PENERAPAN METODE ENKRIPSI RSA DAN LSB UNTUK PENGAMANAN PESAN FORMAT BITMAP

I Putu Herryawan
Program Studi Teknik Informatika, Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam
Universitas Udayana
Email : putu.herry@cs.unud.ac.id

Abstrak : Proteksi pada informasi selama pengiriman data atau pada saat penyimpanan menjadi sebuah hal yang penting untuk dilakukan. Metode steganografi dan kriptografi menjadi salah satu pilihan dalam pengamanan data untuk menghindari akses ilegal oleh orang-orang yang tidak berkepentingan. Steganografi membutuhkan dua properti, wadah penampung dan data rahasia yang akan disembunyikan.

Objek pada penelitian ini adalah sebuah citra bitmap yang akan disisipkan pesan yang terenkripsi. Penyisipan dilakukan dengan menggunakan algoritma LSB dan enkripsi-dekripsi dilakukan dengan algoritma RSA. Pada algoritma LSB, 2 bit terakhir pada nilai merah disetiap *pixel* akan diganti dengan 2 bit dari nilai pesan yang telah di enkripsi. Setelah diganti, nilai merah disetiap *pixel* tersebut akan menjadi nilai merah yang baru pada citra.

Steganografi pada citra bitmap dapat meningkatkan keamanan pesan dengan memperkecil kemungkinan terbacanya pesan tersembunyi dalam citra. Mengkombinasikan metode *Least Significant Bit* dan enkripsi RSA, membuat informasi yang dijaga kerahasiaannya akan lebih terjamin.

Kata Kunci : Steganografi, RSA, Least Significant bit.

RSA And LSB Encryption Method Application For The Security Of The Bitmap Formatted Message.

Abstract : Protection on the information during data transmission or during storage becomes an important thing to do. Methods of steganography and cryptography to be one choice in secure data to prevent unauthorized access by people who are not interested. Steganography requires two properties, the container receptacle and confidential data that will be hidden

Objects in this study is a bitmap image to be inserted an encrypted message. Insertion is done using LSB algorithm and the encryption-decryption is done by the RSA algorithm. In the LSB algorithm, the last 2 bits in each pixel red value will be replaced with 2 bits of the messages that have been encrypted. Once replaced, the red value at each pixel will become the new red on the image.

Steganography on bitmap image can enhance security by minimizing the possibility terbacanya message hidden messages in images. Combining the method and the Least Significant Bit RSA encryption, making information more secure will be kept confidential.

Key : Steganografi, RSA, Least Significant bit.

I. PENDAHULUAN

Data dan informasi bagi sebuah organisasi merupakan sesuatu yang sangat berarti. Apalagi jika data tersebut merupakan data penting yang hanya bisa diketahui oleh orang-orang tertentu saja. Proteksi pada informasi selama pengiriman data atau pada saat penyimpanan menjadi sebuah hal yang penting untuk dilakukan. Metode steganografi dan kriptografi menjadi salah satu pilihan dalam pengamanan data untuk menghindari akses ilegal oleh orang-orang yang tidak berkepentingan.

Steganografi sangat kontras dengan kriptografi. Kriptografi merahasiakan makna pesan sementara eksistensi pesan tetap ada, sedangkan steganografi menutupi keberadaan pesan. Steganografi dapat dipandang sebagai kelanjutan dari kriptografi. Dalam prakteknya, pesan dienkripsi terlebih dahulu, kemudian disembunyikan di dalam media lain sehingga pihak ketiga tidak menyadari keberadaan pesan.

Steganografi membutuhkan dua properti, wadah penampung dan data rahasia yang akan disembunyikan. Dengan menggunakan metode modifikasi LSB (Least Significant Bit) hasil yang didapat cukup memuaskan, karena hasil yang diperoleh tidak dapat dibedakan jika dilihat hanya dengan menggunakan mata manusia.

Steganografi bukanlah pengganti kriptografi. Keduanya saling melengkapi satu sama lain. Untuk dapat mencapai maksud penggunaannya steganografi harus memenuhi beberapa kriteria tertentu. Lebih jauh lagi, untuk dapat diaplikasikan sebagai *watermarking*, ada kriteria lain yang harus dipenuhi, selain kriteria steganografi.

Pada tulisan ini akan dibahas pengembangan metode steganografi dengan memanfaatkan citra bitmap sebagai media penyembunyian data. Pemakaian metode enkripsi juga akan dibahas untuk meningkatkan keamanan data yang disembunyikan.

II. METODE PENELITIAN

2.1 Metode LSB (Least Significant Bit)

Least Significant Bit merupakan bit yang paling tidak berarti dalam suatu byte (misal 1000110**1**, LSB nya adalah **1**). Penyisipan pada *least significant bit* merupakan pendekatan yang umum dan sederhana untuk menanamkan informasi pada file pembungkus. Metode LSB akan mengganti LSB dari byte-byte *pixel* dalam citra dengan informasi yang ingin ditanamkan. Oleh karena citra merupakan salah satu objek multimedia yang sering memiliki representasi yang redundan, pengubahan LSB dari *pixel-pixel* ini tidak akan memperlihatkan degadrasi yang terlihat kasat mata.

Terdapat banyak teknik yang telah dibuat berdasarkan pada steganografi dengan metode LSB. Contoh dari teknik-teknik yang ada adalah *EzStego*, *S-Tools* dan *Steganos* (Nugroho

Sutanto, Arnold. 2010). Secara garis besar, teknik-teknik ini dapat kita kelompokkan sesuai dengan penyebaran penempatan pesan pada citra. Cara pertama adalah dengan penempatan secara teratur (mis. Sekuensial) seperti yang dilakukan oleh *EzStego* dan *Steganos*. Cara kedua, pesan disebarkan secara random pada *pixel-pixel* citra. Dengan cara kedua, kunci steganografi dibutuhkan sebagai *seed* untuk menghasilkan angka-angka acak sebagai posisi *pixel* tempat pesan disimpan

Pada citra *bitmap 24 bits*, setiap *pixel* terdiri atas susunan tiga warna merah, hijau dan biru (RGB) yang masing-masing disusun oleh bilangan 8 bits dari 0 sampai 255 atau dengan format biner 00000000 sampai 11111111. Dengan demikian kita dapat menyisipkan *3 bits* data pada setiap *pixel*. Contohnya huruf A dapat kita sisipkan dalam *3 pixel* (IEEE, Johnson dan Jajodia, 1998), misalnya data *pixel* original adalah sebagai berikut:

(00100111 11101001 11001000)

(00100111 11001000 11101001)

(11001000 00100111 11101001)

Sedangkan nilai ASCII dari huruf A adalah 131 dan nilai binernya adalah 10000011. Dengan menyisipkan-nya pada data *pixel* diatas maka akan dihasilkan:

(0010011’1’ 1110100’0’ 1100100’0’)

(0010011’0’ 1100100’0’ 1110100’0’)

(1100100’1’ 0010011’1’ 11101001)

Terlihat hanya empat bit rendah yang berubah, untuk mata manusia maka tidak akan tampak perubahannya. Secara rata-rata dengan metode ini hanya setengah dari data bit rendah yang berubah, sehingga bila dibutuhkan dapat digunakan bit rendah kedua bahkan ketiga.

Berdasarkan konsep LSB di atas, kita bisa mengembangkan sebuah metode penyisipan yang dipilih berdasarkan perhitungan dari password yang dimasukkan. Kemungkinan komposisi nilai RGB yang akan dibentuk adalah berdasarkan permutasi sebagai berikut:

RGB → diambil 1 elemen = 3!/(3-1)! = 3 kemungkinan

1. Bit data disisipkan pada nilai Red.

2. Bit data disisipkan pada nilai Green.

3. Bit data disisipkan pada nilai Blue.

RGB → diambil 2 elemen = 3!/(3-2)! = 6 kemungkinan

4. Bit data disisipkan pada nilai Red dan Green.

5. Bit data disisipkan pada nilai Red dan Blue.

6. Bit data disisipkan pada nilai Green dan Red.

7. Bit data disisipkan pada nilai Green dan Blue.

8. Bit data disisipkan pada nilai Blue dan Red.

9. Bit data disisipkan pada nilai Blue dan Green.

RGB → diambil 3 elemen = 3! = 6 kemungkinan

10. Bit data disisipkan pada nilai Red, Green dan Blue.

11. Bit data disisipkan pada nilai Red, Blue dan Green..

12. Bit data disisipkan pada nilai Green, Red, dan Blue.

13. Bit data disisipkan pada nilai Green, Blue dan Red.

14. Bit data disisipkan pada nilai Blue, Red dan Green.

15. Bit data disisipkan pada nilai Blue, Green dan Red.

2.2 Metode Enkripsi RSA

RSA di bidang **kriptografi** adalah sebuah **algoritma** pada **enkripsi public key**. RSA merupakan algoritma pertama yang cocok untuk **digital signature** seperti halnya enkripsi, dan salah satu yang paling maju dalam bidang kriptografi public key. RSA masih digunakan secara luas dalam **protokol electronic commerce**, dan dipercaya dalam mengamnkan dengan menggunakan kunci yang cukup panjang. Algortima RSA dijabarkan pada tahun 1977 oleh tiga orang: Ron Rivest, Adi Shamir dan Len Adleman dari Massachusetts Institute of Technology. Huruf RSA itu sendiri berasal dari inisial nama mereka (Rivest—Shamir—Adleman). (Lestriandoko, Novita Hadi., Dkk. 2007:1-2)

III. HASIL DAN PEMBAHASAN

3.1 Pembangkitan Kunci

Langkah -langkah untuk membuat sebuah *public key* dan *private key*:

1. Pilih dua bilangan prima $p \neq q$ secara acak dan terpisah untuk tiap-tiap p dan q. Hitung N = (p.q). N hasil perkalian dari p dikalikan dengan q.

2 Hitung $\varphi = (p-1)(q-1)$.

3. Pilih bilangan bulat (integer) antara satu dan φ, (1 < e < φ) yang juga merupakan coprime dari φ.

4. Hitung d hingga (d.e) ≡ 1 (mod φ).

bilangan prima dapat diuji probabilitasnya menggunakan Fermat's little theorem- a^(n-1) mod n = 1

langkah 3 dan 4 dapat dihasilkan dengan algoritma extended Euclidean atau aritmetika modular.

langkah 4 dapat dihasilkan dengan menemukan integer x sehingga d = (x(p-1)(q-1) + 1)/e menghasilkan bilangan bulat, kemudian menggunakan nilai dari d (mod (p-1)(q-1));

Public key terdiri atas:

- N, modulus yang digunakan.

- e, eksponen publik (sering juga disebut eksponen enkripsi).

Keamanan dan Kekuatan RSA

Keamanan algoritma *RSA* terletak pada tingkat kesulitan dalam memfaktorkan bilangan non prima menjadi faktor primanya, yang dalam hal ini $N = p \times q$.

Sekali N berhasil difaktorkan menjadi p dan q, maka $\phi(N) = (p - 1)(q - 1)$ dapat dihitung. Selanjutnya, karena kunci enkrispi e diumumkan (tidak rahasia), maka kunci dekripsi *SK* dapat dihitung dari persamaan $e \cdot d \equiv 1 \pmod{\phi(N)}$.

Algoritma yang paling ampuh untuk memfaktorkan bilangan yang besar belum ditemukan hingga saat ini. Inilah yang membuat algoritma *RSA* tetap dipakai hingga saat ini. Selagi belum ditemukan algoritma yang ampuh untuk memfaktorkan bilangan bulat menjadi faktor primanya, maka algoritma *RSA* tetap direkomendasikan untuk menyandikan pesan.

Algoritma Penyisipan

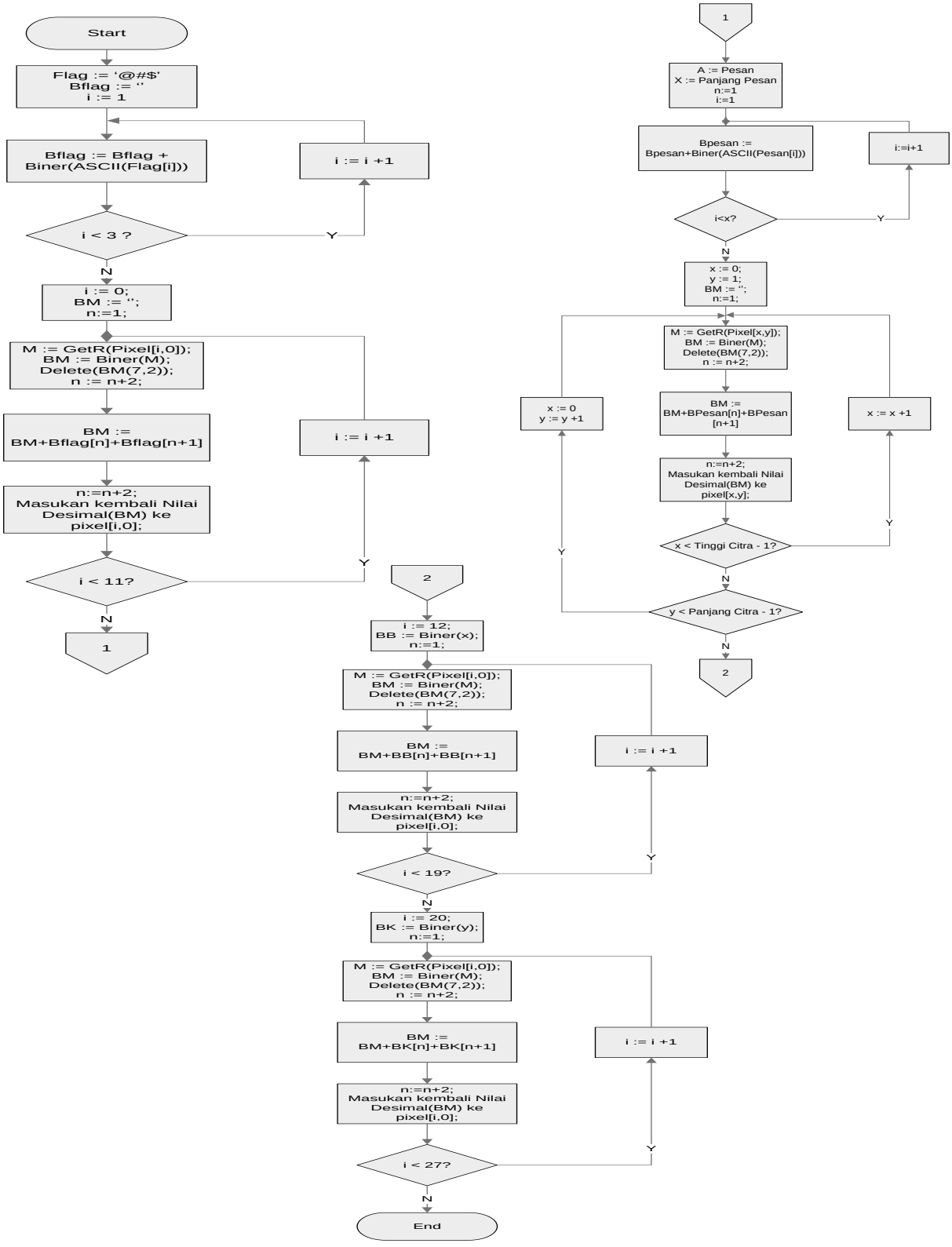
- A ← pesan
- L ← panjang pesan
- n ← 1
- *Looping* dari i ← 1 sampai L
- Begin
- Ubah pesan menjadi ASCII pesan
- Enkripsi ASCII pesan
- Ubah hasil enkripsi menjadi biner pesan
- End Looping
- *Looping* dari y ← 1 sampai tinggi citra
- Begin

- *Looping* dari x ← 0 sampai panjang citra
- Begin
- Ambil nilai merah dari *pixel[x,y]*
- Ubah nilai merah menjadi biner
- Hapus 2 bit biner merah paling kanan
- Tambahkan 2 bit biner pesan ke biner merah
- Ubah biner merah menjadi desimal
- Ganti nilai merah pada citra
- Jika n > panjang biner pesan *Looping* selesai
- End Looping
- Jika n > panjang biner pesan *Looping* selesai
- End Looping
- Simpan informasi panjang dan tinggi pesan tersisip

Algoritma Ekstrak

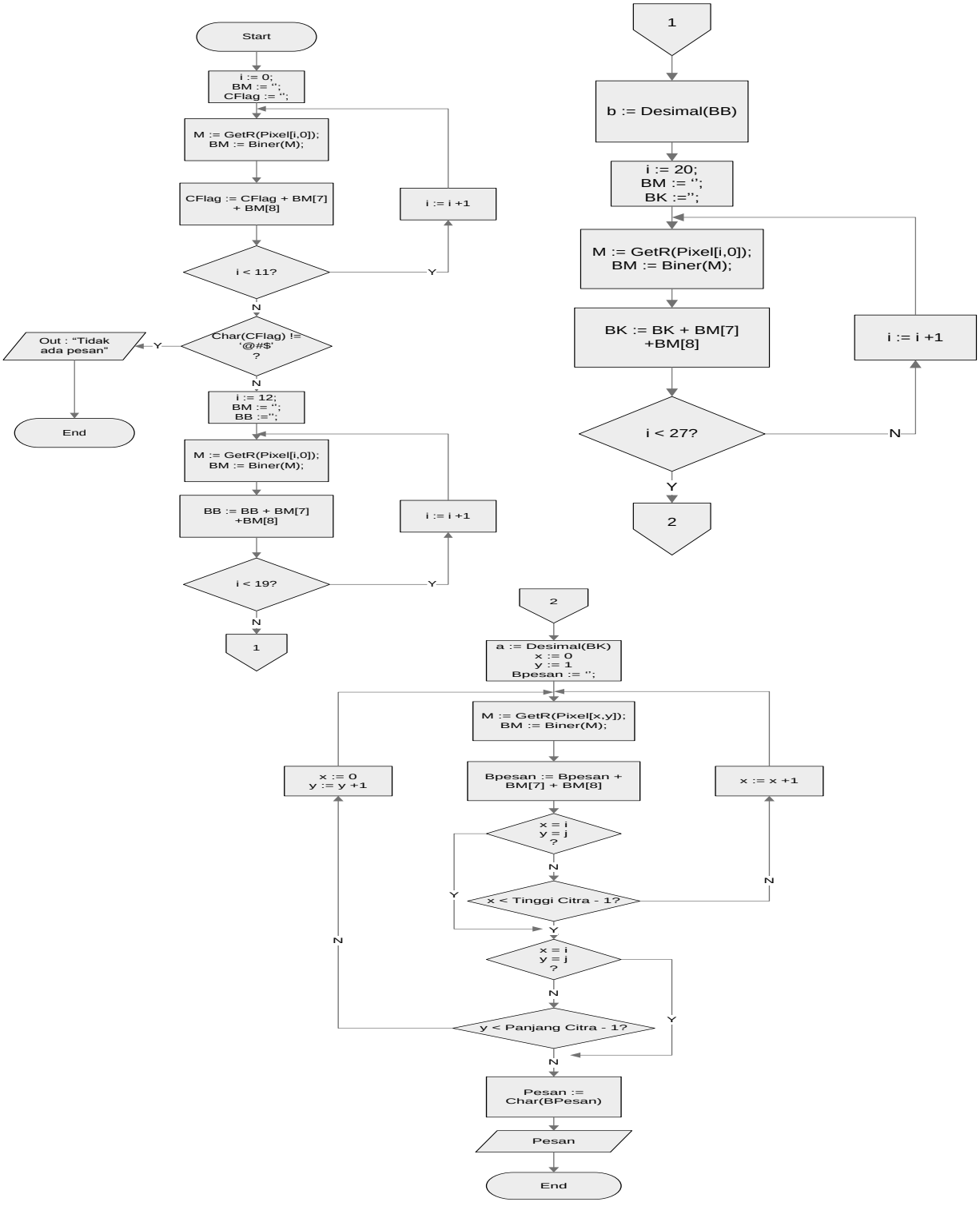
- X ← informasi panjang pesan tersisip
- Y ← informasi tinggi pesan tersisip
- *Looping* dari j ← 1 sampai tinggi citra
- Begin
- *Looping* dari i ← 0 sampai panjang citra
- Begin
- Ambil nilai merah dari *pixel[i,j]*
- Ubah merah menjadi biner merah
- *Copy* 2 bit dari biner merah paling kanan ke biner pesan
- Jika i = x dan j = y maka *Looping* selesai
- End Looping
- Jika i = x dan j = y maka *Looping* selesai
- End Looping
- Ubah biner pesan ke desimal
- Dekripsi nilai desimal pesan
- Kembalikan desimal pesan menjadi pesan teks

3.2 Flochart Penyisipan



Gambar 3.1 Flow chart penyisipan

3.3 *Flowchart* Ekstrak



Gambar3.2 Flowchart Ekstrak

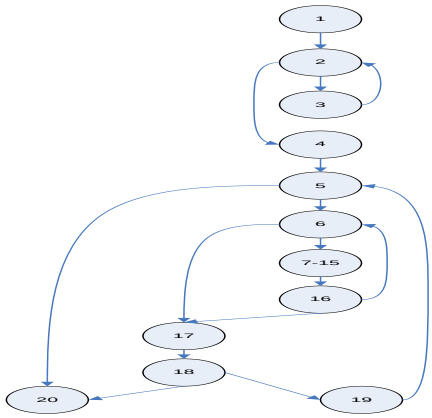
3.4 Impelementasi

Gambar3.3 tampilan program

Pada tampilan awal, citra dan pesan akan diinputkan terlebih dahulu. Karakter sisa akan terlihat pada *field* di bawah citra

Setelah pengisian citra dan pesan, Tombol “Sisip Pesan” diklik. Maka pesan akan disisipkan ke citra. Sebuah kotak pesan akan muncul setelah penyisipan dilakukan.

3.5 Uji White Box



Gambar3.4 Diagram Alir Penyisipan

Cyclomatic Complexity dari diagram adalah :

V(G) = 6 Region

V(G) = 16 Edge – 14 Node + 2 = 6

V(G) = 5 Prediction Node + 1 = 6

Dari hasil perhitungan cyclomatic complexity, terdapat 6 independent path yaitu :

Path 1 : 1-2-3-2-4-5-20

Path 2 : 1-2-3-2-4-5-6-17-18-20

Path 3 : 1-2-3-2-4-5-6-17-18-19-5-20

Path 4 : 1-2-3-2-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-20

Path 5 : 1-2-3-2-4-5-6-7-8-9-10-11-12-13-14-15-16-6-17-18-20

Path 6 : 1-2-3-2-4-5-6-7-8-9-10-11-12-13-14-15-16-6-17-18-19-5-20

3.6 Uji Root Mean Square

Untuk mengetahui kualitas citra yang telah disisipkan pesan, maka dilakukan perhitungan *root mean square.* Perhitungan ini mengambil akar kuadrat dari jumlah selisih warna sebelum dan setelah disisipkan pesan yang telah dikuadratkan pada setiap *pixel*-nya dan dikali dengan 1/MN. Rumusnya adalah sebagai berikut :

$$rms = \sqrt{\frac{1}{MN}\sum_{i=1}^{M}\sum_{j=1}^{N}[f(i,j) - f'(i,j)]^2}$$

Ket. :

M = Tinggi citra

N = Panjang Citra

f(i,j) = *pixel*(i,j) pada citra sebelum disisipkan pesan

f'(i,j) = *pixel*(i,j) pada citra setelah disisipkan pesan

Citra yang diuji, disisipkan pesan dengan jumlah karakter yang berbeda-beda. Rentang nilai RMS dimulai dari 0 hingga 255, sesuai dengan nilai maksimum derajat keabuan. Semakin rendah nilai RMS maka akan semakin mirip citra yang telah disisipkan pesan dengan citra aslinya.

Tabel 3.1 Tabel Pengujian RMS

Jumlah Karakter	Nilai RMS	Persentase kemiripan(%)
44	0,139778231	99,945185007437
256	0,636655349	99,7503312358116
1181	0,687476372	99,7304014226544
3551	1,175309804	99,5390941945206
15073	1,60007735	99,3725186864253

IV. KESIMPULAN

4.1 Kesimpulan

Steganografi pada citra bitmap dapat meningkatkan keamanan pesan dengan memperkecil kemungkinan terbacanya pesan tersembunyi dalam citra. Semakin sedikit karakter pesan yang disisipkan semakin kecil kemungkinan untuk mengetahui adanya pesan pada citra dan semakin mirip pula citra tersebut terhadap aslinya. Penerapan enkripsi RSA juga membuat tingkat keamanan menjadi semakin besar.

Mengkombinasikan metode Least Significant Bit dan enkripsi RSA, membuat informasi yang dijaga kerahasiaannya akan lebih terjamin.

4.2 Saran

Aplikasi steganografi ini dapat dilakukan pengembangan pada metode dan juga teknik enkripsi. Pengembangan juga dapat dilakukan pada pesan atau informasinya. Pesan atau informasi yang disisipkan dapat dikembangkan berupa *file* data, sehingga yang disisipkan tidak hanya berupa *plain text* saja.

V. DAFTAR PUSTAKA

[1] Anneria, Yulia. 2008. “Program Steganalisis Metode LSB pada Citra dengan Enhanced LSB, Uji Chi-Square, dan RS-Analysis”.

[2] Grantham, Beau. 2007. “Bitmap Steganography : An Introduction”. *COT4810 Final Paper Grantham*.

[3] Lestriandoko, Novita Hadi., Dkk. 2007. “Steganografi Pada Multiple Images 24 Bits”.

[4] Munir, Rinaldi. 2006. *Diktat Kuliah IF5054 Kriptografi*.

[5] Pressman, RS. 2002. *Rekayasa Perangkat Lunak*. Ed ke-1. CN Harnaningrum, Penerjemah. Yogyakarta: CV Andi Offset. Terjemahan dari: Software Engineering A Practitioner’s Approach.

[6] Ranova, Denny. 2005. “Pembangunan Pustaka Proteksi Perangkat Lunak dengan Algoritma RSA dan Fungsi Hash MD”.

[7] Sutanto, AN. 2010. “Studi Dan Analisis Teknik-Teknik Pendeteksian Steganografi Dengan Metode LSB Dalam Media Gambar”.