APLIKASI KEAMANAN PESAN MENGGUNAKAN ALGORITMA STEGANOGRAFI DAN KRIPTOGRAFI

Wawan Setiawan, Juwairiah, Herry Sofyan
Prodi Teknik Informatika UPN "Veteran" Yogyakarta
Jl. Babarsari 2 Tambakbayan 55281 Telp (0274)485323
email : juwai_riah@yahoo.com

Abstract

Steganography is a technique of hiding a secret or confidential information in another message so that other people are not aware of the existence of a hidden message. In order for hidden messages or information that it's safer to use the science of cryptography for data encryption. So that the data or messages can not be understood by an unauthorized person who managed to access the data. This application uses methodologies Rappid Guidelines for Application Engineering (Grapple), which consists of Requirement gathering, Analysis, Designs, Development, and Deployment. This application was built using the Java programming language with tools NetBeans IDE 7.0. The algorithm used in this application is the steganography algorithms Least Significant Bit and cryptographic algorithms Vigenere Cipher. The final results of this study is an application of steganography in images by using the method of least significant bit (LSB) and vigenere that can be run on a computer. The image can be used in this application is the image format JPG, PNG, GIF, and BMP. This application can insert text in the image and can perform encryption and decryption at the message you want to insert.

Keywords : application, cryptography, steganography, image, LSB, Vigenere

Steganografi merupakan teknik menyembunyikan pesan atau informasi rahasia ke dalam pesan lain agar orang lain tidak menyadari keberadaan dari pesan yang disembunyikan. Agar pesan atau informasi yang disembunyikan itu lebih aman dapat menggunakan ilmu kriptografi untuk penyandian data. Sehingga data atau pesan tidak dapat dimengerti oleh pihak yang tidak berwenang yang berhasil mengakses data tersebut. Aplikasi ini menggunakan metodologi Guidelines for Rappid Application Engineering (GRAPPLE) yang terdiri dari Requirement gathering, Analysis, designs, Development, dan Deployment. Aplikasi ini dibuat menggunakan bahasa pemrograman java dengan tools NetBeans IDE 7.0. Algoritma yang digunakan dalam aplikasi ini adalah algoritma steganografi Least significant Bit dan algoritma kriptografi Vigenere Cipher. Hasil akhir dari penelitian ini adalah sebuah aplikasi steganografi pada citra dengan menggunakan metode least significant bit (LSB) dan vigenere yang dapat dijalankan pada komputer. Citra yang dapat di digunakan pada aplikasi ini adalah citra berformat JPG, PNG, GIF, dan BMP. Aplikasi ini dapat menyisipkan pesan teks pada gambar serta dapat melakukan proses enkripsi dan dekripsi pada pesan yang ingin disisipkan.

Kata kunci : aplikasi, kriptografi, steganografi, citra, LSB, Vigenere

1. PENDAHULUAN

Dengan seiring berkembangannya kemajuan teknologi saat ini, membuat sebuah informasi sangat penting. Bahkan ada yang mengatakan bahwa masyarakat dunia kini sudah berada pada sebuah *"information-based society".* Sangat pentingnya nilai sebuah informasi menyebabkan seringkali informasi yang ingin disampaikan tidak diterima oleh penerima, melainkan jatuh ditangan orang lain. Untuk mengatasi masalah keamanan informasi tersebut, metode yang bisa digunakan adalah ilmu steganografi dan ilmu kriptografi. Steganografi merupakan teknik menyembunyikan pesan atau informasi rahasia agar orang lain tidak menyadari keberadaan dari pesan yang disembunyikan. Teknik ini menggunakan wadah penampung seperti citra.

Seperti diketahui citra merupakan gambar pada bidang dua dimensi yang dihasilkan melalui proses digitasi. Saat ini peredaran citra di internet sangat banyak sehingga sulit untuk

menemukan file asli citra tersebut. Untuk menjaga keaslian dari suatu citra, bisa juga menggunakan steganografi. Jadi steganografi tidak hanya bisa digunakan untuk menyembunyikan pesan atau informasi, tetapi juga bisa digunakan sebagai proteksi hak cipta dan keaslian suatu citra.

Least significant bit merupakan metode untuk mengimplementasikan steganografi pada citra, yaitu dengan menggantikan bit-bit citra asli dengan bit-bit informasi yang akan disembunyikan, sehingga hasil keluaran akan sama dengan yang aslinya jika dilihat dengan kemampuan indera penglihatan manusia.

Berbeda dengan steganografi, kriptografi merupakan teknik penyandian data. Dengan teknik kriptografi data disandikan atau di enkripsi menjadi data rahasia sehingga data itu tidak akan berarti apa-apa bagi pihak yang tidak berwenang yang berhasil mengakses data tersebut. Data rahasia yang telah di enkripsi dan di terima oleh penerima dapat di ubah kembali atau dideskripsikan ke data asli sehingga dapat di pahami. *Vigenere cipher* merupakan salah satu algoritma kriptografi klasik untuk menyandikan suatu plainteks dengan menggunakan teknik *substitution*.

Penggabungan dua teknik keamanan data yakni kriptografi dengan metode *vigenere* dan steganografi dengan metode *least significant bit* diharapkan mampu mengamankan data. Kriptografi berfungsi untuk mengenkripsikan data atau pesan, sedangkan steganografi berfungsi untuk menyisipkan pesan kedalam sebuah citra.

2. RUMUSAN MASALAH

Berdasarkan latar belakang di atas, maka dapat dibuat rumusan masalah, yaitu : "Bagaimana membuat aplikasi keamanan pesan yang dapat mengimplementasikan steganografi pada citra dengan metode least significant bit (LSB) dan algoritma kriptografi vigenere?"

3. BATASAN MASALAH

Untuk memberikan ruang lingkup yang jelas terhadap suatu objek penelitian, maka dibuat batasan-batasan masalah dalam penelitian ini sebagai berikut :

1. Jenis plainteks yang digunakan adalah dalam bentuk karakter alphabet A - Z.
2. Wadah yang digunakan untuk menyisipkan pesan adalah media dalam bentuk citra atau gambar dengan format bmp, jpg, gif dan png.
3. Hasil file output disimpan dengan format png.
4. Pendeteksian pesan dan keamanan kunci tidak dibahas dalam penulisan skripsi ini.
5. Penelitian ini hanya sampai pada tahap uji coba dan aplikasi bersifat *standalone*.

4. METODOLOGI PENELITIAN

Dalam pengembangan aplikasi ini digunakan metodologi *Guidelines for Rappid Application Engineering* (GRAPPLE) yang terdiri dari *Requirement gathering, Analysis, designs, Development, dan Deployment*. Namun pada pada sistem ini hanya sampai pada tahap *Development*.

5. DASAR TEORI

Least Significant Bit (LSB)

Least significant bit (LSB) merupakan salah satu teknik dalam steganografi. Teknik LSB yaitu menggantikan bit terakhir pada gambar dengan bit yang akan disembunyikan (pesan). Misalkan bit pada gambar dengan ukuran 3 pixel sebagai berikut:

(00111111 11101001 11001000)
(00111111 11001000 11101001)
(11000000 00100111 11101001)

Pesan yang akan disisipkan adalah karakter "A" yang memiliki biner 10000001, stego image yang akan dihasilkan adalah:

(00111111 11101000 11001000)
(00111110 11001000 11101000)
(11000000 00100111 11101001)

Ada dua teknik yang dapat digunakan pada LSB, yaitu penyisipan secara sekuensial dan secara acak. Penyisipan sekuensial dilakukan berurutan sedangkan penyisipan acak dilakukan dengan memasukan kata kunci (*stego key*) (Sukmawan, 2002).

Vigenere Cipher

Vigenere cipher mungkin adalah contoh terbaik dari *cipher* alphabet-majemuk *manual*. Algoritma ini dipublikasikan oleh diplomat (sekaligus seorang kriptologis) Perancis, Blaise de Vigenere pada abad 16, meskipun Giovan Batista Belaso telah menggambarkanya pertama kali pada tahun 1553 seperti ditulis di dalam bukunya *La Cifra del Sig.* Giovan Batista Belaso. *Vigenere cipher* dipublikasikan pada tahun 1586, tetapi algoritma tersebut baru dikenal luas 200 tahun kemudian yang oleh penemunya *cipher* tersebut kemudian dinamakan *vigenere cipher*.

Secara sistematis, misalkan kunci dengan panjang *m* adalah rangkaian $k_1k_1…k_m$, plainteks adalah rangkaian $p_1p_2…p_t$, dan cipherteks dalah rangkaian $c_1c_2….c_t$, maka enkripsi pada Vigenere Cipher dapat dinyatakan sebagai berikut (Munir, 2006):

$$c_i = (pi + k_r) \bmod 26 \quad (1 \le i \le t)$$

$i = r \pmod m$ $(1 \le r \le t)$ dengan indeks A = 0, B = 1,… Z = 25.

Jika panjang kunci lebih pendek dari pada panjang plainteks, maka kunci diulang penggunaanya (sistem periodik). Bila panjang kunci adalah *m*, maka periodenya dikatakan *m.* Sebagai contoh, jika plainteks adalah *THIS PLAINTEKS* dan kunci adalah *SONY*, maka penggunaan kunci secara periodik adalah sebagai berikut:

Plainteks : T H I S P L A I N T E K S
Kunci : S O N Y S ON Y S O N Y S
Cipherteks : L V V Q H Z N G F H R V L

Pada contoh di atas huruf T dienkripsi dengan kunci S menjadi :

(T + S) mod 26 = (19 + 18) mod 26 = 11 = **L**

Dan huruf H berikutnya dienkripsi dengan kunci O menjadi :

(H + O) mod 26 = (7 + 14) mod 26 = 21 = **V**

Demikian seterusnya untuk huruf plainteks lainya.

6. ANALISIS DAN PERANCANGAN

Berdasarkan masalah pada latar belakang, maka diperlukan spesifikasi kebutuhan yang berhubungan dengan kemampuan aplikasi keamanan pesan yang akan dibuat. Tabel 1 menjelaskan daftar kebutuhan pengguna yang akan dibuat dalam bentuk Use Case.

Tabel 1. Kebutuhan Pengguna

No	Requirement	Use Case
1	Pengguna dapat menulis pesan dan mengenkripsi pesan	Penyandian Pesan
2	Pengguna dapat menyisipkan pesan ke dalam sebuah citra atau gambar	Penyisipan Pesan
3	Pengguna dapat membaca atau mengekstrak pesan yang telah disisipkan di dalam sebuah citra atau gambar	Membaca Pesan
4	Penggunadapat mendeksipsikan pesan yang telah diekstrak sebelumnya	Penguraian Pesan

Tabel 2 menjelaskan daftar kelas yang akan dibuat.

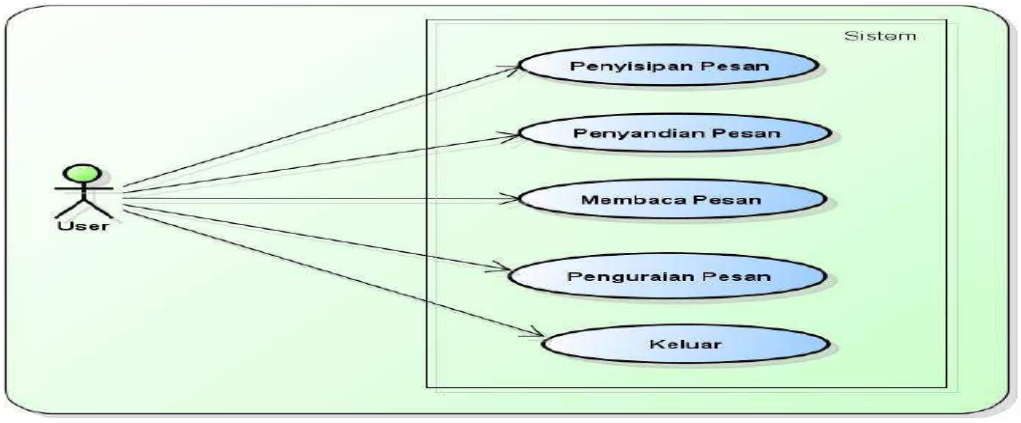
Tabel 2. Daftar Kelas

No	Nama *Class*	Deskripsi
1	Stego_tampil	*Class* ini yang menangani komunikasi antara *actor* dan komponen internal sistem, yaitu dalam hal untuk menulis pesan, menyisipkan pesan, membaca pesan, melihat petunjuk penggunaan aplikasi, about dan keluar dari sistem, semuanya diakses melalui *class* stego_tampil. Di dalam class ini terdapat: *Attribute* : jmenu steganog, jmenu bantu, jmenuitem encode, jmenuitem decode, jmenuitem petunjuk, jmenuitem about, jmenuitem exit, jlabel labeljudul, jbutton decodeButton, jbutton bacapesan, jbutton cancel1, jbutton back1, jbutton bproses1, jbutton bproses2, jbutton bproses3, jlabel image_input, jlabel image_input1 dan jlabel image_output. *Operation* : public stego_tampil, public image_panel, public image1_panel.
2	Filter_gambar	*Class* ini yang menangani komunikasi antara *actor* dan komponen internal sistem, yaitu dalam hal menyaring dan menampilkan format-format gambar tertentu. Di dalam *class* ini terdapat: *Operation* : public filter_gambar, protected filegambar, public Boolean accept, public string getdecription,protected getextention
3	Steganografi	*Class* ini yang menangani penyisipan pesan ke dalam gambar dan ekstrak pesan dari gambar. Di dalam *class* ini terdapat: *Attribute* : teks dan gambar. *Operation* : public steganografi, public Boolean encode, public string decode, private string image_path, private bufferimage getimage, private Boolean setImage, private bufferimage add_text, private byte[] get_byte_data, private byte[] bit_conversion, private byte[] encode_text, private byte[] decode_text.
4	Vigenere	*Class* ini yang menangani enkripsi dan dekripsi pesan. Di dalam *class* ini terdapat: *Attribute* : key, enkripsi, dekripsi, alphabet, mode, pesan dan key. *Operation* : public vigenere, public string mulai, public getAlpahebetPosition, private string enkripsi, private string dekripsi.
5	about	*Class* ini menampilkan informasi dari pembuat aplikasi. Di dalam *class* ini terdapat: *Attribute* : jlabel gtentang, jlabel tentang, dan jlabel tentang1. *Operation* : public about.
6	Petunjuk	*Class* ini menampilkan petunjuk penggunaan aplikasi. Di dalam *class* ini terdapat: *Attribute* : jlabel judul dan jtextarea tampil. *Operation* : public petunjuk dan fileReader Reader.
7	Stego_aksi	*Class* ini yang mengontrol pemanggilan dari semua kelas. Di dalam *class* ini terdapat: *Attribute* : jpanel decode_panel, jpanel tampilG_panel, jlabel labeljudul. *Operation* : public stego_aksi, public void exit actionperformed, public void about actionperformed, public void encode actionperformed, public void decode actionperformed, public void petunjuk actionperformed, public void Bproses1actionperformed, public void decodebutton actionperformed, public void bproses3 actionperformed, public void bproses2 actionperformed, public void back1 actionperformed, public void cancel1 actionperformed, public void bukapesan actionperformed, private void stego_awal, private void decode_view, private void gambar_view, public void update, public static void main.
8	enkripsi	*Class* ini menangani proses enkripsi pesan.*Attribute*: jbutton submitbtn, jbutton submitbtn1, jtextarea area1, jtextarea area2, jtextfield txt1, jbutton submitbtn2. *Operation*: submitbtn actionperformed, submitbtn1actionperformed, submitbtn2 actionperformed, update, private void gambar_view.
9	dekripsi	*Class* ini menangani proses dekripsi pesan. *Attribute*: jbutton submitbtn, jbutton submitbtn1, jtextarea area1, jtextarea area2, jtextfield txt1. *Operation*: submitbtn actionperformed, submitbtn1 actionperformed.

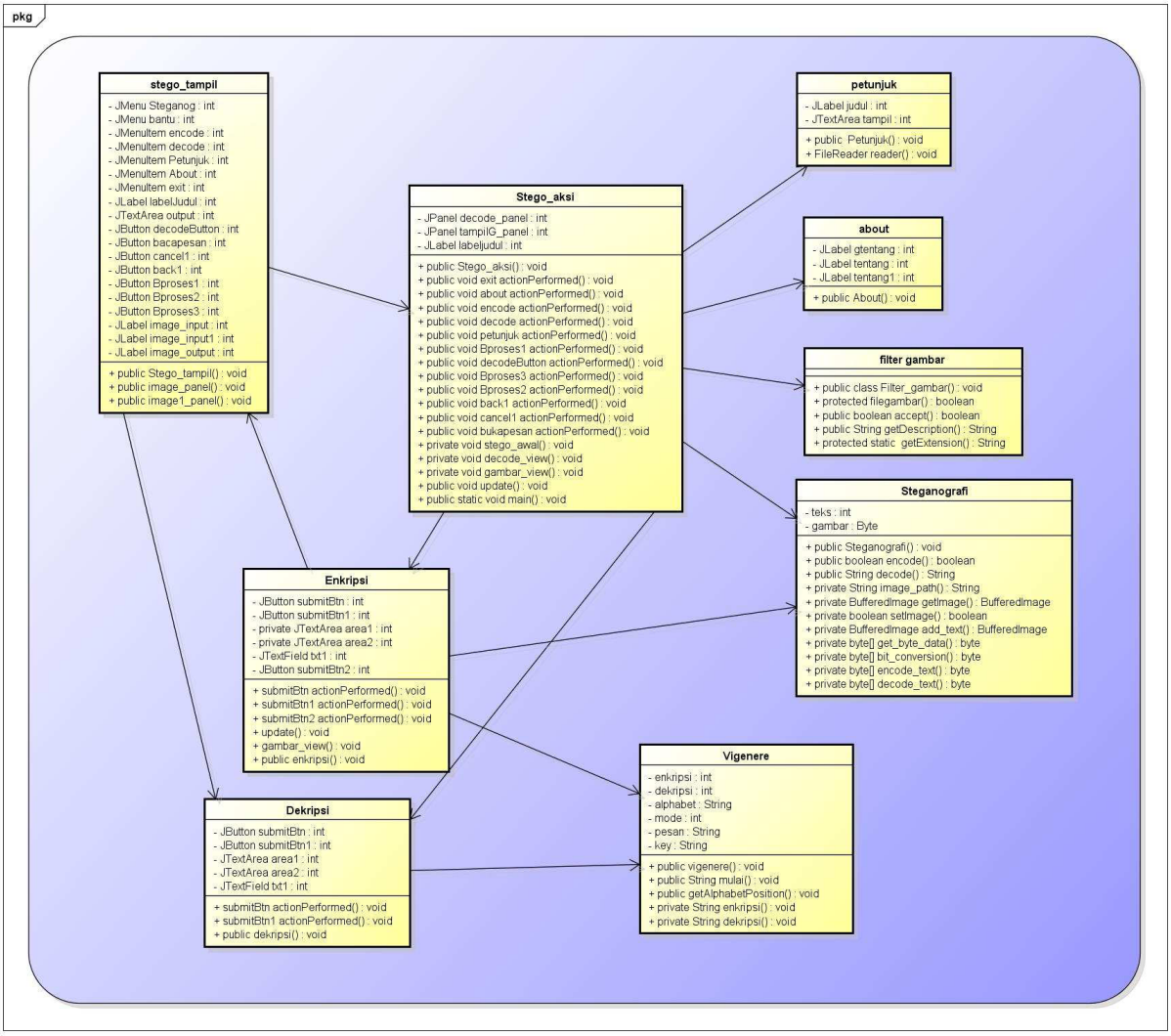
Perancangan

Tahap ini dibuat berdasarkan dari tahap analisis. Dalam perancangan dilakukan 2 tahapan yaitu perancangan diagram use case, diagram sequence, diagram activity, struktur menu dan perancangan antar muka (*interface)*.

Diagram Use-Case



Gambar 1. Diagram Use Case



Gambar 2.Diagram Kelas

7. IMPLEMENTASI

Pembahasan dari pembuatan perangkat lunak ini bersifat umum, yaitu menjelaskan *method*, konstruktor, *class* dan bagian-bagian utama yang dibuat pada tahap *development* dan *deployment*. Bagian utama tersebut dikembangkan dari *class-class* yang terbentuk dari tahap *requirements* hingga tahap *design*. Pada tahap *development* dilakukan pengembangan dan perbaikan *method*, konstruktor dan *class*, agar dapat menghasilkan sistem seperti yang diinginkan. Tahapan *deployment* merupakan tahapan terakhir pada metode GRAPPLE. Seluruh *source code* yang telah disusun mulai dari tahapan perancangan hingga *development* diintegrasikan pada komputer, sehingga seluruh tampilan sistem dapat tampil dan digunakan oleh *user*.

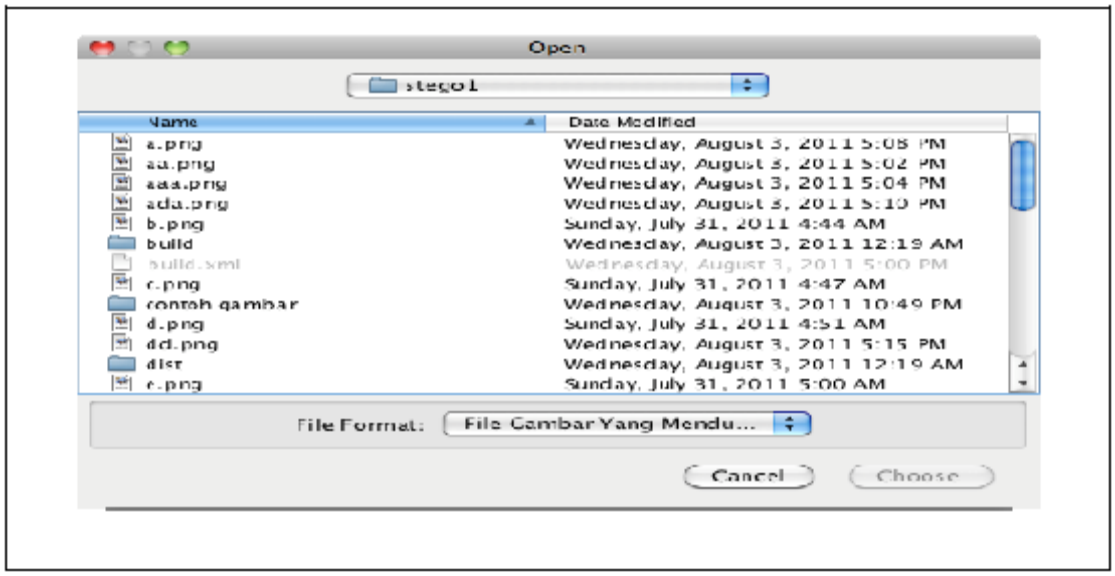
Form Awal

Form awal merupakan *form* utama yang akan tampil setelah aplikasi dijalankan.

Gambar 3. Tampilan Form awal.

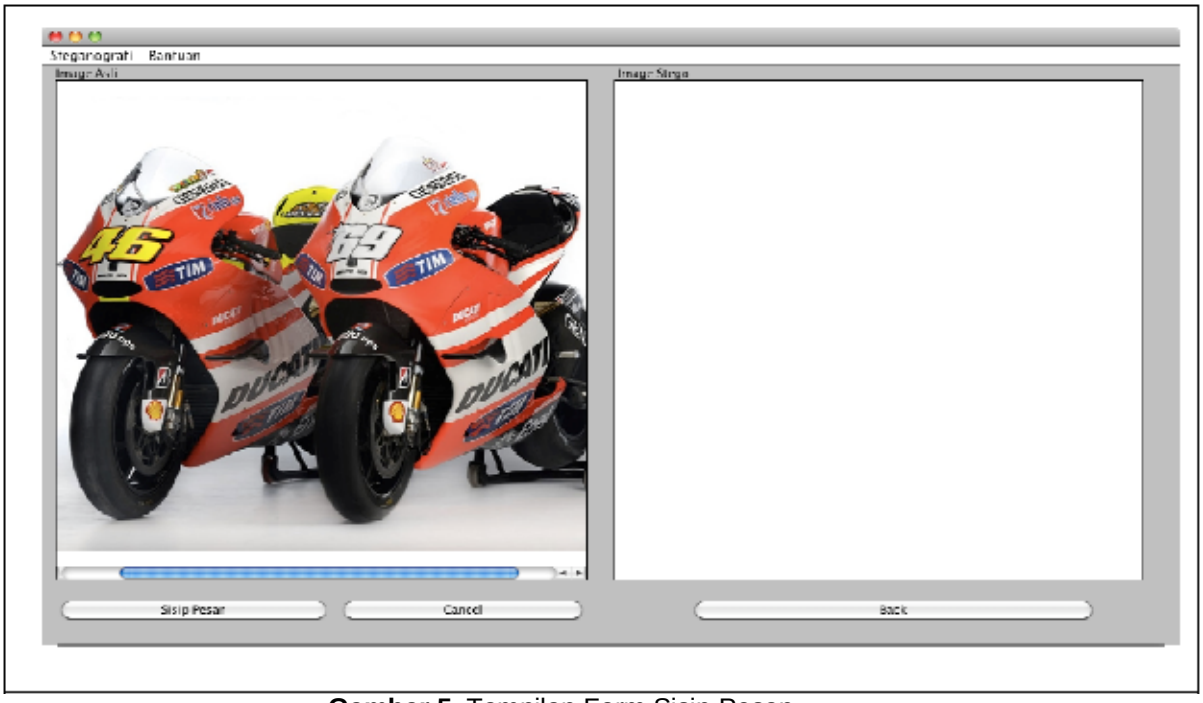
Proses Penyisipan

Ketika *user* memilih menu sisip pesan maka akan tampil *form* open untuk memilih gambar yang akan disisipkan pesan.



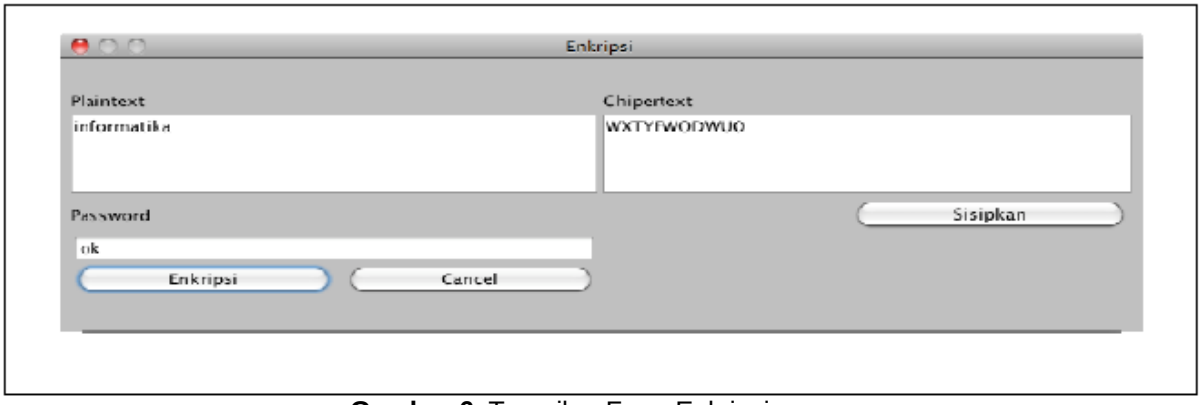
Gambar 4. Tampilan Form Open.

Setelah *user* memilih gambar maka sistem akan menampilkan gambar di *form* sisip pesan.



Gambar 5. Tampilan Form Sisip Pesan

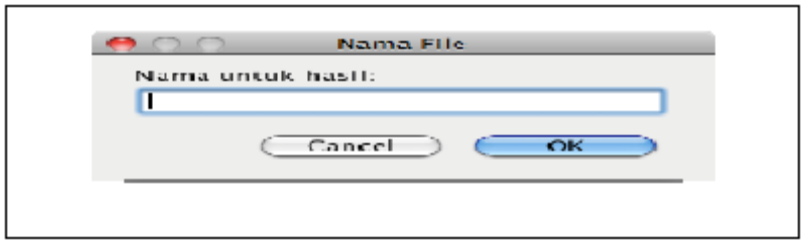
Di dalam *form* sisip pesan terdapat tiga *button* yaitu *button sisip pesan*, *cancel* dan *back*. *Button* sisip pesan akan menampilkan *form* enkripsi, jika *button cancel* akan menampilkan *form* open untuk memilih gambar baru dan *button back* akan kembali ke tampilan awal. Ketika *user* memilih *button* sisip pesan yang terdapat di *form* sisip pesan maka akan tampil *form* enkripsi.



Gambar 6. Tampilan Form Enkripsi

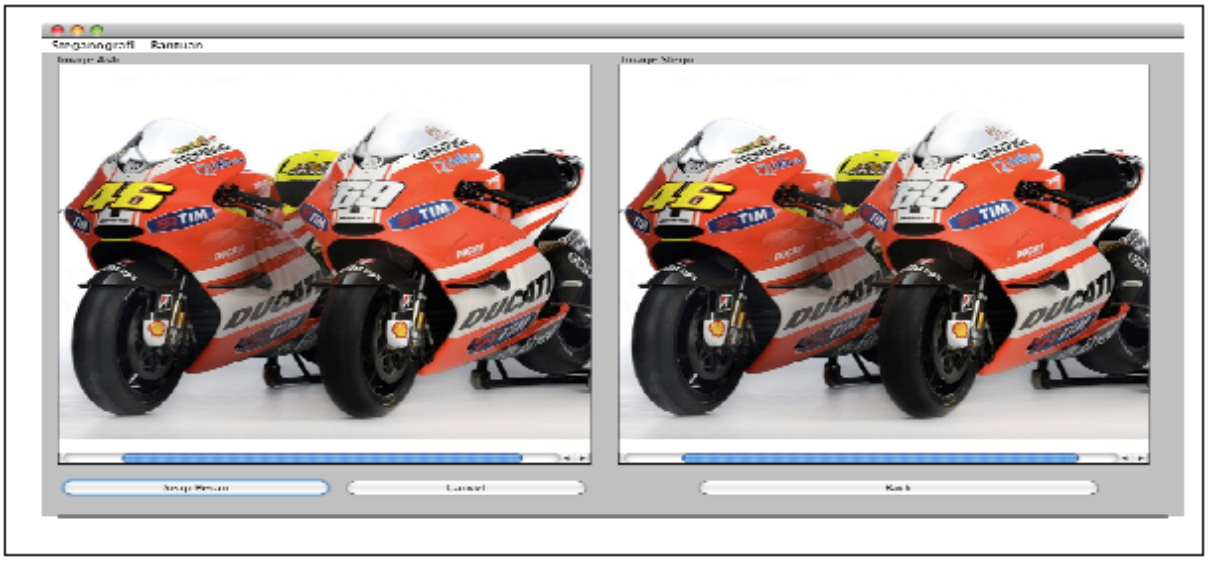
Pada *form* enkripsi terdapat dua *textarea* yaitu *textarea* plaintext dan *textarea* chipertext, satu *textfield* yaitu *textfield* password dan tiga *button* yaitu *button* enkripsi, *button* cancel dan *button* sisipkan. Setelah *user* menginputkan pesan dan *password* maka user harus mengenkripsikan pesan tersebut menjadi pesan chiperteks dengan memilih *button* enkripsi.

Setelah proses enkripsi pesan selesai, selanjutnya sisipkan pesan *chipertext* ke dalam gambar yang telah ditentukan sebelumnya dengan memilih *button* sisipkan. Setelah itu sistem akan meminta *user* menginputkan nama untuk nama gambar yang telah sisipkan pesan.



Gambar 7 Tampilan *Form* untuk Nama Hasil.

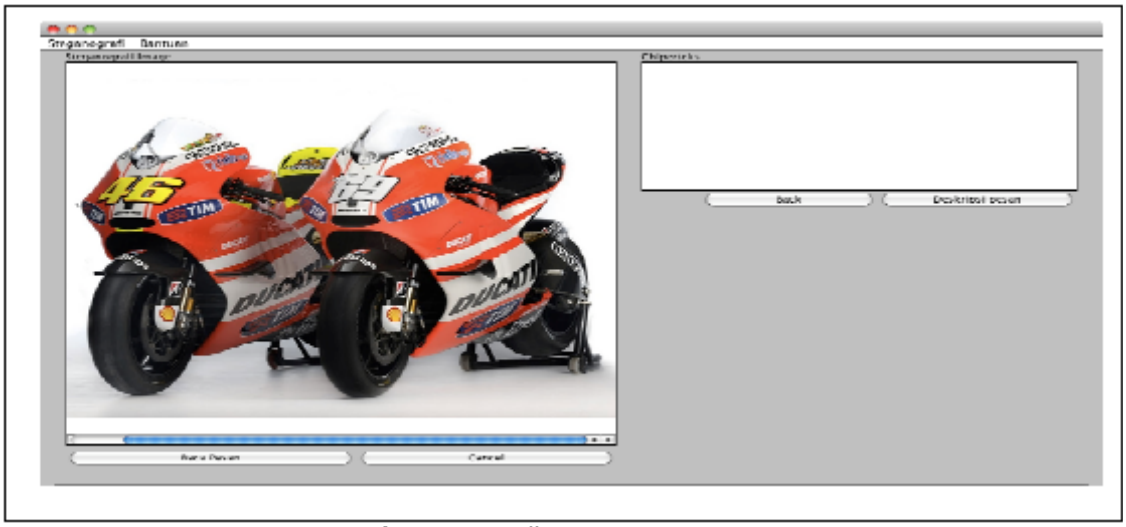
Setelah proses penyisipan selesai, sistem akan menampilkan gambar yang telah disisipkan di dalam *form* sisip pesan.



Gambar 8. Tampilan Setelah Proses Penyisipan

Proses Baca Pesan

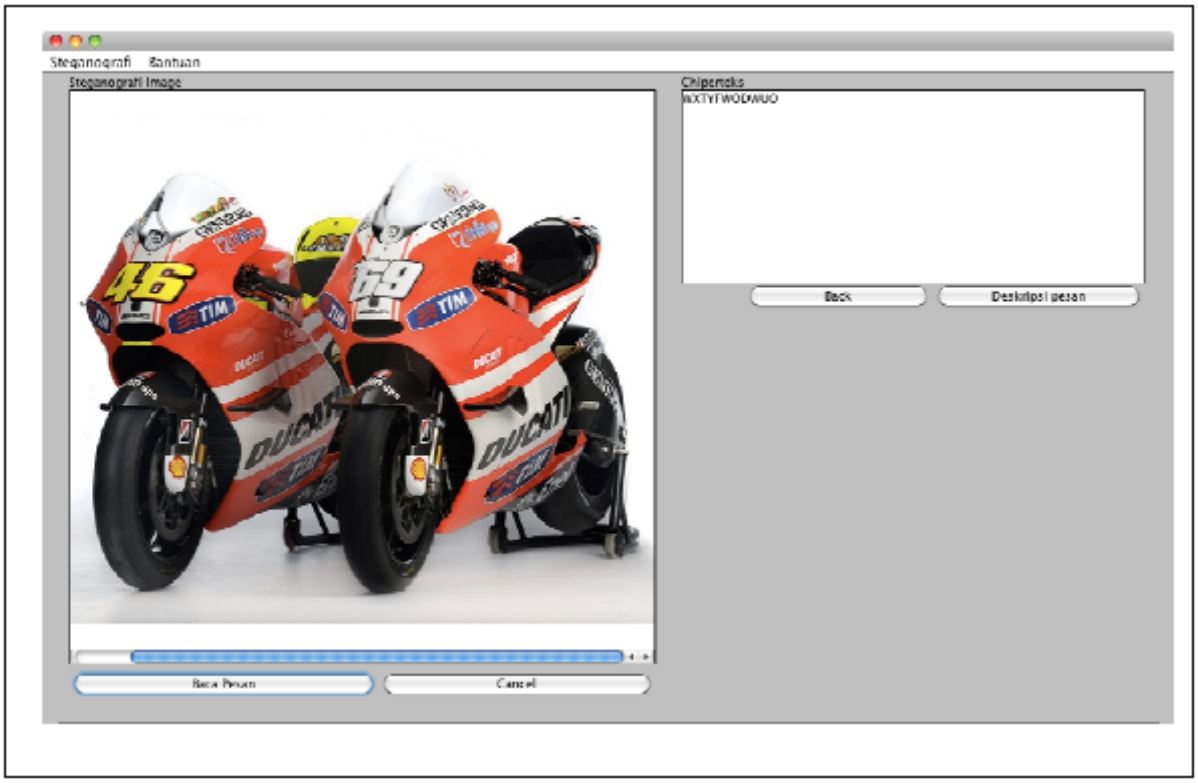
Ketika *user* memilih menu baca pesan maka akan tampil *form* open untuk memilih gambar steganografi. Setelah *user* memilih gambar maka gambar yang dipilih akan ditampilkan di *form* baca pesan.



Gambar 9. Tampilan *Form* Baca Pesan.

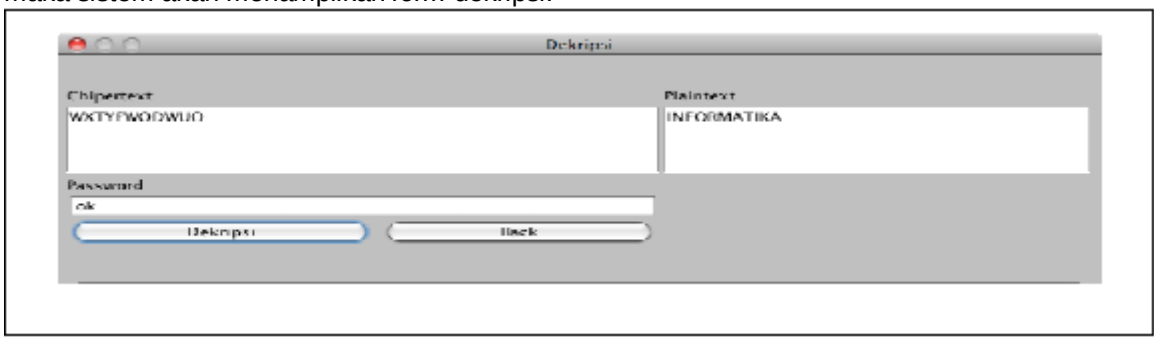
Di dalam *form* baca pesan terdapat empat *button* yaitu *button* baca pesan, cancel, back dan dekripsi Pesan. *Button* baca pesan akan menampilkan pesan yang ada di dalam gambar, jika *button* cancel akan menampilkan *form* open untuk memilih gambar baru, *button* back akan kembali ke tampilan awal dan *button* dekripsi pesan akan menampilkan *form* dekripsi.

Ketika *user* memilih *button* baca pesan maka sistem akan membaca pesan yang ada di dalam gambar dan menampilkannya di *textarea* chiperteks.



Gambar 10. Tampilan Setelah Baca Pesan.

Setelah proses baca pesan selesai maka *user* harus memilih *button* dekripsi pesan agar pesan chiperteks yang ada didekripsikan menjadi pesan plainteks. Ketika *button* dekripsi pesan dipilih maka sistem akan menampilkan *form* dekripsi.



Gambar 11. Tampilan Form Dekripsi.

Pada *form* dekripsi terdapat dua *textarea* yaitu *textarea* chipertext dan *textarea* paintext, satu *textfield* yaitu *textfield* password dan dua *button* yaitu *button* dekripsi dan *button* back. Pada *form* dekripsi, *textarea* chipertext otomatis akan terisi oleh pesan yang ada di *form* baca pesan. Agar pesan chiperteks bisa didekripsikan menjadi plainteks maka *user* harus menginput *password* dan memilih *button* dekripsi.

8. KESIMPULAN DAN SARAN

Kesimpulan yang diperoleh dari analisis perancangan dan implementasi pada penelitian ini adalah:

1. Telah berhasil dibangun sebuah aplikasi steganografi dan kriptografi pada citra dengan menggunakan J2SE dan dapat dijalankan diperangkat komputer.
2. Algoritma yang digunakan dalam aplikasi ini adalah algoritma steganografi *least significant bit (LSB)* dan algoritma kriptografi *vigenere*.
3. Aplikasi ini dapat menyisipkan pesan ke dalam sebuah gambar dengan format jpg, png, gif dan bmp.

Saran

Aplikasi ini dapat dikembangkan oleh peneliti selanjutnya dengan penambahan fasilitas-fasilitas baru seperti:

1. Aplikasi ini dapat dikembangkan dengan menambah algoritma-algoritma lain seperti *playfair cipher, DES, RSA* dan sebagainya.
2. Aplikasi ini tidak hanya dapat menghasilkan *output* png, tetapi dapat menghasilkan *output* dengan format lain.
3. Tampilan *interface* dan *background*-nya dapat dibuat lebih menarik dengan menggunakan *gradient* dan sebagainya.

DAFTAR PUSTAKA

Ariyus, Dony, 2006. *Kriptografi Keamanan Data dan Komunikasi*. Graha Ilmu. Yogyakarta.
Fowler, M., 2005, *UML Distilled Edisi 3 Panduan Singkat Bahasa Pemodelan Objek Standar*, Andi, Yogyakarta.
Hermawan, Benny, 2004, *Menguasai Java 2 & Object Oriented Programming*, Andi Yogyakarta.
Johnson, Neil F. 2001, *Information Hiding Steganography and Watermarking – Attack and Countermeasures, Advanced in Information Security*, Kluwer Academic Publisher, United State.
Kurniawan, Freddy. 2005. *Sistem Digital Konsep dan Aplikasi*. Gava Media. Yogyakarta.
Mohanty, S.P, 1999. *Watermarking of Digital Images*, Indian Institute of Science, india.
Munawar, 2005, *Pemodelan Visual dengan UML*, Graha Ilmu, Yogyakarta
Munir, Rinaldi, 2006. *Kriptografi.* Informatika. Bandung.
Putra, Darma. 2010. *Pengolahan Citra Digital*. Andi. Yogyakarta.
Rosa A dan Shalahuddin M, 2011, *Rekayasa Perangkat Lunak Terstruktur dan Beorientasi Objek*, Modula, Bandung.
Schmuller, Joseph, 1999, *Teach Yourself UML in 24 Hours,* San Publising, Indianapolis.
Sellar, D, 1996, *An Introduction to Steganography*, http://www.cs.utc.ac.za/courses/cs400w/nis/ paper99/ dsellars/stego.htm.
Sukmawan, Budi, 2002 , *Steganografi*, http://www.bimacipta.com/stegano.htm.
http://astah.change-vision.com/en/index.php. (diakses 5 April 2011)
http://ilmukomputer.org/2007/02/27/netbeans-open-source-java-ide-berbasiskan-swing. (diakses 5 April 2011)