Analisis dan Implementasi Elliptic Curve Integrated Encryption Scheme (ECIES)

Dian Syahfitra [1)]

1) Jurusan Teknik Informatika ITB, Bandung 60111, email: if14021@students.if.itb.ac.id

Abstrak *- Makalah ini memberikan pembahasan salah satu implementasi kriptografi berbasis kurva elips (Elliptic Curve Crypthography atau ECC) pada bidang skema enkripsi data atau pesan secara langsung. Skema implementasi ini dikenal dengan Elliptic Curve Integrated Encryption Scheme (ECIES) yang menggabung enkripsi kurva eliptik asimetrik dengan AES yang simetris dan memakai algoritma hash SHA-1 untuk menyediakan skema enkripsi dengan dukungan autentikasi pesan Makalah ini juga akan memberikan cara implementasi ECIES pada bahasa pemograman tingkat tinggi yaitu Java dan membandingkan ECIES dengan RSA dan sebuah algoritma tersendiri berbasis chipper blok pada aspek tertentu. Akan dilakukan juga ekperimen dengan mengganti sebahagian elemen dari skema ECIES.*

Kata Kunci *: ECC, ECIES*

1. PENDAHULUAN

Algoritma kunci publik yang populer saat ini seperti RSA memanfaatkan kesulitan untuk memfaktorkan bilangan besar menjadi faktor-faktor prima nya, keamanan algoritma ini pada masalah pemfaktoran bilangan bulat besar ataupun yang keamananya bergantung pada masalah logaritma diskrit (*discrete logarithm problem*). Begitu pula kriptografi kurva elips keamanan bergantung pada masalah logaritma diskrit hanya yang membedakannya adalah ruang solusi dibatasi dengan titik-titik pada kurva elips. Masalah ini seraing disebut dengan *Elliptic Curve Discrete Logarithm Problem* (ECDLP).

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. ECC dan algoritma-algoritma yang didasarkan padanya diyakini dapat menggantikan kritografi yang didasrkan faktorisasi bilangan ataupun kriptografi daerah finite(bidang yang hanya memiliki elemen bilangan yang terbatas). ECC saat ini sudah banyak turunan nya seperti pada Tanda Tangan Digital (ECDSA, ECPVS) , Pertukaran Kunci (ECMQV, ECDH), Pembangkit Bilangan Acak Random (Dual_EC_DRBG), ataupun pada skema enkripsi data (ECIES). Kekuatan dari ECC dibandingkan dengan kriptografi yang melandasakan pada DLP(*discrete logarithm problem*) adalah dengan panjang kunci yang lebih kecil kita dapat memperoleh tingkat keamanan yang sama. Keuntungan yang bisa didapatkan dengan parameter yang lebih kecil adalah kecepatan komputasi lebih tinggi, ukuran data yang lebih kecil. Hal ini dapat dimanfaatkan ada lingkungan dengan sumber daya komputasi terbatas. Perbandingan tingkat komputasi akan ditunjukkan pada bagian selanjutnya.

Elliptic Curve Integrated Encryption Scheme (ECIES) merupakan skema enkripsi kunci publik berdasarkan ECC. ECIES mengkombinasikan enkripsi kurva eliptik asimetrik dengan AES yang simetris dan memakai algoritma hash SHA-1 untuk menyediakan skema enkripsi dengan dukungan autentikasi pesan. ECIES dirancang untuk secara semantik aman terhadap serangan plaintext terpilih(chosen plaintext attack) dan serangan chipertext terpilih(chosen chipertext attack). ECIES terkadang disebut sebagai Elliptic Curve Augmented Encryption Scheme (ECAEC) atau disebut juga Elliptic Curve Encryption Scheme saja.

Dasar Teori Matematika

Untuk memahami ECC pada umumnya dan ECIES pada khususnya kita perlu mempelajari sedikit dasar teori matematikanya. Kurva ellips yang didefinisikan dengan menggunakan dua tipe daerah finite, daerah finite terdiri dari sejumlah set objek terbatas yang disebut elemen daerah dan bersama dengan deskripsi dari operasi perkalian dan penjumlahan yang dapat dilakukan oleh pasangan elemen-elemen ini. Operasinya harus memiliki karakteristik tertentu. Jika $q = p_m$ dimana p adalah bilangan prima dan m adalah bilangan bulatpositif, maka p disebut karakteristik dari Fq, dan m disebut derajat perpangkatan (*extension degree*) dari Fq. Daerah Finite yang dipakai ada 2 jenis yaitu Fp (*odd prime*) yang merupakan karakteristik pertama dimana q = p, dan F_2m (*even*) yang merupakann karakteristik kedua dimana $q=2^m$. Penulis hanya akan menjelaskan kurva elips pada F_2m karena pada implementasi yang digunakkan adalah pada daerah finite ini.

Sebuah kurva eliptik E pada F_2m didefinisikan sebagai sebagai sebuah persamaan dalam bentuk :

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

Dimana a,b Є F2m , dan $b \neq 0$. Diketahui bahwa F_2m merupakan finite field karakteristik kedua, dan

diketahui $a, b \in F_2m$ dimana $b \neq 0 \in F_2m$. Maka sebuah kurva Elips dengan notasi $E(F_2m)$ di wilayah F_2m dengan parameter $a,b \in F_2m$ terdiri dari himpunan solusi atau kumpulan titik-titik P(x,y) dimana $x,y \in F_2m$ memenuhi persamaan:

$$y^2 + xy = x^3 + ax^2 + b \text{ di } F_2m$$

bersama dengan point tambahan *O* yang disebut titik ketakhinggaan(*point of infinity*), angka dari titik-titik pada $E(F_2m)$ dinotasikan dengan $\#E(F_2m)$. Teorema Hasse menyatkan bahwa :

$$2^m + 1 - 2\sqrt{2^m} \leq \#E(F2m) \leq 2^m + 1 + 2\sqrt{2^m}$$

Kemudian memungkinkan didefinisikan aturan untuk operasi penjumlahan sebagai berikut:

1. Aturan penjumlahan *O* dengan dirinya sendiri :
$$O + O = O$$

2. Aturan untuk menjumlahkan O dengan point lain :
$$O + (x,y) = (x,y) + O = (x,y)$$
untuk semua $(x, y) \in E(F_2m)$

3. Aturan menambahkan dua titik dengan kordinat x yang sama ketika titik-titik tersebut berbeda atau mempunyai kordinat 0 pada sumbu x.
$$(x,y) + (x, x+y) = O$$
untuk semua $(x, y) \in E(F_2m)$

4. Aturan untuk menambahkan 2 titik dengan kordinat x berbeda. Ambil $(x_1, y_1) \in E(F_2m)$ dan $(x_2, y_2) \in E(F_2m)$ dan $x_1 \neq x_2$ kemudian $(x_1, y_1) + (x_2, y_2) = (x_3, y_3)$ dimana :
$$x_3 = \lambda^2 + \lambda + x_1 + x_2 + a \text{ didalam } F_2m,$$
$$y_3 = \lambda(x_1+x_3) + x_1 + y_1 \text{ didalam } F_2m, \text{ dan}$$
$$\lambda = \frac{y1+y2}{x1+x2} \text{ didalam } F_2m$$

5. Aturan untuk menambahkan titik dengan dirinya sendiri (dikali 2). Ambil $(x_1, y_1) \in E(F_2m)$ sebuah titik dengan $x_1 = 0$ kemudian $(x_1, y_1) + (x_1, y_1) = (x_3, y_3)$ dimana :
$$x_3 = \lambda^2 + \lambda + a \text{ didalam } F_2m,$$
$$y_3 = x12 + (\lambda + 1).x_3 \text{ didalam } F_2m, \text{ dan}$$
$$\lambda = x_1 + \frac{y1}{x1} \text{ didalam } F_2m$$

Himpunan titik-titik di $E(F_2m)$ dengan aturan penambahan ini membentuk suatu abelian group. Skema kriptografi pada ECC didasarkan pada perkalian scalar dari titik-titik dikurva elips. Jika suatu titik P di kurva elips dimana $P \in E(F_2m)$ dan P dikali dengan suatu skalar k maka sama dengan menjumlahkan titik P dengan P sendiri sebanyak k kali sesuai aturan diatas.

2. PEMBAHASAN

Implemetasi ECC pada bidang enkripsi data atau pesan dituangkan dalam suatu skema yang disebut dengan ECIES. ECIES atau Elliptic Curve Integrated Encryption Scheme mengkombinasikan enkripsi asimetrik kurva elips dengan algoritma simetrik AES dan juga fungsi hash SHA-1. Sebuah chipertext hasil enkripsi ECIES terdiri dari kunci public EC dinotasikan dengan V, pesan terenkripsi dinotasikan dengan C dan tag autentifikasi dinotasikan dengan T yang dibangkitkan dari pesan (M) dan kunci public penerima (Wi). Penerima pesan dapat mendekripsi pesan dengan kunci privat EC mereka. Skema enkripsi yang penulis implementasikan dapat ditinjau pada draft IEEE P1363a. Perlu diingat bahwa ECIES hanya sebuah skema terintegrasi dan berkaitan dengan pemakaian kurva eliptik lainya seperti standar kurva eliptik Diffie-Hellman yang tidak dijelaskan pada makalah ini. Berikut akan dijelaskan skema ECIES.

Apabila ada seorang pengirim U dan pemerima V ingin bertukar pesan M dengan skema enkripsi ECIES, sebelum melakukan pertukaran pesan dilakukan skema sebagai berikut:

1. V memilih skema penurunan kunci (Key Derivation Function) dengan fungi hash tertentu biasanya SHA-1, V juga memilih skema MAC yang digunakan, V juga memilih skema enkripsi simetrik yang akan digunakan biasanya AES, V juga memutuskan apakah akan mengunakan standar kurva eliptik Diffie-Hellman untuk pertukaran kunci. V juga memutuskan domain parameter yang diinginkan.
2. U memperoleh semua keputusan yang di berikan V dan memdapatkan domain parameter. Pada intinya U dan V harus memiliki skema yang sesuai satu dengan lainya.

Kemudian V mengenkripsi pesan sebagai berikut :

1. Masukan adalah pesan M, pilih pasangan kunci kurva elliptic (k, R) dengan $R= (x_r,y_r)$ yang berasosiasi dengan domain parameter yang telah ditentukan sebelumnya, bangkitkan pasangan kunci dengan menggunakan fungsi pembangkit
2. Konversikan R kedalam string oktet
3. Menggunakan standar kurva eliptik Diffie-Hellman turunkan elemen rahasia $z \in F_2m$
4. Konversikan z kedalam octet string Z dengan menggunakan fungsi konversi elemen ke octet string
5. Menggunakan kunci yang diturunkan dari

KDF bangkitkan data kunci K dengan panjang kunci enkripsi ditambah panjang kunci mac

6. Parsing oktet terkiri dari K sebagai kunci enkripsi EK dan oktet terkanan sebgai kunci MK
7. Gunakan operasi enkripsi simetris contohnya seperti AES untuk mengenkripsi M dengan kunci EK menjadi chipertext EM.
8. Gunakan operasi tag pada skema MAC untuk mengkomputasi tag D dengan kunci MK
9. Outputkan R_k (V), EM_k (C) dan D (T)

Implementasi

Implementasi algoritma diatas penulis lakukan dengan menggunakan bahasa Java, sedangkan fungsi fungsi dasar seperti AES, MAC, dan sebagainya dilakukan dengan memakai library yang disedikan oleh jBorZoi. Untuk proses enkripsi diimplementasikan sebagai berikut:

```
//deklarasi
int P1[] = new int[0];
int P2[] = new int[0];

//bangkitkan z rahasia
Fq z = ECC.ECSVDP_DH (u.dp, u.s, W.W);
int Z[] = Utils.FE2OSP (z);

// 256 bits kunci gabung
int K[] = ECC.KDF2 (Z, 32, P1);

//128 bits kunci enkripsi simetrik
int K1[] = new int[16];

// 128 bit kunci MAC
int K2[] = new int[16];

for (int j=0; j<K1.length;j++) {
       K1[j] = K[j];
}for (int k=0; k<K2.length;k++) {
       K2[k] = K[k+K1.length];
}

//bangkitkan V
V = new ECPubKey (u);
//hasilkan chipertext
C    =    ECC.AES_CBC_IV0_Encrypt    (K1,
Utils.toIntArray(p), 128);
//hasilkan tag dari fungsi MAC
T        =        ECC.MAC1        (K2,
Utils.concatenate(C,P2));
```

Dan untuk proses dekripsi pesan sama seperti enkripsi diimplementasikan sebagai berikut:

```
//deklarasi
int P1[] = new int[0];
int P2[] = new int[0];

//bangkitkan z rahasia
Fq z = ECC.ECSVDP_DH (s.dp, s.s, V.W);
int Z[] = Utils.FE2OSP (z);

//256 bits kunci gabung
int K[] = ECC.KDF2 (Z, 32, P1); // 256
bits

//128 bits kunci enkripsi simetrik
int K1[] = new int[16];

// 128 bit kunci MAC
int K2[] = new int[16];
for (int j=0; j<K1.length;j++) {
       K1[j] = K[j];
}
for (int k=0; k<K2.length;k++) {
       K2[k] = K[k+K1.length];
}

//enkripsi pesan kembali ke M
int M[] =
ECC.AES_CBC_IV0_Decrypt(K1,C, 128);

//periksa tag benar atau tidak
if (!Utils.compare (T, ECC.MAC1 (K2,
Utils.concatenate(C, P2)))) {
       throw new Exception ("ECIES:
Authentication Tag Invalid");
}

return Utils.toByteArray(M);
```

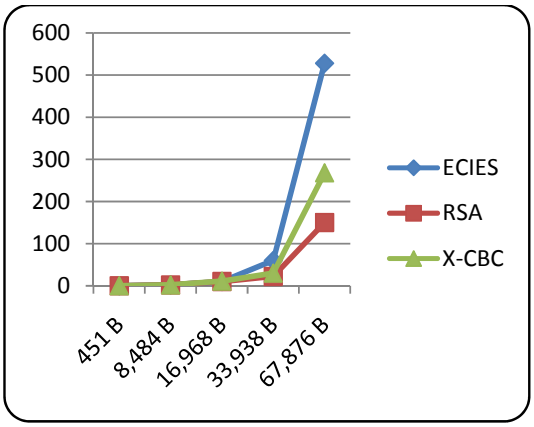
Pengujian, dan Perbandingan

Pengujian dan perbandingan dilakukan terhadap hasil implemetasi ECIES dengan 2 algoritma enkripsi lain. Algoritma yang dipilih adalah algoritma buatan kelompok sendiri dengan block chipper CBC dan algoritma RSA. Pengujian dilakukan terhadap waktu komputasi enkripsi di tambah dengan dekripsi secara langsung.File uji adalah 5 buah file text masing-masing berukuran (451 Bytes, 8,484 Bytes, 16,968 Bytes, 33,938 Bytes, dan 67,876 Bytes). Panjang kunci yang dipakai untuk CBC adalah 8 karakter. Spesifikasi komputer yang dipakai dengan processor AMD Turion 64bit 1.8 GHz dan memory 512 MB. Berikut hasil pengujian.

Tabel 1Hasil Uji 3 Algoritma

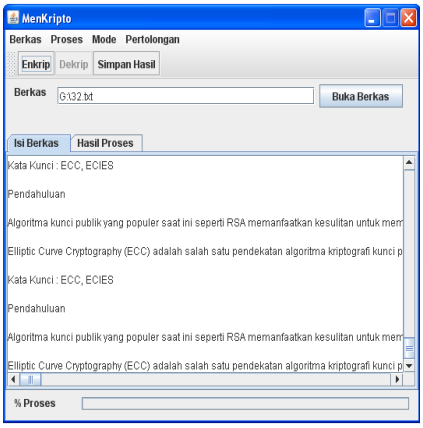
	451 Bytes	8,484 Bytes	16,968 Bytes	33,938 Bytes	67,876 Bytes
ECIES	0 s	2 s	12 s	67 s	546 s
RSA	0 s	2 s	10 s	22 s	150 s
X-CBC	0 s	2 s	11 s	30 s	268 s

Berikut di sajikan dalam grafik pada gambar1:

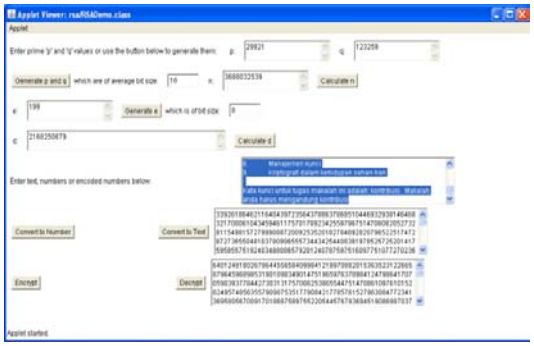


Gambar 1Grafik Hasil Uji 3 Algoritma

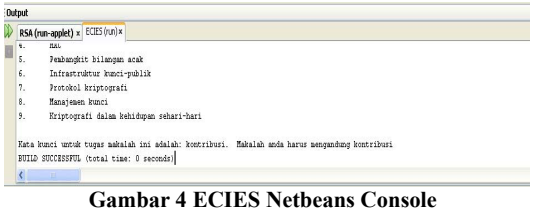
Berikut lampiran screenshot program. Untuk X-CBC digunakan Aplikasi Window Java, untuk RSA digunakan Applet java, sedangkan untuk ECIES dipakai versi console debug dari Netbeans.



Gambar 2 X-CBC Java Application



Gambar 3 RSA Applet Viewer



Gambar 4 ECIES Netbeans Console

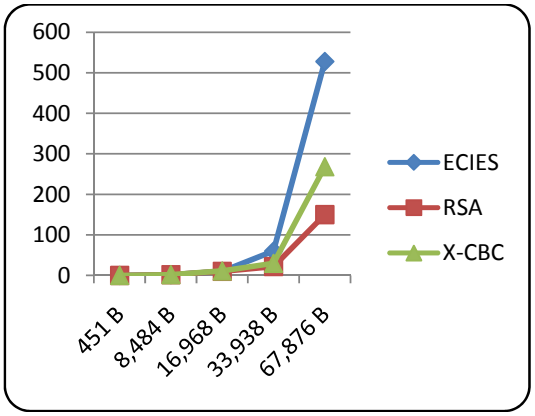
Eksperimen

Penulis melakukan 2 buah ekperimen untuk melihat efek terhadap waktu komputasi .

Eksperimen I dilakukan dengan menggandakan panjang kunci ganda menjadi 512 bits dari yang sebelumnya 256 bits dan mengulang pengujian terhadap ECIES. Hasil yang diperoleh sebagai berikut :

Tabel 2 Tabel Hasil Eksperimen I

	451 B	8,484 B	16,968 B	33,938 B	67,876 B
ECIES	0 s	2 s	11 s	64 s	537 s
RSA	0 s	2 s	10 s	22 s	150 s
X-CBC	0 s	2 s	11 s	30 s	268 s



Gambar 5Grafik Hasil Uji Eksperimen I

Ekperimen II dilakukan mengganti komputasi untuk Tag T menjadi hanya berupa operasi assignment atau dengan kata lain tidak memakai fungsi MAC(meniadakan operasi MAC). Hasil yang diperoleh sebagai berikut :

Tabel 3 Tabel hasil Ekperimen II

	451 B	8,484 B	16,968 B	33,938 B	67,876 B
ECIES	0 s	2 s	10 s	60 s	528 s
RSA	0 s	2 s	10 s	22 s	150 s
X-CBC	0 s	2 s	11 s	30 s	268 s

3. ANALISIS

Dengan mengesampingkan media uji yang digunakan apakah window application, applet viewer ataupun console version kita dapat melihat beberapa hal. Pada hasil pengujian dan ekperimen diperoleh bahwa ECIES memiliki waktu komputasi terlama di bandingkan RSA atau X-CBC. Hal ini sangat mudah dipahamai karena pada ECIES terdapat 3 komputasi yang didalamnya ada operasi penurunan kunci publik dengan SHA, enkripsi data

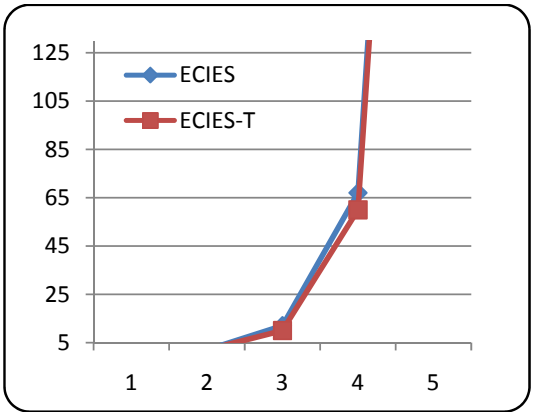
dengan AES 128 bit dan perhitungan Tag T dengan fungsi MAC. Hal ini dibandingkan dengan algoritma lain yang hanya focus pada enkripsi data.

Pada hasil eksperimen pertama ternyata perubahan panjang deklarasi kunci menjadi 2x nya tidak memberikan pengaruh berarti terhdap waktu komputasi hal ini disebabkan pada library yang digunakan penulis ternyata memakai AES dengan panjang kunci tetap yaitu 128 bit. Jadi deklarasi yang lebih tidak akan terpakai.

Pada hasil eksperimen kedua ternyata peniadaan waktu komputasi MAC dengan sebuah operasi assignment sederhana memberikan pengaruh yang cukup signifikan terhadap waktu komputasi seperti pada tabel 4 dibawah ini :

Tabel 4 Perbandingan Awal dan Eksperimen 2

	451 Bytes	8,484 Bytes	16,968 Bytes	33,938 Bytes	67,876 Bytes
ECIES	0 s	2 s	12 s	67 s	546 s
ECIES non T	0 s	2 s	10 s	60 s	528 s



Gambar 6 Grafik sebahagain perbandingan 2 ECIES

Pada table tersebut dapat dilihat bahwa terjadi perubahan 5 -10 % dari pengurangan waktu, dengan kata lain kita dapat menyimpulkan bahwa komputasi MAC mengambil porsi sekitar 5-10% dari seluruh waktu komputasi. Kemungkinan bahwa AES atau fungsi enkripsi data mengambil porsi paling banyak dalam waktu komputasi ini. Sebenarnya ECIES tidak memakai fungsi MAC tidak dapat kita sebut sebagai ECIES namun hal ini hanya untuk keperluan eksperimen waktu komputasi saja.

Operasi dari skema ECIES memberikan banyak pilihan-pilihan fungsi yang dapat dipilih oleh pengguna sesuai dengan situasi dan penggunaan. Pemilihan ini didasarkan pada efisiensi, interoperability, ataupun keamanan. Kita dapat memilih apakah menggunakan primitive Deffie-Hellman atau cofaktornya. Dapat juga menggunakan fungsi Hash yang berbeda.

Serangan Terhadap ECIES

Terdapat sejumlah serangan terhadap skema enkripsi ini yaitu :

- Serangan terhadap ECDLP, karena keamanan ECIES bergantung pada problem ini. Pertukaran domain parameter dari U dan V memungkinkan penyadap untuk mendapatkan nilai rahasia z dari R dan QV yang kemudian dapat digunakan untuk merecover pesan.
- Serangan pada fungsi pembangkit kunci karena masih menggunakan pseudorandom generator
- Serangan terhadap fungsi enkripsi simetrik yang dipilih seperti AES tentu saja berpengaruh pada ECIES
- Serangan terhadap skema MAC yang memungkinkan serangn chosen-chipertext pada ECIES
- Serangan didasarkan pada penggunaan domain parameter yang salah memungkinkan serangan yang dikenal dengan Pohlig-Hellman Attack. Penerima harus memastikan domain parameter yang diterima adalah valid
- Serangan didasarkan pada kunci publik tidak valid jika terjadi kesalahan pada pertukaran kunci Diffie-Hellman atau serangan terhadapnya

Terkadang ada juga serangan non kriptografi seperti fault-based attacks, poweranalysis attacks, and timing-analysis attacks,

4. KESIMPULAN DAN SARAN

Kesimpulan yang diperoleh dari penuisan makalah ini:

1. Skema Enkripsi ECIES memiliki waktu komputasi yang lebih lama dibandingkan skema yang lain karena skema ini melibatkan banyak fungsi. Pemilihan fungsi yang tepat untuk berbagai konteks masalah dan tempat sangat penting dalam pembangunan ECIES.
2. ECIES memiliki tingkat customisasi yang tinggi dengan membebaskan kita untuk memilih fungsi yang sesuai. Akan tetapi perlu diingat bahwa keamanan skema ECIES bergantung pada keamanan fungsi yang digunakannya.
3. Kriptografi berbasis kurva ellips memiliki tingkat keamanan yang baik bahkan pada penelitina tertentu dikatakn lebih baik dan berpeluang untuk terus berkembang dengan dibuatnya fungsi-fungsi baru yang memenuhi daerah finite kurva ellips F_p maupun F_2m

Saran penulis untuk pemakaian ECIES :

1. Dalam pemilihan fungsi hendaknya perlu diperhatikan aspek-aspek efisiensi, interoperability, ataupun keamanan. Perhatikan serangan yang mungkin dan fungsi yang dapat mencegahnya.
2. Perlu dilakukan penelitian lebih lanjut terhadap implementasi diruang finite karekteristik pertama F_p (*odd prime*)

Daftar Referensi :

[1] Certicom. Desember 2007. *ECC Tutorial*. http://www.certicom.com/index.php?action=ecc_tutorial,home

[2] Certicom. 20 September 2000. *Standards For Efficient Cryptography – SEC1 : Elliptic Curve Crypthography*. Certicom Corp

[3] IEEE. May 2000. IEEE P1363A. *Standard Specifications for Public-Key Cryptography: Additional Techniques*.Working Draft.

[4] Munir, Rinaldi. Agustus, 2006. *Kriptografi*. Bandung

[5] Technologies, Dragongate. 12 Agustus 2003. *jBorZoi Manual (Preliminary Draft)*. Dragongate Technologies Ltd.