Implementasi Modifikasi Kompresi *Run-Length Encoding* pada Steganografi

(*Implementation of Run-Length Encoding Compression Modification on Steganography*)

Suwardiman dan Fitri Bimantoro*
Program Studi Teknik Informatika, Universitas Mataram
Jl. Majapahit 62, Mataram, Lombok NTB, INDONESIA
Email: imankidis@gmail.com, [gpsutawijaya, bimo]@unram.ac.id

Penulis korespondensi

***Abstract* - RLE is one of the methods to compress data, however it has disadvantages that the compressed data may become twice larger as the original size. Therefore, in this research RLE will be modified to solve the problem. In the experiment we tested 3 file format i.e. JPG, PNG and BMP. The testing on JPG and PNG shows that conventional RLE method is not able to compress all images because it obtained negative compression ratio with an average compression ratio about -98,2%, Meanwhile the compression with RLE modification shows that the all image successfully compress with an average compression ratio about 0,17%. The testing on BMP shows the conventional RLE method successfully compress 5 files out of 11 tested images with average compression ratio about -34,7%, Meanwhile the compression with modified RLE successfully compress all tested images with average compression ratio about 18,8 %.**

***Key words*: Steganography, *First of File, End of File, compression , Run-length encoding*.**

I. LATAR BELAKANG

Steganografi [1] adalah salah satu metode yang digunakan untuk mengamankan data agar tidak mudah didapatkan serta disalahgunakan oleh orang yang tidak bertanggung jawab. Salah satu aspek steganografi adalah *Imperceptibility* yaitu pengukuran secara kasat mata terhadap suatu file apakah menimbulkan kecurigaan terdapat data rahasia atau tidak [2]. Teknik *End of File* dan *First of File* merupakan salah satu teknik yang digunakan dalam steganografi. Teknik ini digunakan dengan cara menambahkan data atau pesan rahasia seperti gambar maupun teks pada akhir atau awal *file*. Perhitungan ukuran *file* yang telah disisipkan data sama dengan ukuran *file* sebelum disisipkan data ditambah ukuran data rahasia yang telah diubah menjadi encoding *file* [3]. Untuk meningkatkan keamanan dari sisi *imperceptibility* dibutuhkan kompresi data, salah satu metode kompresi adalah RLE. Penelitian tentang *Run-Length-Encoding* dan steganografi menyebutkan *Run-Length-Encoding* menyembunyikan data hampir setengah dari ukuran *file* dan pesan terkompresi dapat didekompresi dengan baik [4].

Terdapat kekurangan yang dimiliki teknik kompresi *Run-Length-Encoding* jika diterapkan pada data pixel warna pada gambar 8 bit maupun dalam bentuk ASCII 8 bit yaitu terdapat kemungkinan hasil kompresi menjadi lebih besar dari ukuran *file* asli. Ini disebabkan oleh proses perhitungan panjang data juga dilakukan pada data-data yang tidak berulang. Oleh karena permasalahan tersebut pada penelitian ini akan dilakukan modifikasi terhadap teknik kompresi *Run-Length-Encoding* agar dapat menghilangkan kemungkinan adanya hasil kompresi yang lebih besar dari ukuran asli sehingga dapat diterapkan dengan maksimal pada steganografi *First of File* dan *End of File.*

II. TINJAUAN PUSTAKA

Penelitian terkait implementasi *Run Length Encoding* pada kompresi citra [5] menunjukkan citra yang mempunyai sebaran nilai piksel tidak merata memiliki rasio kompresi yang relatif besar sedangkan citra dengan nilai piksel yang merata memiliki rasio kompresi yang lebih kecil. Pada penelitian ini kompresi dilakukan pada citra grayscale dengan mengelompokkan pixel-pixel yang seragam, teknik kompresi masih dilakukan dengan menulis jumlah dari setiap pixel meskipun pixel tidak berulang. Perbedaannya dengan penelitian yang akan dilakukan adalah kompresi akan dilakukan pada byte data citra dengan menggunakan kompresi Run Length Encoding yang akan dimodifikasi sehingga kompresi hanya akan dilakukan pada data yang berulang lebih dari 5 kali.

Penelitian tentang teknik steganografi *End of File, First of File* dan kombinasinya berhasil diimplementasikan pada penyisipan citra ke citra [6]. Pada penelitian ini pesan disisipkan ke dalam pixel citra, penyisipan yang dilakukan ke dalam pixel citra membuat perubahan yang tampak secara kasat mata pada cover citra sehingga menimbulkan kecurigaan pada citra hasil penyisipan. Sehingga pada penelitian ini dilakukan penyisipan dengan metode yang

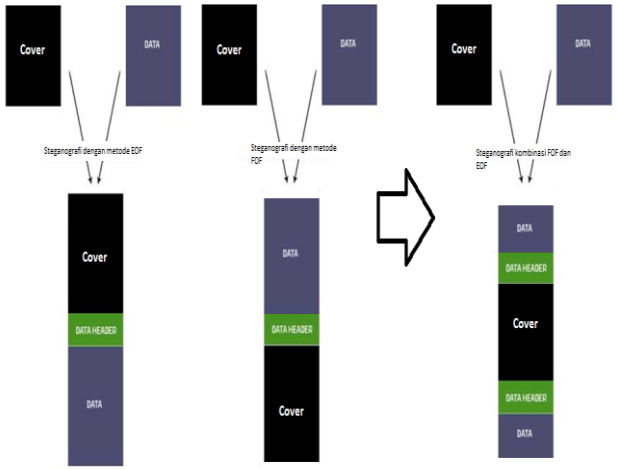
sama tetapi penyisipan dilakukan pada byte data cover dan dilakukan kompresi terlebih dahulu pada pesan.

Penelitian tentang kompresi *Run-Length-Encoding* pada citra digital yang telah dilakukan menunjukkan kompresi Run-Length-Encoding tanpa modifikasi yaitu dengan menuliskan jumlah setiap data walaupun data tersebut tidak berulang akan menyebabkan hasil kompresi menjadi lebih besar dari ukuran file asli. Terlihat pada hasil pengujian terdapat rasio yang -0,08% yang artinya hasil kompresi lebih besar 0,08 % dari ukuran asli. Melihat hasil tersebut pada penelitian ini akan diterapkan Run-Length-Encoding yang akan dimodifikasi sehingga hanya data-data yang berulang lebih dari 5 data yang akan dikompresi [7].

III. STEGANOGRAFI

Metode ini merupakan metode pengembangan LSB. Dalam metode ini pesan disisipkan di akhir berkas. Teknik EOF atau *End of File* merupakan salah satu teknik yang digunakan dalam steganografi. Teknik ini menggunakan cara dengan menyisipkan data pada akhir *file*. Teknik ini dapat digunakan untuk menyisipkan data yang ukurannya sesuai dengan kebutuhan [8]. Ukuran *file* yang telah disisipkan data sama dengan ukuran *file* sebelum disisipkan data ditambah dengan ukuran data yang disisipkan ke dalam *file* tersebut.

Dengan metode EOF, secara umum media steganografi (*file* yang akan disisipi data) memiliki struktur seperti Gambar 1:



Gambar 1. Struktur *File* Steganografi dengan Metode EoF, FOF dan Kombinasi EOF dan FOF [3]

Pada penelitian ini diterapkan steganografi *First of File, End of File* serta kombinasi *First of File* dan *End of File*. Berbeda dengan penelitian sebelumnya [6] yang menyisipkan pesan pada pixel cover, pada penelitian ini akan dilakukan penyisipan pada *byte* data dengan aturan yaitu pada *First of File* data akan disisipkan seluruhnya di awal dari cover. Pada metode *End of File* data akan disisipkan seluruhnya di akhir *file*. Pada metode kombinasi *First of File dan End of File* data akan disisipkan sebanyak 1000 data genap di awal dari file serta menyisipkan 1000 data ganjil dan sisa data lainnya di akhir *file*.

IV. KOMPRESI RUN LENGTH ENCODING

Algoritma RLE menggunakan pendekatan ruang. Algoritma ini cocok digunakan untuk memampatkan citra yang memiliki kelompok-kelompok piksel berderajat keabu-abuan yang sama. Citra yang mempunyai sebaran nilai piksel tidak merata memiliki rasio kompresi yang relatif besar sedangkan citra dengan nilai piksel yang merata memiliki rasio kompresi yang lebih kecil [5]. Metode ini dilakukan dengan menyatakan seluruh baris citra menjadi sebuah baris *run*, lalu menghitung *run – length* untuk setiap derajat keabu-abuan yang berurutan [9].

Contoh, diketahui sebuah citra memiliki *byte* data sebagai berikut dalam vektor:

(1 1 2 1 4 6 3 1 4 4 4 4 4 4 4 4 6 2 7 2 2 3 4 1 1)

Hasil kompresi:

=(1,2)(2,1)(1,1)(4,1)(6,2)(3,1)(1,1)(4,8)(6,1)(2,1)(7,2)(2,2)(3,1)(4,1)(1,2)

= (1 2 2 1 1 1 4 1 6 2 3 1 1 1 4 8 6 1 2 1 7 2 3 1 4 1 1 2)

Sebelum dikompresi = 25 * 1 *byte* = 25 *byte*

Sesudah dikompresi = 28 * 1 *byte* = 28 *byte*

100 % - 28/25 x 100% = -12%

Terlihat pada contoh, hasil kompresi menjadi lebih besar dari ukuran *file* asli sebanyak 12%.

Terdapat tipe RLE lainnya yaitu I2DRLE [10] yang diterapkan pada citra. Pada RLE ini pencarian data pixel yang sama dilakukan dengan blok-blok persegi, garis atau segitiga. Pada RLE ini tidak terdapat penanda dinamis, hanya terdapat informasi blok-blok yang menyusun citra. RLE ini hanya pernah dilakukan pengujian pada citra grayscale yang terdiri dari piksel 1/0 belum ada penelitian terkait RLE tipe ini untuk citra RGB.

V. KOMPRESI RUN LENGTH ENCODING MODIFIKASI

Kompresi RLE tanpa modifikasi pada penelitian-penelitian sebelumnya [7][11] menunjukkan hasil kompresi menjadi lebih besar dari ukuran sebelum dikompresi maka pada penelitian ini dilakukan modifikasi pada algoritma kompresi sehingga tidak terdapat data hasil kompresi yang menjadi lebih besar dari data sebelum dikompresi yaitu dengan mengelompokkan data *byte* yang sama dari citra dan penulisan jumlah data hanya dilakukan pada *byte* yang berulang lebih dari 5 kali. Jika menerapkan teknik modifikasi ini maka diperlukan byte penanda yang digunakan untuk mengetahui bahwa *byte* data tersebut merupakan *byte* yang terkompresi. Penentuan penanda dilakukan dengan mencari *byte* yang memiliki frekuensi kemunculan terkecil dari *byte* pesan, *byte* ini yang akan dijadikan sebagai penanda.

Contoh, diketahui sebuah citra memiliki *byte* data sebagai berikut dalam vektor:

(1 1 2 1 4 6 3 1 4 4 4 4 4 4 4 4 6 2 7 2 2 3 4 1 1)

Hasil kompresi menjadi:

(1 1 2 1 4 6 3 4 0 8 6 2 7 2 2 3 4 1 1)

Sebelum dikompresi = 25 * 1 *byte* = 25 *byte*

Sesudah dikompresi = 20 * 1 *byte* = 20 *byte*

100 % - 20/25 x 100% = 20%

Terlihat hasil kompresi menjadi lebih baik dengan rasio kompresi 20% dengan tidak mengkompresi data yang memiliki panjang data berulang <5. *Byte* data (4 0 8) adalah merupakan hasil kompresi, dengan byte 0 sebagai penanda, penanda didapatkan dari mencari frekuensi kemunculan setiap byte. Seperti yang terlihat pada Tabel I byte 0 memiliki frekuensi 0.

TABEL I. FREKUENSI BYTE

Byte	Frekuensi
0	0
1	6
2	4
3	2
4	10
5	0
6	2
7	1

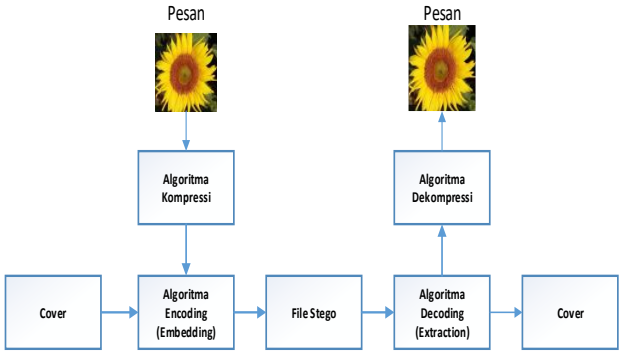
Pemilihan panjang data berulang minimal 5 dilakukan untuk menghindari hasil kompresi yang lebih besar dari ukuran asli, pada modifikasi ini akan terdapat 3 kasus pencarian frekuensi yaitu:

a. Terdapat *byte* data yang berfrekuensi 0 sehingga membutuhkan 1 penanda saja.
b. Penanda hasil kompresi dengan 2 tanda, jika tidak terdapat *byte* berfrekuensi 0 maka akan dilakukan pencarian penanda baru dengan 2 tanda tetapi tetap memperhatikan *byte* data yang memiliki frekuensi terkecil.
c. Penanda hasil kompresi dengan 3 tanda, jika tidak terdapat *byte* kombinasi yang sesuai dengan 2 tanda maka akan dilakukan pencarian penanda baru dengan 3 tanda dan tetap memperhatikan *byte* data yang memiliki frekuensi terkecil.

Dengan melihat kasus ketiga yaitu dengan 3 penanda, maka ada kemungkinan jika mengkompresi data yang memiliki data berulang <5 akan terjadi penambahan jumlah *byte* yang tidak perlu.

VI. IMPLEMENTASI

Aplikasi ini secara garis besar memiliki dua proses utama yaitu penyisipan (*embedding*) pesan dan ekstraksi (*extraction*) pesan yang tersembunyi. Hubungan antara kedua proses tersebut dapat dilihat pada Gambar 2.

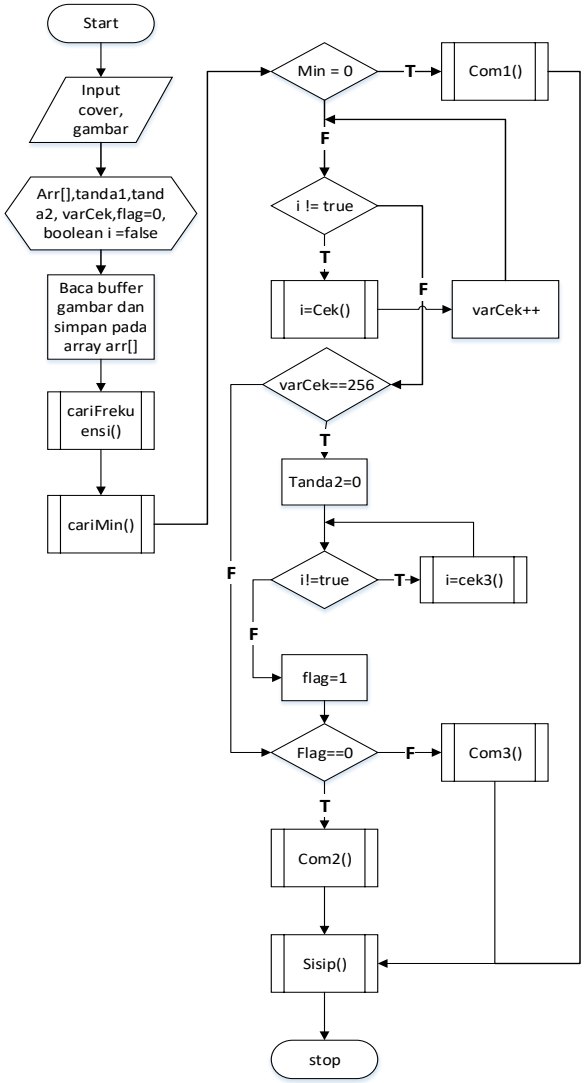


Gambar 2. Proses penyisipan dan ekstraksi pesan

Proses penyisipan pesan dimulai dengan memilih *file* gambar yang akan menjadi pesan rahasia dan dilakukan kompresi gambar untuk mengurangi ukuran dari pesan. Selanjutnya memilih *file* yang menjadi *cover* sebagai tempat penyisipan pesan. Langkah selanjutnya yaitu penyisipan pesan rahasia pada *file cover* (*Embedding*). Berikutnya yaitu ekstraksi pesan rahasia yang dimulai dengan memilih *file* stego. Kemudian mendapatkan kembali informasi pesan rahasia dengan melakukan ekstraksi sesuai dengna algoritma. Data hasil ekstraksi merupakan data yang masih terkompresi dan harus dilakukan dekompressi lagi untuk mendapatkan pesan asli.

A.1. Proses Penyisipan Pesan

Berikut merupakan algoritma dalam tahap untuk menyisipkan pesan rahasia pada citra, dapat dilihat pada Gambar 3.



Gambar 3. Diagram alir penyisipan pesan

Gambar 3. merupakan diagaram alir untuk penyisipan pesan, berikut langkah-langkah penyisipan pesan citra ke dalam cover:

1. Langkah awal yang dilakukan yaitu input *file* gambar yang menjadi pesan rahasia dan *file cover*.
2. Baca *byte file* gambar, lalu hitung frekuensi kemunculan setiap *byte* dari 0 – 255.
   Misalkan diketahui sebuah *file* gambar bertipe PNG memiliki *byte* sebagai berikut:
   [2 2 5 6 9 2 6 8 8 8 8 8 8 8 8 8 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 2 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 9]
3. Dapatkan frekuensi terkecil yang akan digunakan sebagai penanda hasil kompressi. Berdasarkan *byte-byte* di atas didapatkan sebuah tabel frekuensi pada Tabel II.

TABEL II. CONTOH FREKUENSI BYTE

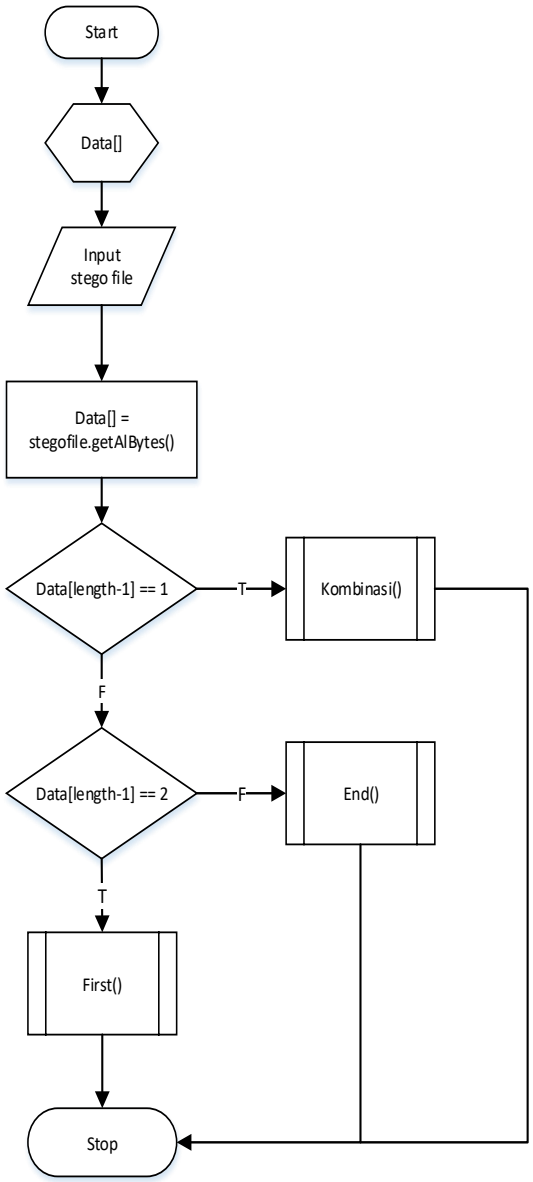
Byte	Frekuensi	Byte	Frekuensi
0	21	45	1
1	3	46	1
2	4	55	2
5	1	56	1
6	3	99	1
7	5	100	1
8	10	123	1
9	3	231	1
22	1	Total	60

4. Kemudian mencari kombinasi yang tidak ada pada *byte* data gambar antara *byte* yang memiliki frekuensi terkecil dengan *byte* 0 – 255.
   Contoh pada poin 2 terdapat byte yang berfrekuensi 0 yang artinya tidak ada pada file pesan oleh karenanya byte berfrekuensi 0 dapat dijadikan penanda hasil kompresi. Pada contoh pada poin 2 penanda yang digunakan adalah 3.Setelah itu lakukan kompressi RLE dengan mencari *byte-byte* data yang memiliki keseragaman lebih dari 5 akan dimampatkan seperti contoh dibawah ini:
   [2 2 5 6 9 5 6 8 8 8 8 8 8 8 8 8 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 5 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 9]
   Hasil kompresi:
   [2 2 5 6 9 5 6 8 3 9 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 5 1 1 1 0 3 21 8 9]
5. Setelah data hasil kompresi didapatkan maka dilakukan penulisan pada *file* cover *byte-byte* hasil compresi, pada kasus ini penyisipan dilakukan menggunakan kombinasi *First of File* dan *End of File* dengan aturan semua index genap akan disisipkan pada *First of File* cover dan index ganjil akan disisipkan pada *End of File* cover. Misalkan *byte-byte* cover sebagai berikut:
   [5 6 2 3 7 8 9 2 4 5 7 1 3 3 7 8 3 4 8 9 0 0 2 3 5 1 7 8 9 5 6 8 0 9 7 6 3 2 3 1 2 3 5 7 8 1 1 3 4 2 7 8 0 9 7 8 2 3]

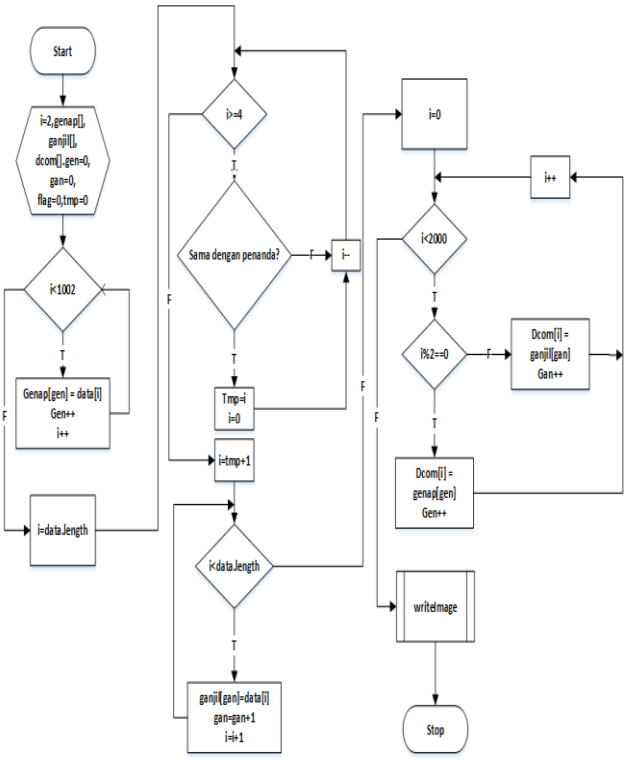
   Hasil penyisipan:
   [3 0 2 5 9 6 3 99 7 9 6 55 7 123 22 46 1 1 3 8 112 110 103 3 3 3 5 6 2 3 7 8 9 2 4 5 7 1 3 3 7 8 3 4 8 9 0 0 2 3 5 1 7 8 9 5 6 8 0 9 7 6 3 2 3 1 2 3 5 7 8 1 1 3 4 2 7 8 0 9 7 8 2 3 3 3 3 2 6 5 8 9 7 55 7 231 56 100 45 7 5 1 0 21 9 0 1]
   Angka berwarna hijau diawal *byte* adalah penanda hasil compresi, angka berwarna biru adalah penanda atau flag batas dari pesan dan cover dan angka hijau sebelum [3 3 3] adalah type *file* pesan yang disisipkan dalam ASCII [112 110 103] yang artinya PNG. Angka [1] pada akhir file adalah penanda pesan disisipkan dengan menggunakan kombinasi *First of File* dan *End of File*. Angka [0] pada length-2 adalah penanda pesan dikompresi dengan 1 penanda.
6. Selesai *file* gambar berhasil disisipkan pada *file* cover.

A.2. Proses Ekstraksi Pesan

Berikut merupakan proses dari proses ekstraksi pesan yang telah disisipkan pada *file* atau citra *stego*, diagram alir dapat dilihat pada Gambar 4.



Gambar 4. Diagram alir ekstraksi pesan



Gambar 5. Diagram alir mendapatkan pesan terkompesi

1. Langkah awal yang dilakukan yaitu input *file stego* yang sudah disisipkan pesan
2. Baca *byte file* stego.
   Misalkan diketahui sebuah *file* stego memiliki *byte* sebagai berikut:
   [3 0 2 5 9 6 3 99 7 9 6 55 7 123 22 46 1 1 3 8 112 110 103 3 3 3 5 6 2 3 7 8 9 2 4 5 7 1 3 3 7 8 3 4 8 9 0 0 2 3 5 1 7 8 9 5 6 8 0 9 7 6 3 2 3 1 2 3 5 7 8 1 1 3 4 2 7 8 0 9 7 8 2 3 3 3 3 2 6 5 8 9 7 55 7 231 56 100 45 7 5 1 0 21 9 0 1]
3. *Byte* terakhir data adalah penanda pesan disisipkan dengan metode *First of File*, *End of File* atau kombinasi keduanya, pada kasus *byte* bernilai [1] artinya pesan disisipkan dengan metode kombinasi *First of File* dan *End of File*.
4. *Byte* ke-0 dan ke-1 adalah merupakan penanda dan bukan merupakan bagian dari *byte* pesan. *Byte-byte* awal dimasukan kedalam array *byte* genap dimulai dari data ke-2 hingga 3 *byte* terakhir sebelum ditemukan data dengan kombinasi [3 3 3] secara berurutan.
   [3 0 2 5 9 6 3 99 7 9 6 55 7 123 22 46 1 1 3 8 112 110 103 3 3 3 5 6 2 3 7 8 9 2 4 5 7 1 3 3 7 8 3 4 8 9 0 0 2 3 5 1 7 8 9 5 6 8 0 9 7 6 3 2 3 1 2 3 5 7 8 1 1 3 4 2 7 8 0 9 7 8 2 3 3 3 3 2 6 5 8 9 7 55 7 231 56 100 45 7 5 1 0 21 9 0 1]
   Maka didapatkan *byte* genap adalah:
   (2 5 9 6 3 99 7 9 6 55 7 123 22 46 1 1 3 8)
5. Melanjutkan mencari data [3 3 3] untuk mendapatkan *byte* ganjil dari pesan, pencarian dilakukan dimulai dari belakang, jika ditemukan maka *byte* setelahnya hingga *byte* akhir merupakan *byte* ganjil.
   [3 0 2 5 9 6 3 99 7 9 6 55 7 123 22 46 1 1 3 8 112 110 103 3 3 3 5 6 2 3 7 8 9 2 4 5 7 1 3 3 7 8 3 4 8 9 0 0 2 3 5 1 7 8 9 5 6 8 0 9 7 6 3 2 3 1 2 3 5 7 8 1 1 3 4 2 7 8 0 9 7 8 2 3 3 3 3 2 6 5 8 9 7 55 7 231 56 100 45 7 5 1 0 21 9 0 1]
   Maka didapatkan *byte* ganjil adalah:
   (2 6 5 8 9 7 55 7 231 56 100 45 7 5 1 0 21 9)
6. *Byte* genap dan *byte* ganjil ditulis disimpan pada satu array secara bergantian dimulai dari *byte* genap hingga akhir. Pada tahap ini sudah didapatkan *byte file* pesan gambar yang masih terkompresi.
   [2 2 5 6 9 5 6 8 3 9 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 5 1 1 1 0 3 21 8 9]
7. Pada byte length-2 adalah bernilai 0 artinya pesan terkompressi dengan menggunakan 1 penanda. Maka dilakukan pencarian byte terkompress dengan memperhitungkan 1 penanda yaitu data ke-0 [3] menjadi:
   [2 2 5 6 9 5 6 8 3 9 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 5 1 1 1 0 3 21 8 9]
8. Dekompresi dilakukan dengan menulis kembali *byte-byte* sebelum penanda sebanyak angka yang ada tepat setelah angka 3, misalkan pada contoh [8 3 9] maka hasil dekompresi adalah [8 8 8 8 8 8 8 8 8] → 8 ditulis sebanyak 9 kali. Langkah 7 berlaku pada *byte-byte* selanjutnya hingga akhir data.
   [2 2 5 6 9 5 6 8 8 8 8 8 8 8 8 8 99 7 7 55 9 7 6 231 55 56 7 100 123 45 22 7 46 5 1 1 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 8 9]
9. *Byte-byte* hasil dekompresi ditulis pada sebuah *file* dan disimpan sesuai dengan type *file* yang sudah didapatkan dari *file* stego. Pada contoh type *file* adalah PNG dengan nilai ASCII 112 110 103.
10. Selesai *file* pesan berhasil didapatkan

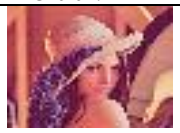
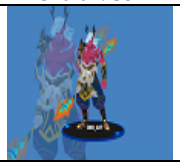
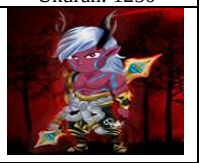
VII. Hasil dan Pembahasan

Pengujian untuk membandingkan dua metode kompresi pada percobaan ini dilakukan dengan mengukur rasio kompresi yang dihasilkan pada berbagai jenis gambar pesan dengan beragam ukuran. Serta pengujian kualitas metode steganografi pada percobaan ini dilakukan dengan beberapa cara yaitu, menghitung nilai MSE dan PSNR file stego gambar dan file pesan setelah recovery.

Bahan yang digunakan untuk melakukan pengujian tersebut menggunakan 33 gambar pesan terdiri dari 11 bertipe JPG,11 bertipe PNG dan 11 bertipa BMP. Terdapat 11 jenis file cover yang digunakan bertipe DOCX, PPTX, XLSX, PDF, JPG dan PNG. Adapun daftar pesan gambar yang digunakan dalam percobaan ini dapat dilihat pada Tabel III.

E-ISSN:2541-0806
P-ISSN:2540-8895

TABEL III. DAFTAR PESAN GAMBAR YANG DIGUNAKAN DALAM PENGUJIAN

		
Id: J1	Id: J2	Id: J3
Type: JPG	Type: JPG	Type: JPG
Ukuran: 51	Ukuran: 113	Ukuran: 60
		
Id: J4	Id: J5	Id: J6
Type: JPG	Type: JPG	Type: JPG
Ukuran: 235	Ukuran: 445	Ukuran:710
		
Id: J7	Id: J8	Id: J9
Type: JPG	Type: JPG	Type: JPG
Ukuran: 1287	Ukuran: 2100	Ukuran: 2171
		
Id: J10	Id: J11	Id: P1
Type: JPG	Type: JPG	Type: PNG
Ukuran: 2486	Ukuran: 5028	Ukuran: 330
		
Id: P2	Id: P3	Id: P4
Type: PNG	Type: PNG	Type: PNG
Ukuran: 621	Ukuran: 729	Ukuran: 307
		
Id: P5	Id: P6	Id: P7
Type: PNG	Type: PNG	Type: PNG
Ukuran: 1246	Ukuran: 1990	Ukuran: 2066
		
Id: P8	Id: P9	Id: P10
Type: PNG	Type: PNG	Type: PNG
Ukuran: 2368	Ukuran: 3059	Ukuran: 3506
		
Id: P11	Id: B1	Id: B2
Type: PNG	Type: BMP	Type: BMP
Ukuran: 3925	Ukuran: 676	Ukuran: 801
		
Id: B3	Id: B4	Id: B5
Type: BMP	Type: BMP	Type: BMP
Ukuran: 1025	Ukuran: 149	Ukuran: 302
		
Id: B6	Id: B7	Id: B8
Type: BMP	Type: BMP	Type: BMP
Ukuran: 384	Ukuran: 907	Ukuran: 1250
		
Id: B9	Id: B10	Id: B11
Type: BMP	Type: BMP	Type: BMP
Ukuran: 2402	Ukuran: 3597	Ukuran: 4006

Selain pesan gambar bahan lainnya yang dibutuhkan pada proses pengujian ini adalah *file cover*. Informasi mengenai *file cover* yang digunakan pada percobaan ini dapat dilihat pada Tabel IV.

TABEL IV. DAFTAR FILE *COVER* YANG DIGUNAKAN DALAM PROSES PENGUJIAN

NO.	FILE *COVER*	JENIS FILE	UKURAN FILE (KB)
1	PPT1	PPTX	242
2	PPT2	PPTX	3444
3	JPG1	JPG	445
4	JPG2	JPG	654
5	DOC1	DOCX	576
6	DOC2	DOCX	622
7	XL1	XLSX	55
8	XL2	XLSX	31
9	PDF1	PDF	1221
10	PDF2	PDF	1321
11	PNG1	PNG	1636

A. Hasil Pengujian

A.1. Rasio Kompresi

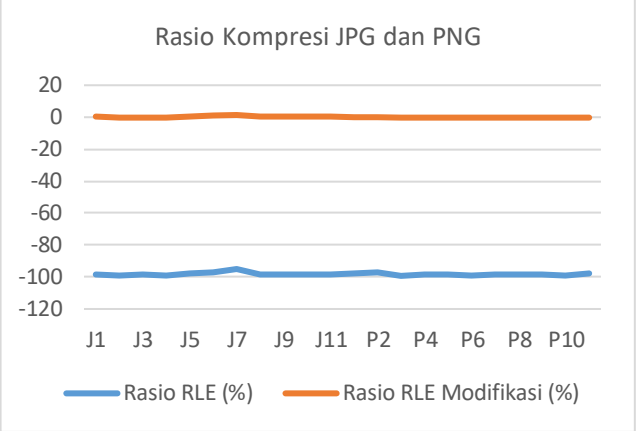
Pengujian Rasio kompresi bertujuan untuk mengetahui dan membandingkan metode kompresi RLE tanpa modifikasi dan RLE dengan modifikasi. Besarnya rasio kompresi merupakan salah satu indikator baik atau tidaknya sebuah algoritma kompresi yang digunakan.

E-ISSN:2541-0806
P-ISSN:2540-8895

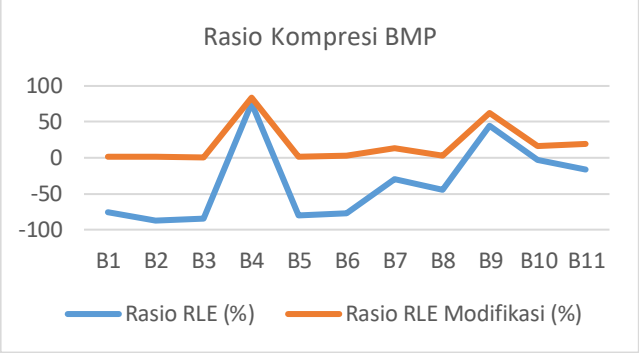
Pengujian dilakukan dengan menyisipkan 33 pesan gambar pada 11 cover Sebelum disisipkan gambar akan terlebih dahulu dikompresi dengan dua metode yaitu RLE modifikasi dan RLE tanpa modifikasi. Hasil pengujian dapat dilihat pada Tabel V.

TABEL V. RASIO KOMPRESI

No.	Pesan	Rasio RLE (%)	Rasio RLE Modifikasi (%)
1	B4	75,81	83,7300
2	B9	43,85	63,0000
3	B11	-16,85	19,6300
4	B10	-3,57	16,4000
5	B7	-30,29	13,9900
6	B8	-44,71	3,5200
7	B6	-77,06	3,2500
8	J7	-95,00	1,3900
9	B5	-80,80	1,2100
10	B1	-75,11	1,0900
11	J6	-96,87	0,8860
12	B2	-87,41	0,8600
13	B3	-85,41	0,4600
14	J5	-97,99	0,4330
15	J8	-98,38	0,2850
16	J9	-98,38	0,2770
17	J10	-98,43	0,2440
18	J11	-98,55	0,1610
19	J1	-98,59	0,1300
20	J2	-98,92	0,0500
21	J3	-98,62	0,0400
22	J4	-99,07	0,0270
23	P5	-98,41	0,0053
24	P8	-98,70	0,0009
25	P7	-98,59	0,0008
26	P10	-98,95	0,0006
27	P3	99,29	0,0006
28	P11	-97,67	0,0003
29	P6	-98,80	0,0002
30	P9	-98,48	0,0001
31	P4	-98,46	0,0001
32	P1	-98,01	0,0000
33	P2	-97,01	0,0000



Gambar 6. Grafik rasio kompresi JPG dan PNG



Gambar 7. Grafik rasio kompresi BMP

Berdasarkan data pada Tabel V rasio kompresi dengan menggunakan metode RLE modifikasi menunjukkan hasil yang lebih baik dari metode RLE tanpa modifikasi, ditunjukkan dengan sebanyak 22 data JPG dan PNG menghasilkan rasio negatif setelah dilakukan kompresi dengan RLE tanpa modifikasi yang artinya data hasil kompresi menjadi lebih besar dari ukuran data sebelum terkompresi. Sebaliknya dengan data yang sama dilakukan kompresi menggunakan RLE modifikasi menunjukkan 22 data memiliki rasio > 0 artinya data hasil kompresi lebih kecil dari data sebelum dilakukan kompresi.

Pengujian algoritma kompresi pada gambar berformat BMP menunjukkan hasil yang sangat baik dengan melihat grafik pada Gambar 6 dan Gambar 7. Ini juga ditunjukkan dengan nilai rasio kompresi dengan menggunakan RLE modifikasi meningkat sangat tinggi pada gambar yang memiliki warna yang seragam begitu juga dengan kompresi dengan RLE tanpa modifikasi juga memiliki rasio yang tinggi, tetapi pada beberapa gambar yang memiliki kombinasi warna yang beragam seperti data uji dengan id B1, B3, B5, B6, dan B7, rasio kompresi RLE modifikasi terlihat lebih baik jika dibandingkan dengan RLE tanpa modifikasi pada gambar yang sama. Ini ditunjukkan dengan dengan semua data yang diuji menghasilkan rasio > 0, sebaliknya pada RLE tanpa modifikasi terdapat beberapa data yang menghasilkan rasio negatif yang artinya data hasil kompresi menjadi lebih besar dari ukuran data sebelum terkompresi.

Kompresi RLE dengan modifikasi menjadi lebih baik dari RLE tanpa modifikasi disebabkan oleh kompresi dengan modifikasi tidak ikut memperhitungkan data yang tidak berulang atau sama. Sedangkan RLE tanpa modifikasi ikut memperhitungkan data yang tidak berulang untuk dikompresi sehingga sangat memungkinkan jika terdapat sebuah data yang memiliki tingkat keragaman *byte/pixel* data sangat tinggi hasil kompresi yang didapatkan dengan RLE tanpa modifikasi menjadi lebih besar dari data sebelum dilakukan kompres.

A.2. Fidelity

Pengujian nilai PSNR (*Peak Signal to Noise Ratio*) dilakukan untuk mengetahui kualitas file stego. Proses pengujian ini akan menghasilkan nilai MSE (*Mean Square Error*) dan nilai PSNR dari sebuah *file* stego. Semakin kecil nilai MSE yang dihasilkan maka semakin baik kualitas *file* stego. Pengujian nilai PSNR pada percobaan ini dilakukan terhadap *file* stego gambar dan *file* pesan hasil proses ekstraksi. Tabel VI merupakan hasil pengujian nilai MSE dan PSNR [12] dalam percobaan ini.

Tabel VI merupakan tabel hasil perhitungan nilai MSE dan PSNR pada *file* stego gambar dan *file* pesan hasil dari proses ekstraksi yang sudah melalui proses dekompresi. Pengujian pada *file* cover ini bertujuan untuk memeriksa kesamaan *file* cover sebelum dan setelah digunakan dalam metode steganografi ini. Selain itu pengujian ini juga digunakan untuk mengukur tingkat keberhasilan algoritma ekstraksi dan dekompresi yang digunakan. Jika hasil pengujian MSE semakin mendekati nol maka akan semakin baik. Terdapat beberapa data yang berlabel NA atau *Not Available* artinya pengujian MSE dan PSNR tidak dapat dilakukan karena bukan merupakan sebuah *file* gambar.

Berdasarkan hasil pengujian *file* stego gambar pada Tabel VI dapat dilihat bahwa seluruh nilai MSE yang dihasilkan adalah 0, MSE dengan nilai 0 menyebabkan hasil perhitungan PSNR Infinity artinya data PSNR dari *file* tersebut tak hingga karena MSE seusai dengan persaman 2-3 menjadi pembagi pada proses perhitungan PSNR. MSE dengan nilai 0 menunjukkan bahwa *file* cover gambar sebelum dan sesudah disisipkan adalah identik. Pada *file* stego PDF, XLSX, DOCX, PPTX, JPG, dan PNG dengan menggunakan metode steganografi *First of File* memiliki kelemahan hanya dapat dibuka dengan beberapa reader. Kelemahan metode *First of File* ini disebabkan oleh penempatan data di awal dari suatu *file* dapat mengganggu isi file dan menyebabkan CRC (*Cyclic Redundancy Check*) *file* rusak. *Cyclic Redundancy Check* (CRC) adalah salah satu fungsi hash yang dikembangkan untuk mendeteksi kerusakan data dalam proses transmisi ataupun penyimpanan. CRC menghasilkan suatu checksum yaitu suatu nilai dihasilkan dari fungsi hash-nya, dimana nilai inilah yang nantinya digunakan untuk mendeteksi error pada transmisi ataupun penyimpanan data. Ketika CRC dari suatu *file* rusak maka CRC tidak dapat lagi menggambarkan isi dari suatu *file*.

TABEL VI. HASIL PERHITUNGAN MSE DAN PSNR

Pesan	Metode	Cover	MSE Pesan Recovery	PSNR Pesan Recovery	MSE Cover	PSNR cover
J5	*End of Fille*	JPG	0	Infinity	0	Infinity
J6		PNG	0	Infinity	0	Infinity
J11		JPG	0	Infinity	0	Infinity
P5		JPG	0	Infinity	0	Infinity
P6		PNG	0	Infinity	0	Infinity
P11		JPG	0	Infinity	0	Infinity
B5		JPG	0	Infinity	0	Infinity
B6		JPG	0	Infinity	0	Infinity
B7		PNG	0	Infinity	0	Infinity
J1		docx	0	Infinity	NA	NA
J2		pptx	0	Infinity	NA	NA
J3		xlxs	0	Infinity	NA	NA
J7		docx	0	Infinity	NA	NA
J8		pptx	0	Infinity	NA	NA
J9		xlxs	0	Infinity	NA	NA
P1		docx	0	Infinity	NA	NA
P2		pptx	0	Infinity	NA	NA
P3		xlxs	0	Infinity	NA	NA
P7		docx	0	Infinity	NA	NA
P8		pptx	0	Infinity	NA	NA
P9		xlxs	0	Infinity	NA	NA
B1		PDF	0	Infinity	NA	NA
B2		PDF	0	Infinity	NA	NA
B3		Docx	0	Infinity	NA	NA
B4		Docx	0	Infinity	NA	NA
B8		Pptx	0	Infinity	NA	NA
B9		pptx	0	Infinity	NA	NA
B10	*End of File*	xlsx	0	Infinity	NA	NA
B11		xlsx	0	Infinity	NA	NA
J4	Kombinasi	PDF	0	Infinity	NA	NA
P4		PDF	0	Infinity	NA	NA
J10	*First of File*	PDF	0	Infinity	NA	NA
P10		PDF	0	Infinity	NA	NA

NA : Not Available

Berdasarkan Tabel VI nilai MSE pada pesan setelah diekstraksi dibandingkan dengan pesan sebelum disisipkan dan dikompresi seluruhnya menghasilkan nilai 0. Hal tersebut menandakan bahwa seluruh *file* pesan hasil

proses ekstraksi dapat kembali ke bentuk sebelumnya seperti *file* pesan sebelum proses penyisipan.

A.3. Imperceptibility

Kriteria ini merupakan kriteria yang diukur dengan cara hanya melihat secara kasat mata, untuk melihat kemampuan steganografi dalam penelitian ini digunakan kuesioner yang dibagikan kepada beberapa orang yang melihat perbedaan cover sebelum dan sesudah dilakukan penyisipan pesan serta pesan sebelum disisipkan dan setelah dilakukan ekstraksi. Untuk melihat dan menguji imperceptibility maka digunakan kuesioner yang dibagikan kepada 30 responden. Pertanyaan-pertanyaan dan hasil dari kuesioner dapat dilihat pada Tabel VII.

TABEL VII. PERTANYAAN DAN HASIL KUESIONER

No	Pertanyaan	1	2	3	4	5
1	Terdapat perbedaan antara file cover dengan file stego?	33,3	56,6	9,9	0	0
2	Terdapat perubahan makna kata atau informasi dan dimensi pada file stego jika dibandingkan dengan cover?	46,6	46,6	6,6%	0	0
3	Terdapat perubahan terhadap pesan gambar sebelum dan sesudah diekstrak?	49,9	36,6	13,3	0	0

Keterangan :
Skor dalam persen (%)
1 = Sangat Tidak Setuju 3 = Cukup 5 = Sangat Setuju
2 = Tidak Setuju 4 = Setuju

Hasil kuesioner pertanyaan satu dan dua menunjukkan pesan berhasil disembunyikan dengan baik pada file cover sehingga ketika dilihat dengan kasat mata orang tidak dapat mengetahui jika *file* sudah disisipkan pesan gambar. Hasil kuesioner pertanyaan 3 menunjukkan pesan berhasil diekstraksi dan didekompresi dengan baik sehingga dapat dibaca dan memiliki tingkat kesamaan yang tinggi antara pesan sebelum dan sesudah disisipkan.

A.4. Robustness

Pengujian *Robustness* atau ketahanan dilakukan untuk mengetahui kemampuan dari *file* stego ketika *file* stego mengalami perubahan apakah data pesan rahasia mengalami gangguan atau bahkan tidak dapat diekstraksi. Tabel VIII merupakan tabel hasil pengujian pada *file-file cover* yang diuji pada penelitian, setelah disisipkan pesan kemudian *file-file* tersebut dilakukan pengubahan terhadap isi dari *file* dan dilakukan ekstraksi terhadap *file* tersebut.

TABEL VIII. HASIL PENGUJIAN *ROBUSTNESS*

No.	File Cover	Berhasil diekstraksi
1.	PDF	Tidak
2.	JPG	Tidak
3.	PNG	Tidak
4.	DOCX	Tidak
5.	PPTX	Tidak
6.	XLSX	Tidak

Berdasarkan Tabel VIII dapat disimpulkan *file* hasil penyisipan atau stego objek dari metode *First of File, End of File* atau kombinasinya akan mengalami kerusakan ketika isi dalam *file* mengalami perubahan sekecil apapun, ini dapat disebabkan oleh perubahan *byte* data pada *file* stego sehingga informasi terkait pesan mengalami perubahan yang dapat menyebabkan pesan tidak dapat diekstraksi kembali.

VIII. KESIMPULAN DAN SARAN

A. Kesimpulan

Dari penelitian yang telah dilakukan kesimpulan yang telah didapatkan yaitu sebagai berikut:

1. Kompresi RLE dengan modifikasi berhasil memperbaiki kekurangan RLE konvensional yaitu tidak terdapat hasil kompresi yang menjadi lebih besar dari ukuran sebelum dikompresi. Ini ditunjukkan dengan rata-rata rasio kompresi dengan RLE pada JPG, PNG dan BMP sebesar -71,03% sedangkan RLE dengan modifikasi sebesar 6,4%.
2. Kompresi dengan menggunakan RLE Modifikasi atau RLE tanpa modifikasi lebih baik jika diterapkan pada gambar BMP dibandingkan dengan gambar JPG dan PNG, ini ditunjukkan oleh data dimana rata-rata hasil kompresi pada *file* berformat BMP lebih besar yaitu 18,8% jika dibandingkan JPG dengan PNG yang memiliki rata-rata 0,17% menggunakan RLE modifikasi dan memiliki rasio -34,7% jika dibandingkan JPG dengan PNG yang memiliki rata-rata -98,2% menggunakan RLE tanpa modifikasi.
3. Metode *First of File* tidak dapat diterapkan pada *file* bertipe JPG, PNG, DOCX, XLSX dan PPTX, ini disebabkan karena penempatan data di awal *file* akan merusak *file* asli karena mengganggu isi *file* asli dan merusak CRC (*Cyclic Redundancy Check*) *file*.
4. Metode steganografi dengan *First of File, End of File* dan kombinasinya memiliki kelemahan terkait ketahanannya terhadap perubahan data, pengujian yang dilakukan pada berbagai format stego setelah disisipkan pesan menunjukkan setelah dilakukan pengubahan pada *file* stego, pesan tidak dapat diekstraksi dari *file*.

B. Saran

Saran untuk penelitian selanjutnya penggunaan metode steganografi *First of File, End of File* dan kombinasi dapat dimodifikasi agar dapat diterapkan pada berbagai file cover. Pada algoritma kompressi RLE dapat dikembangkan lagi konsep penanda pada hasil kompressi

sehingga tidak perlu melakukan komputasi yang tinggi dalam melakukan pencarian penanda.

DAFTAR PUSTAKA

[1] Muslih dan E.H. Rachmawanto, *Penerapan Steganografi Pada Citra / Image Dengan Metode End of File (Eof) Sebagai Aplikasi Pengamanan Data Multimedia*, Semarang, 2014.

[2] M. Edisuryana., R. Isnanto, dan M. Soemantri, "*Aplikasi Steganografi Pada Citra Berformat Bitmap Dengan Menggunakan Metode EOF*", TRANSIENT, Vol.2, No. 3, 2013.

[3] Martono dan Irawan, "P*enggunaan Steganografi dengan Metode End of File (EOF) pada Digital Watermarking"*, TRANSIENT, Vol.2, No. 1.

[4] S. Khan, T. Khan, dan N. Ahmad, "*Run Length Encoding Based Loseless Compressedimage Steganography*", Sindh University Research Journal (Science Series), 2015.

[5] A. Jalaluddin dan Y. Melita, "*Implementasi Metode Run Length Encoding (RLE) untuk Kompresi Citra*", Jurnal Teknika Volume 3 No 2, 2012.

[6] Arfiyah, *Perbandingan Teknik Steganografi Dengan Metode First-Of-File, End-Of-File Dan Kombinasi First-Of File Dan End-Of-File Pada File Bitmap,* Sumatera Utara, 2013.

[7] W. Kusdianti, A. Septiarini, *Kompresi Pada Citra Digital Menggunakan Algoritma Run Length Encoding*, Scan Vol. IX, ISSN 1978-0087, 2014.

[8] Sukrisno dan E. Utami,"*Implementasi Steganografi Teknik EOF, Ridjandel, Shift Cipher, dan Fungsi Hash MD5*", Seminar Nasional Teknologi 2007 (SNT 2007) ISSN: 1978 – 9777: D1-D16, 2007.

[9] C.T. Utari, "*Implementasi Algoritma Run Length Encoding Untuk Perancanganaplikasi Kompresi Dan Dekompresi File Citra*",Jurnal TIMES , Vol. V No 2 : 24-31,2016.

[10] P. Wu, S. Zhou, F. Fang, dan S. Zhou, "*An Improved Two-Dimensional Run-Length Encoding Scheme and Its Application*" vol. 9, no. 4, pp. 339–346, 2016.

[11] K. Fahmi, "*Kompresi Citra Menggunakan Metode Run Length Encoding (Rle) Dan Algoritma Aritmetic Coding* Jurnal INFOTEK, Vol 1, No 2, ISSN 2502-6968, 2016.

[12] G.M. Male, Wirawan, dan E. Setijadi, "*Analisa Kualitas Cttra pada Steganografi untuk Aplikasi e-Government"*, Prosiding Seminar Nasional Manajemen teknologi XV, Program studi MMT-ITS, Surabaya, 2012.