Universitas Maritim Raja Ali Haji
SOJ
Student Online Journal

PENYISIPAN PESAN PADA GAMBAR MENGGUNAKAN ALGORITMA BLOWFISH DAN ALGORITMA F5

Daniel Tua Parlaungan Tampubolon, Ferdi Chahyadi, Muhamad Radzi Rathomi
Daniel.tampubolon2415@gmail.com
Jurusan Teknik Informatika, Fakultas Teknik, Universitas Maritim Raja Ali Haji

Abstract

The issue of security and confidentiality of a message or data is very important, often a message or data is hacked by an irresponsible party, therefore a technique or method is needed to secure the message or data. There are 2 (two) ways of securing data using cryptographic and steganographic techniques. In this research, making a system with the implementation of cryptographic and steganographic techniques, the method used for cryptography is the blowfish algorithm while the steganography method uses the F5 algorithm. The data used is messages in written format, with storage media in the form of images. The maximum character length of the message is 550 words, while for storage media the maximum size is 640 x 640 pixels. From 8 data that have been tested, it produces an average value above 30dB which shows that the level of construction in the resulting stego image has a similarity that is not much different from the original image. The similarity results from the PSNR calculation resulting from the F5 algorithm process shows the average similarity value is good with an average value of 36.65dB (decibels), the similarity between the original image and the stego image can be seen from the RGB histogram results, so this algorithm is suitable for use in inserting messages into images.

Keywords : *Cryptography, Steganography, PSNR, Histogram*

I. Pendahuluan

Permasalahan keamanan dan kerahasiaan sebuah pesan maupun data merupakan hal sangat penting dalam' suatu organisasi yang komersial (perusahaan), perguruan tinggi, lembaga pemerintahan, bahkan dalam hal individual (pribadi). Itu dikarenakan sebuah pesan maupun data yang penting kadang tidak sampai ketangan si penerima(di retas), bahkan sebuah pesan maup un data bisa di ubah informasinya dengan orang – orang yang tidak bertanggung jawab, sehingga si penerima tidak mendapat pesan maupun data tersebut dengan akurat. Hal ini seringkali ditakutkan oleh pihak – pihak yang ingin bertukar data, mereka takut apakah data yang dikirimkan sampai kepada si penerima atau tidak, oleh sebab itu agar sebuah data dapat sampai kepada sipenerima, diperlukan sebuah teknik atau metode untuk merahasiakan data yang dikirim tersebut.

Pengamankan pesan rahasia dapat dilakukan dengan Teknik *cryptography* dan *steganography*. *Cryptography* merupakan ilmu yang mempelajari teknik – teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasian data. Sedangkan *Steganography* merupakan teknik untuk menyembunyikan pesan atau data kedalam sebuah media dengan suatu cara sehingga tidak ada seorang pun yang mengetahui atau menyadari sebenarnya ada suatu pesan atau data rahasia didalam media tersebut.

Di dalam membuat sistem untuk *Image Steganography*, tidak sedikit saat ini yang telah melakukan penelitian dengan berbagai metode yang di terapkan. Adapun metode yang telah digunakan seperti *Least Significant bit insertion (LSB), Algorithms and Transformation, Redundant*

Pattern Encoding, Spread Spectrum Method. Maka penulis membuat penelitian ini sebagai bentuk implementasi salah satu metode *Algorithms and Transformation* yaitu *Algoritma F5*.

Algoritma *F5* di kembangkan oleh *Andreas Westfeld*, yang sebelumnya sudah ada Algoritma *F3* dan Algoritma *F4*. Algoritma *F5* mengimplementasikan pengkodean matriks untuk meningkatkan efisiensi penyematan. Dengan demikian mengurangi jumlah perubahan yang diperlukan. F5 menggunakan permutative straddling untuk menyebarkan perubahan secara merata pada seluruh steganogram.

Algoritma *Blowfish* yang merupakan suatu teknik *cryptography*, untuk mengenkripsikan dan mendeskripsikan isi pesan yang akan disisipkan ke dalam suatu media gambar, agar pesan rahasia yang akan disisipkan kedalam gambar terjaga keamanannya. Mengingat latar belakang di atas, maka penulis membuat judul penelitian ini dengan judul, "Penyisipan Pesan Pada Gambar Menggunakan Algoritma *Blowfish* Dan Algoritma *F5*".

II. Metode Penelitian

2.1 Kriptografi

Menurut Sitinjak, dkk. (2010) Kriptografi secara umum adalah ilmu dan seni untuk menjaga kerahasian berita. Selain pengertian tersebut terdapat pula pengertian ilmu yang mempelajari teknik – teknik matematika yang berhubungan dengan aspek keamanan informasi seperti kerahasian data. Kata Kriptografi berasal dari dua suku kata dalam bahasa Yunani yakni *kriptos* dan *graphia*. Kata *kriptos* memiliki arti *secret* (rahasia) sedangkan kata *graphia* berarti *writing* (tulisan). Jika di lihat dari arti dua suku kata tersebut, kriptografi dapat diartikan sebagai sebuah tulisan rahasia yang maknanya tidak dapat di ketahui oleh orang lain.

2.2 Steganografi

Menurut Amal & Ryano (2015) *Steganography* adalah suatu teknik untuk menyembunyikan informasi yang bersifat pribadi dengan media tertentu sehingga hasilnya akan tampak seperti informasi normal lainnya. Kata Steganography berasal dari bahasa Yunani yaitu *steganos* yang artinya "tersembunyi atau terselubung" dan *graphein* yang artinya "menulis" sehingga dapat di artikan sebagai "menulis (tulisan) tersembunyi". Proses penyisipan pada *steganography* di kenal sebagai proses *encoding*, sedangkan proses ektraksi pesan di kenal dengan proses *decoding*.

2.3 Algoritma Blowfish

Menurut Dimas & Herlina (2016) *Blowfish* merupakan enkripsi yang menggunakan algoritma simetris yang tergolong ke dalam algoritma cipher blok. *Blowfish* dirancang untuk memenuhi kriteria sebagai berikut :

1. Cepat, pada implementasi yang optimal *Blowfish* dapat mencapai kecepatan 26 *clock cycle/byte*.
2. Kompak, *Blowfish* dapat berjalan pada memori kurang dari 5 KB.
3. Sederhana, *Blowfish* hanya menggunakan operasi yang sederhana, yaitu penambahan (*addition*), XOR, dan penelusuran tabel (*table lookup*) pada operand 32 bit.
4. Keamanan yang variabel, panjang kunci *Blowfish* dapat bervariasi dan dapat mencapai 448 bit (56 *Byte*).

2.3.1 Key Expansion

Langkah pertama adalah Key Expansion berfungsi merubah kunci menjadi beberapa array subkunci (subkey) dengan total 4168 Byte (18x32-bit untuk P-array dan 4x256x32-bit untuk S-box sehingga totalnya 33344 bit atau 4168 Byte). Kunci disimpan dalam K-array (Wardoyo, dkk, 2016):

$$K1,K2,\ldots\ldots Kj \; 1 \leq j \leq 14 \qquad (1)$$

Kunci-kunci ini yang dibangkitkan (generate) dengan menggunakan subkunci yang harus dihitung terlebih dahulu sebelum enkripsi atau dekripsi data. Sub-sub kunci yang digunakan terdiri dari:

P-array yang terdiri dari 18 buah 32-bit subkunci,

$$\mathbf{P1,P2,\ldots\ldots,P18} \quad \textbf{(2)}$$

S-box yang terdiri dari 4 buah 32-bit, masing-masing memiliki 256 entri:

$$\begin{matrix}\mathbf{S1,0,S1,1,S1,2,S1,3,\ldots\ldots\ldots\ldots S1,255}\\ \mathbf{S2,0,S2,1,S2,2,S2,3,\ldots\ldots\ldots\ldots S2,255}\\ \mathbf{S3,0,S3,1,S3,2,S3,3,\ldots\ldots\ldots\ldots S3,255}\\ \mathbf{S4,0,S4,1,S4,2,S4,3,\ldots\ldots\ldots\ldots S4,255}\end{matrix} \quad \textbf{(3)}$$

Langkah-langkah perhitungan atau pembangkitan subkunci tersebut adalah sebagai berikut (Dimas & Herlina, 2016) :

1. Inisialisasi P-array yang pertama dan juga 4(empat) S-box, berurutan, dengan string yang telah pasti. String tersebut terdiri dari digit-digit heksadesimal dari phi, tidak termasuk angka (3)tiga di awal.
2. XOR-kan P1 dengan 32-bit awal kunci, XOR-kan P2 dengan 32-bit berikutnya dari kunci, dan seterusnya untuk semua bit kunci. Ulangi siklus seluruh bit kunci secara berurutan sampai seluruh P-array ter-XOR-kan dengan bit-bit kunci. Atau jika disimbolkan :

   $$\mathbf{P1 = P1 \oplus K1,P2 = P2 \oplus K2,P3 = P3 \oplus K3,...P14 = P14 \oplus K14,P15 = P15 \oplus K1,...P18 = P18 \oplus K4.} \quad \textbf{(4)}$$

   Keterangan : $\oplus$ adalah simbol untuk XOR.
3. Enkripsikan string yang seluruhnya nol dengan algoritma Blowfish, menggunakan subkunci yang telah dideskripsikan pada langkah 1 dan 2.
4. Gantikan P1 dan P2 dengan output dari langkah 3.
5. Enkripsikan keluaran langkah 3 menggunakan algoritma Blowfish dengan subkunci yang telah dimodifikasi.
6. Gantikan P3 dan P4 dengan output dari langkah 5..

2.3.1 Tahapan Enkripsi dan Dekripsi

Lanjutkan langkah-langkah 1-6, gantikan seluruh elemen P-*array* dan kemudian keempat S-*box* secara berurutan, dengan hasil keluaran algoritma *Blowfish* yang terus-menerus berubah. Total terdapat 521 iterasi untuk menghasilkan subkunci dan membutuhkan memori sebesar 4KB (Wardoyo, dkk, 2016). Kemudian lakukan langkah enkripsi data dengan langkah – langkah seperti berikut:

1. Bagi X menjadi dua bagian yang masing-masing terdiri dari 32-*bit*: XL, XR. Lakukan langkah berikut :

   $$\begin{matrix}\boldsymbol{For\ i\ =\ 1\ to\ 16}:\\ \boldsymbol{XL\ =\ XL\ \oplus\ Pi}\\ \boldsymbol{XR\ =\ F(XL)\ \oplus\ XR}\end{matrix} \quad \textbf{(5)}$$

2. Tukar XL dan XR. Setelah iterasi ke-16, tukar XL dan XR lagi untuk melakukan membatalkan pertukaran terakhir.
3. Lalu lakukan

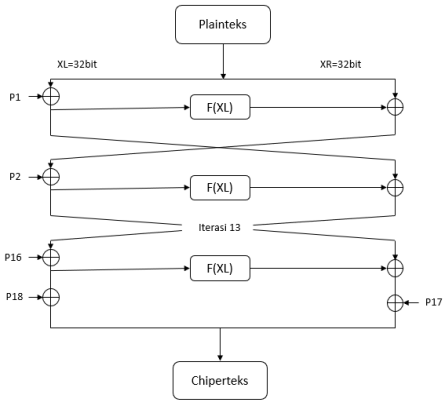
   $$\begin{matrix}\boldsymbol{XR\ =\ XR\ \oplus\ P17}\\ \boldsymbol{XL\ =\ XL\ \oplus\ P18}\end{matrix} \quad \textbf{(6)}$$

4. Terakhir, gabungkan kembali XL dan XR untuk mendapatkan cipherteks.

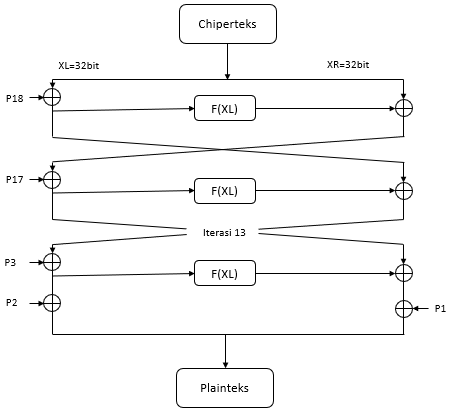
Dekripsi sama persis dengan enkripsi, kecuali bahwa P1, P2,…, P18 digunakan pada urutan yang berbalik (*reverse*). Algoritmanya dapat dinyatakan sebagai berikut :

$$\begin{matrix}\boldsymbol{for\ i\ =\ 1\ to\ 16\ do}\\ \boldsymbol{XRi\ =\ XLi-1\ \oplus\ P19-i;}\\ \boldsymbol{XLi\ =\ F[XRi]\ \oplus\ XRi-1;}\\ \boldsymbol{XL17\ =\ XR16\ \oplus\ P1;}\\ \boldsymbol{XR17\ =\ XL16\ \oplus\ P2;}\end{matrix} \quad \textbf{(7)}$$

Proses enkripsi dan dekripsi dapat dilihat dari gambar di bawah:



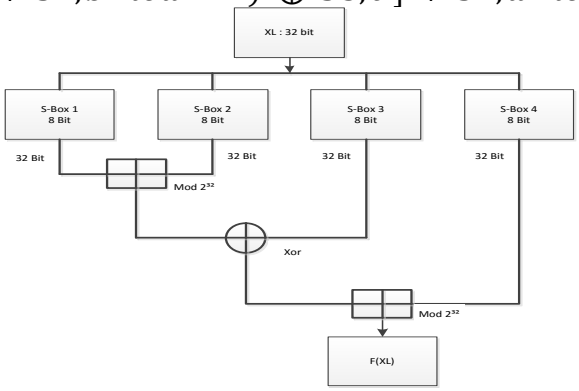
Gambar 1 Langkah Proses Enkripsi Sumber : Indriyono (2019)



Gambar 2 Langkah Proses Dekripsi Sumber : Indriyono (2019)

Fungsi *f(Xl)* pada rumus *blowfish* bisa di lihat dari rumus dibawah, dan proses dari fungsi f(Xl) dapat dilihat dari gambar 5 :

$$\boldsymbol{f(Xl)} = [(\boldsymbol{S1, a} + \boldsymbol{S2, b \bmod 2^{32}}) \oplus \boldsymbol{S3, c}] + \boldsymbol{S4, d \bmod 2^{32}}. \quad \textbf{(8)}$$



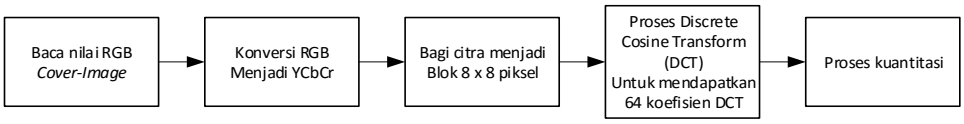
Gambar 3 Langkah Proses fungsi *f(Xl)* Sumber : Indriyono (2019)

2.4 Algoritma F5

F5 merupakan sebuah metode / algoritma untuk steganografi pada citra digital. Algoritma *F5* diajukan oleh Andreas Westfeld dari *Technische Universitat Dresden*, *Institute for System Architecture*, Jerman, ada juga metode Andreas sebelumnya, yaitu *F3* dan *F4*, dimana metode untuk menyisipkan pesan rahasianya menggunakan metode *LSB* (*Least Significant Bit*). Pada Algoritma *F5*, digunakan metode penyebaran *byte-byte* dari citra digital baik yang bernilai positif maupun negatif, dan bukan byte 0. (Adil & Fahrur, 2015)

2.4.1 Tahapan Pre-Processing Citra

Tahapan selanjutnya yaitu *pre-processing* citra sesuai dengan gambar 4 untuk mempersiapkan citra yang digunakan agar dapat di proses pada tahap embedding.



Gambar 4 Tahapan *Pre-Processing* Citra Sumber : Sianturi & Hutagaol (2019)

Nilai *RGB* di baca setelah pengguna memilih citra, kemudian nilai tersebut akan dikonversi menjadi *YCbCr* yang bertujuan untuk membuang komponen yang tidak penting pada citra sesuai dengan tingkat kepekaan mata manusia, sehingga menghemat ruang pada citra. Mata manusia lebih peka terhadap perubahan warna *luminance* (*Y*) daripada warna *chrominance* (*Cb, Cr*) sehingga yang di gunakan untuk masukkan citra adalah warna *Y*. Konversi warna *RGB* menjadi *YCbCr* menggunakan persamaan (9), (10), (11) (Zulfikar, 2018) :

$$\boldsymbol{Y = 0.229\,R + 0.587\,G + 0.114\,B} \quad \textbf{(9)}$$

$$\boldsymbol{Cb = -0.1687R - 0.3313\,G + 0.5\,B + 128} \quad \textbf{(10)}$$

$$\boldsymbol{Cr = 0.5\,R - 0.418\,G - 0.0813\,B + 128} \quad \textbf{(11)}$$

Tahap selanjutnya yaitu membagi citra ke dalam blok 8 x 8 yang berfungsi untuk mempermudah proses *DCT*. Selanjutnya untuk mendapat 64 koefisien *DCT* digunakan persamaan (12):

$$\boldsymbol{F(u,v) = \sqrt{\frac{2}{M}}\sqrt{\frac{2}{N}}C(u)C(v)\sum_{x=0}^{M-1}\sum_{y=0}^{N-1}f(x,y)cos\left[\frac{(2x+1)u\pi}{2M}\right]cos\left[\frac{(2y+1)v\pi}{2N}\right]} \quad \textbf{(12)}$$

Ket:

$F(u,v) = Nilai\ DCT\ pada\ indeks\ ke\ (i,j)$

$N = Ukuran\ baris\ Matriks$

$M = Ukuran\ kolom\ Matriks$

$f(x,y) = Nilai\ pixel\ pada\ indeks\ ke(x,y)$

$C(u), C(v) = 1\ jika\ u,v > 0$

$C(u), C(v) = \frac{1}{\sqrt{2}}\ jika\ u,v = 0$

Pada persamaan di atas, *F(u,v)* berbentuk matriks 2 dimensi 8 x 8, dimana:

a. u, x = 0, 1, 2, ..., n – 1 ; v, y = 0, 1, 2, ..., m – 1;

b. u, v merupakan koordinat frekuensi pada domain transformasi atau koefisien-koefisien DCT;

c. x, y adalah koordinat spasial dari domain asal;

d. $C(u), C(v) = \frac{1}{\sqrt{2}}\ untuk\ u = 0$

Tahapan ini untuk mengidentifikasi dan menghilangkan komponen pada citra berfrekuensi tinggi yang tidak dapat dideteksi oleh mata manusia namun tidak mengurangi kualitas citra. 64 koefisien DCT ini yang akan digunakan untuk menyisipkan bit-bit pesan rahasia yang dimasukkan oleh pengguna. Setelah proses DCT dilakukan, 64 koefisien DCT akan dibagi dengan koefisien tabel kuantisasi luminance dan chrominance sesuai pada tabel 1 seperti dibawah (Zulfikar, 2018):

Tabel 1. Kuantisasi Luminance dan Chrominance

6	1	10	16	24	40	51	61		17	18	24	47	99	99	99	99
2	12	14	19	26	58	60	55		18	21	26	99	99	99	99	99
14	13	16	24	40	57	69	56		24	26	99	99	99	99	99	99
14	17	22	29	51	87	80	62		47	99	99	99	99	99	99	99
18	22	37	56	68	109	103	77		99	99	99	99	99	99	99	99
24	35	55	64	81	104	113	92		99	99	99	99	99	99	99	99
49	64	78	87	103	121	120	101		99	99	99	99	99	99	99	99
72	92	98	98	112	100	103	90		99	99	99	99	99	99	99	99

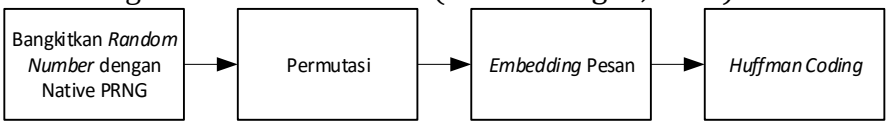
Proses pembagian 64 koefisien DCT dengan tabel 1 menggunakan rumus persamaan (13):

$$\boldsymbol{F_Q(u,v) = Round\left[\frac{F[u,v]}{Q[u,v]}\right]} \quad \textbf{(13)}$$

Koefisien tabel kuantisasi dinyatakan dengan Q[u,v], dan F[u.v] merupakan 64 koefisien DCT. Tujuan dari tahapan ini yaitu untuk membuang koefisien-koefisien hasil DCT yang tidak diperlukan dalam pembentukan citra baru (Abdelhamid, 2018).

2.4.2 Tahapan Embedding Pesan

Selanjutnya di lakukan proses embedding pesan sesuai dengan gambar 4. Pada tahap awal, akan mebangkitkan random number yang seed-nya di peroleh dari masukkan password pengguna atau akan menggunakan default seed jika pengguna tidak memasukkan password. Setelah random number di dapatkan, maka lakukan permutasi menggunakan random number dan koefisien DCT yang telah di peroleh sebelumnya. Setelah itu, tentukan nilai k dari kapasitas embedding pada data citra dan dari panjang data pesan yang akan di sisipkan. Nilai k yang di peroleh akan di gunakan untuk menentukan panjang dari code word dengan rumus $n = 2^k - 1$ (Hafidh & Agus, 2016).



Gambar 4 Tahapan *Embedding* Sumber : Sianturi & Hutagaol (2019)

Tahapan selanjutnya yaitu menyisipkan pesan rahasia dengan (1, n, k) matrix encoding dengan ketentuan (Eder & Perry, 2010):

1. Isi buffer dengan koefisien *non zero*.
2. Lakukan proses hashing pada *buffer* (menghasilkan nilai hash dengan k bit-places).
3. Tambahkan bit k berikutnya dari data pesan pada nilai hash (lakukan bit per bit dengan operator XOR) .
4. Jika hasil yang di dapat sama dengan 0, maka nilai buffer di biarkan tetap dan tidak di ubah. Tetapi, jika hasil yang di dapat sama dengan nilai rentang index buffer yaitu 1 .....n, maka nilai absolut dari elemen pada index tersebut harus di kurangi 1.
5. Lakukan pengecekan jika koefisien yang diubah tidak sama dengan 0. Jika sama, maka terjadi proses shrinkage. Jika peristiwa ini terjadi, maka tambahkan satu koefisien non zero pada buffer dan hilangkan koefisien 0 tadi.
6. Lalu ulangi langkah(i), (vi) jika tidak terjadi peristiwa shrinkage, maka isi buffer dengan koefisien DCT selanjutnya (dimulai dari index koefisien ditambah 1). Jika masih ada data pesan yang akan disisipkan, maka ulangi langkah (i).

Jika semua tahapan telah dilakukan, maka dilanjutkan dengan proses Huffman Coding. Tujuan dilakukannya kompresi ini yaitu untuk melakukan kompresi citra agar ukuran yang dihasilkan tidak terlalu besar. Hasil akhir dari tahapan ini yaitu stego-image yang telah terkompresi

2.4.3 Tahapan Decode Pesan

Selanjutnya adalah tahapan *decode* pesan sesuai dengan gambar 5.

Gambar 5 Tahapan Decode Pesan Sumber : Sianturi & Hutagaol (2019)

Proses dari *decode* pesan adalah kebalikan dari proses *encode*. Syarat melakukan decode pesan yaitu password yang dimasukkan saat *encode* pesan harus sama saat *decode* pesan. Urutan dari proses decode dapat dilihat pada gambar 2.9. Langkah pertama setelah pengguna memasukkan password yang benar yaitu melakukan dekompres pada *stego-image* menggunakan *Huffman Decoder*, selanjutnya dilakukan permutasi terhadap koefisien *DCT* dan *random number* yang dibangkitkan pada saat memasukkan *password*. Selanjutnya pesan akan di *decode* dan menghasilkan uraian pesan (Hafidh & Agus, 2016).

2.5 *Mean Square Error* (MSE) dan *Peak Signal to Noise Ratio* (PSNR)

Peak signal to noise ratio (PSNR) merupakan sebuah parameter yang biasa digunakan dalam proses kompresi *image* untuk menentukan kualitas hasil rekontruksi *image* akhir. Kualitas citra hasil rekontruksi setelah dikompresi dapat dilihat dari nilai PSNR untuk membandingkan tingkat kemiripan antara citra asli dan hasil rekontruksi. Nilai PSNR dipengaruhi oleh nilai MSE suatu jenis gelombang yang diuji. *Mean square error* (MSE) merupakan rata-rata kuadrat nilai kesalahan antara *image* asli dengan hasil kompresi. Semakin rendah nilai MSE hasil pengolahan *image* maka proses rekontruksi sangat baik. Secara matematis MSE dapat dituliskan sebagai berikut (Andrean, 2015) :

$$\boldsymbol{MSE = \frac{1}{HW}\sum_{i=1}^{H}\sum_{j=1}^{W}((P\,(i,j) - S\,(i,j))^2} \qquad \textbf{(14)}$$

Keterangan :

H = dimensi tinggi
W = dimensi lebar
P(i,j) = gambar stego
S(I,j) = gambar sampul

Nilai PSNR dengan tingkat kemiripan tertinggi untuk masing-masing *image* dapat memperngaruhi rasio kompresi warna terhadap gelombamg singkat (*wavelet*). Semakin tinggi nilai kompresi warna, artinya kemiripan hasil rekontruksi sangat baik karena mendekati *image* aslinya. Untuk PSNR bisa dihitung menggunakan rumus berikut (Andrean, 2015) :

$$\boldsymbol{PSNR = 10\log_{10}\frac{L^2}{MSE}} \qquad \textbf{(15)}$$

Dimana L adalah nilai maksimum dari nilai piksel yaitu 255.

III. Hasil dan Pembahasan

3.1 Hasil Pengujian

Bahan yang digunakan dalam penelitian ini berupa pesan rahasia yang akan disisipkan kedalam suatu gambar, pesan rahasia dalam format (.txt) berjumlah 8 (delapan) dengan panjang karakter maksimal 550 kata, dan beberapa gambar dengan format (.jpg) berjumlah 5(lima) dengan ukuran maksimal 640 x 640 pixcel. Setiap pesan *plaintext* akan diubah terlebih dahulu menjadi pesan *chipertext* menggunakan algoritma *blowfish*, data *plaintext* dan data *chipertekxt* dapat dilihat pada Tabel 2.

Tabel 2. Hasil Pengujian Algoritma *Blowfish*

No	*Plaintext*	Ukuran file	Hasil *chipertext*	Ukuran file	Waktu
1	8_bit.*txt*	8 Bytes	Hasil_8_bit.*txt*	8 Bytes	11.2s
2	32_bit.*txt*	32 Bytes	Hasil_32_bit.*txt*	32 Bytes	20.7s
3	48_bit.*txt*	48 Bytes	Hasil_48_bit.*txt*	48 Bytes	16.1s
4	64_bit.*txt*	64 Bytes	Hasil_64_bit.*txt*	64 Bytes	30.5s
5	128_bit.*txt*	128 Bytes	Hasil_128_bit.*txt*	128 Bytes	9.2s
6	256_bit.*txt*	256 Bytes	Hasil_256_bit.*txt*	256 Bytes	12.3s
7	512_bit.*txt*	512 Bytes	Hasil_512_bit.*txt*	512 Bytes	80.1s
8	550_bit.*txt*	550 Bytes	Hasil_550_bit.*txt*	550 Bytes	Null

$Total\ waktu = 180{,}1s(2menit20detik)$

$Rata - rata = \frac{180{,}1}{8} = 22{,}512$

Dari 8 data yang telah diuji untuk enkripsi pesan rahasia berhasil diimplementasikan, pada pengujian ke-8 waktu proses enkripsi tidak muncul namun pesan berhasil di enkripsikan, dengan total waktu yang dibutuhkan untuk mengenkripsi 8 data pesan adalah 180,1s

Setelah didapatkan *chipertext* , selanjutnya melakukan penyisian pesan *chipertext* kedalam gambar sampul dapat dilihat pada Tabel 3.

Tabel 3. Hasil Pengujian Algoritma F5

No	Pesan *Chipertext*	Gambar Asli	Ukuran File	Gambar Stego	Ukuran File
1	Hasil_8_bit.*txt*	Gambar 1-bunga.*jpg*	4,50 KB	Hasil_stego1.*jpg*	2,23 KB
2	Hasil_32_bit.*txt*	Gambar 2-boneka.*jpg*	32,7 KB	Hasil_stego2.*jpg*	13,0 KB
3	Hasil_48_bit.*txt*	Gambar 3-Robot.*jpg*	32,8 KB	Hasil_stego3.*jpg*	34,2 KB
4	Hasil_64_bit.*txt*	*Gambar 4-Smile.jpg*	14,4 KB	Hasil_stego4.*jpg*	13,6 KB
5	Hasil_128_bit.*txt*	*Gambar 5-hitamputih.jpg*	151 KB	Hasil_stego5.*jpg*	44,2 KB
6	Hasil_256_bit.*txt*	*Gambar 2-boneka.jpg*	32,7 KB	Hasil_stego6.*jpg*	12,8 KB
7	Hasil_512_bit.*txt*	*Gambar 3-Robot.jpg*	32,8 KB	Hasil_stego7.*jpg*	34,0 KB
8	Hasil_550_bit.*txt*	*Gambar 4-Smile.jpg*	14,4 KB	Hasil_stego8.*jpg*	12,9 KB

3.2 Hasil Pengujian PSNR dan MSE

Hasil pengujian nilai PSNR dari 8(delapan) gambar sampul dengan pesan rahasia dapat dilihat pada Tabel 4.

Tabel 4. Hasil Pengujian PSNR

No	Gambar Asli	Pesan	Gambar Stego	PSNR
1	 Gambar 1-bunga.*jpg*	Hasil_8_bit.*txt*	 Hasil_stego1.*jpg*	33.80dB
2	 Gambar 2-boneka.*jpg*	Hasil_32_bit.*txt*	 Hasil_stego2.*jpg*	33.79dB
3	 Gambar 3-Robot.*jpg*	Hasil_48_bit.*txt*	 Hasil_stego3.*jpg*	40,25dB

Tabel 4. Tabel Lanjutan

4	 *Gambar 4-Smile.jpg*	Hasil_64_bit.*txt*	 Hasil_stego4.*jpg*	39.00dB
5	 *Gambar 5-hitamputih.jpg*	Hasil_128_bit.*txt*	 Hasil_stego5.*jpg*	36.95dB
6	 Gambar 2-boneka.*jpg*	Hasil_256_bit.*txt*	 Hasil_stego6.*jpg*	33.36dB
7	 Gambar 3-Robot.*jpg*	Hasil_512_bit.*txt*	 Hasil_stego7.*jpg*	39.35dB
8	 *Gambar 4-Smile.jpg*	Hasil_550_bit.*txt*	 Hasil_stego8.*jpg*	36.59dB
TOTAL PSNR				293,09dB

$Total\ PSNR = 293{,}09$

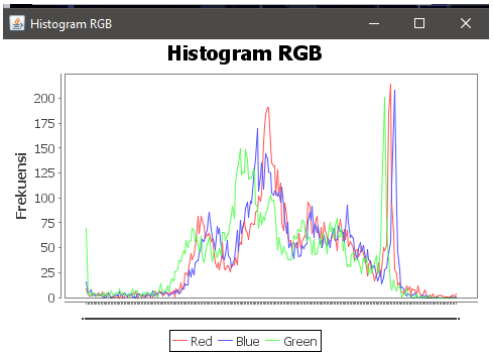
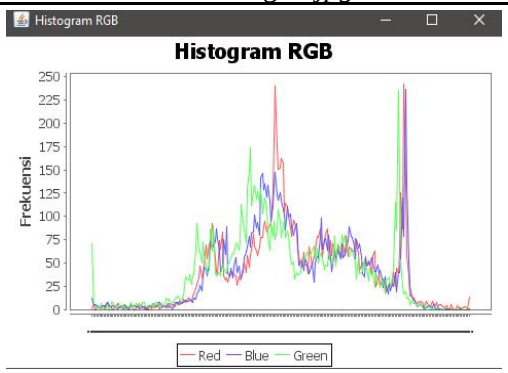
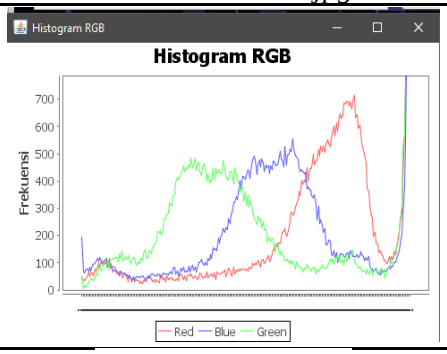
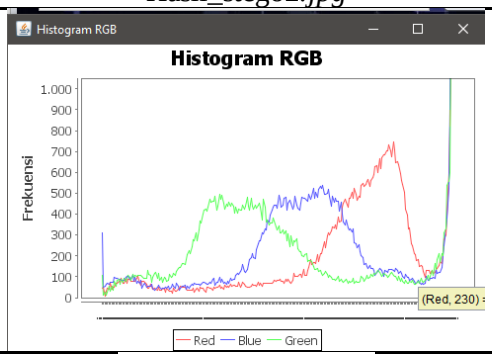
$$Rata - rata = \frac{293.09}{8} = 36{,}63625$$

Dari 8 data yang telah diuji, menghasilkan nilai rata-rata 36,63625 yang mana menunjukkan bahwa tingkat kontruksi pada gambar stego yang dihasilkan memiliki kemiripan yang tidak jauh berbeda dengan gambar sampul. Nilai kontruksi akan semakin baik jika nilai yang dihasilkan oleh PSNR diatas 30.

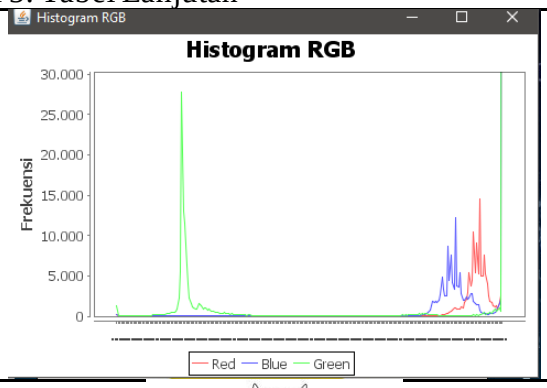
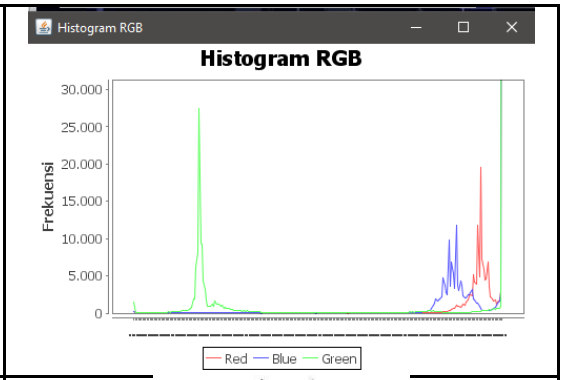
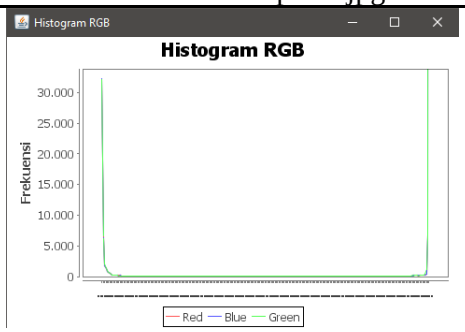
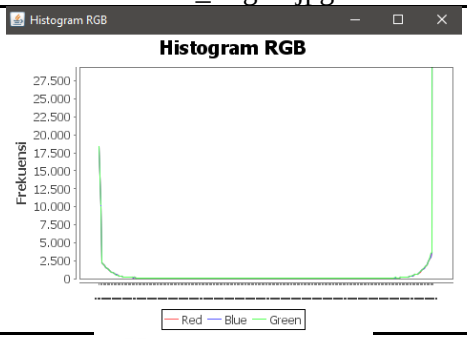
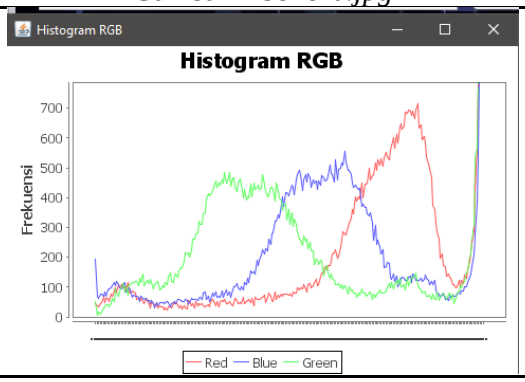
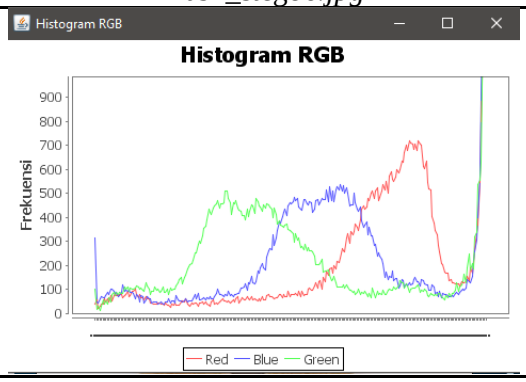
3.3 Hasil Pengujian Histogram

Hasil pengujian histogram bertujuan untuk melihat perbedaan frekuensi pada gambar asli dengan gambar stego, di mana sekilas mata gambar akan sama namun tidak bisa membedakan mana yang terdapat pesan yang sudah disipkan, hasil histogram dapat dilihat pada Tabel 5.

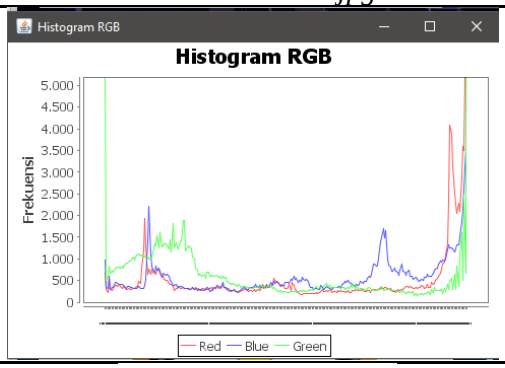
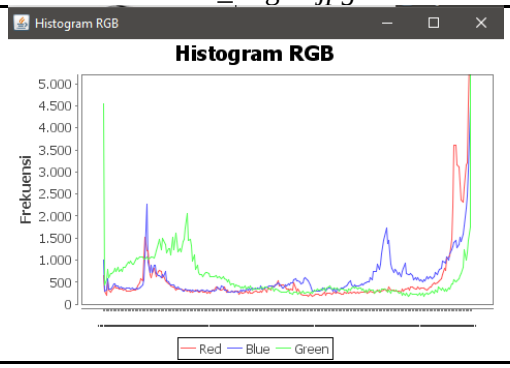
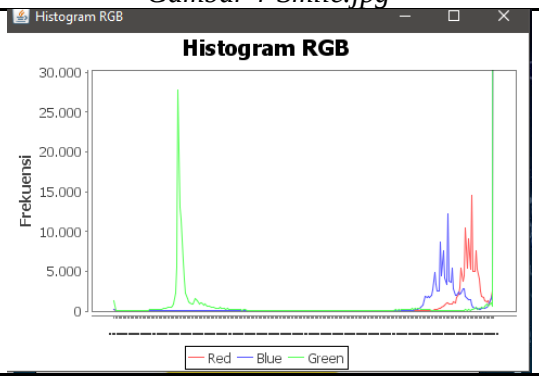
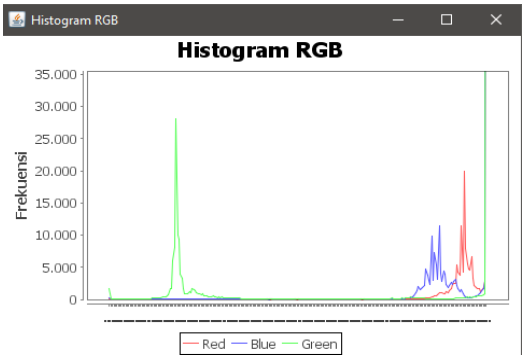
Tabel 5. Hasil Pengujian Histogram

Gambar	 Gambar 1-bunga.*jpg*	 Hasil_stego1.*jpg*
Histogram		
Gambar	 Gambar 2-boneka.*jpg*	 Hasil_stego2.*jpg*
Histogram		
Gambar	 Gambar 3-Robot.*jpg*	 Hasil_stego3.*jpg*

Tabel 5. Tabel Lanjutan

Histogram		
Gambar	 Gambar 5-hitamputih.jpg	 Hasil_stego5.jpg
Histogram		
Gambar	 Gambar 2-boneka.*jpg*	 Hasil_stego6.*jpg*
Histogram		

Tabel 5. Tabel Lanjutan

Gambar	 Gambar 3-Robot.*jpg*	 Hasil_stego7.*jpg*
Histogram		
Gambar	 *Gambar 4-Smile.jpg*	 Hasil_stego8.*jpg*
Histogram		

3.4 Analisa Hasil Pengujian

Analisa pengujian nilai *PSNR* dan *MSE*, untuk mencari nilai *PSNR* dan *MSE* pada 1 gambar cover dan 1 gambar steganografi dengan pengujian 4 kali percobaan. Hasil analisa dapat dilihat pada Tabel 6.

Tabel 6. Analisa Pengujian *PSNR* dan *MSE*

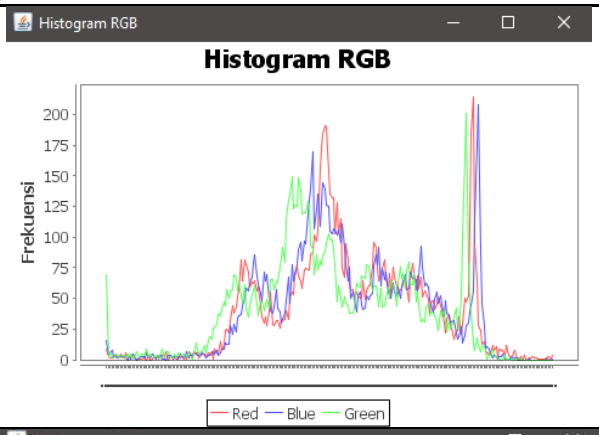
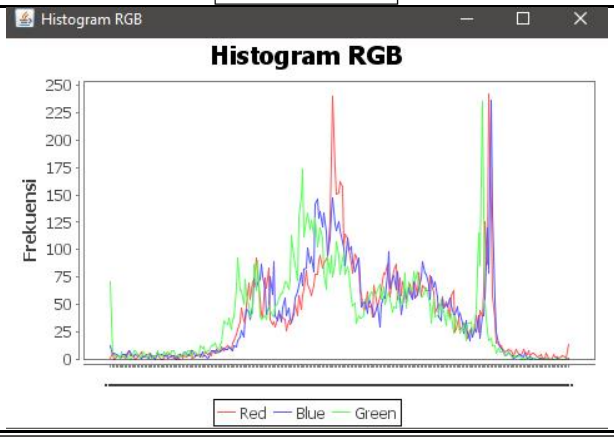
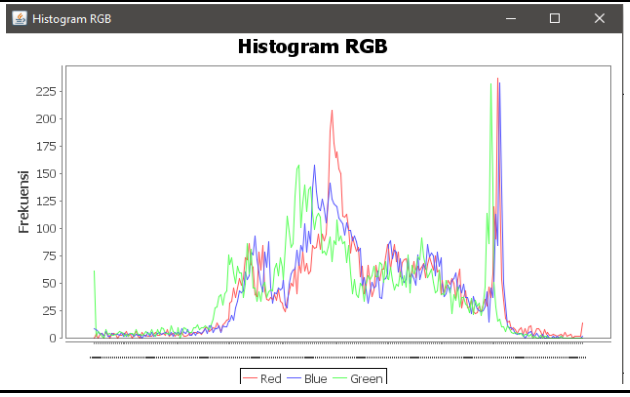
Gambar Cover	Gambar Steganografi	Nilai *PSNR*	Nilai MSE
Gambar 1-bunga.*jpg*	Hasil_stego1.*jpg*	31.80dB	43.00dB
Gambar 1-bunga.*jpg*	Hasil_stego1.*jpg*	31.80dB	43.00dB
Gambar 1-bunga.*jpg*	Hasil_stego1.*jpg*	31.80dB	43.00dB
Gambar 1-bunga.*jpg*	Hasil_stego1.*jpg*	31.80dB	43.00dB

Dari pengujian diatas nilai *PSNR* dan *MSE* tidak berubah dengan nilai 31.80 dB(Decibel) dan nilai MSE 43.00dB.

3.5 Analisa Pengujian Histogram

Analisa pengujian histogram bertujuan untuk melihat kemipiran gambar cover dengan gambar steganografi, analisa akan menggunakan 3(tiga) gambar yaitu gambar *cover*(gambar asli), gambar steganografi pertama(gambar asli disisipkan pesan rahasia 8.bit), dan gambar steganografi kedua(gambar asli disisipkan pesan rahasia 32.bit) hasil analisa dapat dilihat pada tabel 6.

Tabel 6. Analisa Pengujian Histogram

 Gambar Asli	
 Gambar Steganografi Pertama (Pesan 8.Bit)	
 Gambar Steganografi Pertama (Pesan 32.Bit)	

Dari pengujian diatas, 3(tiga) gambar menampilkan histogram yang berbeda, dikarenakan masing - masing gambar sudah ada disisipkan pesan rahasia, walaupun secara kasat mata gambar tetap sama.

IV. Kesimpulan

Berdasarkan hasil penelitian yang dilakukan, dapat di ambil kesimpulan sebagai berikut:

1. Aplikasi penyisipan pesan pada gambar menggunakan algoritma blowfish dan algoritma F5 berhasil diimplementasikan untuk mendapatkan gambar yang sudah disisipkan pesan rahasia(image steganography).
2. Proses enkripsi algoritma blowfish sangat cepat, dengan 8(delapan) data pesan rahasia membutuhkan total waktu 180,1s (2menit20detik).
3. Dari penghitungan PSNR untuk melihat kualitas gambar asli dengan gambar steganografi menunjukkan nilai rata-rata baik dengan nilai 36,63625dB, sehingga metode ini cocok untuk digunakan dalam penyisipan gambar.
4. Analisa gambar asli dengan 2(dua) gambar steganografi memiliki histogram yang berbeda - beda

V. Daftar Pustaka

Abdelhamid. (2018). Data hiding inside JPEG images with high resistance to steganalysis using a novel technique: DCT-M3. *Ain Shams Engineering Journal*, 1965-1974.

Adil, F. A., & Fahrur, R. I. (2015). Implementasi Steganography Menggunakan Algoritma Discrete Cosine Transform. *Jurnal Informatika Polinema*, 35-39.

Amal, V. M., & Ryano, Y. A. (2015). Aplikasi Steganografi pada Citra Digital Menggunakan Algoritma Discrete Cosine Transform. *KALBISCIENTIA, 1*, 77-88.

Andrean, G. (2015). Aplikasi Keamanan Untuk Pengiriman Gambar Dengan Hash Least Significant Bit Berbasis Anroid. *Jurnal Teknologi dan Informasi dan Ilmu Komputer(JTIIK)*.

Dimas, & Herlina. (2015). Analisis Perbandingan Kinerja Algoritma Blowfish Dan Algoritma Twofish Pada Proses Enkripsi Dan Dekripsi. *Jurnal Pseudocode, II*(1), 37-44.

Eder, C., & Perry, J. (2010). F5C: A variant of Faugère's F5 algorithm with reduced Gröbner bases. *Jurnal of Symbolic Computation*, 1442-1458.

Hafidh, Z. D., & Agus, H. (2016). Perbandingan Kapasitas Pesan pada Steganografi DCTSekuensial dan Steganografi DCT F5 dengan Penerapan Point Operation Image Enhancement. *IJCCS*, 35-46.

Indriyono, & Vicky, B. (2019). Integrasi Algoritma Kriptografi Blowfish dengan Stegografi LSB untuk Pengamanan Data pada File MP3. *Jurnal SISFO*, 31-50.

Sianturi, T. N., & Hutagaol, R. G. (2019). Penyisipan Pesan Rahasia Kedalam Audio Menggunakan Algoritma F5. *Seminar Nasional Teknologi Komputer & Sains(SAINTEKS)*, 890-893.

Sitinjak, S., Fauziah, Y., & Juwairiah. (2010). Aplikasi Kriptografi Menggunakan Algoritma Blowfish. *Seminar Nasional Informatika (semnasIF)*, 78-86.

Wardoyo, S., Imanullah, Z., & Fahrizal, R. (2016). Enkripsi dan Dekripsi File dengan Algoritma Blowfish pada Perangkat Mobile Berbasis Anroid. *Jurnal Nasional Teknik Elektro*, 36-44.

Zulfikar, D. H. (2018). Keamanan Pesan Rahasia Menggunakan Steganografi DCT. *JURNAL ILMIAH INFORMATIKA GLOBAL, XI*, 118-123.