Aplikasi Steganografi pada Video dengan Teknik *Least Significant Bit* dan Gabungan Enkripsi *Rivest Chiper 4*

Erwin[1], Gregorius Satia Budhi[2], Rolly Intan[3]
Jurusan Teknik Informatika Fakultas Teknologi Industri Universitas Kristen Petra
Jl. Siwalankerto 121 – 131 Surabaya 60236
Telp. (031) – 2983455, Fax. (031) - 8417658
E-mail: m26409037@john.petra.ac.id[1], greg@petra.ac.id[2], rintan@petra.ac.id[3]

ABSTRAK: Perkembangan teknologi komunikasi dan informasi berkembang dengan pesat. Hal ini mempengaruhi kehidupan manusia dalam proses pertukaran informasi yang semakin mudah. Namun seiring dengan perkembangan tersebut, kejahatan dalam bidang ini pun turut berkembang, sehingga pertukaran informasi menjadi kurang aman.
Aplikasi yang dibuat menggunakan bahasa pemrograman C# dengan bantuan *library* untuk melakukan proses terhadap file video yang digunakan. Untuk menyembunyikan file kedalam video menggunakan teknik *Least Significant Bit* dengan terlebih dahulu menambah keamanan pada file dengan cara melakukan enkripsi *Rivest Chiper 4*.

Hasil akhir dari aplikasi ini adalah sebuah video yang berisikan data rahasia. Untuk mengekstrak data tersebut dibutuhkan *password* yang digunakan saat menyembunyikan data. Kemampuan aplikasi ini terbatas pada kemampuan *library* yang digunakan.

Kata kunci: Steganografi, Least Significant Bit, Rivest Chiper 4

ABSTRACT: *Communication and information technology are growing rapidly. This circumstance affects human life in process of exchanging information which going easier day by day. However simultaneously with this development, the harm is also growing. It's caused exchanging information not safety anymore.*
This application made using C# programming language with library assitance for processing video. To hide file into video, firstly adding security to file with doing Rivest Chiper 4 encryption then using Least Significant Bit method.

Final result of this application is a video which containing hidden file. For extracting hidden file, using similar password which using when hiding file is required.

Keyword:*Steganography, Least Significant Bit, Rivest Chiper 4*

1. PENDAHULUAN

Seiring dengan kemajuan jaman, teknologi komunikasi dan informasi berkembang dengan pesat. Hal ini memiliki pengaruh besar bagi kehidupan manusia, sebagai contoh internet menjadi media pertukaran data. Namun seiring dengan perkembangan yang ada, kejahatan dalam teknologi komunikasi dan informasi juga ikut berkembang.

Ada lima konsep keamanan yang perlu diperhatikan dalam pertukaran informasi, yakni *confidentiality, integrity, availability, authenticity,* dan *non-repudiation* [2]. Kelima konsep keamanan tersebut merupakan sorotan penting dalam perlindungan terhadap pesan atau data yang dikirimkan terhadap interupsi, penyadapan, modifikasi maupun fabrikasi.

Salah satu solusi adalah melalui teknik *hidden message (steganography),* yaitu suatu teknik yang mengijinkan para pengguna untuk menyembunyikan suatu pesan atau data didalam data yang lain. Solusi lainnya adalah menggunakan kriptografi, yaitu suatu ilmu dan seni untuk menjaga keamanan pesan yang dikirim dari suatu tempat ke tempat yang lain.

2. TEORI DASAR

2.1 *Cryptography*

Cryptography berasal dari BahasaYunani: "*cryptós*" artinya "*secret*" (rahasia), sedangkan "*gráphein*" artinya "*writing*" (tulisan). Jadi, kriptografi berarti "*secret writing*" (tulisan rahasia). Ada beberapa definisi kriptografi yang telah dikemukakan di dalam berbagai literatur. Definisi yang dipakai di dalam buku-buku yang lama (sebelum tahun 1980-an) menyatakan bahwa kriptografi adalah ilmu dan seni untuk menjaga kerahasian pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya.

Kriptografi sendiri pembagiannya bermacam – macam. Secara umum berdasarkan kesamaan kuncinya, algoritma sandi dibedakan menjadi :

- kunci-simetris / *symetric-key*, sering disebut juga algoritma sandi konvensional karena umumnya diterapkan pada algoritma sandi klasik. Dalam *symmetric-key* proses enkripsi dan dekripsi dilakukan dengan menggunakan *secret key* yang sama [4].
- kunci-asimetris / *asymetric-key : asymmetric-key cryptography* menggunakan *key* yang berbeda untuk enkripsi dan dekripsi. *Key* ini dinamakan *private key* dan *public key*. *Private key* merupakan *key* yang digunakan untuk dekripsi. Sedangkan *public key* merupakan *key* yang digunakan untuk enkripsi [4].

2.1.1 Rivest Chiper 4

Algoritma kriptografi Rivest Chiper 4 (RC4) merupakan salah satu algoritma kunci simetris dibuat oleh RSA Data Security Inc (RSADSI) yang berbentuk stream chipper. RC4 merupakan salah satu jenis *stream cipher* sehingga RC4 memproses unit atau input data, pesan atau informasi pada satu saat. Unit atau data pada umumnya sebuah byte atau bahkan kadang kadang bit (byte dalam hal RC4) sehingga dengan cara ini enkripsi atau dekripsi dapat dilaksanakan pada panjang yang variabel.

Cara kerja algoritma RC4 yaitu inisialisasi S-Box pertama, S[0],S[1],….S[255], dengan bilangan 0 sampai 255. Pertama isi secara berurutan S[0] = 0, S[1] = 1,….,S[255] = 255. Kemudian inisialisasi array lain (S-Box lain), misal array K dengan panjang 256. Isi array K dengan kunci yang diulangi sampai seluruh array K[0],K[1],…,K[255] terisi seluruhnya. Proses selanjutnya adalah pengacakan terhadap S-Box pertama kemudian dilanjutkan dengan *pseudo-random* untuk menentukan nilai variable yang akan di-XOR-kan dengan *plaintext* ataupun *ciphertext* [1].

2.2 *Steganography*

Steganography adalah seni dan sains untuk menulis pesan tersembunyi dengan cara tertentu sehingga tidak seorangpun selain pengirim dan penerima akan menyadari ada sebuah pesan tersembunyi. Singkatnya, kriptografi mengacak arti dari sebuah pesan, tetapi kriptografi tidak menutupi kenyataan bahwa ada sebuah pesan dalam media tertentu. Kini, menyembunyikan informasi digital di dalam *computer file* masuk ke dalam ruang lingkup steganografi.

Secara umum, sebuah pesan steganografi akan dikenal sebagai sesuatu yang lain, sebuah gambar, sebuah artikel, sebuah daftar belanja, atau bentuk – bentuk pesan yang lain. Bentuk – bentuk pesan ini dikenal dengan nama *cover text*.

Keuntungan steganografi jika dibandingkan dengan kriptografi adalah bahwa pesan – pesan tidak menarik perhatian terhadap pesan itu sendiri, terhadap pengirim, atau terhadap penerima. Sebuah pesan kode yang tidak tersembunyi, tidak peduli serumit apapun pesan tersebut diacak, akan menimbulkan kecurigaan dan keterbatasan karena di beberapa negara kriptografi dicap illegal. Seringkali, steganografi dan kriptografi digunakan secara bersama – sama untuk memastikan keamanan dari sebuah pesan.

2.2.1 *Least Significant Bit*

Least Significant Bit adalah bit yang memiliki nilai terendah dalam barisan biner. Sedangkan bit yang memiliki nilai tertinggi disebut *Most Significant Bit* seperti yang terlihat pada Figure 1.

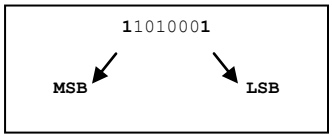

Figure 1. MSB dan LSB

Pada file biasanya terdapat bit-bit LSB yang perannya tidak terlalu penting dan dapat diganti dengan informasi lain tanpa merusak file tersebut. Karena memanfaatkan bit-bit LSB, metode ini tidak digunakan pada media yang mengalami kompresi terutama jenis *lossy compression* karena akan menghilangkan bit-bit LSB tersebut [5]. Penggunaan metode LSB umumnya tidak mengubah ukuran file dan bekerja dengan baik pada file gambar/audio yang memiliki resolusi/bit rate tinggi.
Pada penyisipan pesan dalam berkas bitmap 24-bit, terdapat 3 bit LSB yang dapat kita manfaat dari setiap *pixel* yaitu komponen *Red*, *Green*, dan *Blue*. Pesan yang akan disisipkan cenderung mempunyai panjang yang dinamis. Oleh karena kita membutuhkan sebuah *header* untuk menyimpan panjang pesan yang disisipkan [3].

3. IMPLEMENTASI

3.1 *Steganography*

Proses *steganography* pada *flowchart* pada Figure 2. terlihat adanya proses *next frame*. *Next frame* adalah untuk mendapatkan *frame* yang akan disisipi file *byte* dengan metode *Least Significant Bit* pada *pixel* yang terpilih secara acak dengan *pseudo-random*.

Tujuan penggunaan *pseudo-random* adalah agar ketika pengambilan kembali data tersebut pengacakan pada *pixel* dapat terulang sehingga memungkinkan data yang disembunyikan didapatkan kembali. Byteperframe yang digunakan dalam proses steganografi merupakan batasan sejumlah *byte* yang dapat di-*input*-kan dalam sebuah *frame*. Untuk *message bytes* merupakan isi dari file *message* yang disisipkan dengan dirubah menjadi bentuk *byte*.

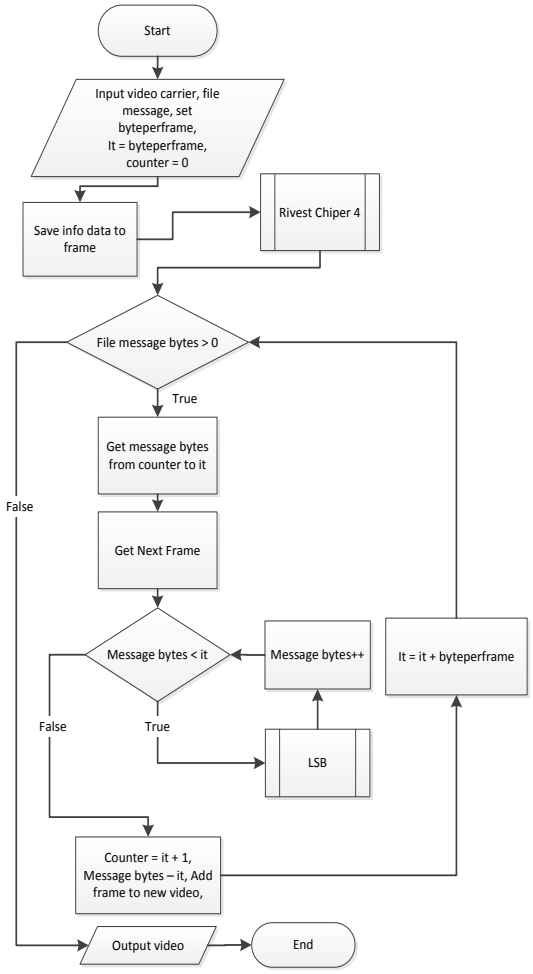

Figure 2. *Steganography Flowchart*

3.1.1 *Encrypt and Decrypt Rivest Chiper 4*

Proses ini merupakan proses untuk memperkuat sekuritas dari data yang akan disembunyikan. Data tersebut akan dienkripsi dengan metode RC4 dengan harapan semakin sulit data tersebut apabila media penyimpanan jatuh ditangan pihak yang tidak diharapkan.
Dapat dilihat dari *flowchart* pada Figure 3. bahwa terdapat 3 hal penting dalam proses enkripsi dengan metode *Rivest Chiper 4*.

Yakni membuat Sbox array sepanjang 256 data, melakukan pengacakan Sbox serta menggunakan Sbox tersebut untuk melakukan *pseudo random* terhadap *byte* data yang akan dienkripsi. Proses ini merupakan bagian dari awal proses keseluruhan steganografi yang berfungsi merusak file yang disisipkan agar mempersulit pihak lain untuk mengambil file tersebut. Kerusakan file tersebut dapat dikembalikan dengan semula dengan menggunakan *password* yang sama ketika melakukan enkripsi pada file yang disisipkan.

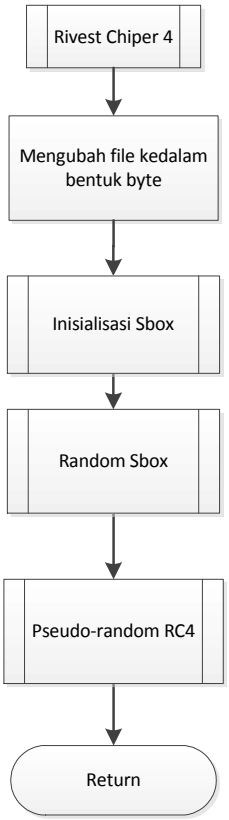

Figure 3. ***Rivest Code 4 Flowchart***

3.1.2 *Least Significant Bit* (LSB)

Proses *Least Significant Bit* adalah proses dimana *byte* yang akan disisipkan pada *frame* diubah menjadi bentuk *bit*-nya dan disisipkan pada beberapa *byte* warna pada *pixel*. Tiap – tiap *pixel* yang terpilih akan diambil RGB dan disisipkan pada *bit* terakhir pada warna tersebut. Data yang dibutuhkan untuk memulai proses steganografi adalah adanya *frame* berbentuk *bitmap* yang akan disisipi serta sebagian *byte* dari *message* yang akan disisipkan pada *frame* tersebut. Untuk menentukan posisi dari *pixel* dilakukan suatu proses *pseudo-random*.

Seperti yang terlihat pada Figure 4 proses LSB (*Least Significant Bit*) membutuhkan *bitmap* yang didapat dari mengambil *frame* pada *video*, *byte* yang akan disisipkan serta posisi *pixel* tempat disisipkan yang didapatkan dari proses *pseudo-random*. Dari *pixel* yang terpilih diekstrak *value red*, *green* dan *blue*. Kemudian *byte* dipecah menjadi bentuk *bit*. Dan disisipkan pada *bit* terakhir dari tiap – tiap warna dan dilakukan perubahan *byte* warna pada *pixel* tersebut. Kemudian proses dari steganografi dilanjutkan dengan mendapatkan *pixel* selanjutnya untuk disisipi lagi oleh *bit* dari *message* berikutnya hingga *byte message* tersisipkan semua pada *frame*.

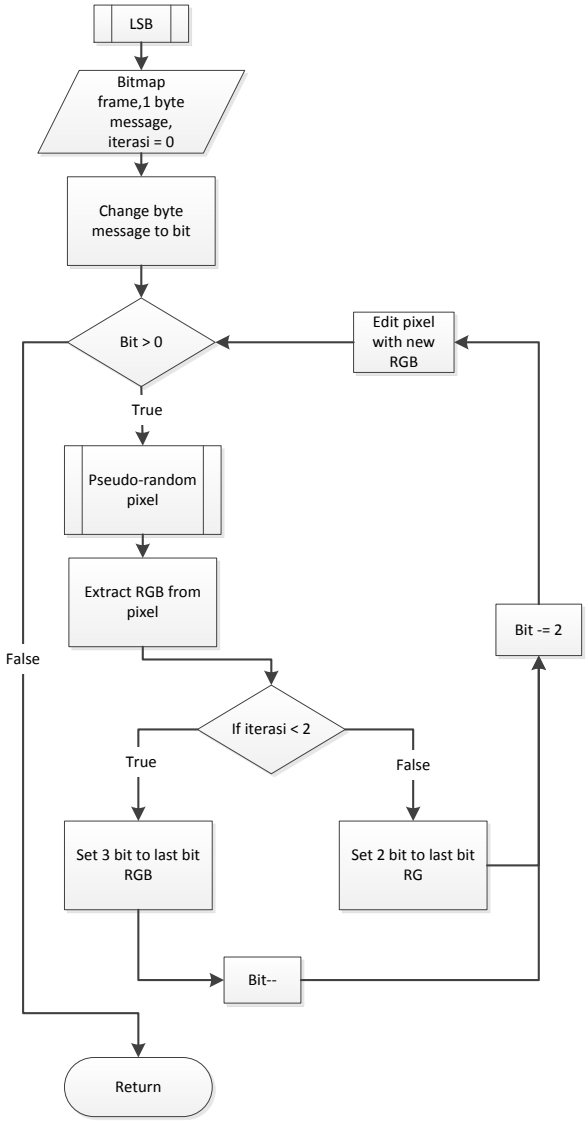

Figure 4. LSB ***Flowchart***

3.2 *Extracting Data*

Untuk proses *extracting data* sebenarnya hanya mengulang proses steganografi dengan beberapa perubahan, yakni yang semula disisipkan menjadi pengambilan data serta tidak membuat *video* baru melainkan membuat file dari susunan *byte* yang didapatkan dari antara *frame*.

Pada Figure 5 terdapat proses LSB dan *Rivest Chiper 4*, yang dilakukan dalam proses ini adalah sama seperti proses yang dilakukan ketika penyembunyian file.

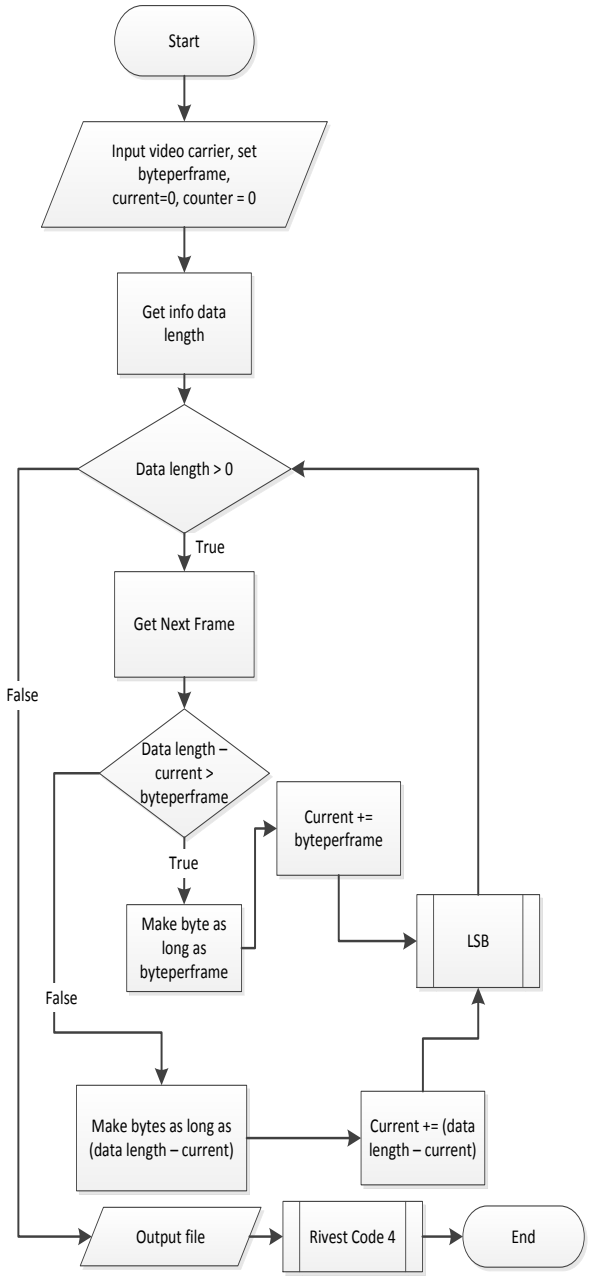

Figure 5. ***Extracting Flowchart***

Hanya saja pada proses LSB *extracting* yang dilakukan adalah mengambil tiap – tiap *bit* dari warna *pixel* yang terpilih secara *pseudo-random* dan mengumpulkannya hingga menjadi sebuah *byte*. Sedangkan pada proses *Rivest Chiper* fungsi yang dilakukan sama karena merupakan jenis *stream cipher*. Sehingga ketika *ciphertext* di-XOR-kan pada penghitungan *Rivest Chiper* tersebut akan menghasilkan *plaintext* dengan kata lain file yang dienkripsi dengan *Rivest Chiper 4* sebelumnya akan dikembalikan dalam bentuk semula.

4. PENGUJIAN

4.1 Pengujian ukuran file

Untuk video yang dihasilkan memiliki *size* yang cukup berbeda dikarenakan pembuatan video ulang menggunakan *standard* AVI yang cukup besar dalam segi ukuran file dan juga menggunakan file gambar *bitmap* untuk mengisikan *frame* dari video baru. Hal ini mengakibatkan perubahan ukuran yang besar pada file video yang disisipi file *message* disamping adanya penambahan *hidden* file kedalam video.

4.2 Pengujian hasil video

Walaupun dari segi ukuran file cukup berbeda drastis namun dari segi tampilan video tidak berubah ketika menggunakan video dengan panjang *frame* 772 seperti yang terlihat pada Figure 6.

Figure 6. Pengujian Video 1

Percobaan berikutnya dilakukan pada video yang lebih besar dalam jumlah *frame*-nya sebanyak 1673 *frame*. Pada hasil dari video ini hal yang terjadi sama dalam peningkatan file. Namun terjadi peningkatan yang lebih besar dikarenakan dari resolusi gambar dari video yang lebih besar dari sebelumnya.

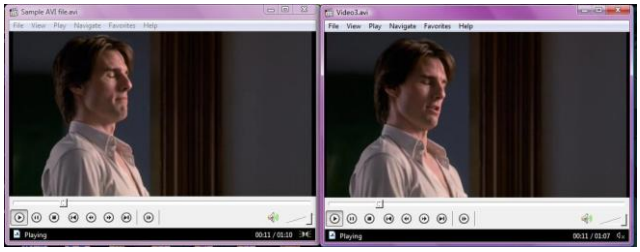

Figure 7. Pengujian Video 2

Dari Figure 7 di atas terlihat bahwa adanya perbedaan pada *frame*. Hal ini dikarenakan olek penggunaan *library* yang menggunakan standard fps tertentu sehingga mengakibatkan adanya kesalahan. Permasalahan yang muncul adalah perhitungan fps yang salah mengakibatkan adanya pertambahan pada *frame*. Namun hasil video tersebut berakhir pada *frame* yang sama. Untuk *hidden* file pun dapat diekstrak dengan sempurna.

Kesimpulan dari percobaan terhadap tiga video yang berbeda baik dalam hal jumlah *frame* maupun ukuran dari gambar per *frame* menghasilkan hasil yang berbeda dan dapat dilihat pada Tabel 1.

Tabel 1. Perbandingan Video

Frame	Size	New Frame	New Size	Extract
772	2.90 Mb	774	170 Mb	Berhasil
1673	12.6 Mb	1674	1.05 Gb	Berhasil
7478	19.1 Mb	6361	1.36 Gb	Berhasil

4.3 Efisiensi byteperframe

Untuk menghitung efisiensi terbaik untuk batasan dari *byteperframe* menggunakan perhitungan PSNR (*Peak Signal to Signal Noise Ratio*) perhitungan ini membandingkan dua citra dan dihitung perbedaan dalam *noise*- nya. Batas aman dari hasil perhitungan PSNR adalah berkisar 40 dB, dan mendapatkan hasil lebih baik lagi apabila diatas 40 dB. Pengujian pertama dengan menyembunyikan pada *frame* sebanyak 30% dari *dimension frame*

Figure 8. Frame Original

Figure 9. Frame 30% hidden file

Dari gambar asli (Figure 8) dan gambar dengan *hidden* file (Figure 9) apabila dilakukan penghitungan PSNR didapatkan hasil 41.6555 dB yang berarti merupakan ukuran yang baik untuk penyembunyian *hidden* file dikarenakan batas dari perhitungan PSNR yakni 40 dB atau lebih besar. Batas teratas dari percobaan efisiensi adalah pada 32% yakni mendapatkan hasil sebesar 41.6268 dB. Percobaan untuk efisiensi lebih daripada 32% mendapatkan permasalahan, yaitu lamanya proses dalam mendapatkan *pixel* dengan fungsi *pseudo-random* untuk mengisi sebanyak 32% dari bagian *frame*.

5. KESIMPULAN DAN SARAN

Berdasarkan hasil pengujian maka dapat diambil beberapa kesimpulan sebagai berikut:

- Aplikasi steganografi ini berhasil dalam menyembunyikan file maupun melakukan ekstrasi file. Hanya saja dibutuhkan *codec* yang sesuai agar proses yang berhubungan dengan video dan audio dapat berjalan lancar.
- Karena metode steganografi yang digunakan adalah *Least Significant Bit* (LSB) maka file video (*carrier)* harus dikembalikan ke bentuk tampa kompresi. Hal ini mengakibatkan file *carrier* menjadi lebih besar dibandingkan file aslinya.
- Semakin banyak *byte per frame* dari file video yang digunakan sebagai *carrier*, maka semakin lama pula waktu yang diperlukan untuk proses steganografi.

Setiap aplikasi pasti memiliki kekurangan, aplikasi steganografi ini pun tidak terlepas dari kekurangan. Beberapa saran yang berkaitan dengan pengembangan aplikasi steganografi :

- Membutuhkan metode steganografi lain, sebab hasil video dengan menggunakan metode ini tidak dapat di *compressi* akibat dari adanya file data yang disembunyikan pada LSB dari warna pada *pixel*.
- Dalam pengerjaan aplikasi steganografi pada file video perlu diperhatikan *codec* dari komputer yang dipakai. Beberapa *codec* mempengaruhi proses kerja pembuatan video.

6. REFERENCES

[1] Ariyus, Dony. (2006). Pengantar Ilmu Kriptografi. Bandung: Informatika.

[2] Cole, Eric. (2003). *Hiding in Plain Sight.* New York: John Willey & Sons, Inc Publishers.

[3] Katzenberisser, Stefan., & Petitcolas, Fabien. A. (2000). *Information Hiding Techniques for Steganography and Digital Watermaking.* London: Artech House.

[4] Scheneier, Bruce. (1996). *Applied Cryptographic Second Edition.* New Jersey: John Willey & Sons, Inc Publishers.

[5] Wayner, Peter. (2009). *Disappearing Crytography: Information Hiding Steganography & Watermaking Third Edition.* USA: Morgan Kaufmann