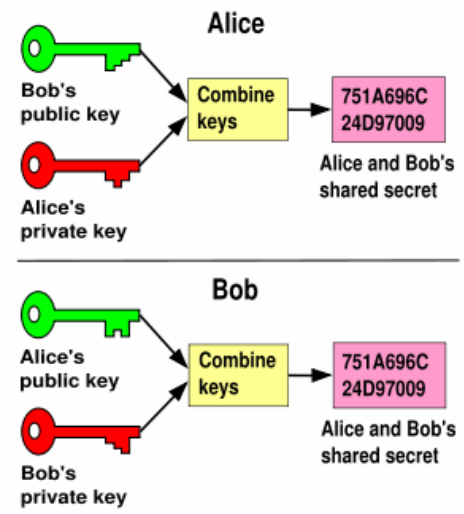
Pertama, nomer random yang sangat besar digunakan untuk membuat pasangan kunci publik dan kunci privat.



Kedua, siapapun dapat mengenkripsi pesan dengan menggunakan kunci publik, tetapi hanya dapat didekripsi dengan kunci privat. Keamanan dari algoritma kunci publik ini terletak pada kerahasiaan kunci privatnya.



Ketiga, dengan menggunakan kunci privat untuk mengenkripsi, siapapun dapat mengecek validitas dari pesan. Validitas ini bergantung pada kerahasiaan kunci privat.



Keempat, dengan menggunakan kunci privat sendiri dan kunci publik orang lain, akan didapat satu kunci rahasia yang sama, yang kedua orang tersebut yang hanya tahu. Kunci yang sama ini dapat digunakan untuk kriptografi kunci simetri.

Aplikasi kriptografi kunci publik dapat dibagi menjadi 3 kategori:

1. Kerahasiaan data
   Seperti pada kriptografi kunci simetri, kriptografi kunci publik dapat digunakan untuk menjaga kerahasiaan data melalui mekanisme enkripsi dan dekripsi. Contoh algoritma untuk aplikasi ini adalah RSA, Knapsack, Rabin, ElGamal, Elliptic Curve Cryptography (ECC). ECC akan dibahas pada makalah ini.

2. Tanda tangan digital
   Tanda tangan digital (*digital signature*) dengan menggunakan algoritma kriptografi kunci publik dapat digunakan untuk membuktikan otentikasi pesan maupun otentikasi pengirim. Contoh algoritmanya untuk aplikasi ini adalah RSA, DSA, dan ElGamal.

3. Pertukaran kunci (*key exchange*)
   Algoritma kriptografi kunci publik dapat digunakan untuk pengiriman kunci simetri. Contoh algoritmanya adalah RSA dan Diffie-Hellman.

Beberapa algoritma kriptografi kunci publik dapat digunakan untuk ketiga macam kategori aplikasi (misalnya RSA), beberapa algoritma

hanya ditujukan untuk aplikasi spesifik (misalnya DSA untuk *digital signature*).

II. Elliptic Curve Cryptography (ECC)

Kriptografi kurva eliptik termasuk kedalam sistem kriptografi kunci publik yang mendasarkan keamanannya pada permasalahan matematis kurva eliptik. Tidak seperti permasalahan matematis logaritma diskrit (*Discrete Logarithm Problem*, DLP) dan pemfaktoran bilangan bulat (*Integer Factorization Problem*, IFP), tidak ada algoritma waktu subeksponensial yang diketahui untuk memecahkan permasalahan matematis logaritma diskrit kurva eliptik (*Elliptic Curve Discrete Logarith Problem*, ECDLP). Karena alasan tersebut, algoritma kriptografi kurva eliptik mempunyai keuntungan jika dibandingkan dengan algoritma kriptografi kunci publik lainnya yaitu dalam hal ukuran panjang kunci yang lebih pendek tetapi memiliki tingkat keamanan yang sama. Ada tiga protokol ECDLP yang diketahui saat ini yaitu *Elliptic Curve Digital Signature Algorithm* (ECDSA), *Elliptic Curve Diffie Hellman* (ECDH), dan *Elliptic Curve ElGamal* (ECElgamal).

Elliptic Curve Cryptography (ECC) adalah salah satu pendekatan algoritma kriptografi kunci publik berdasarkan pada struktur aljabar dari kurva ellips pada daerah finite. Penggunaan kurva ellips dalam kriptografi dicetuskan oleh Neal Koblitz dan Victor S. Miller pada tahun 1985.

Kurva ellips juga digunakan pada beberapa algoritma pemfaktoran integer yang juga memiliki aplikasinya dalam kriptografi, seperti *Lenstra Elliptic Curve Factorization.*

Algoritma kunci publik berdasarkan pada variasi perhitungan matematis yang terbilang sangat sulit dipecahkan tanpa pengetahuan tertentu mengenai bagaimana perhitungan tersebut dibuat. Pembuat algoritma menyimpan kunci privat dan menyebarkan kunci publiknya. Algoritma kunci publik digunakan untuk mengenkripsi pesan dimana hanya pembuat algoritma yang dapat memecahkannya. Sistem kunci publik awal, seperti algoritma RSA, menggunakan dua bilangan prima yang sangat besar: pengguna memilih dua bilangan prima random yang besar sebagai kunci privatnya, dan mempublikasikan hasil dari perhitungannya sebagai kunci publik. Pemfaktoran bilangan-bilangan besar yang sangat sulit dapat menjaga kerahasiaan kunci privat dari publik.

Persoalan lain menyangkut perhitungan aljabar $a^b = c$, dimana a dan c diketahui. Perhitungan semacam itu menyangkut bilangan kompleks atau real yang dapat dengan mudah dipecahkan menggunakan logaritma. Tetapi dalam kumpulan bilangan finite yang besar, menemukan solusi untuk perhitungan semacam itu sangat sulit dan dikenal sebagai "*discrete logarithm problem*".

Kurva ellips dapat ditulis dengan perhitungan matematis sebagai berikut:

$$y^2 = x^3 + ax + b$$

Kumpulan titik pada kurva dapat membentuk kumpulan abelian (dengan titik pada tak terhingga sebagai elemen identitas). Jika nilai x dan y dipilih dari daerah finite yang besar, solusi akan membentuk suatu kumpulan abelian finite. Permasalahan logaritma diskrit pada kumpulan kurva ellips tersebut dipercaya lebih sulit dibandingkan permasalahan yang sama (perkalian bilangan tidak nol) dalam daerah finite. Selain itu, kunci dalam algoritma kriptografi kurva ellips dapat dipilih lebih pendek panjangnya untuk keamanan yang cukup tinggi.

Sebagai salah satu kriptosistem kunci publik, belum ada pembuktian matematis untuk tingkat kesulitan dari ECC yang telah dipublikasikan sampai tahun 2006. Tetapi *U.S. National Security Agency* telah mengesahkan teknologi ECC sebagai algoritma kriptografi yang dianjurkan. Walaupun paten RSA telah kadaluwarsa, masih dibutuhkan usaha mendapatkan paten untuk algoritma ECC.

<u>Pengenalan matematis ECC:</u>

Kurva ellips yang digunakan dalam kriptografi didefinisikan dengan menggunakan dua tipe daerah finite: daerah karakteristik ganjil (F_p, dimana $p > 3$ adalah bilangan prima yang besar) dan daerah karakteristik dua (F_2^m). Karena perbedaan menjadi tidak begitu penting, kedua daerah finite tersebut dapat ditunjukkan sebagai F_q, dimana $q = p$ atau $q = 2^m$. Elemen dari F_p adalah integer ($0 \leq x < p$) dimana elemen tersebut dapat dikombinasikan menggunakan modular aritmetik. Untuk F_2^m sedikit lebih kompleks: salah satu mengandung representasi yang berbeda dari elemen daerah sebagai bitstring untuk tiap pilihan polinomial f(x) biner yang *irreducible* untuk derajat m.

<u>Bidang Terbatas:</u>

Bidang terbatas (*finite field*) atau yang biasa disebut dengan *Galois Field* (GF) adalah bidang yang hanya memiliki elemen bilangan yang terbatas. Derajat (*order*) dari *finite field* adalah banyaknya elemen yang ada di dalam bidang. Jika q adalah pangkat prima (*prime power*), maka hanya ada satu bidang terbatas dengan derajat q. Bidang tersebut dilambangkan dengan F_q atau GF(q).

Banyak cara untuk merepresentasikan elemen dari F_q, jika $q=p^m$, dimana p adalah bilangan prima dan m adalah bilangan integer positif, maka p disebut sebagai karakteristik dari F_q dan m disebut sebagai derajat perluasan (*extension degree*) dari F_q.

Bidang terbatas yang digunakan dalam kriptografi adalah q=p, dimana p adalah bilangan prima ganjil, yang dilambangkan dengan F_p (*odd prime*), dan $q=2^m$, dimana m adalah integer lebih besar dari satu, yang dilambangkan dengan F_2^m (*characteristic two or even*).

Bidang Terbatas F_p:

Bidang Terbatas F_p merupakan sebuah bidang yang beranggotakan bilangan integer {0,1,...,p-1}, dan p merupakan bilangan prima, setiap perhitungan dikalkulasikan dengan modulo p agar hasilnya tetap berada dalam daerah F_p. Operasi yang berlaku dalam bidang terbatas F_p adalah:

1. Penjumlahan (*Addition*), jika a,b Є F_p, maka a + b = r, dimana r adalah sisa pembagian a + b dengan bilangan prima p, $0 \leq r \leq p-1$. Penjumlahan seperti ini disebut penjumlahan modulo p (mod p).
2. Perkalian (*Multiplication*), jika a,b Є F_p, maka a• b = s, dimana s adalah sisa pembagian a • b dengan bilangan prima p, $0 \leq s \leq p-1$. Perkalian seperti ini disebut perkalian modulo p (mod p).

Bidang Terbatas F_2^m :

Bidang terbatas F_2^m biasa disebut dengan bidang terbatas biner (*binary finite field*), dapat dipandang sebagai ruang vektor berdimensi m pada F_2. Karena itu ada himpunan yang beranggotakan m elemen $\{\alpha_0, \alpha_1, ... \alpha_{m-1}\}$di dalam F_2^m sedemikian rupa msehingga setiap a Є F_2^m dapat ditulis secara unik ke dalam bentuk:

$a = a_0\alpha_0+a_1\alpha_1+...+a_{m-1}\alpha_{m-1}$, untuk ai Є {0,1}

Salah satu cara untuk merepresentasikan elemen-melemen pada F_2^m adalah dengan representasi basis polinomial. Pada representasi basis polinomial elemen pada F_2^m merupakan polinomial dengan derajat lebih kecil dari m, dengan koefisien bilangan 0 atau 1.

$\{a_{m-1}x^{m-1} + ... + a_2x^2 + a_1x^1 + a_0x^0 \mid a_i : 0,1\}$

Operasi yang berlaku dalam bidang terbatas F_2^m representasi basis polinomial:

1. Penjumlahan (*Addition*), $(a_{m-1}...a_1a_0) + (b_{m-1}..b_1b_0) = (c_{m-1}...c_1c_0)$ dimana $c_i = a_i + b_i$. Operasi penjumlahan dapat menggunakan deretan komponen $(a_{m-1}...a_1a_0)$ yang di XOR-kan dengan $(b_{m-1}..b_1b_0)$.
2. Perkalian (*Multiplication*), $(a_{m-1}...a1a0) \bullet (bm-1..b1b0) = (r_{m-1}...r_1r_0)$ dimana $r_{m-1}x^{m-1} + ... + r_1x + r_0$ adalah sisa dari pembagian $(a_{m-1}x^{m-1} + ... + a_1x + a_0) . (b_{m-1}x^{m-1} + ... + b_1x + b_0)$ dibagi dengan polinomial f(x) pada F_2 (setiap koefisien polinomial di reduksi ke modulo 2).

Kurva Eliptik Pada Bidang Terbatas:

Ada beberapa cara untuk mendefinisikan persamaan kurva eliptik bergantung kepada bidang terbatas yang digunakan apakah F_p atau F_2^m. Persamaan Weierstrass yang digunakan untuk kedua bidang terbatas tersebut berbeda.

Kurva Eliptik Pada Bidang Terbatas F_p:

Misalkan p > 3 adalah bilangan prima ganjil, dan a,b Є F_p memenuhi

$$4a^3 + 27b^2 \neq 0 \pmod p$$

maka sebuah kurva eliptik $E(F_p)$ pada F_p merupakan himpunan titik-titik P(x,y), dimana x,y Є F_p ,yang memenuhi persamaan :

$$y^2 = x^3 + ax + b,$$

dan sebuah titik khusus φ(∞,∞) yang merupakan titik tak hingga. Operasi penjumlahan pada $E(F_p)$ didefinisikan sebagai berikut :

1. P + φ = φ + P = P untuk setiap P Є $E(F_p)$. Jika P(x.y) Є $E(F_p)$, maka (x,y) + (x,-y) = φ (titik (x,-y) Є $E(F_p)$ dinotasikan sebagai -P, disebut sebagai negatif dari P)
2. Misalkan $P(x_1,y_1)$ Є $E(F_p)$, $Q(x_2,y_2)$ Є $E(F_p)$, dan P ≠ ± Q, maka P + Q = (x_3,y_3) dimana :

$$x_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)^2 - x_1 - x_2$$

$$y_3 = \left(\frac{y_2 - y_1}{x_2 - x_1}\right)(x_1 - x_3) - y_1$$

3. Misalkan $P(x_1,y_1)$ Є $E(F_p)$, maka P + P = 2P = (x_3,y_3), dimana :

$$x_3 = \left(\frac{3x_1^{\,2} + a}{2y_1}\right)^2 - 2x_1$$

$$y_3 = \left(\frac{3x_1^{\,2} + a}{2y_1}\right)(x_1 - x_3) - y_1$$

Operasi di atas disebut dengan penggandaan titik (*doubling a point*) .

Kehebatan dari operasi penjumlahan pada kurva eliptik adalah jika menjumlahkan dua buah titik yang merupakan elemen dari kelompok kurva eliptik, maka hasil penjumlahannya adalah titik lain yang juga merupakan elemen dari kelompok kurva eliptik tersebut.

Kurva Eliptik Pada Bidang Terbatas F_2^m :

Sebuah kurva eliptik E pada F_2^m didefinisikan sebagai sebagai sebuah persamaan dalam bentuk :

$$y^2 + xy = x^3 + ax^2 + b ,$$

dimana a,b Є F_2^m , dan b ≠ 0. Set E(F_2^m) terdiri dari seluruh titik (x,y), x Є F_2^m, y Є F_2^m yang memenuhi persamaan kurva eliptik tersebut, bersamaan dengan titik khusus φ(∞,∞) yang disebut titik tak hingga (*point at infinity*).

Sebagaimana kurva-kurva eliptik pada F_p, ada aturan-aturan untuk menjumlahkan titik–titik pada kurva eliptik E(F_2^m) untuk mendapatkan sebuah titik ketiga kurva eliptik. Rumus aljabar untuk menjumlahkan dua titik dan menggandakan dua titik adalah sebagai berikut:

1. P + φ = φ + P = P untuk seluruh P Є E(F_2^m). Jika P = (x,y) Є E(F_2^m), kemudian (x,y) + (x, x+y) = φ. (Titik (x,x+y) dinotasikan dengan –P, dan disebut negatif P.
2. Misalkan P = (x_1,y_1) Є E(F_2^m) dan Q = (x_2,y_2) Є E(F_2^m) , dimana P ≠ ± Q. Kemudian P + Q = (x_3,y_3) , dimana

$$x_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)^2 + \frac{y_1 + y_2}{x_1 + x_2} + x_1 + x_2 + \alpha$$

$$y_3 = \left(\frac{y_1 + y_2}{x_1 + x_2}\right)(x_1 + x_3) + x_3 + y_1 .$$

3. Penggandaan titik (*Point doubling*) Misalkan P = (x_1,y_1) Є E(F_2^m), kemudian 2P = (x_3,y_3), dimana :

$$x_3 = x_1^{\,2} + \frac{b}{x_1^{\,2}}$$

$$y_3 = x_1^{\,2} + \left(x_1 + \frac{y_1}{x_1}\right)x_3 + x_3$$

ECDSA:

Dalam protokol ECDSA, pihak yang akan melakukan tanda tangan digital, mempunyai parameter domain kurva eliptik berupa D = {q,FR,a,b,G,n,h} dan pasangan kunci kunci rahasia d_A dan kunci publik Q_A. Kemudian pihak yang akan melakukan verifikasi terhadap tanda tangan, memiliki salinan dokumen D yang otentik dan kunci publik Q_A. Proses-proses yang terjadi adalah sebagai berikut :

Key Generation

1. Memilih sebuah bilangan bulat random d_A, yang nilainya diantara [1,n-1]
2. Menghitung $Q_A = d_A \bullet G = (x_1,y_1)$
3. Kunci rahasia = d_A, dan kunci publik = Q_A.

Signing

1. Memilih sebuah bilangan bulat random k, yang nilainya diantara [1,n-1].
2. Menghitung $Q_A = k \bullet G = (x_1,y_1)$ dan r = x_1 mod n, jika r = 0, maka kembali ke langkah 1.
3. Menghitung k^{-1} mod n
4. Menghitung e = Hash(m)
5. Menghitung $s = k^{-1} \{e+d_A \bullet r\}$ mod n tanda tangan Alice untuk *message* m adalah (r,s)

Verifying

1. Memverifikasi bahwa r dan s adalah bilangan bulat yang antara [1,n-1]
2. Menghitung e = Hash (m)
3. Menghitung $w = s^{-1}$ mod n
4. Menghitung u_1 = ew mod n dan u_2 = rw mod n
5. Menghitung $u_1 \bullet G + u_2 \bullet Q_A = (x_1,y_1)$
6. Menghitung v = x_1 mod n
7. Menerima tanda tangan jika dan hanya jika v = r

III. Smart Card

Salah satu aplikasi *Public Key Infrastructure* yang tumbuh sangat pesat adalah kartu cerdas (*smart card*). Kartu cerdas yang mirip dengan kartu kredit dapat melayani banyak fungsi, mulai dari otentikasi sampai penyimpanan data. Dengan menggunakan kartu cerdas, pengguna dapat mengakses informasi dari

berbagai peralatan dengan kartu cerdas yang sama.

Kartu cerdas yang paling populer adalah *memory card* dan *microprocessor card*. *Memory card* mirip dengan *floppy disk*, sedangkan *microprocessor card* mirip dengan komputer kecil dengan sistem operasi, sekuriti, dan penyimpanan data. Kartu cerdas mempunyai beberapa jenis antarmuka (*interface*) yang berbeda. Jenis antarmuka yang umum adalah *contact interface*, yang dalam hal ini kartu cerdas dimasukkan ke dalam alat pembaca (*card reader*) dan secara fisik terjadi kontak fisik antara alat dan kartu.

Kartu cerdas menyimpan kunci privat, sertifikat digital, dan informasi lainnya untuk mengimplementasikan PKI. Kartu cerdas juga menyimpan nomor kartu kredit dan informasi kontak personal (nomer telepon). Sertifikat digital ditandatangani oleh *card issuer* (CA) untuk mensertifikasi kunci publik pemilik kartu.

Penggunaan kartu cerdas dikombinasikan dengan PIN (*Personal Identification Number*). Jadi, ada dua level yang harus dari penggunaan kartu cerdas, yaitu memiliki kartu cerdas itu sendiri dan mengetahui PIN yang mengakses informasi yang disimpan di dalam kartu. Komputer *server* mengotentikasi kartu dengan cara mengirimkan suatu nilai atau *string* (yang disebut *challenge*) ke kartu untuk ditandatangani dengan menggunakan kunci privat (yang tersimpan di dalam kartu), lalu tanda-tangan tersebut diverifikasi oleh mesin dengan menggunakan kunci publik pemilik kartu. Komputer *server* perlu menyimpan kunci publik *card issuer* untuk memvalidasi sertifikat digital.

Banyak peralatan *mobile* yang menggunakan kartu cerdas untuk otentikasi. Namun kartu cerdas masih tidak menjamin keamanan secara total. Jika peralatan *mobile* hilang atau dicuri, sertifikat digital dan kunci privat di dalam kartu cerdas (yang terdapat di dalam peralatan tersebut) berpotensi diakses oleh pencuri untuk mengakses informasi rahasia. Telepon seluler dengan teknologi GSM memiliki kartu cerdas yang terintegrasi di dalam *handphone*. Pemilik *handphone* memiliki opsi untuk men-set PIN untuk proteksi tambahan, sehingga jika *handphone* hilang atau dicuri, *handphone* tidak dapat digunakan tanpa mengetahui PIN tersebut.

Kartu cerdas *Wireless Identity Module (WIM)* termasuk di dalam *Wireless Application Protocol (WAP)*. Kartu WIM memproteksi komunikasi dan transaksi *mobile* dengan tanda-tangan digital. Kartu WIM menyediakan keamanan untuk sertifikat digital, manajemen kode PIN, kunci, dan tanda-tangan digital. WIM menyimpan algoritma enkripsi yang diperlukan di dalam kartu cerdas. Semua fungsi yang diperlukan untuk sistem PKI dimasukkan ke dalam kartu cerdas.

IV. Aplikasi ECC untuk Smart Card

Kurva ellips menghasilkan kriptosistem kunci publik berdasarkan kesulitan dari permasalahan logaritma diskrit kurva ellips, yang disebabkan kesamaannya dengan permasalahan logaritma diskrit / *discrete logarithm problem* (DLP) melalui modulo integer a dan primer p. Kesamaan ini menyebabkan kebanyakan prosedur kriptografi yang menggunakan kriptosistem yang berdasarkan pada DLP yang menggunakan integer modulo p dapat juga menggunakan kriptosistem kurva ellips. Kegunaan lain dari ECC adalah dapat menggunakan panjang kunci yang lebih pendek daripada kriptosistem kunci publik lainnya untuk menghasilkan level keamanan yang sama. Sebagai contohnya, 160 bit kriptosistem kurva ellips (ECCs) dipercaya dapat menghasilkan level keamana yang sama dengan 1024 bit RSA. Selain itu juga peningkatan ukuran kunci untuk mendapatkan tingkat keamanan yang lebih tinggi sangat kecil dibandingkan peningkatan ukuran kunci pada logaritma diskrit berbasis integer (DL) atau RSA hanya untuk peningkatan keamanan yang sama. ECCs juga dapat menghasilkan implementasi yang lebih cepat daripada RSA atau sistem DL, dan juga menghasilkan penggunaan bandwith dan power yang lebih sedikit. Hal tersebut dapat dipakai untuk aplikasi krusial seperti *smart card*.

Pada beberapa tahun terakhir ini, kepercayaan pada tingkat keamanan ECCs terus meningkat, sampai dimana ECC digunakan atau dianjurkan dalam standar internasional yang diakui (secara spesifik yaitu IEEE Std 1363-2000, WAP (*Wireless Application Protocol*), ANSI X9.62, ANSI X9.63 dan ISO CD 14888-3). Selain itu juga, kriptografi kurva ellips diset untuk menjadi bagian integral dari aplikasi ringan pada masa berikutnya.

Sebuah kurva ellips yang melalui daerah Galois dengan elemen p, GF(p), dimana p adalah bilangan prima dan $p > 3$ dapat didefinisikan sebagai titik-titik (x,y) pada persamaan kurva $E : y^2 = x^3 + ax + b \pmod p$,

dimana a dan b adalah konstanta yang memenuhi $4a^3 + 27b^2 \neq 0 \pmod p$. Sebagai tambahan titik-titik yang memenuhi persamaan kurva E, sebuah titik pada daerah tak terhingga, Φ, juga didefinisikan. Dengan definisi yang cocok dari penambahan titik, hal tersebut dapat membuat titik-titik pada sebuah kurva ellips untuk membentuk sebuah grup dengan penambahan titik pada kumpulan operasi, dan titik pada daerah tak hingga yang menjadi elemen identitas. Selanjutnya kita dapat mendefinisikan lebih jauh perkalian skalar dari sebuah titik P dengan sebuah skalar k sebagai hasil dari penambahan titik P tersebut terhadap dirinya sendiri sebanyak k kali (kP = P + P + … + P (k kali)).

Logaritma diskrit kurva ellips didefinisikan sebagai berikut: diberikan bilangan prima modulus p, konstanta kurva a dan b dan dua titik P dan Q, temukan sebuah skalar k sehingga Q = kP. Permasalahan ini tidak mungkin dapat dikerjakan dengan mudah untuk kurva ellips yang aman, dan perkalian skalar adalah operasi kriptosistem yang mendasar dari sebuah kurva ellips.

Untuk mendapatkan implementasi yang efisien, pertama-tama daerah aritmetik yang efisien (penambahan modular, pengurangan, perkalian, dan invers) harus ada. Operasi-operasi ini kemudian digunakan alam algoritma untuk penambahan dan pendobelan titik. Sebagai gantinya, operasi penambahan dan pendobelan tersebut harus efisien agar perkalian skalar yang menggunakan mereka dapat menjadi efisien juga. Hal tersebut dapat memungkinkan penambahan dan pendobelan titik dalam berbagai variasi sistem koordinat. Pilihan sistem koordinat memiliki hasil yang dapat dipertimbangkan pada operasi perkalian skalar final.

Pada masa lampau, banyak penelitian yang terfokus pada kurva yang melalui daerah Galois dengan elemen-elemen 2^m, $GF(2^m)$, karena hal tersebut dapat menghasilkan implementasi hardware yang efisien. Tetapi karena kelebihan kecepatan yang ada pada kurva ellips yang melalui daerah GF(p) dibandingkan $GF(2^m)$ ketika sebuah *coprosesor crypto* untuk aritmetik modular ditemukan, dan karena patent yang sudah ada pada kurva yang melalui daerah $GF(2^m)$, penelitian dilanjutkan pada kurva yang melalui GF(p).

Smart card adalah salah satu alat yang paling populer untuk penggunaan ECC. Banyak perusahaan yang memproduksi *smart card* menggunakan algoritma *digital signature* kurva ellips. Perusahaan-perusahaan ini antara lain adalah Phillips, Fujitsu, MIPS Technologies, dan DataKey; selain itu juga vendor-vendor yang menjual *smart card* ini adalah Funge Wireless dan Entrust Technologies. *Smart card* adalah alat yang fleksibel dan dapat digunakan di banyak situasi dan kondisi. Contohnya, *smart card* dapat digunakan pada kartu kredit bank (*credit debit*), tiket elektronik, dan kartu personal identification (atau registrasi personal).

Smart card yang digunakan pada penelitian adalah Motorola M-Smart Jupiter™ berdasarkan pada teknologi Java Card™ 2.1 dan sebuah ARM prosesor dengan ukuran word 32 bit, 64 KB ROM, 32KB EEPROM, 3KB RAM dan sebuah *accelerator* aritmetik modular. Seluruh operasi ECC diimplementasikan dalam bahasa pemrograman C, dan testing dilakukan pada simulasi *smart card ARM Software Development Toolkit*.

Selain itu juga terdapat beberapa algoritma yang digunakan untuk penambahan dan pendobelan titik pada beberapa variasi sistem koordinat dengan beberapa detail mengenai beberapa variasi kecepatan, penggunaan RAM, ukuran *tradeoff* parameter dan kode yang mungkin. Juga diberikan waktu relatif untuk operasi signature dan verifikasi pada ECDSA (*Elliptic Curve Digital Signature Algorithm*) dan RSA pada tingkat kemananan yang sama. Seluruhnya diujicoba pada komputer PC Pentium III 450 MHz.

Implementasi:

Untuk membangun sebuah kriptosistem kurva eliptik ada tiga hal yang harus diperhatikan, yaitu :

1. Pemilihan bidang terbatas F_q dan representasi elemen dari F_q, implementasi yang dipilih : F_p dengan representasi elemen berupa bilangan bulat yang sangat besar.
2. Pemilihan Kurva Eliptik E pada F_q, tidak semua kurva eliptik 'aman' digunakan untuk kriptografi. Menurut [9] , syarat yang harus dipenuhi adalah :
    a. Jumlah titik pada kurva E atau derajat kurva E, $\#E(F_q)$, harus dapat dibagi oleh sebuah bilangan prima n yang cukup besar.
    b. $\#E(F_q) \neq q$

c. n tidak membagi q^k -1 untuk semua $1 \leq k \leq 20$
d. Implementasi yang dipilih : kurva eliptik yang dibangkitkan dengan cara metoda perkalian komplek (*Complex Multiplication Method*).

3. Penentuan protokol kurva eliptik, implementasi yang dipilih : protokol ECDSA.

Perangkat lunak yang diimplementasikan diberi nama EDiS (*Elliptic Curve Digital Signature*), disamping itu juga dikembangkan penandatanganan dokumen elektronis dengan menggunakan sistem kriptografi kunci publik lainnya yaitu penandatanganan dokumen menggunakan algoritma RSA. Kemudian kedua algoritma ini diperbandingkan tingkat keamanan dan performansinya.

ECDSA *Smart Card* Simulation Data:

Tabel 1 menunjukkan verifikasi dan signature waktu, ukuran kode, dan jumlah variabel yang disimpan (tidak digunakan) dari ECDSA (*Elliptic Curve Digital Signature Algorithm*) untuk tiap efisiensi pilihan yang berbeda. Data ini juga ditunjukkan secara grafis pada Gambar 1, 2, dan 3.

Tabel 2, 3, dan 4 menunjukkan algoritma untuk penambahan dan mendobelan dalam koordinat dan variabel Jacobian.

Perbandingan ECDSA dan RSA:

Tabel 5 merupakan perbandingan antara kecepatan signing dan verifikasi ECDSA (*Elliptic Curve Digital Signature Algorithm*) dengan kecepatan signing dan verifikasi dari RSA pada *smart card*. Dapat diambil kesimpulan bahwa waktu verifikasi RSA lebih cepat karena hanya sedikit exponen yang digunakan. Pada tabel tersebut dipaparkan waktu yang digunakan sebagai persentase dari waktu signature EC (*Elliptic Curve*) pada platform yang ada. Hasilnya menunjukkan bahwa rasio waktu yang digunakan untuk operasi EC dengan waktu yang digunakan untuk signature RSA adalah sama untuk *smart card* dan Pentium. Tetapi pada *smart card*, signature RSA kebanyakan dilakukan pada hardware, walaupun sebenarnya kebanyakan komputasi EC seharusnya dilakukan di software, yang dimana akan memperlambat EC. Waktu signature RSA pada *smart card* lebih lama dibandingkan waktu pada Pentium, dimana hal tersebut dapat dilihat bahwa ECDSA sangat cocok diimplementasikan pada smart card dibandingkan RSA.

Tingkat Keamanan:

Yang dimaksud dengan tingkat keamanan pada sistem kriptografi kunci publik adalah berapa waktu yang diperlukan untuk memecahkan suatu kunci rahasia berdasarkan persamaan matematis yang dimiliki oleh algoritma kriptografinya. RSA termasuk ke dalam persamaan matematis *Integer Factorization Problem* (IFP) sedangkan ECDSA termasuk ke dalam *Elliptic Curve Discrete Logarithm Problem* (ECDLP). Tingkat keamanan dihitung berdasarkan panjang kunci dari masing-masing algoritma kriptografi, parameter kunci RSA yang digunakan adalah panjang bit n , yaitu perkalian antara faktor prima p dan q, sedangkan untuk ECDSA parameter kunci yang digunakan juga panjang bit n, tetapi merupakan orde dari titik basis yang digunakan dalam persamaan kurva eliptik.

Untuk memecahkan persamaan matematis tersebut harus digunakan software dan hardware yang terbaik. Menurut [24] algoritma terbaik yang diketahui untuk menyelesaikan IFP pada RSA adalah algoritma *General Purposed Number Field Sieve* yang memiliki kompleksitas algoritma $O = \exp [1{,}923 (\ln n)^{1/3} (\ln \ln n)^{2/3}]$, sedangkan untuk menyelesaikan ECDLP pada ECDSA adalah *Pollard Rho Method Attacks* yang memiliki $O = 2^{n/2}$. Jika diasumsikan hardware yang digunakan mampu menjalankan 1000000 instruksi per detik (1 MIPS (*Million Instruction per Second*)) maka akan dihitung tingkat keamanan kunci ECDSA sebagai berikut : Misalkan untuk n = 149 bit , maka tingkat keamanan dihitung sebagai berikut :
MIPS = $2^{149/2}$ / 1000000.3600.24.365 = 598981035 MIPS years

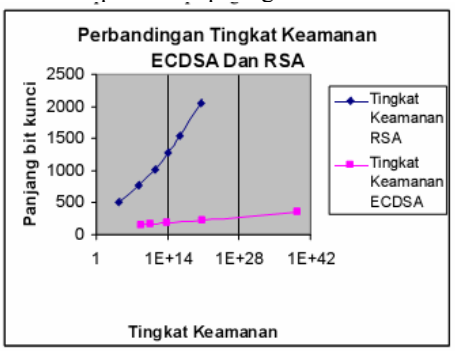
Dengan cara yang sama dihitung tingkat keamanan untuk kunci dengan panjang bit n yang berbeda-beda sehingga diperoleh tabel hubungan panjang kunci ECDSA dengan tingkat keamanannya sebagai berikut :

No	Ukuran n (bit)	Tingkat Keamanan (MIPS year) (≈)
1	149	6.10^{8}
2	160	$3{,}8.10^{10}$
3	183	$7{,}9.10^{13}$
4	229	$6{,}6.10^{20}$
5	353	3.10^{39}
6	512	$3{,}7.10^{63}$
7	768	$1{,}2.10^{102}$
8	1024	$4{,}3.10^{140}$
9	1280	$1{,}4.10^{179}$
10	1536	$4{,}9.10^{217}$
11	2048	$5{,}7.10^{294}$

Sedangkan untuk hubungan panjang kunci RSA dan tingkat keamanannya menurut [25] dapat dilihat dalam tabel berikut ini:

No	Ukuran n (bit)	Tingkat Keamanan (MIPS year)
1	512	30000
2	768	2.10^{8}
3	1024	3.10^{11}
4	1280	1.10^{14}
5	1536	3.10^{16}
6	2048	3.10^{20}

Hubungan antara tingkat keamanan RSA dan ECDSA dapat dilihat pada grafik berikut ini :



Performansi:

Untuk membahas tingkat performansi dari ECDSA maupun RSA ada tiga kriteria yang menjadi pertimbangan yaitu :

Ukuran pajang kunci, kunci publik RSA adalah pasangan (n,e), dimana n adalah modulo sedangkan e adalah kunci publik. Jika sistem kriptografi RSA yang dibangun 1024 bit, maka tentunya n juga mempunyai panjang 1024 bit dan kunci publik yang digunakan adalah e = 2^{16} + 1=65537. Ukuran kunci publik RSA yang diperlukan adalah 128 bytes untuk modulo dan 3 bytes untuk kunci publik, sehingga totalnya adalah 131 bytes. Sedangkan jika sistem kriptografi ECDSA menggunakan 160 bit, maka panjang kunci publik ECDSA adalah sebuah titik pada kurva Q(x,y) yang masing masing elemennya , x dan y, juga mempunyai panjang 160 bit. Sehingga total ukuran kunci publik ECDSA adalah 160 bit = 40 bytes, yang jauh lebih kecil jika dibandingkan dengan RSA.

Ukuran panjang tanda tangan digital, panjang tanda tangan digital ECDSA = 320 bit (2 x 160 bit, tanda tangan merupakan pasangan r dan s yang masing-masing panjangnya 160 bit), atau 40 bytes. Sedangkan ukuran tanda tangan digital RSA adalah 1020 bit ≈ 1024 bit = 128 bytes. Sehingga total ukuran tanda tangan digital ECDSA jauh lebih kecil dari RSA.

Kecepatan proses signing dan verifying, waktu yang diperlukan untuk proses signing dan verifying dapat dilihat pada tabel berikut ini :

Proses	ECDSA	RSA
Operasi Kunci Rahasia (pembangkitan tanda tangan digital)	Cepat	Lambat
Operasi Kunci Publik (verifikasi tanda tangan digital)	Lambat	Cepat

Berdasarkan tabel di atas ECDSA baik digunakan untuk proses yang banyak menggunakan pembangkitan tanda tangan digital, misalnya digunakan oleh orang yang sering menggunakan webmail karena di setiap suratnya, dia harus selalu menandatanganinya. Sebaliknya RSA baik digunakan untuk proses yang sering melakukan verifikasi tanda tangan digital, misalnya pihak CA (*Certification Authority*) yang hanya menadatangani sertifikat kunci publik sekali saja tetapi sertifikat tersebut nantinya akan sering diverifikasi orang lain.

V. Kesimpulan

Kesimpulan yang dapat ditarik sehubungan dengan tingkat keamanan dan performansi dari algoritma kriptografi ECDSA maupun RSA adalah :

1. ECDSA dengan panjang kunci 160 bit mempunyai tingkat keamanan yang relatif sama dengan RSA dengan panjang kunci 1024 bit. Jadi algoritma kriptografi kurva eliptik mempunyai keuntungan berupa ukuran panjang kunci yang lebih kecil jika dibandingkan dengan algoritma kunci publik lainnya (RSA) tetapi sudah memiliki tingkat keamanan yang relatif sama., sehingga algoritma kriptografi kurva eliptik cocok untuk

diimplementasikan pada peralatan perangkat keras yang memiliki daya dan memori yang terbatas

2. Dari kriteria ukuran panjang tanda tangan digital, algoritma kriptografi kurva eliptik memiliki performansi yang lebih baik karena menghasilkan tanda tangan digital yang mempunyai ukuran lebih kecil. Sedangkan dari kriteria kecepatan proses signing dan verifying, performansi kriptografi kurva eliptik akan lebih baik jika proses signing lebih sering dilakukan. Sebaliknya performansi kriptografi RSA akan lebih baik jika proses verifying lebih sering dilakukan.

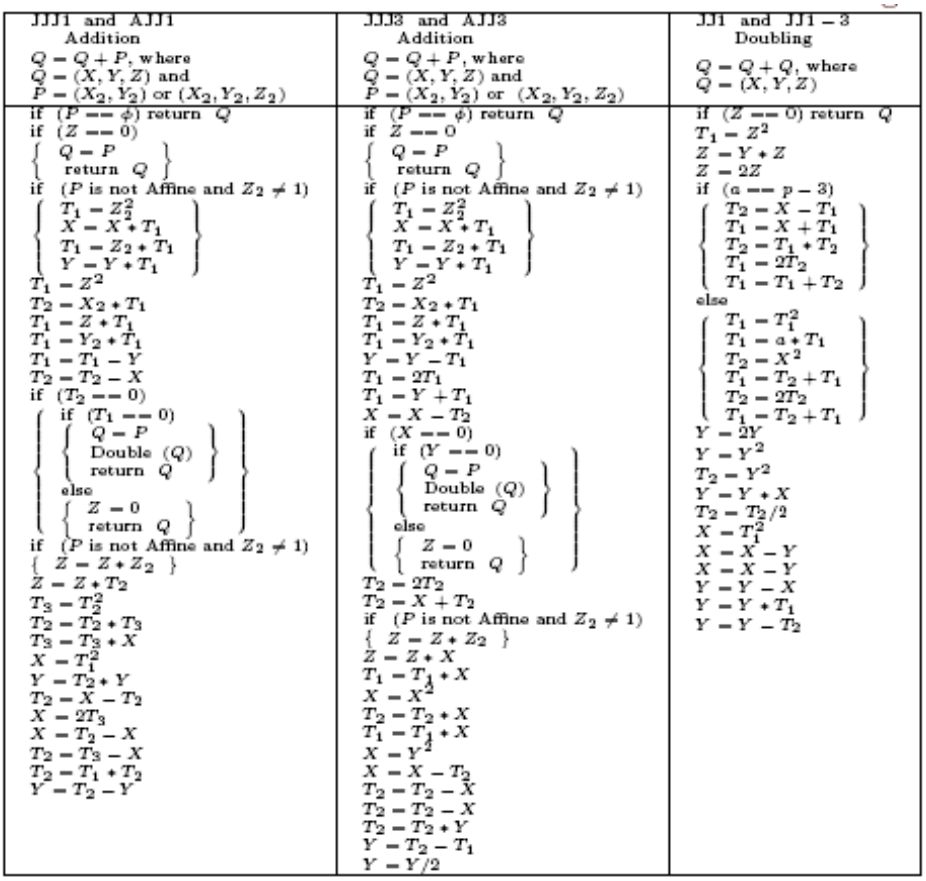
VI. Saran

Saran untuk pengembangan di masa mendatang :

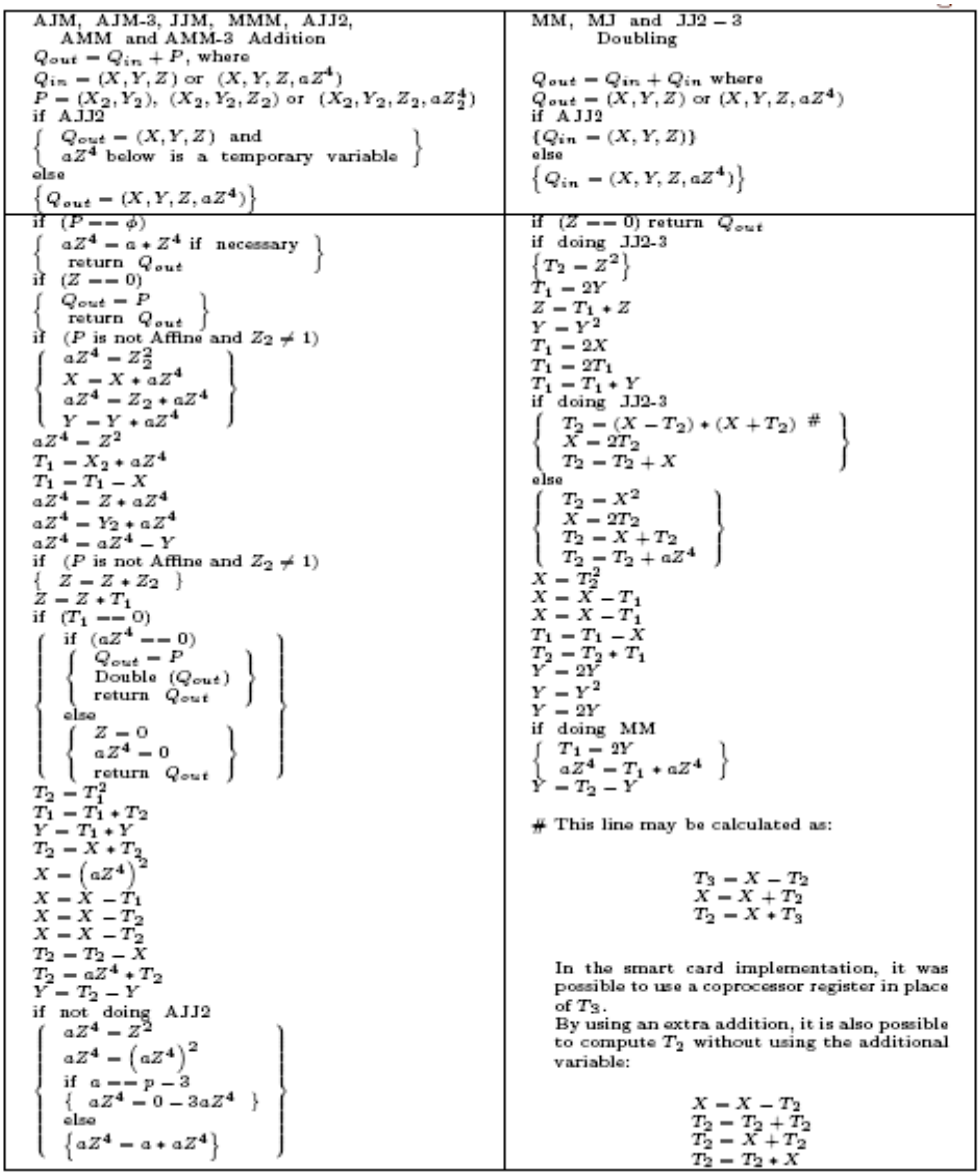
1. Kriptografi kurva eliptik memiliki dua buah bidang terbatas yaitu F_p dan F_2^m , dan yang diimplementasikan adalah bidang terbatas F_p, saran penulis adalah bidang terbatas F_2^m juga diimplementasikan, kemudian dicoba untuk dibandingkan bagaimana performansi keduanya.
2. Algoritma kriptografi kurva eliptik lain yang dapat digunakan untuk penandatanganan dokumen elektronis adalah ECElGamal, saran penulis adalah algoritma kriptografi ECElGamal tersebut juga diimplementasikan untuk kemudian diketahui bagaimana performansinya jika dibandingkan dengan algoritma kriptografi kurva eliptik ECDSA.

2-in-1 Multiplication	Number of Temporary Points	Signed (NAF) Scalar Used	Addition Algorithm	Doubling Algorithm(s)	ECDSA Signature Time (Simulated)	ECDSA Verification Time (Simulated)	Code Size (bytes)	Variables Saved in Signature	Variables Saved in Verification
no	0	no	AJM-3	MJ/MM	42.9%	84.3%	5324	3	7
no	0	no	AJJ2	JJ2-3	42.3%	86.1%	5060	3	7
no	0	no	AJM	MJ/MM	42.7%	85.9%	5280	2	6
no	0	no	AJJ1	JJ1-3	44.4%	90.0%	5068	3	7
no	0	no	AJJ1	JJ1	47.7%	93.7%	5120	2	6
no	0	no	AJJ3	JJ1-3	46.9%	96.1%	5156	4	8
no	0	no	AJJ3	JJ1	51.2%	100.0%	5204	3	7
no	0	yes	AJM-3	MJ/MM	40.1%	77.1%	5640	1	6
no	0	yes	AJJ2	JJ2-3	40.7%	78.3%	5380	1	6
no	0	yes	AJM	MJ/MM	40.3%	76.9%	5592	0	5
no	0	yes	AJJ1	JJ1-3	42.6%	82.3%	5388	1	6
no	0	yes	AJJ1	JJ1	45.0%	86.6%	5436	0	5
no	0	yes	AJJ3	JJ1-3	44.6%	86.3%	5472	2	7
no	0	yes	AJJ3	JJ1	47.2%	90.5%	5520	1	6
yes	0	no	AJM-3	MJ/MM	42.9%	59.0%	5332	3	7
yes	0	no	AJJ2	JJ2-3	42.4%	60.0%	5068	3	7
yes	0	no	AJM	MJ/MM	42.7%	60.9%	5288	2	6
yes	0	no	AJJ1	JJ1-3	44.4%	62.1%	5076	3	7
yes	0	no	AJJ1	JJ1	47.8%	63.5%	5124	2	6
yes	0	no	AJJ3	JJ1-3	46.9%	68.2%	5164	4	8
yes	0	no	AJJ3	JJ1	51.3%	69.4%	5212	3	7
yes	0	yes	AJM-3	MJ/MM	40.2%	50.6%	5712	1	5
yes	0	yes	AJJ2	JJ2-3	40.8%	50.9%	5456	1	5
yes	0	yes	AJM	MJ/MM	40.2%	50.6%	5664	0	4
yes	0	yes	AJJ1	JJ1-3	42.7%	53.0%	5464	1	5
yes	0	yes	AJJ1	JJ1	45.1%	56.1%	5512	0	4
yes	0	yes	AJJ3	JJ1-3	44.7%	57.7%	5548	2	6
yes	0	yes	AJJ3	JJ1	47.3%	60.1%	5596	1	5
yes	1	no	AJM-3	MJ/MM	42.9%	52.3%	5380	3	5
yes	1	no	AJJ2	JJ2-3	42.4%	52.4%	5112	3	5
yes	1	no	AJM	MJ/MM	42.7%	52.5%	5332	2	4
yes	1	no	AJJ1	JJ1-3	44.4%	55.1%	5120	3	5
yes	1	no	AJJ1	JJ1	47.8%	57.7%	5172	2	4
yes	1	no	AJJ3	JJ1-3	46.9%	59.9%	5208	4	6
yes	1	no	AJJ3	JJ1	51.2%	62.2%	5256	3	5
yes	1	yes	AJM-3	MJ/MM	40.1%	49.4%	5924	1	3
yes	1	yes	AJJ2	JJ2-3	40.8%	50.0%	5652	1	3
yes	1	yes	AJM	MJ/MM	40.3%	49.3%	5876	0	2
yes	1	yes	AJJ1	JJ1-3	42.7%	52.2%	5660	1	3
yes	1	yes	AJJ1	JJ1	45.1%	54.6%	5704	0	2
yes	1	yes	AJJ3	JJ1-3	44.7%	56.9%	5740	2	4
yes	1	yes	AJJ3	JJ1	47.2%	58.4%	5792	1	3
yes	2	yes	AJM-3	MJ/MM	40.0%	49.5%	5828	1	1
yes	2	yes	AJJ2	JJ2-3	40.7%	50.2%	5548	1	1
yes	2	yes	AJM	MJ/MM	40.3%	49.2%	5780	0	0
yes	2	yes	AJJ1	JJ1-3	42.6%	52.2%	5556	1	1
yes	2	yes	AJJ1	JJ1	45.1%	55.0%	5612	0	0
yes	2	yes	AJJ3	JJ1-3	44.6%	56.5%	5640	2	2
yes	2	yes	AJJ3	JJ1	47.2%	58.4%	5696	1	1

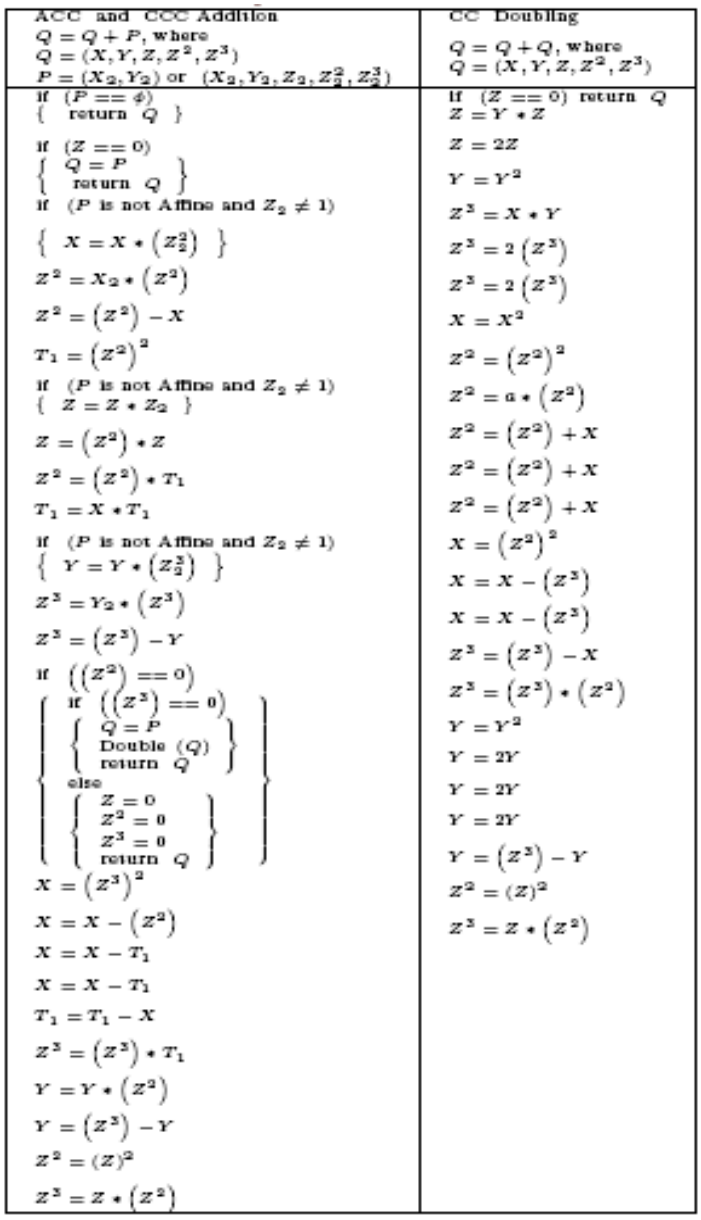
Tabel 1 Ukuran kode, jumlah variabel yang disimpan, dan waktu pada smart card.



Tabel 2 Jacobian 1 dan 3 Addition dan Jacobian 1 Doubling



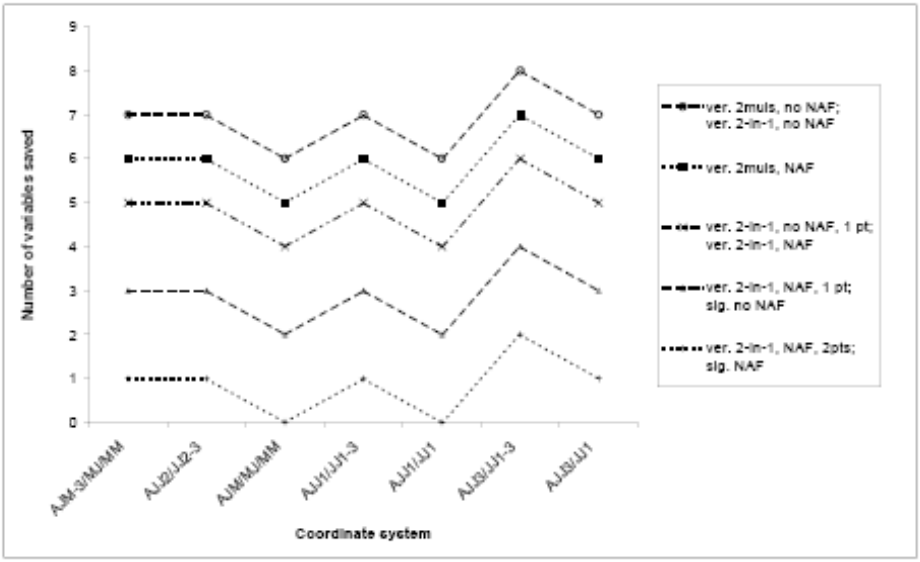
Tabel 3 Jacobian yang telah dimodifikasi dan Variants Addition dan Doubling.



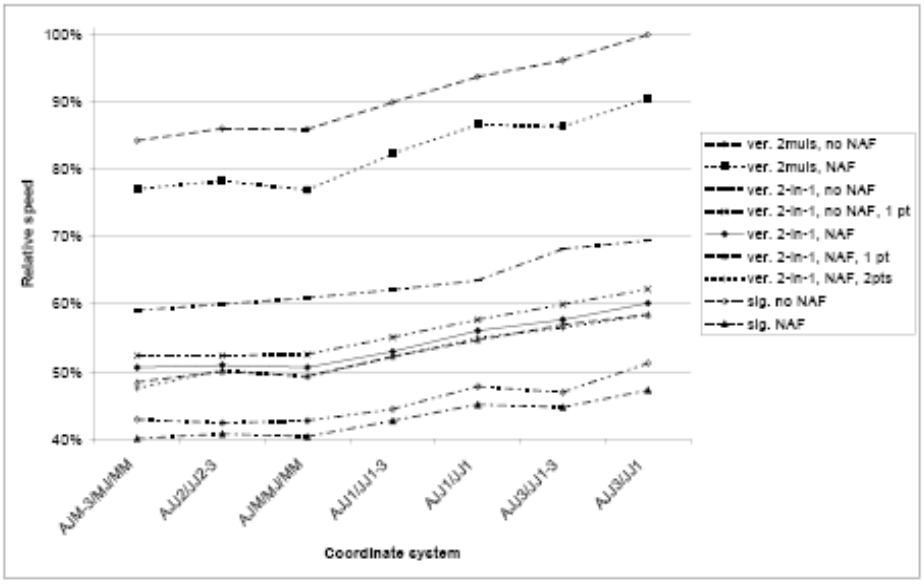
Tabel 4 Chudnovsky Jacobian Addition dan Doubling

Algorithm		MIRACL Library Pentium III 450 MHz	Our library Smart Card
160 bit ECDSA	Signature	100%	100%
	Verification	121%	122%
1024 bit RSA	Signature	220%	236%
	Verification	29%	24%

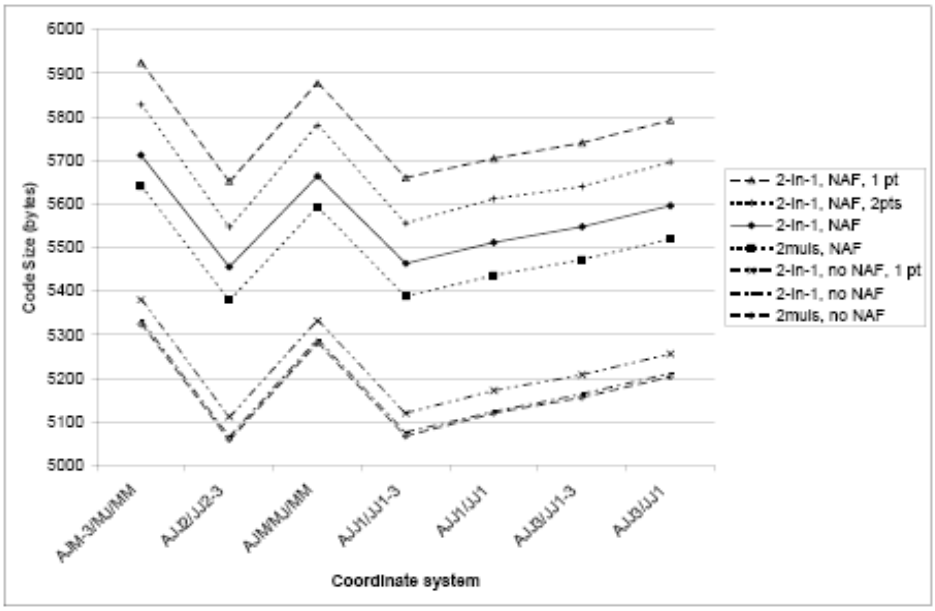
Tabel 5 Perbandingan waktu ECDSA dan RSA pada smart card dan Pentium



Gambar 1 Jumlah variabel yang disimpan untuk tiap pilihan berbeda pada ECDSA



Gambar 2 ECDSA signature dan verification timings pada simulator smart card.



Gambar 3 Ukuran code untuk ECDSA signature dan verification pada smart card.

VII. Referensi

[1] V. Miller, *Use of elliptic curves in cryptography*, CRYPTO 85, 1985.

[2] A. Menezes, T. Okamoto, and S.A. Vanstone, *Reducing elliptic curve logarithms to logarithms in a finite field,* IEEE Transactions on Information Theory, Volume 39, 1993.

[3] NIST, *Recommendation for Key Management*, Special Publication 800-57, August 2005.

[4] Y. Hitchcock, E. Dawson, A. Clark, and P. Montague, *Implementing an efficient elliptic curve cryptosystem over GF(p) on a smart card*, 2002.

[5] N. Koblitz, *Elliptic curve cryptosystems*, in *Mathematics of Computation* 48, 1987, pp. 203–209.

[6]G. Lay and H. Zimmer, *Constructing elliptic curves with given group order over large finite fields,* Algorithmic Number Theory Symposium, 1994.

[7] I. Semaev, *Evaluation of discrete logarithm in a group of P-torsion points of an elliptic curve in characteristic P,* Mathematics of Computation, number 67, 1998.

[8] N. Smart, *The discrete logarithm problem on elliptic curves of trace one,* Journal of Cryptology, Volume 12, 1999.

[9] Lopez, Julio. Dahab, Ricardo. (2000). *An Overview of Elliptic Curve Cryptography*. http://citeseer.ist.psu.edu/cache/papers/ , Agustus 2004.

[10] H. Cohen, A. Miyaji, T. Ono, *Efficient Elliptic Curve Exponentiation Using Mixed Coordinates*, ASIACRYPT 1998.

[11] M. Brown, D. Hankerson, J. Lopez, and A. Menezes, *Software Implementation of the NIST Elliptic Curves Over Prime Fields.*

[12] M. Hedabou, P. Pinel, and L. Beneteau, *A comb method to render ECC resistant against Side Channel Attacks*, 2004.

[13] Atmel Corporation. *Motorola chooses Atmel technology for its M-Smart Jupiter smart card platform*, Press release. http://www.armltd.co.uk/sitearchitek/news.ns4/916d9b71de2065938025694400358dec/cd91a1cbf6c291228025692f002a0018!OpenDocument

[14] Daniel V. Bailey and Christof Paar. *Optimal extension fields for fast arithmetic in public-key algorithms*. In Advances in Cryptology—Crypto '98, volume 1462 of Lecture Notes in Computer Science, pages 472–485. Springer-Verlag, 1998.

[15] Ian Blake, Gadiel Seroussi, and Nigel Smart. *Elliptic Curves in Cryptography*, volume 265 of London Mathematical Society Lecture Note Series. Cambridge University Press, Cambridge, 1999

[16] Alfred J. Menezes, Paul C. van Oorschot, and Scott A. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[17] Rudolf Lidl and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge University Press, 1986.

[18] Toshio Hasegawa, Junko Nakajima, and Mitsuru Matsui. *A practical implementation of elliptic curve cryptosystems over GF(p) on a 16-bit microcomputer*. In Public Key Cryptography – PKC '98,Proceedings, volume 1431 of Lecture Notes in Computer Science, pages 182–194. Springer-Verlag, 1998.

[19] Erik De Win, Serge Mister, Bart Preneel, and Michael Wiener. *On the performance of signature schemes based on elliptic curves*. In Algorithmic Number Theory: Third International Symposium, ANTS-III, Proceedings, volume 1423 of Lecture Notes in Computer Science, pages 252–266. Springer-Verlag, 1998.

[20] Certicom Corp. *The elliptic curve cryptosystem for smart cards*, The seventh in a series of ECC white papers. http://www.certicom.com/research.html

[21] M. Brown, D. Hankerson, J. L´opez, and A. Menezes. Software implementation of the NIST elliptic curves over prime fields. In Topics in Cryptology—CT-RSA 2001, volume 2020 of Lecture Notes in Computer Science, pages 250–265. Springer-Verlag, 2001. http://www.cacr.math.uwaterloo.ca/~ajmeneze/research.html

[22] Munir, Rinaldi. Diktat Kuliah IF5054 Kriptografi. STEI. 2006

[23] T. Satoh and K. Araki, *Fermat quotients and the polynomial time discrete log algorithm for anomalous elliptic curves,* Commentarii Mathematici Universitatis Sancti Pauli, Volume 47, 1998.

[24] A Certicom Whitepaper. (2000). *The Elliptic Curve Cryptosystem*. http://www.certicom.com , Agustus 2004.

[25] Schneier, Bruce.(1996). *Applied Cryptography, second edition*. John Wiley & Sons, inc.

[26] Johnson, Don. Menezes, A. Vanstone, S. (2000). *The Elliptic Curve Digital Signature Algorithm (ECDSA)*. http://www.certicom.com , Desember 2004.