Perbandingan *Steganografi* Metode *Spread Spectrum* dan *Least Significant Bit* (LSB) Antara Waktu Proses dan Ukuran *File* Gambar

M.A. Ineke Pakereng , Yos Richard Beeh, Sonny Endrawan
Fakultas Teknik Program Studi Teknik Informatika
Universitas Kristen Duta Wacana Yogyakarta
Email: inekep200472@yahoo.com, yos.fti.uksw@gmail.com, onnyendrawan@gmail.com

Abstrak :

Steganografi adalah salah satu cara mengamankan pengiriman pengiriman pesan. Spread spektrum dan *least significant bit* (LSB) adalah metode steganografi umum yang sering digunakan. Masukan dari proses *embedding* tersebut akan menjadi file gambar jenis BMP dan JPEG dan file teks, sedangkan keluaran akan menjadi sebuah file teks. Hasil file gambar pembanding antara masukan dan keluaran tidak menunjukkan perubahan yang signifikan, seperti untuk file teks. Metode LSB memiliki proses *embedding* dan proses ekstraksi lebih cepat daripada *spread spectrum*. Namun untuk *spread spektrum,* keamanan lebih baik dibandingkan dengan LSB. Hasil dari proses *embedding* gambar jenis BMP tidak akan jauh berbeda dari gambar aslinya jika dibandingkan dengan JPEG karena jenis BMP telah dipadatkan.

Kata Kunci : *Steganography, Least Significant Bit, Spread Spectrum*

1. Pendahuluan

Teknik dalam melakukan steganografi ada bermacam-macam, antara lain *Least Significant Bit* (LSB), *Algorithm and Transformation*, *Redundant Pattern Encoding*, dan *Spread Spectrum*. Metode LSB dan *spread spectrum* adalah dua metode yang sering sekali digunakan dan merupakan metode yang cukup sederhana dalam melakukan proses steganografi. Selain itu proses *embedding* dan *ekstraksi* dari metode ini juga *relative* cepat, sehingga banyak orang yang menggunakan metode ini untuk menyisipkan pesan.

Metode LSB dan *spread spectrum* merupakan dua metode yang memiliki kelebihan dan kekurangan sendiri-sendiri, oleh karena itu penulis mencoba untuk membandingkannya. Penulis ingin mengetahui bagaimana perbandingan waktu proses *embedding* dan *ekstraksi*, ukuran *file* gambar sebelum dan sesudah disisipkan pesan serta isi dari *file* teks yang akan disisipkan.

2. Tinjauan Pustaka

Penelitian mengenai steganografi telah banyak dilakukan. Seperti penelitian berjudul "Metoda Steganografi Berbasis Least Significant Bit dengan Penyisipan Variable-Size dan Penambahan Redudant Gaussian Noise" yang mengungkapkan bahwa steganografi merupakan suatu teknik menyembunyikan pesan yang telah dienkripsi sedemikian rupa menggunakan metoda kriptografi untuk kemudian pesan tersebut diletakkan di dalam *cover carrier* sehingga pesan tersebut tidak dapat dilihat walaupun dipancarkan pada komunikasi public seperti jaringan komputer. Dalam penelitian ini membahas tentang pemecahan masalah *common-cover-carrier attack* dengan menggunakan penyisipan *variable-size* dan penambahan *redundant Gaussian Noise, stego-images* dibuat sebagai metoda yang digunakan agar dapat mencakup baik dalam hal sistem visualisasi pada manusia maupun *common-cover-carrier attack* [1].

Penelitian lain yang pernah dilakukan adalah "*Spread Spectrum* Steganografi" yang membahas tentang steganografi dengan metode *spread spectrum*, yaitu mengungkapkan tentang kelebihan metode *spread spectrum* dibandingkan dengan metode lainnya walaupun metode *spread spectrum* bukan merupakan metode yang paling ideal. Namun dalam penelitiannya, dinyatakan bahwa metode *spread spectrum* dapat menjawab kebutuhan akan steganografi dan *watermarking* yang tangguh terhadap berbagai macam serangan [2].

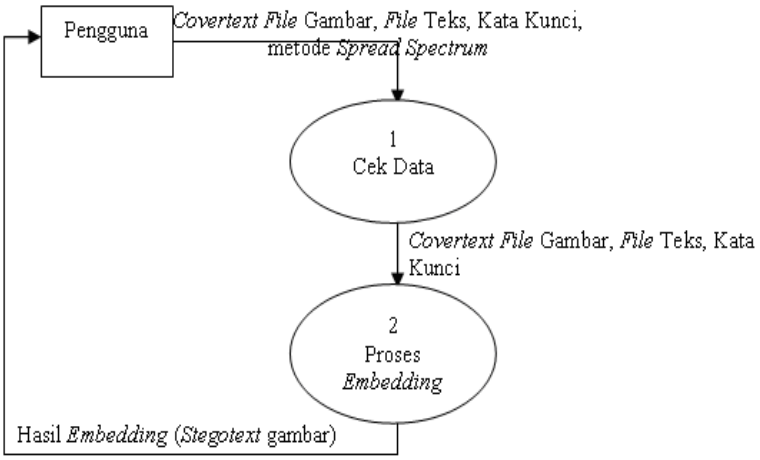
Sementara itu, dalam penelitian " Studi dan Implementasi Steganografi Metode LSB dengan *Preprocessing* Kompresi Data dan Ekspansi Wadah" mengungkapkan tentang cara mengatasi kelemahan metode LSB dalam menyisipkan data. Metode pertama untuk mengatasi masalah tersebut adalah melakukan prepossessing terhadap berkas data yang akan disimpan yaitu dengan jalan memampatkan data tersebut, kedua adalah melakukan *preprosessing* terhadap berkas wadah (*cover*) dengan cara memperbesar berkas wadah (*stretching image*), dan yang ketiga adalah menggabungkan metode pertama dan kedua sekaligus [3].

Penelitian yang pernah dilakukan juga adalah "*Steganografi Melalui Media Gambar dengan Metode Spread Spectrum*" yang isinya tentang steganografi mengunakan media gambar dengan metode *spread spectrum,* tetapi dalam tulisannya ini hanya terbatas pada *file* gambar *.BMP [4]. Berdasarkan pada penelitian sebelumnya yang pernah dilakukan dengan metode *spread spectrum* dan *least significant bit*, maka penulis akan mencoba membandingkan steganografi dengan metode *spread spectrum* dan *least significant bit* pada *file* gambar dengan format *.BMP dan *.JPG.

3. Perancangan Sistem

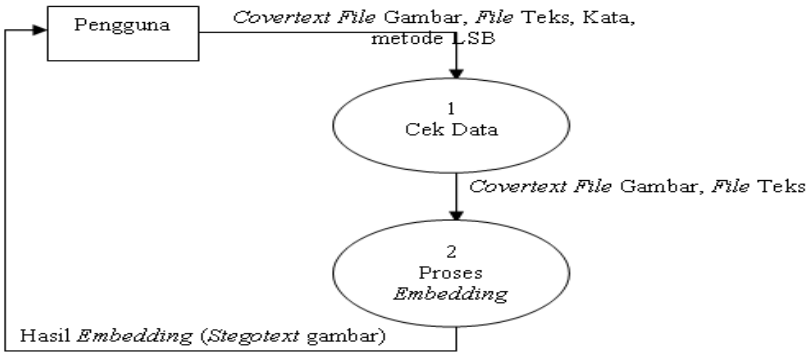
Proses embedding dengan metode Spread Spectrum pada Gambar 1 merupakan proses penyisipan data (embedding) ke dalam covertext dengan memilih gambar, file teks, metode Spread Spectrum dan kata kunci. Pertama dilakukan dahulu pengecekkan data baik covertext gambar dan file teks. Langkah berikutnya adalah memilih metode Spread Spectrum dan

memasukkan kata kunci, selanjutnya menggabungkan kedua file yaitu file gambar dan file teks menjadi hasil embedding (stegotext).



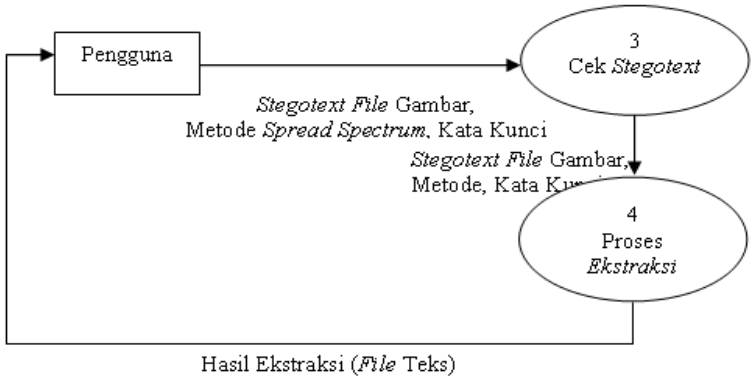
Gambar 1 DFD Level Satu Proses Embedding Metode Spread Spectrum

Proses embedding dengan metode LSB pada Gambar 2 merupakan proses penyisipan data (embedding) ke dalam covertext dengan memilih gambar, file text dan metode steganografi LSB. Pertama dilakukan dahulu pengecekkan data baik covertext gambar dan file teks. Langkah berikutnya adalah memilih metode steganografi LSB, selanjutnya menggabungkan kedua file yaitu file gambar dan file teks menjadi hasil embedding (stegotext).



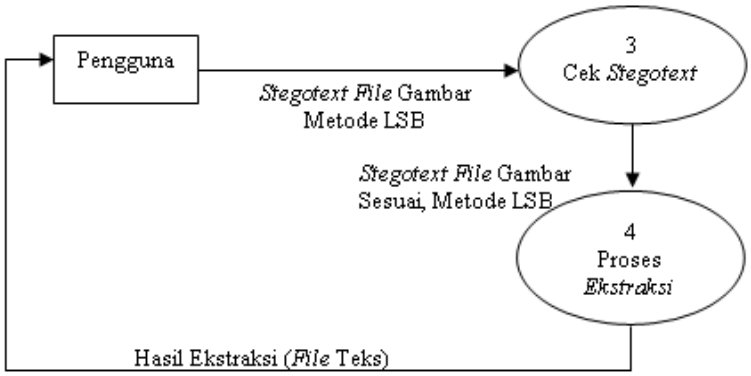
Gambar 2 DFD Level Satu Proses Embedding LSB

Proses ekstraksi dengan metode Spread Spectrum pada Gambar 3, file stegotext terlebih dahulu dicek dengan pembacaan data stegotext dari kemungkinan keberadaan file teks yang disembunyikan ke dalam stegotext. Setelah ditemukan tag tersebut maka dilakukan proses ekstraksi untuk dapat memisahkan antara file yang disisipkan dengan file gambar.



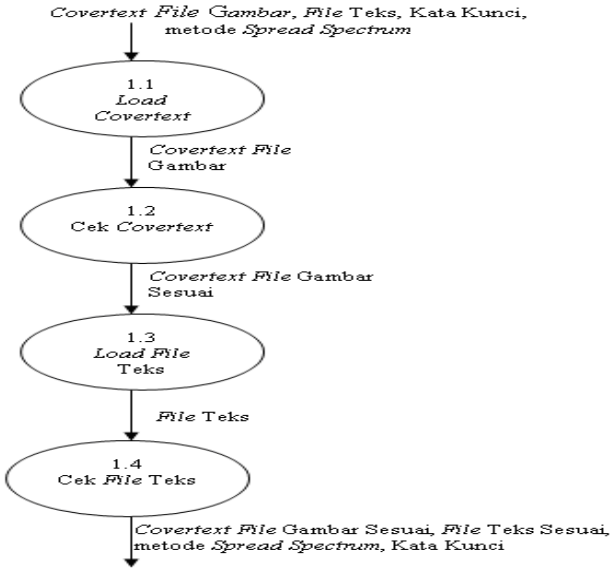
Gambar 3 DFD Level Satu Proses Ekstraksi Metode Spread Spectrum

Proses ekstraksi dengan metode LSB pada Gambar 4, file stegotext terlebih dahulu dicek dengan pembacaan data stegotext dari kemungkinan keberadaan file teks yang disembunyikan ke dalam stegotext. Setelah ditemukan tag tersebut maka dilakukan proses ekstraksi untuk dapat memisahkan antara file yang disisipkan dengan file gambar.

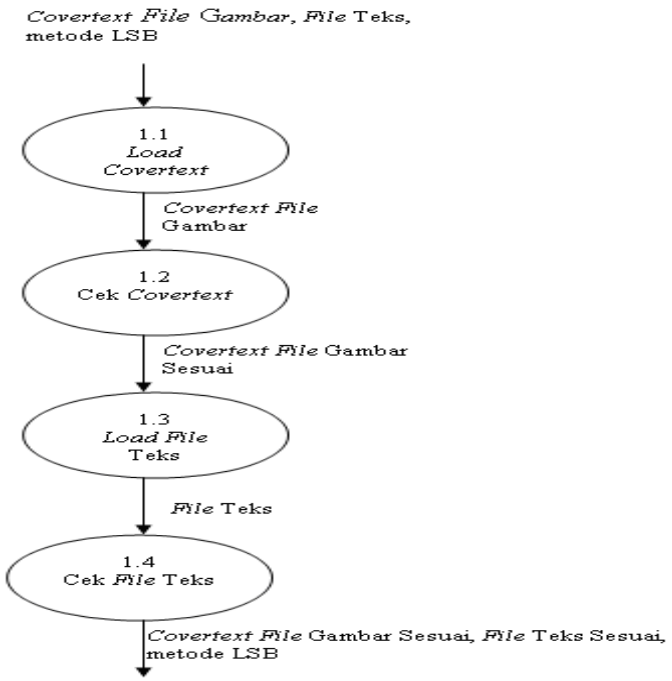


Gambar 4 DFD Level Satu Proses Ekstraksi LSB

Pada DFD Level Dua proses cek data Gambar 5 dan Gambar 6 dilakukan pembacaan terhadap covertext yaitu file gambar dan file teks yang akan disisipkan, apakah ukuran file teks yang akan disisipkan sesuai (tidak melebihi) ukuran file gambar. Jika sesuai, baru proses embedding dapat dilakukan.



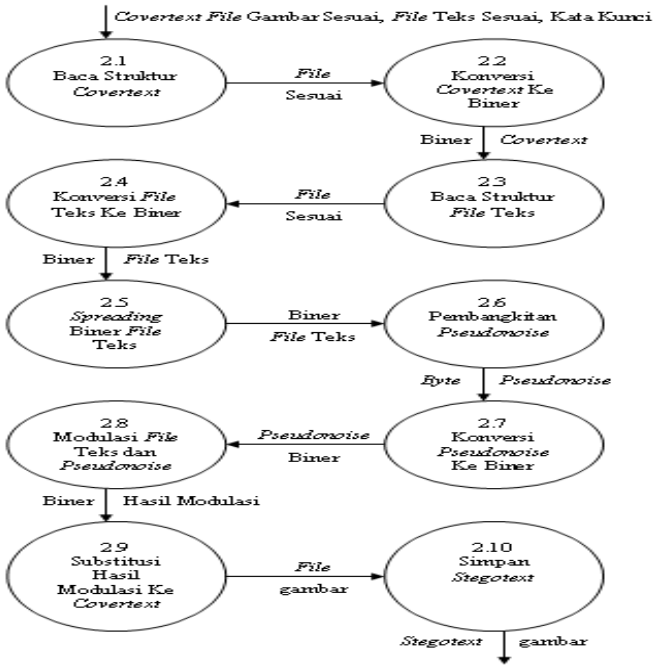
Gambar 5 DFD Level Dua Proses Cek Data Metode Spread Spectrum



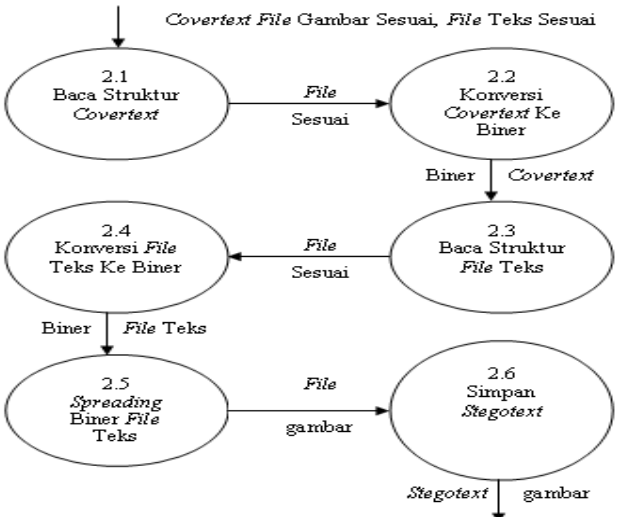
Gambar 6 DFD Level Dua Proses Cek Data Metode Least Significant Bit (LSB)

Pada DFD Level Dua Proses Embedding Gambar 7 dan Gambar 8 hal pertama yang dilakukan adalah proses pembacaan struktur covertext gambar untuk memisahkan antara header gambar dengan data gambar yang sesungguhnya. Kemudian setelah itu data gambar diubah ke dalam bentuk biner. Hal yang sama juga dilakukan terhadap file teks yang akan

disisipkan, namun bedanya adalah biner file teks akan di-spreading dengan parameter faktor pengali yang sudah ditentukan. Langkah berikutnya adalah pembangkitan bilangan acak untuk menciptakan pseudonoise. Pseudonoise ini juga akan diubah ke dalam bentuk biner. Kemudian biner file teks hasil spreading akan dimodulasi dengan biner noise menjadi biner file teks termodulasi. Setelah itu barulah proses substitusi biner file teks hasil modulasi ke dalam covertext dilakukan. Jika proses substitusi selesai dilakukan, stegotext hasil substitusi biner file teks hasil modulasi ke covertext akan disimpan ke dalam bentuk file gambar baru.

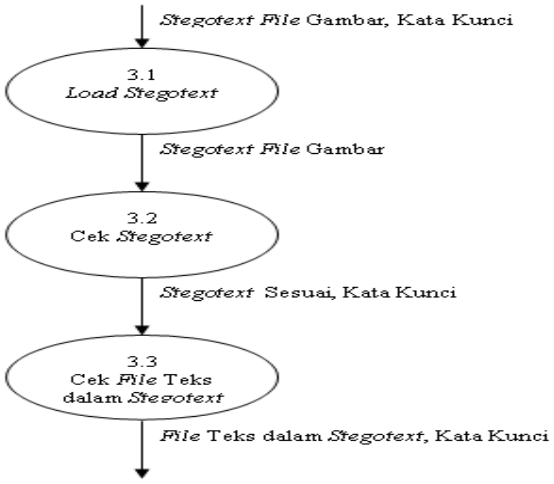


Gambar 7 DFD Level Dua Proses Embedding Spread Spectrum

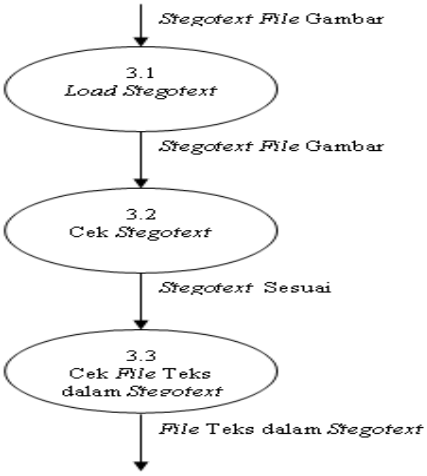


Gambar 8 DFD Level Dua Proses Embedding LSB

Pada DFD Level Dua proses cek stegotext Gambar 9 dan Gambar 10 dilakukan pembacaan terhadap stegotext yang berupa file gambar apakah terdapat tag yang mendeteksi keberadaan file yang disembunyikan atau tidak. Jika ada maka akan dilakukan proses selanjutnya yaitu ekstraksi.

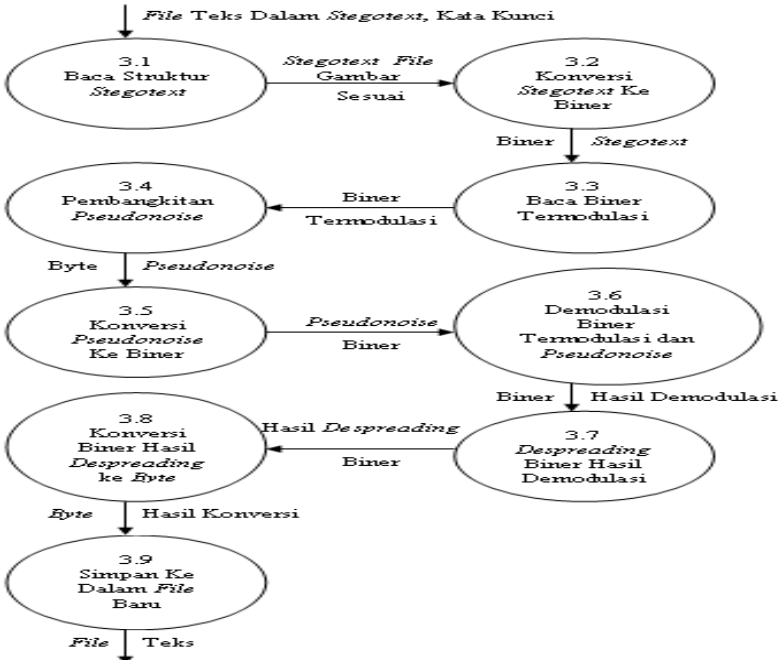


Gambar 9 DFD Level Dua Proses Cek Stegotext Spread Spectrum

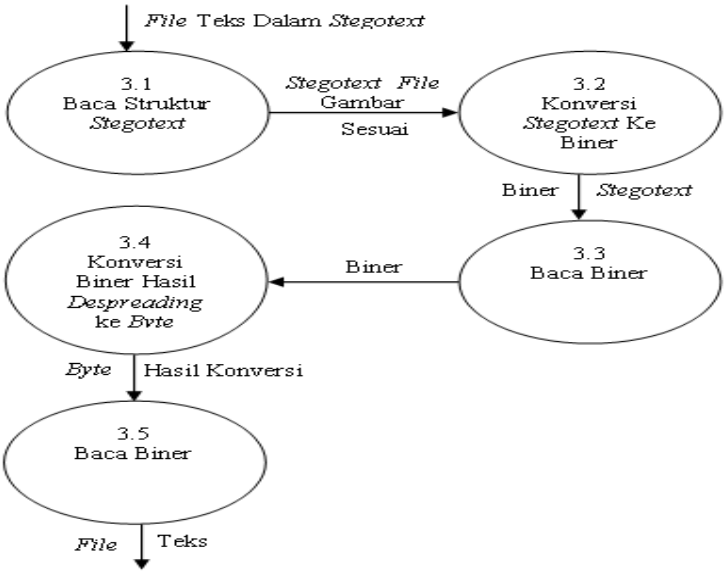


Gambar 10 DFD Level Dua Proses Cek Stegotext LSB

Pada DFD Level Dua Proses Ekstraksi Gambar 11 dan Gambar 12 hal pertama yang dilakukan adalah proses pembacaan struktur covertext gambar untuk mendeteksi apakah ada kemungkinan pesan yang disembunyikan. Jika ditemukan tag, maka setelah itu data gambar diubah ke dalam bentuk biner. Langkah berikutnya adalah pengambilan biner-biner pada biner gambar untuk mendapatkan bilangan-bilangan termodulasi. Pembangkitan bilangan acak dilakukan untuk menciptakan pseudonoise seperti pada proses embedding. Pseudonoise ini kemudian diubah ke dalam bentuk biner. Lalu biner-biner termodulasi akan di-demodulasi dengan biner-biner pseudonoise agar menghasilkan biner-biner hasil spreading. Setelah itu barulah proses despreading dilakukan. Proses despreading akan menyusutkan biner-biner hasil spreading menjadi biner-biner file tersisipkan yang sesungguhnya. Bila proses despreading selesai dilakukan, hasil despreading akan diubah ke dalam bentuk byte dan disimpan sebagai file teks yang baru.

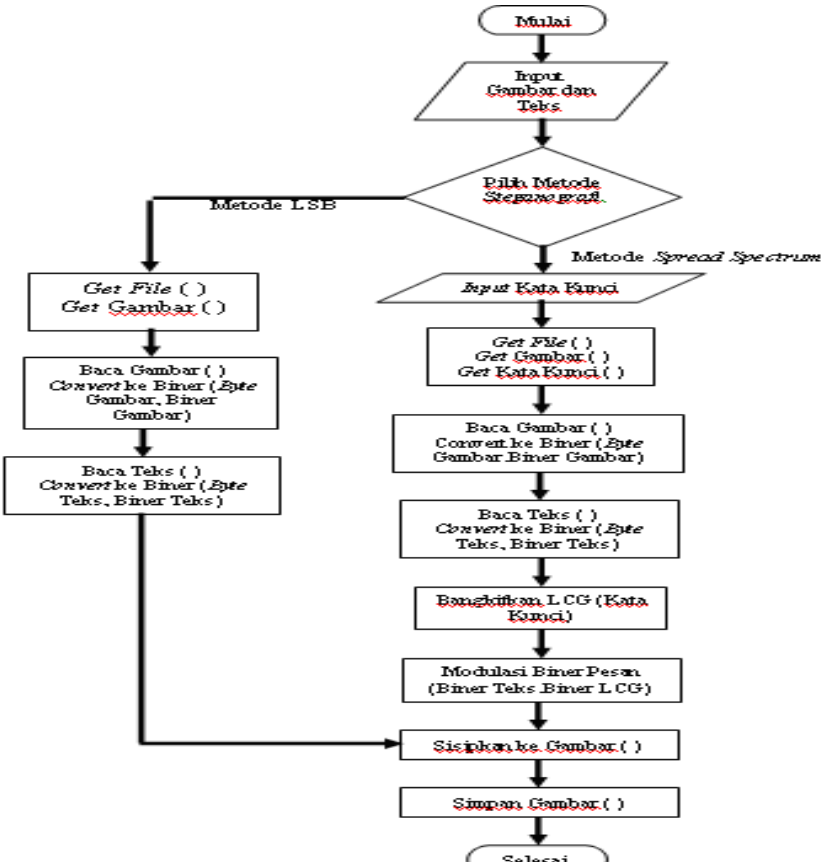


Gambar 11 DFD Level Dua Proses Ekstraksi Spread SPectrum



Gambar 12 DFD Level Dua Proses Ekstraksi LSB

Proses penyisipan pesan dirancang dengan menggunakan metode spread spectrum dan LSB. Adapun bentuk rancangan Flowchart-nya dapat dilihat pada Gambar 13

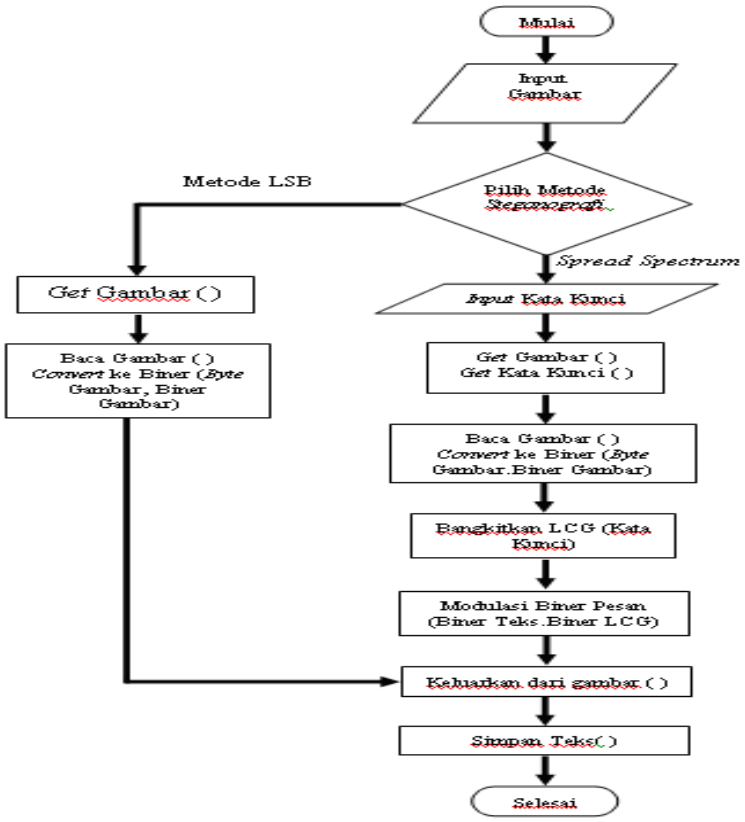


Gambar 13 Flowchart Embedding

Dari flowchart Gambar 13, dapat dijelaskan mengenai alur program yaitu, Pertama setelah proses dimulai sebagai inputan diperlukan file teks dan file gambar; Kedua, kemudian dilakukan pemilihan metode steganografi Spread Spectrum atau Least Significant Bit (LSB); Ketiga, apabila memilih menggunakan metode Spread Spectrum maka sebagai inputan perlu ditambahkan sebuah nilai kunci. Sebelum dilakukan proses embedding, akan didapat panjang kunci, tinggi, lebar gambar dan jumlah piksel gambar serta kapasitas file atau jumlah byte teks. Proses pertama adalah pembacaan secara urut piksel pada gambar yang dibentuk menjadi sebuah array byte dan diubah ke array biner. Proses selanjutnya adalah pembacaan nilai tiap byte dalam teks yang dibentuk menjadi sebuah array byte lalu diubah ke array biner dan disebar (spreading). Kemudian setelah itu dilakukan pembangkitan bilangan random dengan bibit awal pembangkitan berupa masukan dari kata kunci pengguna dan diubah ke array biner. Langkah berikutnya adalah modulasi antara array biner teks dengan array biner bilangan random; Keempat, apabila memilih menggunakan metode LSB maka sebelum dilakukan proses embedding, akan didapat tinggi, lebar dan jumlah piksel gambar serta kapasitas file atau jumlah

byte teks. Proses pertama adalah pembacaan secara urut piksel pada gambar yang dibentuk menjadi sebuah array byte dan diubah ke array biner. Proses selanjutnya adalah pembacaan nilai tiap byte dalam teks yang dibentuk menjadi sebuah array byte lalu diubah ke array biner dan disebar (spreading); Kelima, setelah itu file biner hasil proses diubah ke bentuk byte dan dibentuk menjadi sebuah file gambar baru untuk kemudian disimpan.

Untuk mengembalikan pesan teks yang telah disisipkan, struktur flowchart proses ekstraksi yang digunakan sehingga menghasilkan pesan teks dari gambar stegotext yang diterima adalah seperti pada Gambar 14



Gambar 14 Flowchart Ekstraksi

Dari flowchart Gambar 14, dapat dijelaskan mengenai alur program yaitu, pertama setelah proses dimulai kemudian sebagai masukan diperlukan sebuah gambar yang telah disisipi teks (stegotext); Kedua, kemudian pilih metode yang akan digunakan. Apabila menggunakan metode spread spectrum maka masukkan inputan kata kunci. Sebelum dilakukan proses ekstraksi, akan didapat panjang kunci, tinggi dan lebar gambar serta jumlah piksel gambar. Proses pertama adalah pembacaan secara mendatar dari gambar yang akan diterjemahkan

hingga terbentuk sebuah untaian byte dan dimasukkan ke dalam array. Kemudian setelah itu dilakukan pembangkitan bilangan random dengan bibit awal pembangkitan berupa masukan dari kata kunci pengguna dan diubah ke array biner. Langkah berikutnya adalah modulasi antara array biner gambar dengan array biner bilangan random; Ketiga, proses pembacaan secara mendatar dari gambar yang akan diterjemahkan hingga terbentuk sebuah untaian byte dan dimasukkan ke dalam array. Kemudian dilakukan proses ekstraksi, akan didapat panjang kunci, tinggi dan lebar gambar serta jumlah piksel gambar; Keempat, proses selanjutnya adalah pengambilan bit yang telah diproses yang dibentuk menjadi sebuah array biner yang baru lalu di-despreading; Kelima, yang terakhir adalah mengubah array biner hasil despreading ke dalam bentuk string dan proses ekstraksi berakhir.

3. Implementasi dan Pengujian

Jalan aplikasi adalah gambaran mengenai perhitungan yang terjadi di dalam program sehingga program dapat berjalan. Pada proses *embedding* dapat digambarkan sebagai berikut; Pertama sebagai masukkan dipilih gambar Smile.bmp; Kedua,masukkan pesan yang akan disisipkan, misalnya test.txt; Kelima, pilih metode *spread spectrum* sebagai contoh dan masukkan kata kunci “sonny”, kemudian pilih *embedding*; Keenam, pada proses *embedding* ini pertama-tama aplikasi akan membaca pesan yang dimasukkan dan mengecek ukuran pesan yang dimasukkan apakah lebih kecil dari pada ukuran gambar yaitu dengan memasukkan ke dalam rumus:

“Panjang pesan = ((ukuran pesan) + 28) * 4* 8”

Angka 28 adalah untuk *tag* pemberian tanda pada gambar yang sudah disisipkan, angka 4 adalah besar faktor pengali yang berguna untuk penyebaran bit serta angka 8 adalah bit gambar. Setelah mengecek ukuran *file* selesai kemudian dilakukan pengecekan ukuran gambar, metode steganografi yang digunakan dan kata kunci, jika semua syarat sudah terpenuhi akan dilanjutkan kedalam proses penyisipan; Ketujuh, sebelum penyisipan dilakuan aplikasi akan membaca gambar dan mengambil *header* dari gambar Smile.bmp, kemudian *body* dari gambar ini nanti yang akan disisipi pesan. Sebelum penyebaran proses yang dilakukan adalah mengubah pesan kebentuk biner. Pesan test.txt berisi tulisan “test” dan hasil dari pengkonversian di temukan biner dari pesan “test” adalah “01110100 01100101 01110011 01110100”. Kemudian biner pesan disebar dengan besaran skalar pengalinya empat, akan menghasilkan segmen baru yaitu **:**

000011111111111110000111100000000
000011111111000000000111100001111
000011111111111110000000011111111
000011111111111110000111100000000

Langkah selanjutnya adalah pembangkitan *pseudonoise* dengan bibit pembangkitan yang ditentukan berdasarkan kunci masukan yaitu ”sonny”.

s = 01110011
o = 01101111
00011100
n = 01101110
01110010
n = 01101110
00011100
y = 01111001
01100101 ⟶ 101 (decimal)

Setelah mendapatkan nilai dari kata kunci (101) kemudian nilai tersebut digunakan sebagai bibit awal pembangkitan bilangan acak. Perhitungan pembangkitan bilangan acak sesuai rumus pembangkitan bilangan acak LCG adalah seperti berikut :

$$X_{n+1} = (aX_n + c) \bmod$$

a = 17
c = 7
m = 84
Xn = Bilangan bulat ke-n

Perhitungannya adalah sebagai berikut:

X_1 = (17 * 101 + 7) mod 84 hasilnya X_1 = 44

X_2 = (17 * 44 + 7) mod 84 hasilnya X_2 = 83

X_3 = (17 * 83 + 7) mod 84 hasilnya X_3 = 74

X_4 = (17 * 74 + 7) mod 84 hasilnya X_4 = 5

X_5 = (17 * 5 + 7) mod 84 hasilnya X_5 = 8

Demikian seterusnya untuk X_6, X_7, X_8.....X_n

Sebagi contoh dilakukan lima kali penyebaran dan hasilnya adalah ”44 83 74 5 8” jika diubah dalam bentuk biner menjadi ”00101100 01010011 01001010 00000101 00001000”

Untuk mendapatkan hasil modulasi, segmen pesan akan dimodulasi dengan *pseudonoise signal* menggunakan fungsi XOR (*Exclusive OR*).

Segmen pesan :

00001111111111110000111100000000
00001111111100000000111100001111
00001111111111110000000011111111
00001111111111110000111100000000

Pseudonoise signal :

00101100010100110100101000000101000010000

Maka hasil proses modulasi antara segmen pesan dengan *pseudonoise signal* menggunakan fungsi XOR adalah :

00100011101011000100010100000101
00000111111100000000111100001111
00001111111111110000000011111111

00001111111111111000011110000000

Hasil dari proses modulasi inilah yang akan disisipkan ke bit-bit gambar. Sebagai contoh, misalkan mengambil sepuluh *pixel* pertama dari gambar Smile.bmp dan mengambil tiga puluh bit pertama dari modulasi antara segmen pesan dan *pseudonoise signal*.

Red = 180 186 185 182 181 183 186 184 184 187
Green = 166 172 174 171 170 173 176 174 176 179
Blue = 163 169 172 169 168 171 175 173 174 177

Kemudian diubah menjadi biner dan disisipkan hasil proses modulasi antara segmen pesan dengan *pseudonoise signal* menjadi sebagai berikut:

Red	**Green**	**Blue**
1 0 1 1 0 1 0 **0**	1 0 1 0 0 1 1 **0**	1 0 1 0 0 0 1 **1**
1 0 1 1 0 1 0 **0**	1 0 1 0 1 1 0 **0**	1 0 1 0 1 0 0 **0**
1 0 1 1 1 0 0 **1**	1 0 1 0 1 1 1 **1**	1 0 1 0 1 1 0 **1**
1 0 1 1 0 1 1 **0**	1 0 1 0 1 0 1 **1**	1 0 1 0 1 0 0 **0**
1 0 1 1 0 1 0 **1**	1 0 1 0 1 0 1 **1**	1 0 1 0 1 0 0 **0**
1 0 1 1 0 1 1 **0**	1 0 1 0 1 1 0 **0**	1 0 1 0 1 0 1 **1**
1 0 1 1 1 0 1 **0**	1 0 1 1 0 0 0 **0**	1 0 1 0 1 1 1 **0**
1 0 1 1 1 0 0 **1**	1 0 1 0 1 1 1 **0**	1 0 1 0 1 1 0 **1**
1 0 1 1 1 0 0 **0**	1 0 1 1 0 0 0 **0**	1 0 1 0 1 1 1 **0**
1 0 1 1 1 0 1 **0**	1 0 1 1 0 0 1 **0**	1 0 1 1 0 0 0 **1**

Langkah tersebut berlanjut sampai modulasi antara segmen pesan dan *pseudonoise signal* disisipkan semua. Proses terakhir setelah proses penyisipan adalah mengembalikan *header* gambar supaya gambar tidak mengalami kerusakan; Kedelapan, simpan gambar sebagai gambar baru dengan nama “Smile SS.bmp dan proses selesai.

Pada proses ekstraksi prosesnya adalah kebalikan dari proses *embedding*, untuk lebih jelasnya prosesnya sebagai berikut; Pertama, sebagai masukkan dipilih gambar Smile SS.bmp; Kedua, pilih metode *spread spectrum* dan masukkan kata kunci “sonny” seperti pada waktu proses *embedding*, kemudian pilih proses “Ekstraksi”; Ketiga, langkah pertama yang dilakukan saat proses ekstraksi adalah membaca gambar apakah di dalam gambar Smile SS.bmp terdapat *tag* untuk mendeteksi apakah gambar tersebut sudah pernah disisipi gambar atau belum. Apabila belum kemudian aplikasi akan mengambil *header* gambar terlebih dahulu, selanjutnya pada *body* gambar dilakukan proses penyaringan agar mendapatkan bit-bit hasil modulasi. Hasil dari proses penyaringan yang dilakukan akan mendapatkan bit-bit sebagai berikut :

00100011101011000100010100000101
00000111111100000000111100001111
00001111111111111000000011111111

000011111111111110000111100000000

setelah semua bit-bit hasil modulasi diperoleh, kemudian dilakukan proses demodulasi dengan *pseudonoise signal* dari kata kunci yang sama pada proses modulasi agar memperoleh bit-bit yang berkorelasi. Hasil penyaringan :

001000111010110001000101000001 01
000001111111000000000111100001111
000011111111111110000000011111111
000011111111111110000111100000000

Pseudonoise signal :

00101100010100110100101000000101000010 00

Hasil demodulasi :

000011111111111110000111100000000
000011111111100000000111100001111
000011111111111110000000011111111
000011111111111110000111100000000

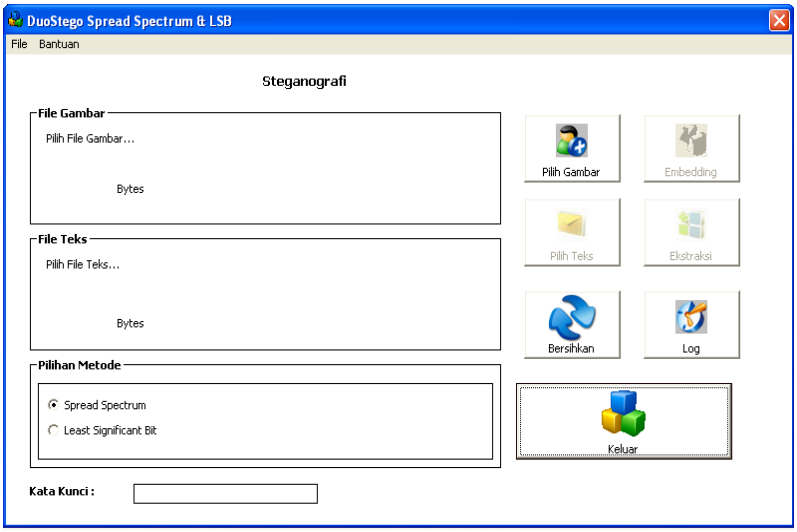
Proses berikutnya adalah *de-spreading* yaitu dengan membagi empat hasil demodulasi, yang berguna untuk menyusutkan hasil demodulasi menjadi isi pesan yang sebenarnya. Proses penyusutan (*de-spreading*) segmen tersebut menjadi :

01110100 01100101 01110011 01110100

Hasil akhir “01110100 01100101 01110011 01110100” merupakan segmen pesan yang sama ketika disembunyikan pada proses *embedding*. Hasil tersebut kemudian diubah kebentuk karakter akan menjadi menjadi ”test”; Keempat, kemudian simpan hasil ekstraksi dengan nama baru, misalnya “Test SS.txt”

4. Implementasi Program

Form utama adalah *form* yang tampil pertama saat program ini dijalankan, *form* ini menyediakan pilihan bagi *user* untuk melakukan proses penyisipan (*embedding*) *file* teks ke dalam gambar ataupun melakukan *ekstraksi file* teks dari gambar melalui dua metode *spread spectrum* dan *Least significant bit* (LSB) serta LOG untuk menyimpan proses yang dijalankan dengan aplikasi ini. Di dalam *form* ini terdapat komponen *picturebox* untuk menempatkan identitas gambar, sebuah *textbox* untuk menempatkan identitas *file* teks yang akan disisipkan, sebuah *metodebox* untuk memilih metode yang akan digunakan, serta sebuah *textbox* untuk menuliskan kata kunci apabila menggunakan metode *spread spectrum*. Selain itu pada *form* ini juga memiliki tujuh buah tombol yaitu PILIH GAMBAR untuk membuka *file* gambar, PILIH TEKS untuk membuka *file* teks, *EMBEDDING* untuk melakukan proses penyisipan, *EKSTRAKSI* untuk melakukan proses *ekstraksi*, BERSIHKAN untuk membersihkan *form*, LOG untuk melihat proses dari program ketika di jalankan serta tombol KELUAR untuk keluar dari aplikasi. Tampilan antarmuka *form* utama ditunjukkan pada Gambar 15.



Gambar 15 *Form* Utama

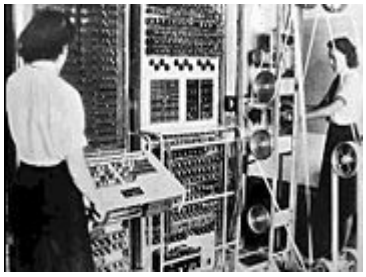
Selain itu *form* utama juga mempunyai dua buah menu yaitu menu *FILE* dan BANTUAN. Menu *FILE* digunakan untuk melakukan proses memilih gambar, memilih teks, membersihkan *form* dan menutup tampilan antar muka *form* utama, kemudian menu BANTUAN digunakan untuk menampilkan jendela bantuan yang berisi cara melakukan *embedding* dan *ekstraksi* serta menampilkan jendela pembuat program.

Perbandingan Proses *Embedding* dan *Ekstraksi*

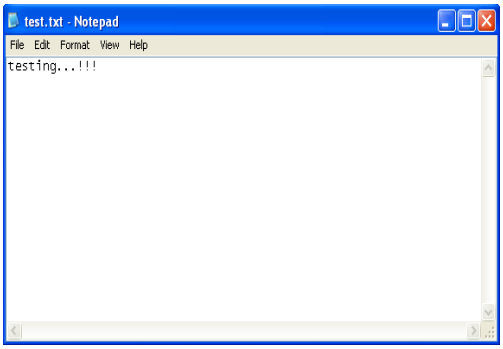
Pada perbandingan ini, dibutuhkan dua buah *file* gambar yaitu *file* gambar BMP gambar JPEG dan sebuah *file* teks. Pada proses *embedding*, gambar *covertext* dan *file* teks digunakan sebagai *input,* dan menghasilkan *output* berupa gambar *stegotext.* Sedangkan pada proses *ekstraksi*, gambar *stegotext* sebagai *input* dan sebagai *output* adalah *file* teks dengan menggunakan metode *spread spectrum* dan LSB.

Dari Gambar 16 dan Gambar 17 nantinya akan dilakukan pengujian terhadap resolusi, waktu dan besar *file* gambar sebelum dan sesudah dilakukan proses *embedding* dan *ekstraksi*.

Gambar 16 Smile.bmp



Gambar 17 Colosus.jpg



Gambar 18 *File* Teks

Hasil Pengujian

Tabel 1 Pengujian Gambar Smile.bmp dan Colosus.jpg

	Gambar BMP	Gambar JPEG
Resolusi Awal Gambar	104 x 121	180 x 119
Ukuran Awal *File* Gambar	37 KB	9.07 KB
Besar *File* Teks Awal (*bytes*)	13	13
Resolusi Setelah *Embedding*	104 x 121	180 x 119
Ukuran *File* Gambar *Spread Spectrum*	37 KB	63.3 KB
Ukuran *File* Gambar LSB	37 KB	63.3 KB
Besar *File* Teks Ekstraksi (*bytes*)	13	13
Waktu *Embedding Spread Spectrum*	40.83 detik	6.42 detik
Waktu *Embedding* LSB	16.14 detik	6.32 detik
Waktu Ekstraksi *Spread Spectrum*	40.63 detik	5.61 detik
Waktu Ekstraksi LSB	14.30 detik	5.07 detik

Tabel 2 Pengujian SNR Gambar Smile.bmp

Ukuran File teks (byte)	SNR
1	99.63 %
10	98.87 %

50	97.12 %
100	95.76 %
150	94.03 %

Tabel 3 Pengujian SNR Gambar Colosus.jpg

Ukuran File teks (byte)	SNR
1	99.32 %
10	98.45 %
50	96.89%
100	94.43 %
150	91.11 %

Hasil dari pengujian dapat diketahui bahwa Aplikasi "Perbandingan Steganografi Metode *Spread Spectrum* dan *Least Significant Bit* (LSB) Antara Waktu Proses dan Ukuran *File* Gambar" berjalan dengan baik karena gambar hasil dari *embedding* tidak mengalami kerusakan dan pesan *file* teks hasil ekstraksi juga tidak mengalami perubahan isi maupun ukuran, resolusi dari gambar *stegotext* juga tidak mengalami perubahan.

5. Kesimpulan dan Saran

Pada percobaan yang sudah dilakukan dapat di simpulkan bahwa, pertama aplikasi "Perbandingan Steganografi Metode *Spread Spectrum* dan *Least Significant Bit* (LSB) Antara Waktu Proses dan Ukuran *File* Gambar" berjalan dengan baik karena gambar hasil dari *embedding* tidak mengalami kerusakan dan pesan *file* teks hasil ekstraksi juga tidak mengalami perubahan isi maupun ukuran, resolusi dari gambar *stegotext* juga tidak mengalami perubahan; Kedua, warna, ukuran dan kualitas gambar asli tidak rusak ataupun berubah setelah melalui proses *embedding* karena bit gambar yang disisipkan menggunakan bit paling kecil; Ketiga, metode LSB memiliki proses *embedding dan ekstraksi* yang lebih cepat dari metode *Spread Spectrum* karena proses metode *Spread Spectrum* harus melalui peng-XOR-an antara pesan dan kata kunci, sedangkan LSB langsung menyisipkan pesan ke dalam gambar; Keempat, proses *embedding* dan ekstraksi gambar JPEG lebih cepat dari BMP karena *file* JPEG mempunyai ukuran dan resolusi *file* yang lebih kecil; Kelima, dari gambar tabel pengujian SNR menunjukkan bahwa adanya perubahan bit-bit karena adanya penyisipan pesan kedalam gambar sehingga mengalami perubahan prosentase, tetapi perubahan pada bit gambar tidak membuat gambar asli mengalami perubahan yang significant karena yang terubah hanya satu bit terakhir dari gambar atau maksimal perubahan bit adalah dua belas setengah persen.

Daftar Pustaka

[1] Krisnandi, Dikdik, 2004, "*Metoda Steganografi Berbasis Least Significant Bit dengan Penyisipan Variable-Size dan Penambahan Redundant Gaussian Noise*", Program Magister Teknik Elektro, Institut Teknologi Bandung.

[2] Vembrina, Yus Gias., 2006, "*Spread Spectrum Steganografi"*, Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

[3] Hakim, Muhammad, 2007, "*Studi dan Implementasi Steganografi Metode LSB dengan Preprocessing Kompresi data dan Ekspansi Wadah",* Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika, Institut Teknologi Bandung.

[4] Putranto, Adam, 2009, "*Steganografi Melalui Media Gambar dengan Metode Spread Spectrum"*, Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Kristen Satya Wacana.