Perbandingan Algoritma Kunci Nirsimetris ElGammal dan RSA pada Citra Berwarna

Whilda Chaq - 13511601
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13511601@std.stei.itb.ac.id

***Abstract*— Pada makalah kali ini penulis akan membandingkan dua buah algoritma kriptografi kunci nirsimetris RSA dan ElGammal. Alasan pemilihan kedua algoritma tersebut karena RSA merupakan algoritma kriptografi kunci publik yang paling populer hingga saat ini dan ElGammal yang dinilai memiliki keunikan dalam mengenkripsi sebuah plainteks menjadi 2 buah cipherteks. Perbandingan dua buah algoritma ini akan diujikan pada citra berwarna sehingga hasil enkripsi dapat dilihat secara visual. Proses enkripsi maupun dekripsi dilakukan dengan memproses setiap pixel – pixel pada citra berwarna yang direpresentasikan sebagai integer RGB. Pada algoritma ElGammal akan menghasilkan 2 buah citra berwarna karena memang algoritma ElGammal mempunyai karakteristik demikian. Hasil perbandingan nantinya berupa citra hasil enkripsi, kecepatan eksekusi algoritma, ditinjau dari segi keamanan algoritma, dll.**

***Index Terms*—Asimetris, Citra , ElGammal, Kriptografi, RSA.**

I. PENDAHULUAN

Pada tahun 1970an, dunia kriptografi hanya mengenal sistem kriptografi menggunakan satu kunci yang sama untuk melakukan enkripsi maupun dekripsi sebuah pesan yang dikenal dengan istilah kriptografi kunci simetris. Sistem kunci simetris seperti ini masih memiliki masalah yang cukup besar yaitu bagaimana kit menjamin dapat mengirimkan kunci rahasia ke penerima dengan aman. Mengirim kunci melalui media yang ada saat itu juga tidak dapat menjamin kerahasiaan pesan. Para pakar kriptografi akhirnya menemukan sebuah soludi untuk permasalahan kriptografi kunci simetris yaitu kriptografi simetris.

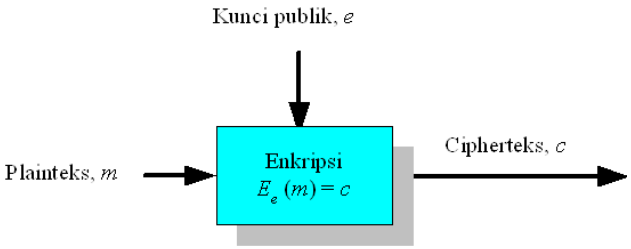
Tepatnya tahun 1976 makalah pertama mengenai kriptografi kunci nirsimetris ditulis oleh dua orang yang bernama Whitfield Diffie dan Martin Hellman. Kriptografi kunci nirsimetris ini sering disebut juga kriptografi kunci publik karena pada prinsipnya terdapat 2 buah kunci untuk melakukan proses enkripsi dan dekripsi pesan yang dikenal dengan kunci publik dan kunci privat.

II. DASAR TEORI

A. Kriptografi Kunci Nirsimetris / Publik

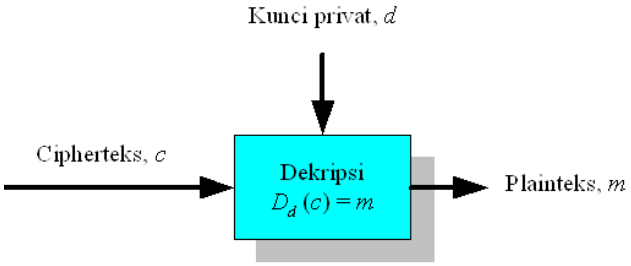
Pada kriptografi kunci-publik, masing-masing pengirim dan penerima mempunyai sepasang kunci:

1. Kunci publik: untuk mengenkripsi pesan
   $E_e(m) = c$



Bagan 1 Skema Enkripsi Kriptografi Kunci Publik

2. Kunci privat: untuk mendekripsi pesan.
   $D_d(c) = m$



Bagan 2 Skema Dekripsi Kriptografi Kunci Publik

B. ElGammal

Dibuat oleh Taher Elgamal (1985). Keamanan algoritma ini terletak pada sulitnya menghitung logaritma diskrit.

Masalah logaritma diskrit: Jika *p* adalah bilangan prima dan *g* dan *y* adalah sembarang bilangan bulat. carilah *x* sedemikian sehingga

$$g^x \equiv y \pmod{p}$$

Properti algoritma ElGamal:

1. Bilangan prima, p (tidak rahasia)
2. Bilangan acak, g ($g < p$) (tidak rahasia)
3. Bilangan acak, x ($x < p$) (rahasia, kc. privat)
4. $y = g^x \bmod p$ (tidak rahasia, kc. publik)
5. m (plainteks) (rahasia)
6. a dan b (cipherteks) (tidak rahasia)

Algoritma pembangkit kunci untuk Elgammal adalah sebagai berikut :

1. Pilih sembarang bilangan prima p (p dapat di-*share* di antara anggota kelompok)
2. Pilih dua buah bilangan acak, g dan x, dengan syarat $g < p$ dan $1 \le x \le p - 2$
3. Hitung $y = g^x \bmod p$.

Hasil dari algoritma ini:
- Kunci publik: tripel (y, g, p)
- Kunci privat: pasangan (x, p)

Algoritma Enkripsi adalah sebagai berikut :

1. Susun plainteks menjadi blok-blok m_1, m_2, …, (nilai setiap blok di dalam selang $[0, p - 1]$.
2. Pilih bilangan acak k, yang dalam hal ini $1 \le k \le p - 2$.
3. Setiap blok m dienkripsi dengan rumus
   $a = g^k \bmod p$
   $b = y^k m \bmod p$

Pasangan a dan b adalah cipherteks untuk blok pesan m. Jadi, ukuran cipherteks dua kali ukuran plainteksnya.

Algoritma Dekripsinya adalah sebagai berikut :

1. Gunakan kunci privat x untuk menghitung
   $(a^x)^{-1} = a^{p-1-x} \bmod p$
2. Hitung plainteks m dengan persamaan:
   $m = b/a^x \bmod p = b(a^x)^{-1} \bmod p$

C. RSA

RSA merupakan algoritma kunci-publik yang paling terkenal dan paling banyak aplikasinya. Ditemukan oleh tiga peneliti dari *MIT* (*Massachussets Institute of Technology*), yaitu Ron Rivest, Adi Shamir, dan Len Adleman, pada tahun 1976. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima.

Properti algoritma RSA:

1. p dan q bilangan prima (rahasia)
2. $n = p \cdot q$ (tidak rahasia)
3. $\phi(n) = (p - 1)(q - 1)$ (rahasia)
4. e (kunci enkripsi) (tidak rahasia)
   Syarat: PBB(e, $\phi(n)$) = 1
5. d (kunci dekripsi) (rahasia)
   d dihitung dari $d \equiv e^{-1} \bmod (\phi(n))$
6. m (plainteks) (rahasia)
7. c (cipherteks) (tidak rahasia)

Pembangkitan kunci RSA adalah sebagai berikut :

1. Pilih dua bilangan prima, p dan q (rahasia)
2. Hitung $n = pq$.
3. Hitung $\phi(n) = (p - 1)(q - 1)$.
4. Pilih sebuah bilangan bulat e untuk kunci publik, sebut, e relatif prima terhadap $\phi(n)$.
5. Hitung kunci dekripsi, d, dengan persamaaan
   $ed \equiv 1$ (**mod** $\phi(n)$) atau $d \equiv e^{-1} \bmod (\phi(n))$

Hasil dari algoritma di atas:
- Kunci publik adalah pasangan (e, n)
- Kunci privat adalah pasangan (d, n)

Algoritma Enkripsi RSA adalah sebagai berikut :

1. Nyatakan pesan menjadi blok-blok plainteks: m_1, m_2, m_3, … (syarat: $0 < m_i < n - 1$)
2. Hitung blok cipherteks c_i untuk blok plainteks p_i dengan persamaan
   $c_i = m_i^e$ **mod** n
   yang dalam hal ini, e adalah kunci publik.

Proses dekripsi dilakukan dengan menggunakan persamaan

$m_i = c_i^d$ **mod** n,

yang dalam hal ini, d adalah kunci privat.

D. Citra Berwarna

Pada dunia komputer citra berwarna dikenal dengan istilah Bitmap. Bitmap adalah mapping dari sebuah domain tertentu, dalam persoalan kali ini domain yang dimaksud adalah pixel. Bitmap sendiri nampak seperti objek 2D (Gambar) yang terdiri dari pixel – pixel. Pixel direpresentasikan sebuat warna yang dibangun dengan pendekatan RGB (Red, Green, Blue). Untuk setiap pixel merupakan kombinasi nilai dari nilai red, green, dan blue yang masing masing bernilai 1 byte / 8 bit.

III. LINGKUNGAN IMPLEMENTASI

A. Hardware
- Laptop
  - Processor : Intel I5 4 CPU @ 2.5 GHz
  - RAM : 4096 MB

B. Software
- OS : Windows 7
- Bahasa : c#
- IDE : Microsoft Visual Studio 2010

IV. IMPLEMENTASI ALGORITMA PADA CITRA BERWARNA

Untuk mengimplementasikan algoritma ElGammal maupun RSA yang perlu diperhatikan adalah representasi pixel yang di enkripsi maupun didekripsi. Untuk kedua algoritma tersebut menggunakan representasi pixel yang sama.

Sebuah citra berwarna merupakan objek 2D yang tersusun dari pixel-pixel. Proses enkripsi dan dekripsi dengan cara menelusuri setiap pixel secara iteratif sehingga dapat mengakses seluruh pixel. Kemudian untuk setiap pixel di dapat warna yang di representasikan sebagai RGB, masing masing 1 byte. Sebagai contoh, warna hitam total memiliki nilai red=255, green=255, dan blue=255.

Agar pixel yang diakses dapat di terapkan sebagai message maka dari pixel yang didapat dibentuk menjadi sebuah bilangan integer. Bilangan yang dimaksud didapat dari mengubah nilai red, green, dan blue pada sebuah pixel menjadi biner 8 bit untuk setiap nilainya. Masing masing nilai kemudian di append 1 sama lain dimulai dari red, green, kemudian blue. Setelah mendapatkan biner 24 bit maka segera dikonversikan menjadi bilangan integer, bilangan inilah yang nantinya menjadi message untuk diterapkan dalam sebuah algoritma ElGammal maupun RSA.

Sebagai contoh diberikan sebuah citra berwarna dengan ukuran 600 x 400 pixel. Kemudian diambil pixel ke (0,0) dan didapat pixel dengan warna yang hitam (red=255,green=255,blue=255). Masing – masing di konversi menjadi biner 8 bit dan nilainya menjadi red=11111111,green=11111111, dan blue=11111111. Ketiga biner yang didapat kemudian digabung menjadi satu dimulai dari reed, green, blue dan menjadi pixel =111111111111111111111111. Biner pixel lalu dikonversi menjadi bilangan integer. Pixel hitam mempunyai integer = 16777215. Bilangan tersebut yang nantinya sebagai message yang akan di enkripsi. Cara sebaliknya dapat digunakan untuk membentuk sebuah pixel berwarna dari hasil enkripsi.

Pada prinsipnya proses dekripsi melakukan hal yang sama dengan ketika mengenkripsi sebuah pixel, yang membedakan adalah algoritma dekripsinya saja. Pixel yang ingin di dekripsi harus diubah dalam bentuk yang dapat di masukkan ke dalam algoritma kemudian hasilnya dibentuk ulang menjadi sebuah pixel berwarna.

A. ElGammal

Sebelum memulai mengenkripsi citra berwarna menggunakan algoritma elgammal, yang pertama dilakukan adalah pembangkitan kunci :

P = 16777213
G = 14872311
X = 9735281
Y = 4881904

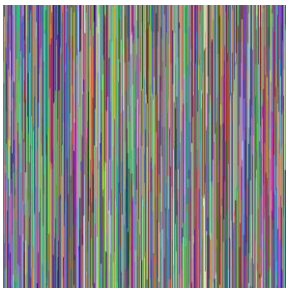
Pertama yang dilakukan adalah menentukan P. Nilai P diassign manual 16777213 dengan alasan karena sebuah message yang akan di enkripsi maupun dekripsi harus diantara 0 sampai (P - 1) dan P adalah bilangan prima. Supaya tidak mendapatkan hasil yang diluar batas, maka di cari bilangan prima terdekat dan lebih kecil dari 16777215.

Kemudian untuk G dan X dirandom sesuai syarat masing masing. yang terakhir nilai Y dihitung sesuai aturannya.

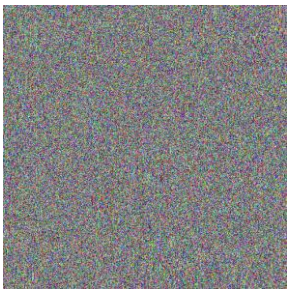
Setelah mendapatkan properti algoritma proses enkripsi citra dimulai. Iterasi setiap pixel, enkripsi pixel tersebut, chiper pixel disusun sesuai plain pixel dan membentuk image baru.

Bagan 3 Lenna.jpg (Plainimage)

Setelah enkripsi gambar dengan cara iterasi setiap pixelnya berhasil, maka akan didapat 2 buah cipher image sesuai dengan karakteristik algoritma Elgammal.



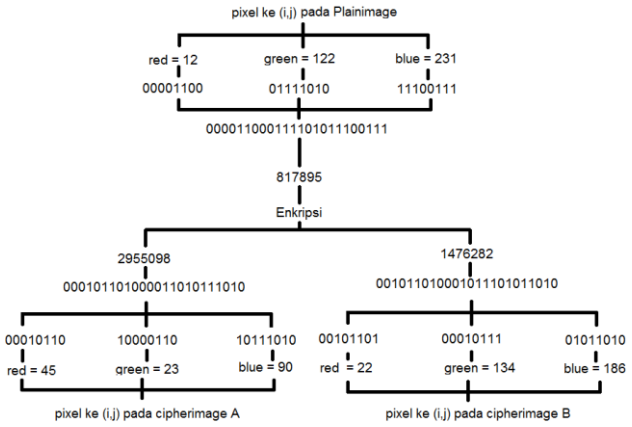
Bagan 4 Cipherimage A



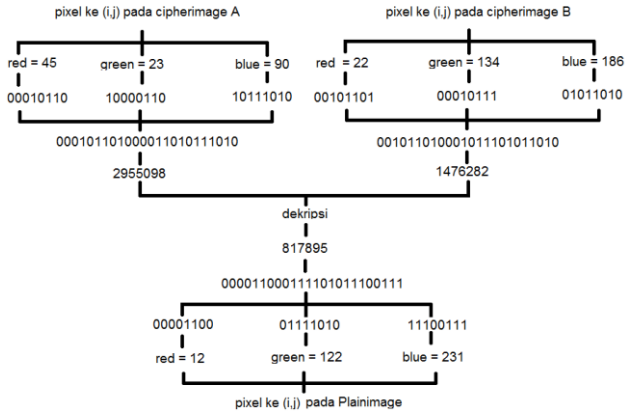
Bagan 5 CipherImage B

Cipherimage A lebih terlihat garis – garis sebagai penyusunnya dan Cipher B terlihat lebih random. Ini dikarenakan untuk mendapatkan Cipherimage B memuat message yang bernilai tidak menentu, sehingga image yang dihasilkan terlihat lebih random dan plainimage sudah tidak dapat dikenali.

Pixel ke (i,j) dari plainimage di enkripsi menjadi pixel ke (i,j) cipherimage A dan pixel ke (i,j) cipherimage B. berikut ilustrasi enkripsi dan dekripsi pixel ke (i,j).



Bagan 6Ilustrasi Enkripsi Pixel ElGammal



Bagan 7Ilustrasi Dekripsi Pixel ElGammal

Setelah dilakukan dekripsi pada cipherimage A dan cipherimage B, maka didapat plainimage kembali.

B. RSA

Sebelum melakukan enkripsi menggunakan algoritma RSA, yang pertama dilakukan adalah pembangkitan kunci terlebih dulu, berikut properti algoritma yang digunakann :

P = 4093
Q = 4099
N = 16777207
Phi = 16769016
E = 15769739
D = 15225251

Pada persoalan kali ini, penentuan besar kunci tidak dapat dilakukan secara random sepenuhnya, kita harus memahami batasan batasan kunci dan yang paling penting adalah seluruh message yang kita punya dapat dienkripsi dengan kunci yang kita bangkitkan.

Perlu di tekankan bahwa pada algortima RSA, message yang dapat di enkripsi yaitu antara 0 sampai N – 1. Itu sebabnya penentuan P dan Q pada persoalan kali ini di cari pasangan bilangan prima dari 2 sampai 99991 yang mempunyai nilai dibawah 16777215 (batas message) dan berjarak paling sedikit dengan nilai tersebut.

Effort yang cukup besar untuk menentukan kunci E dan D. Berikut ditampilkan pseudo code untuk membangkitkan kunci E dalam bentuk notasi algoritmik:

```
Function GeneratorE(Phi : Key) → key
{ mengembalikan key E sesuai dengan aturannya }
Kamus :
    E : Key
Algoritma :
    do
        E ← Random bilangan dari 2 sampai Phi
    while (GCD(E,Phi) ≠ 1)
    → E
```

Dibawah ini juga ditampilkan pseudo code untuk pembangkitan kunci D dalam bentuk notasi algoritmmik:

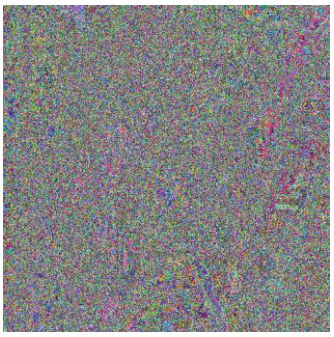
```
Function GeneratorD(Phi : Key, E : Key) → key
{ mengembalikan key D sesuai dengan aturannya }
Kamus :
    D : Key
    I : integer
    Flag : boolean
    Temp : Key
Algoritma :
    I ← 1
    Flag ← true
    Temp ← 0
    While (Flag) do
        Temp ←1 + (I * Phi)
        If (Temp mod E = 0) then
            D ← Temp / E
            Flag ← false;
        I ← I + 1
    →D
```

Setelah mendapatkan seluruh properti yang dibutuhkan, maka sudah dapat dilakukan enkripsi image dengan menggunakan algoritma RSA.

Berikut merupakan plainimage yang digunakan :

Bagan 8 Plainimage

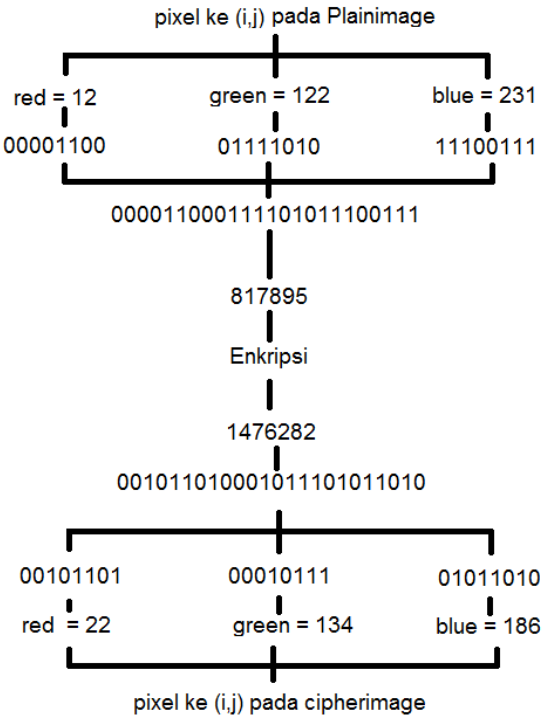
Plain image diatas kemudian dienkripsi menggunakan algoritma RSA dan menghasilkan cipherimage seperti berikut :



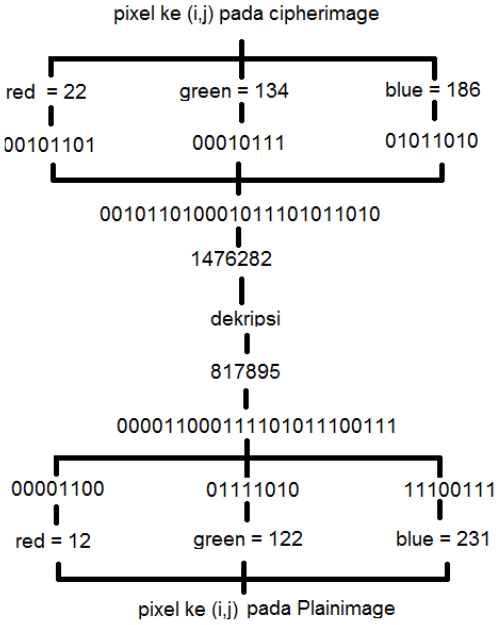
Bagan 9 Cipherimage RSA

Hasil yang didapat terlihat random disetiap pixelnya dan tidak membentuk pola sehingga gambar asli benar – benar sudah tidak dapat dilihat kembali.

Proses enkripsi image dilakukan dengan cara menelusuri tiap pixel pada plainimage dan enkripsi dilakukan tiap pixel sehingga menghasilkan pixel baru yang disusun menjadi sebuah cipherimage baru. Berikut ditampilkan ilustrasi enkripsi dan dekripsi tiap pixelnya :



Bagan 10 Ilustrasi Enkripsi RSA



Bagan 11 Ilustrasi Dekripsi RSA

Untuk mendapatka kembali plainimage cukup mendekripsi cipherimage.

V. PERBANDINGAN

Pada prinsipnya, Kriptografi kunci nirsimetris semua memiliki ide yang sama, dan memiliki karakteristik kunci yang digunakan menggunakan digit yang besar. Namun pada persoalan ini, untuk membatasi agar pesan dan hasil enkripsi tidak keluar dari batas yang sudah ditentukan sebelumnya maka kunci yang digunakan tergolong kecil. Terlepas dari itu, berikut merupakan perbandingkan altara algoritma ElGammal dan RSA dilihat dari beberapa aspek:

A. Pembangkitan Kunci

Pada persoalan kali ini, pembangkitan kunci pada ElGammal dirasa lebih sederhana, hanya pada bagian penentuan P saja yang membutuhkan perhatian lebih. Selebihnya hanya random dan melakukan komputasi kecil.

Berbeda dengan algoritma ElGammal, pada algoritma RSA membutuhkan effort yang lebih ketika pembangkitan kunci untuk kasus ini. Pertama penentuan bilangan prima P dan Q sehingga P * Q memiliki jarak yang seminimal mungkin dari 16777215 agar message dan hasil tidak keluar batas dan kedua penentuan kunci E dan D dilakukan secara random dengan syarat yang rumit sehingga harus dilakukan random berulang kali untuk mendapatkan nilai yang pas.

B. Enkripsi

Proses enkripsi jelas ElGammal memiliki metode enkripsi paling rumit dari segi komputasinya.

Komputasi yang berat membutuhkan resource yang besar pula untuk mengeksekusinya. Untuk Mengenkripsi Bagan 3 dibutuhkan waktu 19.28 detik menggunakan Algoritma Elgammal dan membutuhkan waktu 9.06 detik dengan algoritma RSA. Dari segi waktu eksekusi terlihat jelas algoritma Elgammal membutuhkan waktu yang lebih besar.

C. Hasil Enkripsi

Hasil enkripsi dari algoritma Elgammal menjadi 2 buah cipherimage yang saling berkaitan. Berbeda dengan algoritma kriptografi lainnya yang menghasilkan 1 buah cipher. Hasil yang diperolehpun terlihat sangat random sehingga gambar asli sudah tidak dapat dikenalilagi.

Untuk algoritma RSA, hasil yang didapat terlihat random dan gambar asli sudah tidak dapat dikenali.

Satu satunya hal yang terlihat beda adalah elgammal menghasilkan 2 buah cipherimage sedangkan RSA menghasilkan 1 buah cipherimage.

D. Dekripsi

Proses dekripsi berhubungan juga dengan kerumitan enkripsi. Untuk mendekripsi cipherimage pada bagan 4 dan 5 dibutuhkan waktu 9.73 detik dengan algoritma ElGammal dan untuk mendekripsi cipher image pada bagan 9 membutuhkan waktu 6.98 detik dengan algoritma RSA.

Algoritma RSA lebih cepat dalam proses enkripsi dan dekripsinya.

Dalam kasus ini, akurasi ketepatan hasil dekripsi dibanding dengan image asli dipengaruhi sepenuhnya oleh penentuan kunci. Oleh karena itu, ElGammal memiliki akurasi lebih tinggi dibandingkan RSA seperti yang sudah dijelaskan sebelumnya.

E. Keamanan Algoritma

Semakin besar kunci, semakin rumit algoritma pembangkitan kunci, semakin rumit algoritma enkripsi dan dekripsinya maka keamanan algoritma tersebut semakin tinggi. Dalam hal ini, Algoritma kriptografi kunci publik ElGammal dapat dikatakan lebih aman dari Algoritma RSA.

VI. KESIMPULAN

Pada penyusunan makalah ini, penulis dapat menarik beberapa kesimpulan. Antara lain :

a. Teknik kriptografi kunci nirsimetris memiliki kekuatan pada kuncinya. Sehingga kerumitan algoritma pembangkitan kunci menjadi kekuatan sebuah algoritma.
b. ElGammal memiliki kekuatan yang lebih tinggi dibandingkan algoritma RSA yang cenderung lebih sederhana.
c. Disamping kesederhanaannya, algoritma RSA tergolong algoritma kriptografi yang kuat sehingga pada nyatanya algoritma ini lebih sering digunakan.

REFERENCES

[1] Munir, Rinaldi, Kriptografi Kunci Publik, Program Studi Teknik Informatika.
Waktu akses: 13 Maret 2013 pukul 17.00.

[2] Munir, Rinaldi, Algoritma RSA, Program Studi Teknik Informatika.
Waktu akses : 12 Mei 2013 pukul 19.00.

[3] Munir, Rinaldi, Algoritma ElGamal, Program Studi Teknik Informatika.
Waktu akses : 11 Mei 2013 pukul 15.38.

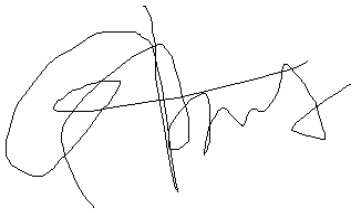
[4] http://www.tataelxsi.com/whitepapers/pub_key2.pdf
Waktu akses : 29 April 2013

[5] http://www.cimt.plymouth.ac.uk/resources/topics/art003.pdf
Waktu akses : 29 April 2013

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 13 Mei 2013



Whilda Chaq - 13511601