PEMANFAATAN *PIXEL INDICATOR TECHNIQUE* (PIT) DAN *PSEUDO RANDOM NUMBER GENERATOR* (PRNG) DALAM PENYISIPAN PESAN TEKS PADA GAMBAR

Dani Gunawan[1],M. Fadly Syahputra[2], Amira Akhmad Nasution[3]
[1,2,3]Program Studi S1 Teknologi Informasi Fakultas Ilmu Komputer dan Teknologi Informasi
Universitas Sumatera Utara

ABSTRAK

Dengan perkembangan teknologi informasi memudahkan pengguna melakukan pertukaran data dalam kapasitas yang relatif besar. Perkembangan tersebut dapat memicu kejahatan terhadap pencurian data pada saat pengiriman data. Email merupakan salah satu alternatif pertukaran data antar pengguna. Namun untuk mencegah pencurian data oleh pihak yang tidak diinginkan diperlukan integrasi aplikasi steganografi pada email client. Steganografi merupakan teknik penyisipan pesan kedalam suatu media, salah satunya adalah gambar. Salah satu teknik steganografi adalah Pixel Indicator Technique (PIT). PIT merupakan teknik penyembunyian pesan yang menggunakan indikator dan channel. PIT memanfaatkan channel warna 24-bityang terdiri dari Red, Green dan Blue (RGB) dalam setiap pixel. PIT dapat dimodifikasi dengan menerapkan Pseudo Random Number Generator (PRNG) yang merupakan sebuah fungsi matematika yang menghasilkan bilangan acak yang berpola.Implementasi steganografi menggunakan PIT dan PRNG dilakukan pada email clientMozilla Thunderbird. Nilai bilangan acak (PRNG) merupakan indikator yang digunakan untuk menentukan channel warna yang akan digunakan untuk penyisipan pesan. Dengan menerapkan PRNG terhadap PIT dapat disimpulkan bahwa kapasitas dari PIT meningkat dan perbedaan gambar cover dengan gambar stego tidak terlihat perbedaannya secara kasat mata.

Kata Kunci; Steganografi, Pixel Indicator Technique (PIT), Pseudo Random Number Generator (PRNG).

PENDAHULUAN

Pada era digital ini, perkembangan teknologi semakin pesat. Seiring dengan hal tersebut, perkembangan jaringan internet juga semakin luas. Dengan adanya jaringan internet tersebut, memudahkan pengguna (user) saling bertukar informasi (data) dalam kapasitas yang relatif besar. Salah satu alternatif pertukaran tersebut adalah email. Seiring dengan perkembangan tersebut, pencurian data melalui media internet pun semakin marak dilakukan oleh penyadap pada saat proses pengiriman data dan banyak user yang tidak menyadarinya sehingga mengirim data penting melalui email.

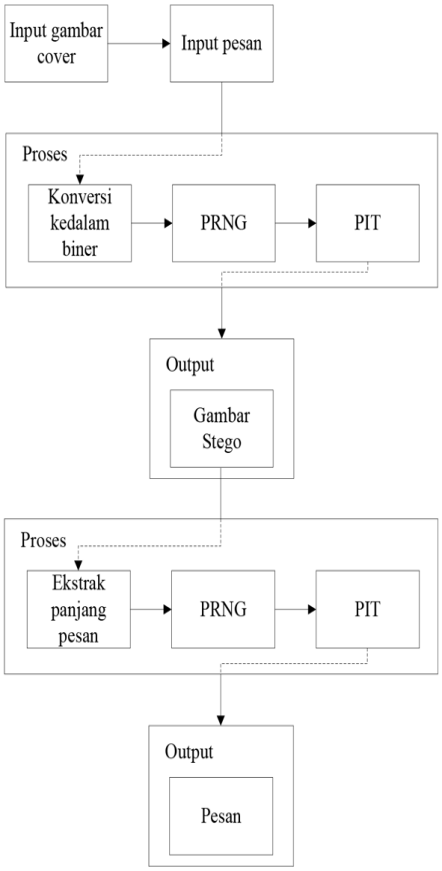
Untuk menghindari terjadinya kejahatan maka diperlukan sebuah metode pengamanan data. Steganografi adalah salah satu metode alternatif pengamanan pertukaran data dengan kapasitas yang relatif besar. Steganografi mengamankan data dengan cara menyisipkan data kedalam suatu media. Pada perkembangannya, penyembunyian data dengan cara steganografi bisa menggunakan beberapa teknikdiantaranya *Least Significant Bit* (LSB), *Stego Color Cycle* (SCC), dan *Pixel Indicator Technique* (PIT). LSB adalah teknik steganografi dengan menyisipkan pesan pada bit paling tidak signifikan atau bit akhir dari sebuah pixel pada gambar. SCC merupakan penyisipan pesan kedalam gambar dengan perputaran channel RGB yang digunakan untuk menyisipkan pesan [3]. PIT merupakan pengembangan dari LSB dimana penyisipan pesan dilakukan pada 2 bit akhir pada channel warna berdasarkan indikator warna dengan menyisipkan minimal 0 bit dan maksimum 4 bit.

METODE PENELITIAN

Arsitektur Umum

Sistem yang dibangun merupakan sebuah add-ons. Add-ons merupakan fitur

yang digunakan untuk meningkatkan dan menyesuaikan aplikasi berbasis Mozilla. Mozilla Thunderbird merupakan *email client* yang dikembangkan oleh Mozilla Foundation dan bersifat open sources. Dalam hal ini, add-ons yang dibangun hanya bisa digunakan pada *email client* mozilla thunderbird.



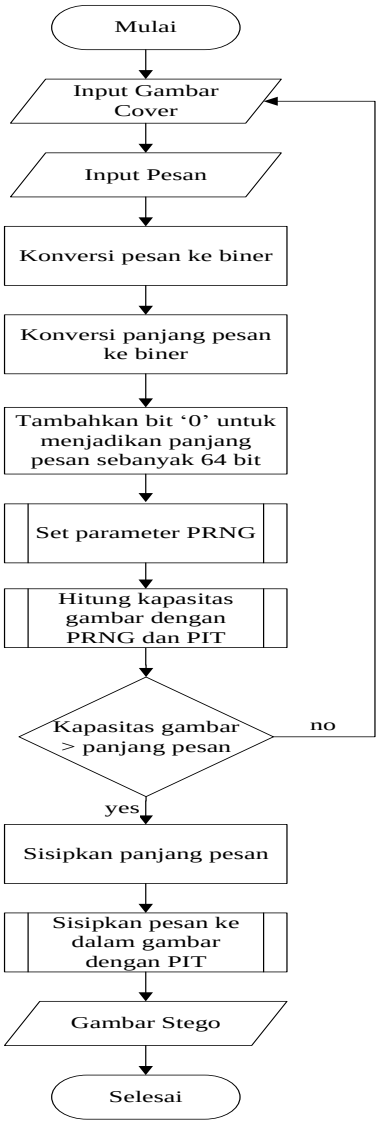
Gambar 1. Arsitektur Umum Penyisipan dan Ekstraksi Pesan

Pada Gambar 1 terlihat proses penyisipan pesan pada sistem yang akan dibangun adalah dimulai dengan meng-input gambar cover dan pesan yang akan disisipi. Dilanjutkan dengan proses konversi pesan ke bilangan biner, bangkitkan nilai PRNG yang digunakan untuk mengacak nilai parameter dan sisipkan pesan dengan algortima PIT. Hasil yang keluar setelah proses selesai adalah gambar stego.

Proses pengekstraksian pesan yang dimulai dengan input yang berupa gambar stego yang diilustrasikan pada Gambar 1. Dilanjutkan dengan mengambil panjang pesan dari 8 bytes pertama dari gambar, lalu bangkitkan nilai PRNG untuk mendapatkan nilai acak parameter dan dilanjutkan dengan mengambil pesan yang disisipkan dengan PIT. Hasilnya adalah pesan yang disisipkan ke dalam gambar.

Proses Penyisipan pesan

Proses penyisipan pesan seperti pada Gambar 2dimulai meng-*input* gambar dilanjutkan meng-*input* pesan. Setelah pesan di-*input*, pesan dikonversi ke dalam bilangan biner. Dihitung total bilangan biner dari hasil konversi pesan untuk diperoleh panjang pesan dalam bentuk desimal. Panjang pesan tadi dikonversikan kembali kedalam bilang biner dan ditambahkan bit ‘0’ didepan bilangan biner dari konversi panjang pesan tersebut. Jumlah bit ‘0’ yang ditambahkan harus menghasilkan jumlah bit sebanyak 64 bit. Hal ini dikarenakan kapasitas yang disediakan untuk menyisipkan panjang pesan sebanyak 8 bytes. Disisipkan dengan metode LSB dimana bit panjang pesan disisipkan pada bit akhir dari setiap channel pada gambar.

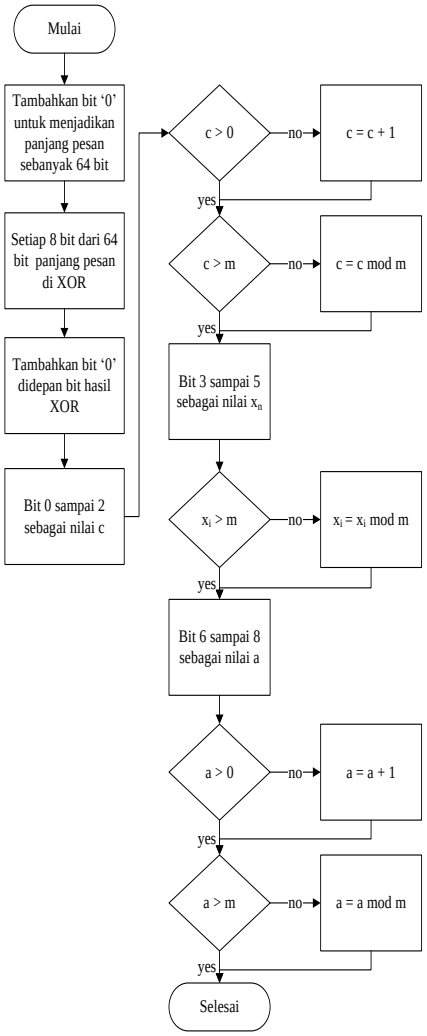


Gambar 2. Proses Penyisipan Pesan

Setelah itu proses dilanjutkan dengan membangkitkan nilai PRNG untuk mengeset bilangan acak parameter. Untuk mendapatkan jumlah banyaknya pesan yang bisa disisipkan ke dalam gambar digunakan perhitungan bilangan acak PRNG dan algoritma PIT lalu bandingkan kapasitas gambar dengan panjang pesan. Jika panjang pesan lebih besar dari kapasitas gambar, maka proses diulang kembali dari proses meng-input gambar. Namun, jika kapasitas gambar lebih besar dari panjang pesan maka proses bisa dilanjutkan dengan menyisipkan pesan ke dalam gambar dengan algoritma LSB seperti yang telah disebutkan diatas. Pesan yang akan disembunyikan disisipkan ke dalam gambar dengan algoritma PIT. Setelah semua proses selesai, maka akan dihasilkan gambar stego.

Set Parameter dengan PRNG

PRNG merupakan sebuah fungsi matematika yang menghasilkan bilangan acak. Bilangan acak ini digunakan untuk mengacak parameter warna yang akan digunakan untuk menyisipkan pesan. Bilangan acak ini akan dibangkitkan pada setiap pixel seperti yang terlihat pada Gambar 3.



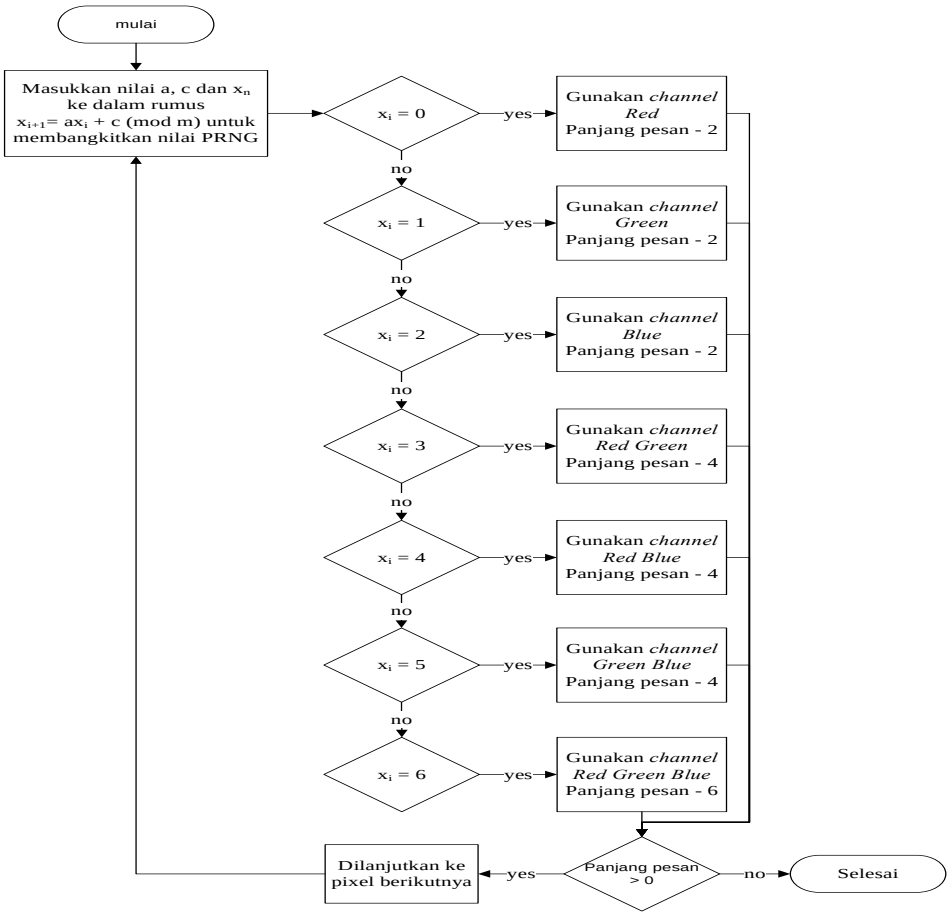
Gambar 3. Proses Membangkitkan PRNG

Untuk memperoleh nilai c, xn dan a, terlebih dahulu bit panjang pesan yang

berjumlah 64 bit dipisah setiap 8 bit. Setiap 8 bit panjang pesan tersebut di XOR. Setelah hasil XOR diperoleh, tambahkan bit '0' di depan 8 bit hasil XOR tersebut. Bit '0' ditambahkan untuk membuat pembagian nilai antara c, xi dan a sama banyaknya. Setiap 3 bit dari jumlah bit tersebut dijadikan sebagai nilai c, a dan nilai awal xi. Nilai c diperoleh dari bit 0 sampai bit 2 dan nilai c harus lebih besar dari 0. Nilai xi diperoleh dari bit 3 sampai bit 5 dan nilai a diperoleh dari bit 6 sampai 8 dan nilai a harus lebih besar dari 0. Nilai c, xi dan a harus lebih besar dari m. Jika tidak, nilai c, xi dan a akan di mod m untuk mendapatkan nilainya. Nilai m disini ditetapkan bedasarkan banyaknya kemungkinan yang diperoleh dari channel red, green, dan blue. Dalam hal ini nilai m = 7.

Penyisipan dengan PIT

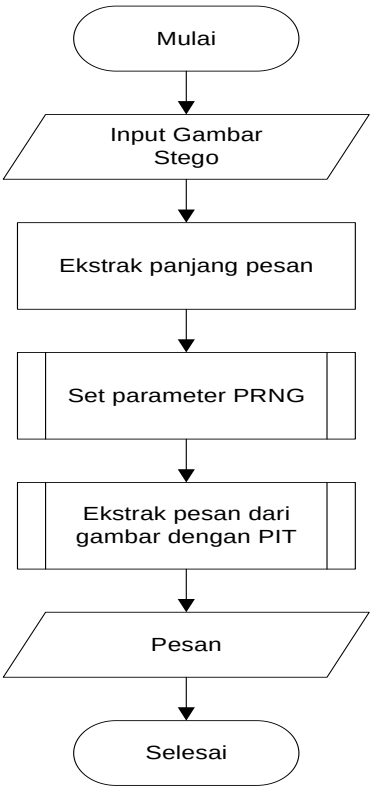
Untuk membangkitkan nilai PRNG, proses seperti pada Gambar 3 terus diulang untuk membangkitkan PRNG untuk memperoleh parameter channel yang akan disisipkan pesan. Seperti yang terlihat pada Gambar 4. Jika nilai xi = 0 maka channel red dari pixel tersebut yang digunakan untuk menyisipkan pesan sebanyak 2 bit. 2 bit akhir dari bit channel red yang diganti dengan bit pesan. Setiap 1 channel disisipkan 2 bit pesan. Panjang pesan akan dikurang seiring dengan disisipkannya pesan ke dalam gambar. Pengurangan panjang pesan sesuai dengan banyaknya pesan yang disisipkan pada 1 pixel. Proses ini terus berulang sampai pengecekan panjang pesan > 0. Jika ya, maka proses penyisipan dilanjutkan ke pixel berikutnya. Jika tidak, maka proses telah selesai.



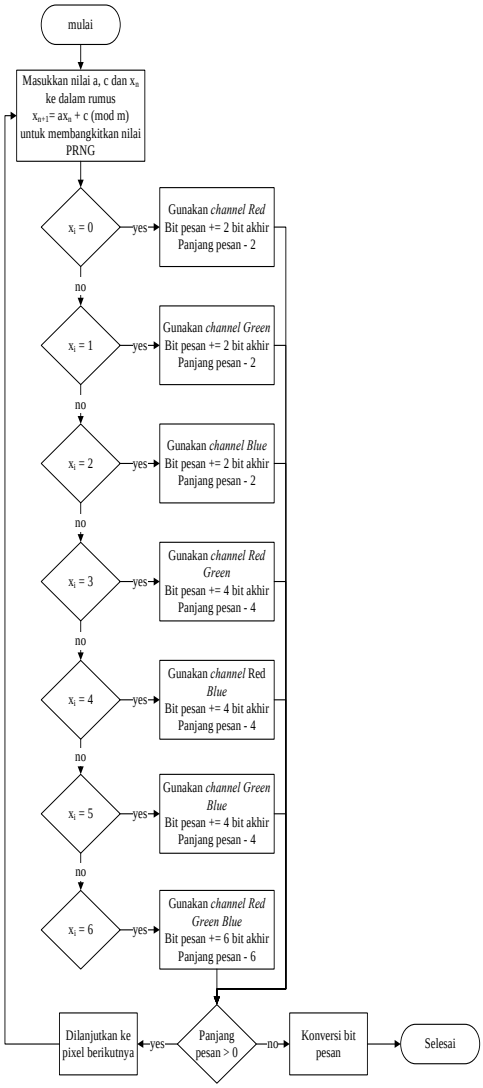
Gambar 4. Penyisipan Pesan Menggunakan PIT

Proses Ekstraksi Pesan

Pada Gambar 5 dapat dilihat alur proses pengekstraksian pesan yang disisipkan pada gambar stego sampai pesan tersebut dapat diketahui oleh si penerima. Proses dimulai dengan meng-*input* gambar stego dan dilanjutkan dengan mengekstrak panjang pesan. Panjang pesan disisipi sebanyak 64 bit pertama pada gambar dengan algortima Least Significant Bit (LSB) dimana pesan disisipi pada 1 bit terakhir atau bit yang paling tidak signifikan pada gambar. Setelah panjang pesan sudah diekstrak, proses dilanjutkan dengan mengeset parameter untuk mengetahui channel apa saja yang digunakan pada setiap setiap pixel. Proses ekstraksi tersebut dilanjutkan dengan mengambil bit pesan dari gambar menggunakan algoritma PIT. Apabila semua proses sudah selesai dilakukan maka pesan yang telah disisipkan pada gambar akan muncul.



Gambar 5. Proses Ekstraksi Pesan



Gambar 6. Ekstraksi Pesan Menggunakan PIT

Ekstraksi dengan PIT

Agar mendapatkan channel yang digunakan untuk menyisipkan pesan pada setiap pixel, nilai PRNG harus dibangkitkan. Seperti terlihat pada Gambar 6 untuk mendapatkan nilai xi maka nilai a, c dan xn harus dimasukkan ke dalam rumus PRNG. Untuk memperoleh nilai - nilai tersebut, penjelasan alur prosesnya sudah dijelaskan pada Gambar 3. Setelah nilai PRNG untuk mengeset parameter diperoleh, proses dilanjutkan dengan mengambil bit pesan yang disembunyikan pada gambar. bit yang diambil adalah 2 bit terakhir dari

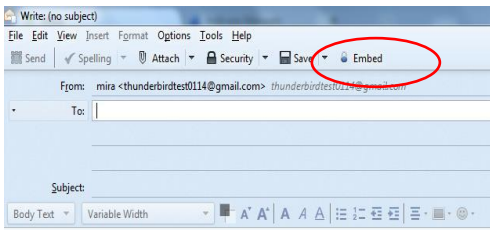
channel warna yang digunakan. Misal nilai xi = 0, maka *channel* yang digunakan adalah *channel red*. Pada *channel red* tersebut diambil 2 bit akhir yang merupakan bit pesan. 2 bit pesan tersebut disimpan di tempat sementara sampai semua bit pesan terpenuhi. Panjang pesan dikurangi dengan 2 karena terdapat 2 bit yang diambil dari *channel red* tersebut. Proses ini terjadi pada 1 pixel. Selanjutnya panjang pesan dibandingkan. Apakah panjang pesan lebih besar dari 0 atau tidak. Jika panjang pesan > 0, maka lanjut ke pixel berikutnya. Proses diulang lagi dari memasukkan nilai a, c dan xi samapai bit pesan yang diambil dari gambar dan panjang pesan dikurangi. Jika panjang pesan <= 0 maka bit pesan yang disimpan sementara sudah terpenuhi maka bit pesan tersebut di konversi ke karakter.

HASIL DAN PEMBAHASAN

Implementasi Sistem

1. Menyisipkan Pesan

Penyisipan pesan dilakukan menggunakan menu 'Embed'. Menu ini diintegrasikan dengan menu standar yang ada di Mozilla Thunderbird. Tampilan menu dapat dilihat pada Gambar 7.



Gambar 7. Menu Embed untuk Menyisipkan Pesan

2. Penulisan Pesan

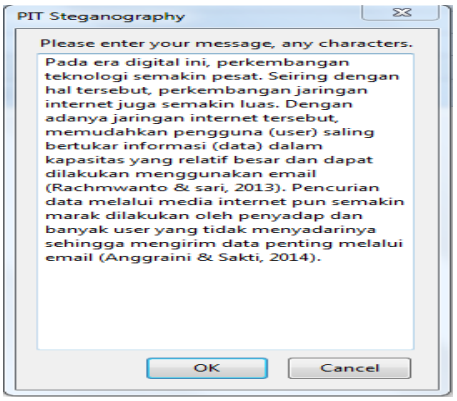
Penulisan pesan dilakukan melalui jendela yang berupa *text dialog* seperti pada Gambar 8. Pada *text dialog* tersebut dapat dituliskan pesan apapun yang ingin disisipkan ke dalam gambar. Tampilannya dapat dilihat pada Gambar 8.

3. Informasi Penyisipan Pesan

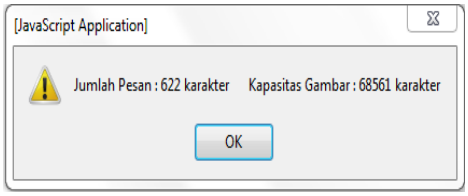
Setelah pesan yang disisipkan berhasil disisipkan, maka akan muncul *dialog box* yang menampilkan banyaknya karakter yang disembunyikan dan maksimal pesan yang bisa dimasukkan ke dalam gambar cover. Tampilan dialog box ini dapat dilihat pada Gambar 9.

4. Ekstraksi Pesan

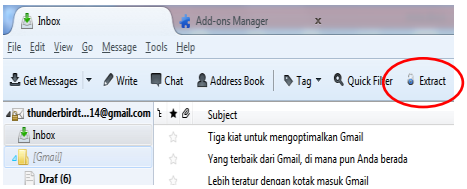
Menu 'Extract' digunakan untuk mengekstraksi pesan yang telah disisipkan ke dalam gambar*cover*. Menu ini dapat dilihat pada Gambar 10.



Gambar 8. Menulis Pesan yang akan Disisipkan



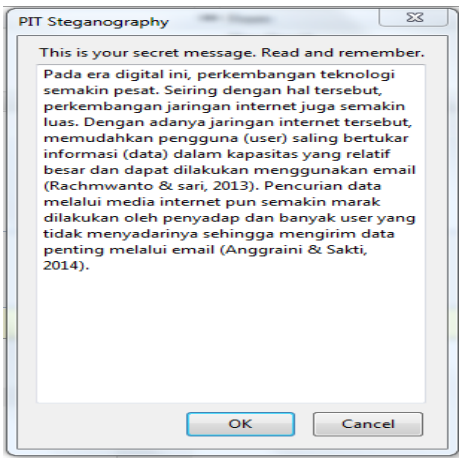
Gambar 9. Informasi Penyisipan Pesan



Gambar 10. Menu untuk Ekstraksi Pesan

5. Hasil Pesan Tersembunyi yang Telah Diekstraksi

Tampilan ini menunjukkan hasil pesan yang diekstraksi sama dengan hasil pesan yang disisip. Tampilan dapat dilihat pada Gambar 11.



Gambar 11. Pesan Telah Diekstraksi

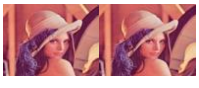
Pengujian Sistem

Pengujian sistem ialah hasil dari perbandingan data yang dilakukan terhadap sistem yang dibangun. Pada pengujian ini akan ditampilkan nilai PSNR dan histogram, kapasitas pesan yang bisa disisipkan pada gambar dan ketahanan dari gambar yang telah disisipi pesan. Hasil pengujian akan ditampilkan sebagai berikut:

1. Hasil Pengujian Perbandingan Gambar yang Dinyatakan dengan Nilai PSNR

Tabel 1 akan menampilkan hasil pengujian nilai PSNR penyisipan pesan menggunakan algoritma PIT dan PRNG. PSNR digunakan untuk mengetahui perbandingan kualitas citra sebelum dan sesudah disisipkan pesan [8]. Semakin tinggi nilai PSNR, maka semakin tidak terlihat perbedaan gambar cover dengan gambar stego.

Tabel 1. Tabel Nilai PSNR Menggunakan PIT dan PRNG

No	Gambar cover	Gambar Stego	Jumlah pesan yang disisipkan	Nilai PSNR
1			294 karakter	56.08689 db
2			294 karakter	58.12177 db
3			294 karakter	58.996 db
4			294 karakter	58.605 db

2. Hasil Pengujian Kapasitas

Pengujian ini dilakukan untuk mengetahui banyaknya pesan yang bisa disisipi pada sebuah gambar. Tabel pengujian kapasitas dapat dilihat pada Tabel 2.

Tabel 2. Kapasitas Maksimum Gambar Menggunakan Algoritma PIT dan PRNG

No	Nama Gambar	Ukuran Gambar	Jumlah pesan yang disisipkan (karakter)	Maksimal pesan yang bisa disisipkan pada gambar (karakter)
1	Gambar 1	100 x 100	294	4275
2	Gambar 2	200 x 200	294	17133
3	Gambar 3	300 x 300	294	19447
4	Gambar 4	400 x 400	294	40200

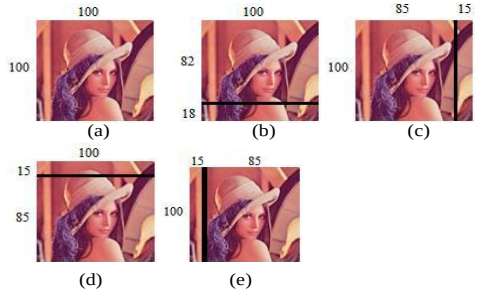
Hasil maksimal pesan menggunakan PIT dengan PRNG lebih banyak yang bisa disispi karena pada penggabungan algoritma PIT dengan fungsi matematika PRNG dihasilkan setiap *pixel*-nya dapat disisipi minimal 2 bit dan maksimal 6 bit.

3. Ketahanan

Pengujian ketahanan ini dilakukan untuk mengetahui apakah pesan masih bisa diekstrak ketika gambar stego sudah dipotong (*crop*), ukurannya diubah (*resize*), diputar (*rotate*), diputar balik (*flip*), dipadatkan (*compress*) dan *grayscale*.

a. Pemotongan (*crop*)

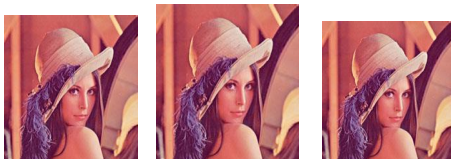
Gambar 12 (a) merupakan gambar stego asli, sedangkan Gambar 12 (b) merupakan gambar stego yang bagian bawah dipotong secara horizontal yang masih bisa diekstrak karena panjang pesan yang disisipi pada 8 *bytes* pertama gambar tidak terpotong. Gambar 12 (c), Gambar 12 (d) dan Gambar 12 (e) merupakan gambar stego yang dipotong secara *vertical* mauun *horizontal* namun tidak bisa diekstrak karena panjang pesan terpotong. Hasil pengujian ini menghasilkan bahwa gambar stego yang dipotong masih bisa diekstrak selama panjang pesan yang disisipkan pada gambar tidak dipotong. Jika panjang pesan dipotong maka tidak ada hasil apapun dari pengekstrakan pesan.



Gambar 12. Pengujian dengan Memotong Gambar

b. Mengubah ukuran (*resize*)

Pada Gambar 13 dapat dilihat bahwa pengujian dilakukan pada gambar dengan ukuran berbeda. Gambar 13 (a) merupakan gambar stego asli yang bisa diekstrak. Gambar 13 (b) adalah gambar stego yang diperbesar ukurannya dan gambar 13 (c) adalah gambar yang diperkecil ukurannya. Hasil pengujian menyatakan bahwa gambar stego yang ukurannya diubah tidak berhasil diekstrak karena bit pada gambar akan berubah.

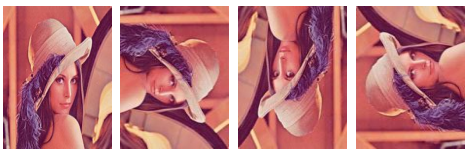


(a) (b) (c)

Gambar 13. Pengujian dengan Mengubah Ukuran Gambar

c. Perputaran (*rotate*)

Pada Gambar 14 (a) adalah gambar stego asli. Gambar 14 (b), Gambar 14 (c) dan Gambar 14 (d) adalah gambar stego diputar ke kanan sebanyak 90, 180 dan 270 derajat. Hasil dari perputaran tersebut membuat gambar stego tidak bisa diekstrak karena *8 bytes* pertama yang disisipi panjang pesan nilai bitnya telah berubah.



(a) (b) (c) (d)

Gambar 14. Pengujian dengan Memutar Gambar

d. Putar balik (*flip*)

Gambar 15 (a) merupakan gambar stego asli yang bisa diekstrak. Gambar 15 (b) dan Gambar 15 (c) merupakan gambar stego yang di *flip* secara *horizontal* dan *vertical*. Gambar stego yang diputar balik juga tidak bisa diekstrak pesan yang disisipi pada gambar karena nilai bit untuk mendapatkan panjang pesan sudah berubah.

(a) (b) (c)

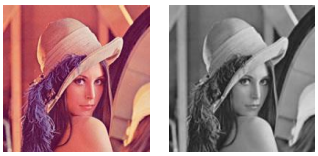
Gambar 15. Pengujian dengan Memutar Balik Gambar

e. Memampatkan (*compress*)

Pada pengujian pemampatan ukuran file gambar, pesan tidak berhasil diektraksi. Haisl ini diakibat oleh ukuran gambar yang berubah menjadi lebih kecil ketika dimampatkan sehingga nilai bit setiap pixel akan berubah. Oleh karena itu pesan yang disisipi pada gambar tidak bisa diekstrak.

f. *Grayscale*

Gambar 16 (a) merupakan gambar stego asli. Gambar 16 (b) merupakan gambar stego yang di *grayscale*. Gambar stego yang di *grayscale* juga tidak bisa diekstrak dikarenakan nilai bit yang telah berubah.



(a) (b)

Gambar 16. Pengujian dengan Membuat Gambar Menjadi Grayscale

PENUTUP

Simpulan

Penyisipan pesan pada gambar menggunakan *Pixel Indicator Technique* (PIT) dan *Pseudo Random Number Generator* (PRNG) yang berupa *add-on* pada *email client*Mozilla Thunderbird menghasilkan kesimpulan sebagai berikut:

1. Hasil maksimal pesan menggunakan PIT dengan PRNG lebih banyak yang bisa disispi pada

gambar daripada menggunakan algoritma PIT saja karena pada penggabungan algoritma PIT dengan fungsi matematika PRNG dihasilkan setiap pikselnya dapat disisipi minimal 2 bit dan maksimal 6 bit sedangkan penyisipan pesan dengan PIT dapat disisipi minimal 0 bit dan maksimal 4 bit pada setiap piksel.

2. Kualitas citra berdasarkan dari nilai PSNR tidak begitu baik namun secara kasat mata gambar stego tersebut tidak terlihat perbedaannya dengan gambar cover.
3. Pengujian ketahanan gambar stego terhadap pemotongan (crop) gambar, putar (rotate) gambar, memutar balik (flip) gambar, mengubah ukuran (resize) gambar, dan grayscale menghasilkan ekstraksi pada pesan gagal karena bit pada gambar telah berubah nilainya sehingga bit yang akan dibaca untuk mengekstraksi pesan tidak diperoleh. Namun, jika gambar stego yang di-crop hanya bagian bawah secara horizontal, pesan masih bisa diekstraksi selama panjang pesan yang disisipi pada gambar tidak terpotong.

Saran

Agar sistem yang dikembangkan dapat memperoleh hasil yang lebih baik dan maksimal, maka diperlukan saran dari semua pihak untuk melengkapi kekurangan yang ada. Saran untuk penelitian selanjutnya adalah:

1. Kekurangan penelitian ini adalah ketahanan gambar stego tidak berhasil diekstraksi jika terjadi perubahan pada gambar stego sehingga pada penelitian selanjutnya masalah ketahanan dapat ditingkatkan.
2. Algoritma pada penelitian ini hanya bisa menggunakan model warna aditif (RGB) sehingga pada penelitian selanjutnya disarankan bisa menggunakan model warna subtraktif (CMYK) dengan algoritma lain.

DAFTAR PUSTAKA

Rachmanto, E.H. & Sari, C.A. 2014. Kriptografi Dengan Algoritma Vernam Chiper dan Steganografi dengan Metode End Of File (eof) Untuk Keamanan Data Email. Techno.COM, Vol. 13 (3): 150-157

Anggraini, Y. & Sakti, D.V.S.Y. 2014. Penerapan Steganografi Metode End Of File (Eof) Dan Enkripsi Metode Data Encryption Standard (Des) Pada Aplikasi Pengamanan Data Gambar Berbasis Java Programming. Konferensi Nasional Sistem Informasi: 1744 – 1753.

Gutub, A.A.. 2010. Pixel Indicator Technique For RGB Image Steganography. Journal of Emerging Technologies in Web Intelligence Vol 2 (1): 57 - 64.

Gutub, A., Anker, M., Abu-Ghalioun, M.,Shaheen, A. & Alvi, A. 2008. Pixel Indicator High Capacity Technique for RGB Image Based Steganography. Saudi Arabia: King Fahd University of Petroleum & Minerals.

Andika, Y. 2012. Pengembangan Random Number Generator dengan Video dan Suara. Skripsi. Institut Teknologi Bandung.

Barus, B.H.D. 2012. Perancangan Add On Keamanan E-mail Mozilla Thunderbird dengan Algoritma Kriptografi XOR dan Three Pass Protocol serta Kompresi Lempel Ziv Welch. Skripsi. Universitas Sumatera Utara.

Sanjeev, M., Mayank, D. & Singh. S.B. 2010. Customized and Secure Image Steganography Through Random Numbers Logic. Signal Processing: An International Journal Vol 1 (1): 1 – 16.

Alatas, P. 2009. Implementasi Teknik Steganografi Dengan Metode LSB Pada Citra Digital. Skripsi. Universitas Gunadarma.