Perbandingan Sistem Kriptografi Kunci Publik RSA dan ECC

Abu Bakar Gadi – NIM : 13506040[1)]
1) Jurusan Teknik Informatika ITB, Bandung, email: abu_gadi@students.itb.ac.id

Abstrak – *Makalah ini akan membahas topik yang berhubungan dengan sistem kriptografi, yaitu masalah algoritma RSA (Rivest-Shamir-Adleman) dan algoritma ECC (Elliptic Curve Cryptography). Kedua algoritma tersebut merupakan contoh dari sistem kriptografi nirsimetri, sehingga biasa disebut sebagai algoritma nirsimetri.*
Sistem kriptografi nirsimetri disebut juga sebagai kriptografi kunci-publik. Sistem kriptografi kunci publik, pada saat ini merupakan salah satu teknologi yang sangat diperlukan, mengingat betapa pesatnya perkembangan dunia teknologi informasi, yang tentu saja sangat mengutamakan masalah kerahasiaan dari tiap penggunanya.
Algoritma RSA saat ini adalah algoritma yang umum dipakai dalam sistem kriptografi kunci publik. Selain algoritma RSA, telah dikenal juga algoritma ECC yang menjadi alternatif yang paling bepotensi untuk menggantikan algoritma RSA, karena performansi dan efisiensinya yang lebih unggul dibandingkan algoritma RSA. Untuk lebih jelasnya akan penulis jabarkan perbandingan antara kedua algoritma tersebut pada bagian-bagian selanjutnya.

Kata Kunci*: Aritmetika Modulo, Kriptografi, Enkripsi, Dekripsi, RSA, ECC.*

1. PENDAHULUAN

Aritmetika modulo memainkan peranan yang penting dalam komputasi integer, terutama pada sistem kriptografi. Operator yang digunakan dalam aritmetika modulo ini adalah *mod*. Operator *mod* adalah operator utama dalam bidang kriptografi. Definisi dari operator *mod* adalah sebagai berikut:
Misalkan x adalah bilangan bulat dan m adalah bilangan bulat > 0. Operasi x **mod** m (dibaca ”x modulo m”) memberikan sisa jika x dibagi dengan m. Dengan kata lain, x **mod** m = r sedemikian sehingga $x = mq + r$, dengan $0 \leq r < m$, q dan r bilangan bulat.

Selain aritmetika modulo, hal lain yang juga berperan penting dalam kriptografi adalah teori bilangan prima. Suatu bilangan positif *p* disebut bilangan prima jika dan hanya jika *p* hanya habis dibagi oleh 1 dan *p*. Untuk menentukan keprimaan suatu bilangan, dapat digunakan **Teorema Fermat** sebagai berikut:

Jika *p* adalah bilangan prima dan *q* adalah bilangan bulat yang tidak habis dibagi dengan *p*, maka

$$q^{p-1} \equiv 1 \ (\mathbf{mod}\ p) \qquad (1)$$

(catatan : teorema ini tidak berlaku untuk bilangan prima semu terhadap basis 2 , yaitu bilangan *n* sedemikian sehingga $2^{n-1} \equiv 1$ (**mod** n))

Seperti yang telah penulis ungkapkan diatas, aritmetika modulo dan bilangan prima sangat berperan penting dalam kriptografi. Kriptografi adalah ilmu sekaligus seni untuk menjaga keamanan pesan [6]. Keamanan pesan dapat diperoleh dengan menyandikannya menjadi sesuatu yang tidak dapat dimengerti maknanya. Pada era berkembangnya teknologi informasi seperti saat ini, kerahasiaan informasi menjadi sangat penting. Untuk merahasiakan suatu informasi dari orang lain, diperlukan teknik untuk menyamarkan informasi tersebut. Karena kebutuhan akan menyamarkan informasi yang bersifat rahasia, dikembangkanlah suatu teknik yang kita kenal sebagai kriptografi.

Suatu informasi yang dirahasiakan dinamakan **plaintext** (tulisan yang jelas maksudnya), sedangkan informasi yang telah disandikan disebut **ciphertext** (tulisan yang tersandi). Dalam kriptografi, dikenal dua buah proses utama, yaitu proses **enkripsi** (encryption) dan **dekripsi** (decryption). Proses enkripsi adalah proses yang menyandikan plaintext menjadi ciphertext. Proses dekripsi adalah proses kebalikan dari enkripsi, yaitu membalikkan ciphertext menjadi plaintext.

Sebagai contoh, sebuah informasi rahasia (berupa plaintext) sebagai berikut :

Aku Suka Matematika Diskrit

disandikan menjadi sebuah ciphertext dengan suatu teknik kriptografi tertentu menjadi :

k*jad7qeo@8disuo1%a92kbnaivmz

Ciphertext hasil dari proses enkripsi, walaupun tidak dirahasiakan, namun isinya menjadi tidak bermakna dan tidak dapat dimengerti artinya.

Proses kriptografi dapat kita temukan dalam dua hal, yaitu proses pengiriman data melalui saluran komunikasi dan proses penyimpanan informasi pada media penyimpanan (*storage media*). Pada proses pengiriman data, data yang berupa informasi, ditransmisikan melalui saluran komunikasi dalam bentuk ciphertext. Sehingga pada bagian penerima data, harus dilakukan proses dekripsi untuk mengembalikan ciphertext menjadi plaintext.

Pada proses penyimpanan data dalam media penyimpanan, data berupa informasi yang akan disimpan, dienkripsi sehingga menjadi ciphertext saat disimpan. Untuk membaca data tersebut, hanya orang yang berhak yang dapat mendekripsi data tersebut. Hal-hal yang menyangkut dua proses tersebut banyak kita temui dalam kehidupan sehari-hari. Sebagai contoh dari proses enkripsi dalam pengiriman data adalah perdagangan elektronis (e-commerce) yang saat ini telah dipergunakan secara luas dikalangan masyarakat umum. Dalam e-commerce, proses pembayaran umumnya menggunakan kartu kredit, sehingga diperlukan suatu sistem pengamanan berupa penyamaran identitas untuk menjaga kerahasiaan segala identitas pengguna. Contoh dari penggunaan enkripsi dalam penyimpanan data dalam media penyimpanan adalah pada kartu elektronis, seperti kartu chip yang berisi data rahasia contohnya. Untuk alasan keamanan dan privasi, dapat digunakan proses enkripsi dalam penyimpanan informasi rahasia dalam chip tersebut.

2. ALGORITMA KRIPTOGRAFI

Misalkan ciphertext sebagai C dan plaintext sebagai P, maka fungsi enkripsi E memetakan P ke C,

$$E(P) = C \quad (2)$$

Pada proses dekripsi, fungsi D memetakan C ke P,

$$D(C) = P \quad (3)$$

Berarti fungsi D adalah inversi dari E, atau $D = E^{-1}$.

Algoritma kriptografi adalah fungsi matematika yang digunakan untuk proses enkripsi dan dekripsi. Kekuatan suatu algoritma kriptografi ditentukan dari banyaknya kerja yang harus dilakukan untuk dapat memecahkan suatu informasi tersandi menjadi informasi aslinya. Kerja ini dapat disetarakan dengan waktu. Semakin banyak usaha yang diperlukan, berarti makin lama waktu yang dibutuhkan untuk memecahkan informasi tersandi, berarti pula makin kuat algoritma kriptografinya, maka semakin aman untuk digunakan dalam penyandian pesan.

Apabila kekuatan kriptografi ditentukan dengan penjagaan kerahasiaan algoritmanya, maka algoritma kriptografi tersebut dinamakan algoritma *restricted*. Sebagai contoh dari algoritma *restricted*, misalkan dalam suatu tim agar kerahasiaan informasi terjaga, disepakati untuk menyandikan setiap pesan yang ada dengan suatu algoritma tertentu. Sehingga untuk dapat mendekripsinya diperlukan algoritma yang hanya diketahui oleh anggota dari tim tersebut. Namun nampaknya saat ini penggunaan metode semacam ini tidak lagi dapat diterapkan. Karena apabila ada anggota tim yang keluar, maka kerahasiaan pesan dalam tim tidak dapat lagi diandalkan.

Kriptografi modern tidak lagi mendasarkan kekuatannya pada kerahasiaan algoritma yang digunakan. Algoritma yang digunakan kini boleh diketahui oleh umum, karena kekuatan kriptografinya bergantung pada sebuah kunci, yaitu berupa sederetan karakter ataupun bilangan bulat. Kunci inilah yang dapat melakukan enkripsi dan dekripsi terhadap informasi yang ada. Sebenarnya, ide penggunaan metode kunci seperti ini telah ada sejak masa kekaisaran Romawi. Metode yang digunakan kaisar pada saat itu adalah dengan penggeseran huruf sejauh *k*. Metode ini kita kenal sebagai *Caesar Cipher*, yang secara umum fungsinya adalah sebagai berikut :

$$c = E(p) = (p + k) \bmod 26 \quad (4)$$

sebagai fungsi enkripsi dan

$$p = D(c) = (c - k) \bmod 26 \quad (5)$$

sebagai fungsi dekripsi. *k* dalam kedua fungsi di atas berperan sebagai kunci yang harus dirahasiakan.

Secara matematis, fungsi enkripsi dan dekripsi dapat dituliskan sebagai berikut :

$$E_{K1}(P) = C \quad (6)$$

dan

$$D_{K2}(C) = P \quad (7)$$

Kedua fungsi ini memenuhi

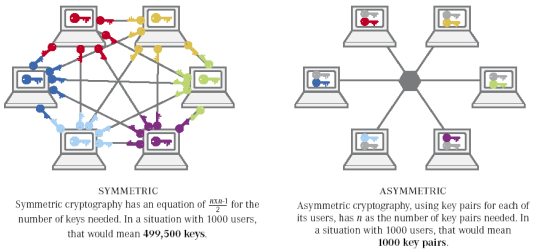
$$D_{K2}(E_{K1}(P)) = P \quad (8)$$

K1 dan K2 pada fungsi tersebut adalah kunci-kunci yang digunakan sebagai pengekripsi dan pendekripsi. Apabila K1 dan K2 yang digunakan sama (kunci untuk enkripsi sama dengan kunci untuk dekripsi), maka sistem kriptografinya dinamakan **sistem kriptografi simetri / kunci-pribadi** (symmetric / private-key cryptography). Algoritma pada sistem kriptografi ini dikenal sebagai algoritma simetri, contohnya adalah DES (Data Encryption Standard). Kunci pada sistem ini bersifat rahasia, sehingga pihak sumber pesan harus memiliki cara tersendiri untuk memberitahukan kunci kepada pihak penerima pesan. Hal tersebut tentu saja akan menjadi sangat tidak fleksibel. Selain itu, makin banyak penerima pesan, maka pihak sumber pesan harus menyediakan makin banyak pula kunci-kunci untuk tiap penerima, yang setara dengan $n(n-1)/2$ pasang kunci, dengan *n* sebagai banyaknya pengguna.

Apabila K1 tidak sama dengan K2 (kunci enkripsi berbeda dari kunci dekripsi), maka sistem kriptografinya dinamakan **sistem kriptografi nirsimetri / kunci-publik** (asymmetric / public-key cryptography). Algoritma pada sistem kriptografi ini dikenal sebagai algoritma nirsimetri, contohnya adalah RSA (Rivest-Shamir-Adleman) dan ECC (Elliptic Curve Cryptography). Kunci pada sistem ini ada dua macam, yaitu public-key (kunci publik) untuk proses enkripsi dan secret-key (kunci rahasia) untuk proses dekripsi. Pihak pengirim pesan dapat mengenkripsi pesan yang akan dikirim menggunakan kunci publik dari pihak penerima, sehingga pihak penerima pesan dapat mengetahui isi pesan yang dikirim dengan mendekripsinya menggunakan kunci rahasia yang dimiliki, yang tentu saja sesuai dengan kunci publik

yang digunakan oleh pengirim pesan.
Dalam hal ini, jumlah kunci yang dibutuhkan setara dengan *n* pasang kunci, dimana *n* adalah jumlah pengguna.

Berikut ini adalah ilustrasi yang menunjukkan bahwa sistem kriptografi simetri lebih rumit dan kurang efisien bila dibandingkan dengan sistem kriptografi nirsimetri:



SYMMETRIC
Symmetric cryptography has an equation of $\frac{n(n-1)}{2}$ for the number of keys needed. In a situation with 1000 users, that would mean **499,500 keys**.

ASYMMETRIC
Asymmetric cryptography, using key pairs for each of its users, has *n* as the number of key pairs needed. In a situation with 1000 users, that would mean **1000 key pairs**.

Dalam makalah ini, penulis akan fokus kepada sistem kriptografi kunci publik, khususnya algoritma RSA dan ECC. Pembahasan lebih lanjut mengenai kedua algoritma tersebut ada pada bagian selanjutnya.

3. RSA (RIVEST-SHAMIR-ADLEMAN)

Algoritma RSA (Rivest-Shamir-Adleman) mulai diperkenalkan pada tahun 1976 oleh tiga orang peneliti dari *MIT* (Massachussets Institute of Technology), yaitu Ronald Rivest, Adi Shamir, dan Leonard Adleman. Seperti sistem kriptografi kunci publik lainnya, algoritma RSA ini berdasar dari suatu permasalahan sukar (*hard problem*) teori matematika, yang dalam hal ini, RSA berdasar pada permasalahan pencarian faktor prima dari sebuah bilangan yang sangat besar.
Kunci enkripsi dan dekripsi pada algoritma ini keduanya merupakan bilangan bulat. Kunci untuk enkripsi merupakan kunci publik, sedangkan kunci untuk dekripsi merupakan kunci rahasia yang hanya diketahui oleh penerima pesan. Untuk membuat pasangan kunci publik dan rahasia, langkah-langkah dasarnya adalah sebagai berikut :

1.) Pilih dua buah bilangan prima *p* dan *q*. Dari dua bilangan prima tersebut, dapat kita peroleh bilangan modulus $n = p.\ q$.
2.) Pilih bilangan ketiga, *e*, yang relatif prima (faktor pembagi terbesarnya = 1) terhadap hasil perkalian $(p-1)(q-1)$.
3.) Hitung sebuah nilai *d* dari persamaan $(ed-1)/[(p-1)(q-1)]$. Bilangan *d* tersebut adalah kunci dekripsinya.

Bilangan yang merupakan kunci publik adalah pasangan bilangan (n,e).

Untuk melakukan enkripsi sebuah pesan S, buat ciphertext dengan kunci publik yang telah diketahui menggunakan persamaan berikut :

$$C = S^e \bmod n \qquad (9)$$

Kemudian pihak penerima pesan dapat mendekripsi ciphertext tersebut dengan kunci rahasia yang telah didapatkan sebelumnya, menggunakan persamaan berikut :

$$S = C^d \bmod n \qquad (10)$$

Walaupun nilai *n* dan *e* diketahui, namun akan sangat sulit untuk menemukan nilai *d* apabila nilai *p* dan *q* yang dipilih diawal cukup besar.

Agar lebih jelas, diberikan contoh sebagai berikut :

1. Pilih p = 3 dan q = 5.
2. Dari p dan q kita dapatkan modulus n = pq = 15.
3. Nilai e harus relatif prima terhadap (p-1)(q-1) = (2)(4) = 8. Oleh karena itu dipilih e=11.
4. Nilai dari d harus memenuhi (ed-1)/[(p-1)(q-1)] sehingga menghasilkan integer. Sehingga (11d-1)/[(2)(4)] = (11d-1)/8 harus menghasilkan sebuah integer. Hitung salah satu nilai yang mungkin, dipilih d=3.
5. Misalnya, akan dikirimkan pesan yang berupa sebuah kata SECRET. Ubah bentuk karakter menjadi nilai ASCII nya, yang sesuai dengan 83 69 67 82 69 84.
6. Pihak pengirim mengenkripsi tiap digit menggunakan kunci publik (e,n) = (11,15). Sehingga akan menghasilkan karakter ciphertext $Ci = Mi^{11} \bmod 15$. Sehingga plainteks yang dikirim, 0x836967826984, akan ditransmisikan sebagai ciphertext 0x2c696d286924.
7. Pihak penerima lalu mendekripsinya menggunakan kunci rahasia (d,n) = (3,15). Sehingga tiap karakter plainteks $Mi = Ci^3 \bmod 15$. Digit-digit masukannya yang berupa ciphertext 0x2c696d286924 akan dikonversi menjadi 0x836967826984, lalu dengan mengembalikannya sebagai karakter, akan didapatkan pesan rahasia semula, yaitu SECRET.

Kekuatan dari algoritma ini, ditentukan dari besarnya dua bilangan prima yang dipilih untuk membangkitkan kunci-kuncinya. Permasalahan tersebut didasarkan pada kesulitan akan pencarian faktor-faktor prima dari sebuah bilangan yang besar. Dalam implementasi yang sesungguhnya, nilai faktor prima yang dipilih sangatlah besar, bahkan mencapai 100 angka (digit), sehingga akan menghasilkan suatu bilangan modulus yang dapat mencapai 200 angka, sehingga untuk mencari faktor-faktor primanya akan sangat susah untuk dilakukan, bahkan hampir tidak mungkin.

4. ECC (ELLIPTIC CURVE CRYPTOGRAPHY)

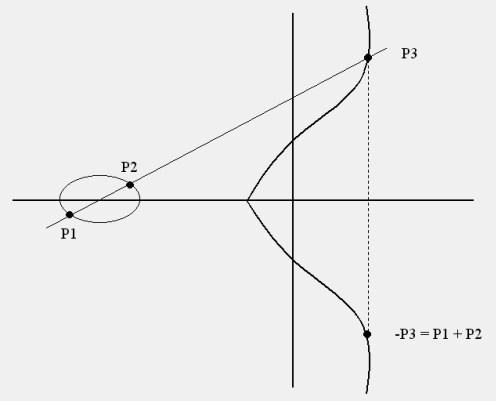
Algoritma ECC (Elliptic Curve Cryptography) diperkenalkan tahun 1985 oleh dua orang ahli kriptografi, Victor Miller (IBM) dan Neal Kobitz (University of Washington). Dasar dari sistem kriptografi ini, seperti sistem kriptografi kunci publik lainnya, yaitu berdasar dari suatu permasalahan sukar (*hard problem*) teori matematika. Permasalahan sukar yang digunakan algoritma ECC ini adalah permasalahan kesukaran penyelesaian *Elliptic Curve Discrete Logarithm Problem* (ECDLP) atau permasalahan logaritma diskrit kurva eliptik.

Kurva eliptik terdiri dari himpunan bilangan real (x,y) dimana x dan y memenuhi persamaan berikut :

$$y^2 = x^3 + ax + b \qquad (11)$$

himpunan dari seluruh solusi persamaan tersebut akan membentuk kurva eliptik. Perubahan nilai a dan b pada persamaan tersebut, selain akan mengubah bentuk kurva eliptik, juga akan mengakibatkan perubahan yang signifikan pada himpunan solusinya.

Berikut ini adalah contoh dari kurva eliptik :



Keterangan: P1 dan P2 adalah titik pada kurva eliptik, sedangkan P3 adalah hasil penjumlahan dari P1 dan P2

Operasi matematika yang dominan dilakukan dalam kriptografi ECC adalah operasi *point multiplication* (sebagai catatan, teori *point multiplication* tidak penulis jabarkan dalam makalah ini), operasi ini sebenarnya sangat sederhana, namun untuk melakukan proses inversi, akan menjadi sangat sulit. Dalam sistem kriptografi ECC, proses pemilihan kunci publik dan kunci rahasia adalah sesederhana operasi *point multiplication*, sebagai ilustrasi, akan diberikan contoh kasus sebagai berikut :

Misalkan Adi dan Badu sepakat untuk menggunakan salah satu kurva eliptik dan memilih salah satu titik F pada kurva tersebut. Langkah selanjutnya, Adi memilih bilangan random A_s dengan $1 \leq A_s \leq (n-1)$, dimana *n* adalah orde (jumlah dari titik yang elemen absis dan ordinatnya adalah integer) dari titik F. Kemudian Adi menghitung $A_p = A_s * F$. Titik A_p pasti merupakan titik yang berada pada kurva, karena A_p adalah *multiple* dari F. Hal ini mudah untuk dilakukan. A_p berlaku sebagai kunci publik bagi Adi dan A_s adalah kunci rahasianya. Sekarang, apabila Badu ingin mengetahui kunci rahasia milik Adi, maka Badu harus menghitung nilai integer A_s dari persamaan $A_p = A_s * F$, dimana Badu telah mengetahui F dan A_p , yang keduanya merupakan titik pada kurva eliptik. Hal ini, menurut teori, sangat sukar untuk dapat dilaksanakan. Dalam permasalahan ini, untuk menghitung nilai A_s dari A_p dan F, secara perhitungan kasar diperlukan $2^{n/2}$ operasi, dengan *n* adalah panjang (dalam bits) dari orde kurva yang merupakan bilangan prima. Sekarang, apabila F panjangnya 160 bits, maka untuk dapat menghasilkan kunci rahasianya, diperlukan 2^{80} operasi, yang apabila kapasitas proses adalah 1 milyar per detiknya, maka waktu yang diperlukan adalah 38 juta tahun [4].

5. PERBANDINGAN RSA DENGAN ECC

Sistem kriptografi kunci publik dengan algoritma RSA telah dipercaya untuk digunakan sebagai algoritma utama dalam proses kriptografi selama lebih dari dua dekade. Namun, algoritma ECC kini nampak berpotensi untuk menghasilkan tingkat pengamanan yang setara dengan RSA dengan panjang kunci-kunci yang lebih kecil. Berikut ini diberikan tabel perbandingan antara panjang kunci yang diperlukan oleh algoritma RSA dan ECC untuk memperoleh tingkat keamanan tertentu [3]:

Ukuran Kunci RSA (bits)	Waktu Untuk Menemukan Kunci Rahasia (MIPS years)	Ukuran Kunci ECC (bits)	Perbandingan Ukuran Kunci RSA : ECC
512	10^4	106	5:1
768	10^8	135	6:1
1.024	10^{11}	160	7:1
2.048	10^{20}	210	10:1
21.000	10^{78}	600	35:1

Pada Sistem kriptografi kunci publik, proses pembuatan kunci (*key generation*) dapat juga kita jadikan tolak ukur dari efisiensi algoritma kriptografinya, berikut ini adalah tabel yang menunjukkan perbandingan waktu yang dibutuhkan untuk menghasilkan pasangan kunci dengan tingkat keamanan yang setara antara algoritma RSA dan ECC [3] :

Panjang Kunci (bits)		Waktu (detik)	
ECC	**RSA**	**ECC**	**RSA**
163	1024	0.08	0.16
233	2240	0.18	7.47
283	3072	0.27	9.80
409	7680	0.64	133.90
571	15360	1.44	679.06

Tabel di atas menunjukkan bahwa proses *key generation* dalam ECC mengungguli RSA. Karena dalam proses menghasilkan kunci, ECC tidak membutuhkan waktu tersendiri untuk membuat (*generate*) bilangan-bilangan prima. Waktu yang dibutuhkan oleh ECC untuk membuat kunci, berkembang secara linear terhadap ukuran kuncinya, sedangkan pada algoritma RSA, waktu berkembang secara eksponensial terhadap ukuran kunci.

Dalam penggunaan sistem kriptografi kunci publik pada jaringan internet, berikut ini diberikan diagram perbandingan waktu yang dibutuhkan server untuk memenuhi permintaan dari user yang menggunakan sistem kriptografi ECC maupun RSA dengan ukuran kunci dan ukuran file yang bervariasi [5]:

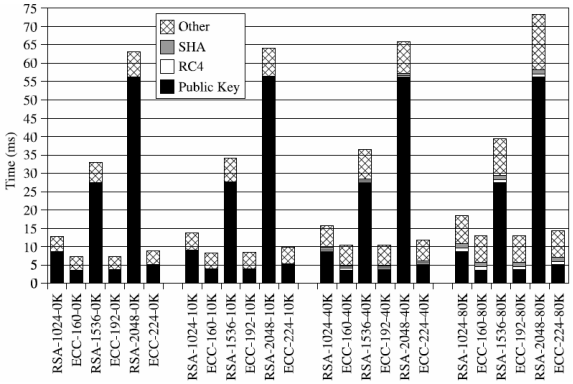


Diagram di atas menunjukkan bahwa waktu yang dibutuhkan oleh server untuk memenuhi permintaan file dari user, apabila menggunakan RSA, nampak perbedaan penggunaan waktu yang signifikan jika dibandingkan dengan ECC.

6. KESIMPULAN

Algoritma RSA dan ECC adalah contoh dari algoritma kriptografi kunci publik, karena kedua algoritma tersebut memiliki dua macam kunci, yaitu kunci publik untuk enkripsi dan kunci rahasia untuk dekripsi. Kekuatan algoritma RSA didasarkan pada kesukaran pemfaktoran suatu integer yang besar menjadi faktor-faktor primanya, sedangkan pada ECC, kekuatannya ada pada permasalahan yang dikenal sebagai *Elliptic Curve Discrete Logarithm Problem* (ECDLP). Karena ukuran kunci dari ECC jauh lebih kecil jika dibandingkan dengan RSA untuk tingkat keamanan yang sama, maka hal tersebut berdampak pada kecepatan pemrosesan sekaligus konsumsi dari memory maupun bandwith. Kelebihan ECC dalam hal tersebut, sangat menonjol pada piranti-piranti yang kecil, karena piranti-piranti kecil hanya memiliki memory terbatas disamping kemampuan komputasi yang juga rendah. Pada penggunaan pada umumnya, algoritma kriptografi ECC juga lebih unggul bila dibandingkan dengan algoritma RSA, seperti yang telah penulis berikan penjabarannya pada bab sebelumnya. Hingga saat makalah ini dibuat, algoritma ECC masih terus dikembangkan, sehingga pada saatnya nanti, ECC dapat menggantikan posisi algoritma RSA yang dinilai berat dan lebih lambat.

DAFTAR REFERENSI

[1] Munir, Rinaldi. (2006). Matematika Diskrit, Edisi Ketiga. Penerbit Informatika. Bandung.

[2] Kessler, Gary. (1998). An Overview of Cryptography. http://mia.ece.uic.edu . Tanggal akses : 29 Desember 2007 pukul 12.34

[3] Jansma, Nicholas. (2004). Performacne Comparison of Elliptic Curve and RSA Digital Signatures. http://engin.unich.edu . Tanggal akses : 29 Desember 2007 pukul 12.00

[4] Engelfriet, Arnoud. (2005). Elliptic Curve Cryptography. http://www.iusmentis.com . Tanggal akses : 31 Desember 2007 pukul 10.21

[5] Gupta, Vipul. Speeding Up Secure Web Transactions Using Elliptic Curve Cryptography (ECC). http://www.isoc.org . Tanggal akses : 31 Desember 2007 pukul 11.51

[6] Schneier, Bruce, *Applied Cryptography 2nd* , John Wiley & Sons, 1996