Aplikasi Penyembunyian Data Rahasia Ke Citra Digital dengan menggunakan Metode Rivest Shamir Adleman dan Least Significant Bit

Feliso Zalukhu
STMIK Budi Darma Medan, Jl. SM.Raja No.338 Sp.Limun Medan, Sumut, Indonesia
E-Mail: zalukhufeliso@gmail.com

ABSTRACT
Steganography is the science of art and science to make hidden messages in a certain way so that other people besides the sender and recipient will realize there is a hidden message. Cryptography is science and art to learn about processing and securing messages safely. This study will discuss the combination of the Rivest Shamir Adleman (RSA) and Least Significant Bit (LSB) methods on how to hide a secret message in a digital image. A text message before being inserted into a digital image is first encrypted with the RSA Algorithm, the results of the encryption are inserted digital imagery with LSB Algorithm. The length of the text message depends on the size of the container media. Applications designed using Microsoft Visual Basic 2008 also provide interfaces for generating randomly needed keys. Applications can be used to hide secret messages in digital images where the color changes of input images with results are not clearly visible.

Keywords: Steganography, Cryptography, Digital Image

PENDAHULUAN

Kemudahan dalam penggunaan dan semua fasilitas yang lengkap dan merupakan keunggulan yang dimiliki oleh internet dan bukan menjadi satu rahasia umum lagi dikalangan masyarakat pengguna internet pada saat sekarang ini. Seiring dengan berkembangnya media online/internet dan aplikasi menggunakan teknologi internet semakin pasti bertambah pula kejahatan dalam sistem informasi. Dengan berbagai metode teknik pecurian informasi yang berkembang, banyak mencoba untuk mengakses informasi yang bukan haknya.

Berbeda dengan teknik kriptografi, steganografi menyembunyikan pesan rahasia sehingga pihak lain selain penerima pesan tidak menyadari keberadaan pesan yang disembunyikan dalam menghindari kecurigaan orang dan menghindari keinginan orang untuk mengetahui isi pesan rahasia tersebut. Informasi rahasia tersebut akan disimpan di dalam suatu file penampung informasi yang dapat berbentuk berbagai jenis *file* multimedia digital seperti teks, citra, audio, video. Salah satu metode steganografi yang paling populer adalah metode *Least Significant Bit* (LSB). Pada metode LSB, ukuran data yang akan disembunyikan bergantung pada ukuran wadah penampung. Perubahan pada LSB ini akan terlalu kecil untuk terdeteksi oleh mata manusia sehingga pesan dapat disembunyikan secara efektif. Proses ekstraksi pesan dapat dengan mudah dilakukan dengan mengekstrak LSB dari masing-masing *pixel* dan menuliskannya ke *output file* yang akan berisi pesan tersebut. Keuntungan metode LSB adalah mudah dalam pengimplementasian dan proses *encoding* yang cepat.

Sementara itu, untuk meningkatkan sekuritas dari informasi yang disembunyikan, maka sebelum disisipkan, informasi tersebut dapat dienkripsi terlebih dahulu. Metode enkripsi yang populer dan banyak digunakan adalah metode Rivest Shamir Adleman (RSA). Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (*Massachussets Institute of Technology*) pada tahun 1976, yaitu Ron (R)ivest, Adi (S)hamir dan Leonard (A)dleman. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemaktoran dilakukan untuk memperoleh kunci privat. Selama pemaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang bagus, maka selama itu pula keamanan algoritma RSA tetap terjamin.

Agar pembahasan dapat terfokus, maka dilakukan pembatasan masalah sebagai berikut:

1. Data yang diinput berupa citra sampul dalam format JPG,BMP dan PNG.
2. *Input* dokumen teks sebagai pesan rahasia yang akan disisipkan memiliki format TXT, RTF, DOC dan DOCX dimana data yang terbaca hanya data *plaintext*

saja, tanpa adanya format dan tidak mencakup gambar ataupun tabel.
3. Ukuran citra yang dapat diproses memiliki batasan minimal 100 x 100 dan maksimal 1000 x 1000.
4. Panjangnya pesan yang dapat disisipkan tergantung pada ukuran citra digital yang digunakan.

LANDASAN TEORI

Steganografi adalah seni dan ilmu membuat pesan secara tersembunyi atau menyembunyikan sebuah pesan dengan suatu cara sehingga selain pengirim dan penerima, sehingga tidak semua orang mengetahui atau menyadari bahwa ada suatu pesan rahasia. Sebaliknya, kriptografi menyamarkan arti dari suatu pesan, tapi tidak menyembunyikan bahwa ada suatu pesan. Kata "steganografi" berasal dari bahasa Yunani steganos, yang artinya “tersembunyi atau terselubung”, dan graphein, “menulis” [1].

2.1. Metode *Least Significant Bit* (LSB)

Least Significant Bit (LSB) adalah Metode yang digunakan untuk menyembunyikan pesan pada media digital Contohnya, pada berkas *image* pesan dapat disembunyikan dengan menggunakan cara menyisipkannya pada bit rendah atau bit yang paling kanan (LSB) pada data piksel yang menyusun *file* tersebut. Pada berkas bitmap 24 bit, setiap piksel (titik) pada gambar tersebut terdiri dari susunan tiga warna merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bit (byte) dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Dengan demikian, pada setiap piksel berkas bitmap 24 bit kita dapat menyisipkan 3 bit data [1].

Kekurangan dari LSB *Insertion* adalah dapat secara drastis mengubah unsur pokok warna dari piksel. Ini dapat menunjukkan perbedaan yang nyata dari *cover image* menjadi *stego image*, sehingga tanda tersebut menunjukkan keadaan dari Steganografi. Variasi warna kurang jelas dengan 24 bit *image*, bagaimanapun file tersebut sangatlah besar. Antara 8 bit dan 24 bit *image* mudah diserang dalam pemrosesan *image*, seperti *cropping* (kegagalan) dan *compression* (pemampatan).

Keuntungan yang paling besar dari algoritma LSB ini adalah cepat dan mudah. Dan juga algoritma tersebut memiliki *software* Steganografi yang mendukung dengan bekerja di antara unsur pokok warna LSB melalui manipulasi *pallete* (lukisan).

Cara paling umum untuk menyembunyikan pesan adalah dengan memanfaatkan *Least Significant Bit* (*LSB*). Walaupun terdapat kekurangan pada metode ini, tetapi kemudahan implementasinya membuat metode ini tetap digunakan sampai sekarang. Contoh ilustrasinya sebagai berikut: jika digunakan *image* 24 *bit* warna sebagai media, sebuah *bit* dari masing-masing komponen *Red*, *Green*, dan *Blue*, dapat digunakan sehingga 3 *bit* dapat disimpan pada setiap *pixel*. Sebuah *image* 800x 600 *pixel* dapat digunakan untuk menyembunyikan 1.440.000 *bit* (180.000 bytes) data rahasia. Misalnya, di bawah ini terdapat 3 *pixel* dari *image* 24 *bit* warna :

(00100111 11101001 11001000)
(00100111 11001000 11101001)
(11001000 00100111 11101001)

jika diinginkan untuk menyembunyikan karakter A (**100000011**) dihasilkan :

(0010011**1** 1110100**0** 1100100**0**)
(0010011**0** 1100100**0** 1110100**0**)
(1100100**0** 0010011**1** 1110100**1**)

dapat dilihat bahwa hanya 3 *bit* saja yang perlu diubah untuk menyembunyikan karakter A ini.

Jika pesan = 10 *bit*, maka jumlah *byte* yang digunakan = 10 *byte*.

Contoh susunan *byte* yang lebih panjang :

00110011 10100010 11100010 10101011 00100110
10010110 11001001 11111001 10001000 10100011

Pesan : **1110010111**

Hasil penyisipan pada *bit* LSB :

0011001**1** 1010001**1** 1110001**1** 1010101**0** 0010011**0**
1001011**1** 1100100**0** 1111100**1** 1000100**1** 1010001**1**

2.2. Kriptografi

Kata kriptografi (*cryptography*) berasal dari bahasa Yunani, yaitu *κρυπτο* (kripto), yang artinya tersembunyi, dan *γραφία* (grafia), yang artinya sesuatu yang tertulis. Jika digabungkan dapat diartikan sebagai sesuatu yang tertulis secara rahasia. Jadi kriptografi adalah ilmu dan seni penyimpanan pesan, data, atau informasi secara aman. Kriptografi mempelajari tentang bagaimana merahasiakan suatu informasi penting kedalam suatu bentuk yang tidak dapat dibaca oleh siapapun serta mengembalikannya menjadi informasi semula dengan menggunakan berbagai macam teknik yang telah ada sehingga informasi tersebut tidak dapat diketahui oleh pihak manapun yang bukan pemilik atau yang tidak berkepentingan. Dengan perkembangan bidang kriptografi, pembagian antara apa yang termasuk kriptografi dan apa yang tidak telah menjadi

kabur. Dewasa ini kriptografi dapat dianggap sebagai perpaduan antara studi teknik dan aplikasi yang tergantung kepada keberadaan masalah-masalah sulit**.**

2.3. Metode RSA (*Rivest Shamir Adleman*)

Algoritma RSA dibuat oleh 3 orang peneliti dari MIT (Massachussets Institute of Technology) pada tahun 1976, yaitu Ron (R)ivest, Adi (S)hamir dan Leonard (A)dleman. Keamanan algoritma RSA terletak pada sulitnya memfaktorkan bilangan yang besar menjadi faktor-faktor prima. Pemaktoran dilakukan untuk memperoleh kunci privat. Selama pemaktoran bilangan besar menjadi faktor-faktor prima belum ditemukan algoritma yang bagus, maka selama itu pula keamanan algoritma RSA tetap terjamin [1].

Algoritma RSA (*Rivest, Shamir dan Adleman*) memiliki besaran-besaran sebagai berikut:

1. p dan q bilangan prima (rahasia)
2. n = p . q (tidak rahasia)
3. $\phi(n) = (p - 1)(q - 1)$ (rahasia)
4. e (kunci enkripsi) (tidak rahasia)
5. d (kunci dekripsi) (rahasia)
6. m (plainteks) (rahasia)
7. c (cipherteks) (tidak rahasia)

2.3.1 Proses Pembentukan Kunci Pada Algoritma RSA

Algoritma untuk membangkitkan pasangan kunci pada algoritma RSA adalah sebagai berikut:

1. Pilih dua buah bilangan prima sembarang, p dan q.
2. Hitung n = p . q (sebaiknya $p \neq q$, sebab jika p = q maka $n = p^2$ sehingga p dapat diperoleh dengan menarik akar pangkat dua dari n).
3. Hitung $\phi(n) = (p - 1)(q - 1)$
4. Pilih kunci publik, e, yang relatif prima terhadap $\phi(n)$.
5. Bangkitkan kunci privat dengan menggunakan persamaan $e . d \equiv 1 \pmod{\phi(n)}$.

Hasil dari algoritma di atas:

Kunci publik adalah pasangan (e, n) dan Kunci privat adalah pasangan (d, n).

Contoh proses pembentukan kunci pada metode RSA adalah sebagai berikut :

1. Misalkan : p = 31; q = 47, maka n = 31 * 47 = 1457.
2. Ø(n) = 30 * 46 = 1380.
3. Pilih d secara *random* dalam *range* (48,1379); misalkan : d = 107.
4. Hitung nilai $e \equiv d^{-1} \pmod{(Ø(n))}$ dengan menggunakan tabel *Extended Euclidean*. Sesuai dengan contoh di atas, maka diperoleh e = 503.

2.3.2 Proses Enkripsi Pada Algoritma RSA

Algoritma enkripsi dari RSA (*Rivest, Shamir dan Adleman*) dapat dirincikan sebagai berikut:

1. Ambil kunci publik penerima pesan, e dan modulus n.
2. Nyatakan plainteks m menjadi blok-blok m_1, m_2, …, sedemikian sehingga setiap blok merepresentasikan nilai di dalam selang [0, n – 1].
3. Setiap blok m, dienkripsi menjadi blok c_i, dengan rumus $c_i = m_i^e \bmod n$.

2.3.3 Proses Dekripsi Pada Algoritma RSA

Proses dekripsi pada algoritma RSA ini cukup sederhana, yaitu setiap blok cipherteks c_i didekripsi kembali menjadi blok m_i dengan rumus $m_i = c_i^d \bmod n$.

PEMBAHASAN

3.1 Analisa Masalah

Analisa sistem adalah pembelajaran sebuah sistem dan komponen-komponennya sebagai prasyarat *system design* / desain sistem dan spesifikasi sebuah sistem yang baru. Berpindah dari definisi klasik analisa sistem ini ke suatu yang lebih kontemporer, analisa sistem adalah sebuah istilah yang secara kolektif mendepenelitiankan fase-fase awal pengembangan sistem.

Dalam proses analisa ini, seorang penganalisa akan melakukan beberapa tahapan kerja berikut:

1. Menganalisa proses kerja dari sistem yang akan dibuat.
2. Menjabarkan menganalisa input, proses, dan output secara sistematika
3. Menggambarkan model dari sistem yang akan dibuat.

3.2 Analisa Proses Metode *Rivest Shamir Adleman* (RSA)

Didalam metode RSA ada dua proses yang perlu dilakukan yaitu proses pembentukan kunci dengan tujuan untuk mendapatkan nilai *public key* dan *private key* dan proses *enkripsi* dengan tujuan untuk mendapatkan nilai cipheteksnya [4]. Agar dapat lebih memahami mengenai prosedur kerja dari sistem, maka diberikan sebuah contoh sederhana berikut ini:

1. Proses pembentukan kunci
   a. Pilih dua buah bilangan prima acak, misalkan dipilih p = 17, q = 19.
   b. Hitung:
      N = p * q , N = 17 * 19 N = 323
   c. Hitung:

T(n) = (p – 1) * (q – 1)
T(n) = (17 – 1) * (19 – 1)
T(n) = 16 * 18
T(n) = 288

d. Pilih kunci privat, d, secar acak, misalkan dipilih d = 23.
e. Hitung:
$e = d^{-1} \bmod T(n)$
$e = 23^{-1} \bmod 288$
e = 263
f. Output:
Private key: (d, n) = (23, 323), Public key: (e, n) = (263, 323)

2. Proses enkripsi
Pesan = 'ABC'
'**A**' = 65 = 0100 0001 '**B**' = 66 = 0100 0010 '**C**' = 67 = 0100 0011
Berdasarkan proses pembentukan kunci, diperoleh n = 323, sehingga:
$2^b \leq n$
$2^b \leq 323 \rightarrow b = 8$ (karena $2^8 = 256 \leq 323$ yang merupakan nilai terdekat ke 323)
Hal ini berarti bahwa panjang bit dari subblok adalah 8.
M(1) = 0100 0001 = 65
M(2) = 0100 0010 = 66
M(3) = 0100 0011 = 67
Hitung:
$C(1) = M(1)^e \bmod N$
$C(1) = 65^{263} \bmod 323$
Proses:
263 = 256 + 4 + 2 + 1
Pangkat 1 : 65 mod 323 = 65 [dipilih]
Pangkat 2 : 65^2 mod 323 = 26 [dipilih]
Pangkat 4 : 26^2 mod 323 = 30 [dipilih]
Pangkat 8 : 30^2 mod 323 = 254
Pangkat 16 : 254^2 mod 323 = 239
Pangkat 32 : 239^2 mod 323 = 273
Pangkat 64 : 273^2 mod 323 = 239
Pangkat 128 : 239^2 mod 323 = 273
Pangkat 256 : 273^2 mod 323 = 239 [dipilih]
65^263 mod 323 = (65 x 26 x 30 x 239) mod 323
65^263 mod 323 = 278
C(1) = 278
$C(2) = M(2)^e \bmod N$
$C(2) = 66^{263} \bmod 323$
Proses:
263 = 256 + 4 + 2 + 1
Pangkat 1 : 66 mod 323 = 66 [dipilih]
Pangkat 2 : 66^2 mod 323 = 157 [dipilih]
Pangkat 4 : 157^2 mod 323 = 101 [dipilih]
Pangkat 8 : 101^2 mod 323 = 188
Pangkat 16 : 188^2 mod 323 = 137
Pangkat 32 : 137^2 mod 323 = 35
Pangkat 64 : 35^2 mod 323 = 256
Pangkat 128 : 256^2 mod 323 = 290
Pangkat 256 : 290^2 mod 323 = 120 [dipilih]
66^263 mod 323 = (66 x 157 x 101 x 120) mod 323
66^263 mod 323 = 195
C(2) = 195
$C(3) = M(3)^e \bmod N$
$C(3) = 67^{263} \bmod 323$
Proses:
263 = 256 + 4 + 2 + 1
Pangkat 1 : 67 mod 323 = 67 [dipilih]
Pangkat 2 : 67^2 mod 323 = 290 [dipilih]
Pangkat 4 : 290^2 mod 323 = 120 [dipilih]
Pangkat 8 : 120^2 mod 323 = 188
Pangkat 16 : 188^2 mod 323 = 137
Pangkat 32 : 137^2 mod 323 = 35
Pangkat 64 : 35^2 mod 323 = 256
Pangkat 128 : 256^2 mod 323 = 290
Pangkat 256 : 290^2 mod 323 = 120 [dipilih]
67^263 mod 323 = (67 x 290 x 120 x 120) mod 323
67^263 mod 323 = 33
C(3) = 33
Agar proses dekripsi dapat dilakukan dengan mudah, maka panjang dari nilai cipher harus sama semua. Sehingga diperoleh:
C(1) = 278
C(2) = 195
C(3) = **0**33
Nilai cipher yang diperoleh = 278195033

2.3. Analisa Proses Metode *Least Significant Bit* (LSB)

Penyembunyian pesan rahasia dapat dilakukan dengan metode *Least Significant Bit* (LSB) [5]. Untuk proses penempelan deretan bit dari *cipher* ke citra digital, maka kita akan menempelkan 4 bit ke sebuah nilai warna piksel.
Nilai cipher = 278195033.
Konversikan nilai setiap digit ke bentuk biner, maka diperoleh:
2 = 0010 , 7 = 0111 , 8 = 1000 , 1 = 0001 , 9 = 1001 , 5 = 0101
0 = 0000 , 3 = 0011 , 3 = 0011

Untuk lebih memahami proses penyisipan pesan rahasia yang telah dienkripsikan dengan metode RSA maka 4 bit nilai cipher dari contoh proses metode RSA diatas disisipkan kesebuah citra, tetapi hanya 3x3 pixel saja yang dijadikan sampel sebagai tempat penyisipan pesan:
Sebagai contoh, digunakan sebuah citra berukuran 3 x 3 dengan warna piksel berikut:

243	186	157
112	119	223
215	123	142

1. Proses penyisipan

Proses penyisipannya adalah sebagai berikut:

Nilai piksel pertama adalah 243 = 1111 0011, maka akan ditempelkan nilai 2 = 0010 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1111 **0010** = 242.

Proses yang sama akan dilakukan terhadap nilai piksel sebelumnya.

Nilai piksel kedua adalah 186 = 1011 1010, maka akan ditempelkan nilai 7 = 0111 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1011 **0111** = 183.

Nilai piksel ketiga adalah 157 = 1001 1101, maka akan ditempelkan nilai 8 = 1000 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1001 **1000** = 152

Nilai piksel keempat adalah 112 = 0111 0000, maka akan ditempelkan nilai 1 = 0

001 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 0111 **0001** = 113.

Nilai piksel kelima adalah 119 = 0111 0111, maka akan ditempelkan nilai 9 = 1001 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 0111 **1001** = 121.

Nilai piksel keenam adalah 223 = 1101 1111, maka akan ditempelkan nilai 5 = 0101 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1101 **0101** = 213.

Nilai piksel ketujuh adalah 215 = 1101 0111, maka akan ditempelkan nilai 0 = 0000 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1101 **0000** = 208.

Nilai piksel kedelapan adalah 123 = 0111 1011, maka akan ditempelkan nilai 3 = 0011 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 0111 **0011** = 115.

Nilai piksel kesembilan adalah 142 = 1000 1110, maka akan ditempelkan nilai 3 = 0011 ke dalam citra digital, maka nilai piksel pertama berubah menjadi 1000 **0011** = 131.

Citra stego yang diperoleh adalah sebagai berikut:

242	183	152
113	121	213
208	115	131

2. Proses ekstraksi

Citra stego yang diperoleh dari proses penempelan cipher di atas akan digunakan untuk mengekstrak keluar bit cipher yang tersimpan di dalamnya.

242	183	152
113	121	213
208	115	131

Konversikan setiap nilai piksel ke bentuk representasi biner, seperti terlihat pada perincian berikut:

242 = 1111 **0010** 183 = 1011 **0111**

152 = 1001 **1000** 113 = 0111 **0001**

121 = 0111 **1001** 213 = 1101 **0101**

208 = 1101 **0000** 115 = 0111 **0011**

131 = 1000 **0011**

Ambil 4 bit akhir dari nilai piksel (nilai LSB), ubah ke bentuk nilai desimal dan digabungkan menjadi nilai cipher.

0010 = 2 **0111 = 7**
1000 = 8 **0001 = 1**
1001 = 9 **0101 = 5**
0000 = 0 **0011 = 3**
0011 = 3

Nilai cipher yang diperoleh = 278195033

3. Proses dekripsi

Nilai cipher yang diperoleh diatas akan didekripsi dengan menggunakan kunci privat dari proses pembentukan kunci. Proses dekripsi dapat dirincikan sebagai berikut:

Nilai cipher yang diperoleh = 278195033

Nilai n = 323, memiliki panjang digit = 3, maka nilai cipher dikelompokkan menjadi subblok dengan panjang 3, sehingga diperoleh:

C(1) = 278
C(2) = 195
C(3) = 33

Hitung:

$M(1) = C(1)^d \bmod N$

$M(1) = 278^{23} \bmod 323$

Proses:

23 = 16 + 4 + 2 + 1

Pangkat 1 : 278 mod 323 = 278 [dipilih]
Pangkat 2 : 278^2 mod 323 = 87 [dipilih]
Pangkat 4 : 87^2 mod 323 = 140 [dipilih]
Pangkat 8 : 140^2 mod 323 = 220
Pangkat 16 : 220^2 mod 323 = 273 [dipilih]

278^23 mod 323 = (278 x 87 x 140 x 273) mod 323

278^23 mod 323 = 65

M(1) = 65

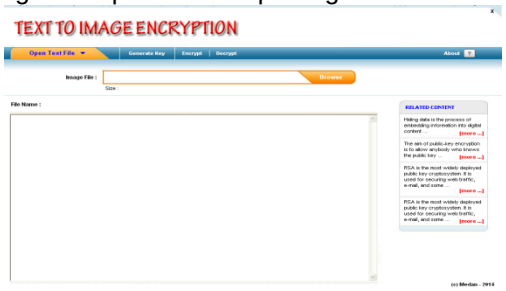
$M(2) = C(2)^d \bmod N$

$M(2) = 195^{23} \bmod 323$

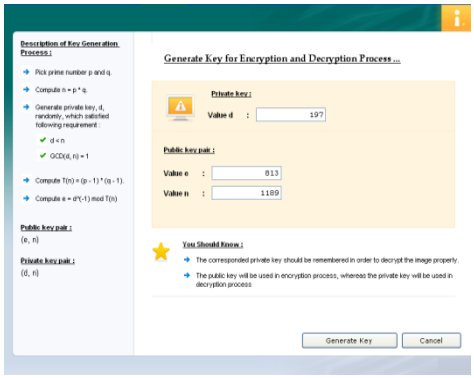
Proses:
23 = 16 + 4 + 2 + 1
Pangkat 1 : 195 mod 323 = 195 [dipilih]
Pangkat 2 : 195^2 mod 323 = 234 [dipilih]
Pangkat 4 : 234^2 mod 323 = 169 [dipilih]
Pangkat 8 : 169^2 mod 323 = 137
Pangkat 16 : 137^2 mod 323 = 35 [dipilih]
195^23 mod 323 = (195 x 234 x 169 x 35) mod 323
195^23 mod 323 = 66
M(2) = 66
M(3) = $C(3)^d$ mod N
M(3) = 33^{23} mod 323
Proses:
23 = 16 + 4 + 2 + 1
Pangkat 1 : 33 mod 323 = 33 [dipilih]
Pangkat 2 : 33^2 mod 323 = 120 [dipilih]
Pangkat 4 : 120^2 mod 323 = 188 [dipilih]
Pangkat 8 : 188^2 mod 323 = 137
Pangkat 16 : 137^2 mod 323 = 35 [dipilih]
33^23 mod 323 = (33 x 120 x 188 x 35) mod 323
33^23 mod 323 = 67
Karena n = 323, maka diperoleh nilai b = 8 ($2^b \leq$ n, lihat penjelasan pada proses pembentukan kunci diatas), sehingga subblok pesan diperoleh:
M(1) = 65 = 0100 0001
M(2) = 66 = 0100 0010
M(3) = 67 = 0100 0011
Gabungkan semua bit pesan, sehingga diperoleh:
0100 0001 0100 0010 0100 0011
Kelompokkan bit pesan menjadi subblok dengan panjang 8 bit dan konversikan ke bentuk karakter, sehingga diperoleh pesan semula.
M(1) = 0100 0001 = 65 = 'A'
M(2) = 0100 0010 = 66 = 'B'
M(3) = 0100 0011 = 67 = 'C'
Pesan rahasia yang diperoleh = 'ABC'

IMPLEMENTASI

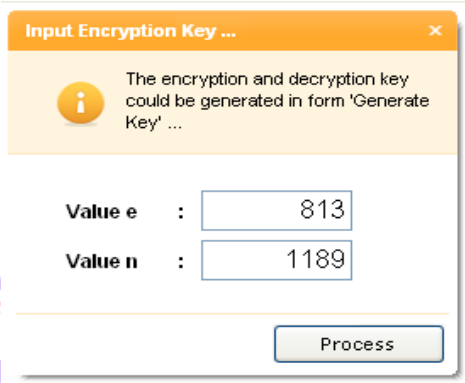
Untuk menggunakan perangkat lunak ini, jalankan *file* "Text to Image Encryption.EXE", maka akan ditampilkan tampilan utama dari program seperti terlihat pada gambar berikut:



Gambar 1 Tampilan Utama



Gambar 2 Tampilan Form Generate Key

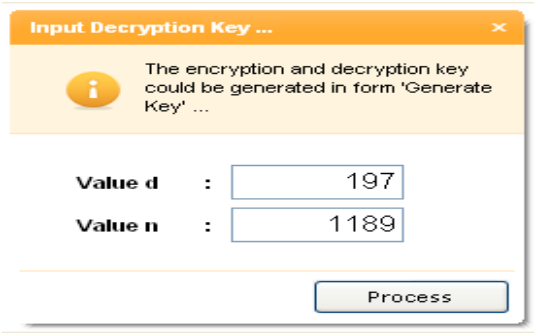


Gambar 3 Tampilan Encrypt

Gambar 4 Tampilan Layar Setelah Proses Enkripsi Berhasil

\

Gambar 5 Tampilan Main Setelah Pemilihan Gambar



Gambar 6 Tampilan Decrypt

Gambar 7 Tampilan Layar Setelah Proses Dekripsi Berhasil

KESIMPULAN

Setelah menyelesaikan penelitian ini, penulis dapat menarik beberapa kesimpulan sebagai berikut:

1. Teknik steganografi dengan metode *Least Significant Bit* (LSB) untuk menyembunyikan pesan teks pada citra JPG, BMP, PNG dapat diaplikasikan dengan menggunakan *Microsoft Visual Basic 2008*
2. Sebuah pesan yang akan disisipkan pada sebuah citra terlebih dahulu di enkripsikan dengan Algoritma RSA
3. Aplikasi dapat digunakan untuk menyembunyikan pesan rahasia pada citra digital dimana perubahan warna citra input dengan hasil tidak kelihatan jelas.

DAFTAR PUSTAKA

1. Munir. R, 2006, Kriptografi, Bandung, Informatika Bandung
2. Sutoyo.T dkk, 2009, Teori Pengolahan Citra Digital, Yogyakarta, ANDI dan UDINUS
3. Rahmat.B, Fairuzabadi.M, 2010, Steganogrfi menggunakan metode Least significant Bit Dengan Kombinasi Kriptografi Vigenere dan RC4, Yogyakarta, Dinamika Informatika
4. Iqbal, Mohammad, 2006. Studi Teknis Metode Enkripsi RSA dalam Perhitungannya, Bandung, Institut Teknologi Bandung.
5. Noertjahyana. A, Hartono. S, Gunadi. K,2012, Aplikasi Metode Steganografy Pada citra digital dengan Metode metode Least significant Bit (LSB) , Universitas Kristen Petra.
6. Wikipedia, Table ASCII, <http://www.ascii.cl/htmlcodes.htm>, 08 Juli 2014, 5:46:10