

Nama : Andi Kurniawan
NIM : 1410652011
URL : https://aksara.pcr.ac.id/page/read_pdf.php?name=paper-aes.pdf&id=43

Implementasi Steganografi untuk Penyembunyian Pesan pada Video dengan Metode LSB

1 PENDAHULUAN

Kemajuan teknologi memungkinkan kita untuk berkomunikasi dan bertukar informasi melalui media digital. Namun penggunaan media digital meningkatkan resiko penyadapan terhadap informasi/pesan. Hal ini merupakan permasalahan penting dalam menjaga kerahasiaan informasi/pesan.

Salah satu cara untuk mencegah penyadapan informasi/pesan adalah dengan melakukan penyandian yang dikenal dengan proses enkripsi. Dalam proses enkripsi, pesan/informasi akan diubah menggunakan algoritma tertentu untuk menyembunyikan makna sebenarnya. Ada berbagai algoritma yang digunakan dalam proses enkripsi, salah satunya adalah algoritma Blowfish.

Algoritma Blowfish merupakan algoritma kunci privat yaitu menggunakan kunci yang sama untuk mengacak dan mengembalikan data. Blowfish memiliki keunggulan dari segi kecepatan dan memori yang digunakan. Algoritma ini diciptakan oleh Bruce Schneier dan hingga saat ini belum ada metode yang efektif untuk memecahkan algoritma ini[1]. Akan tetapi, pengamanan pesan dengan menggunakan enkripsi memiliki kelemahan yaitu menghasilkan kumpulan karakter/symbol acak yang tidak memiliki makna sehingga dapat menimbulkan kecurigaan.

Cara lain untuk mengamankan pesan adalah dengan steganografi. Steganografi merupakan teknik penyembunyian pesan dalam suatu media sehingga orang lain tidak menyadari keberadaan pesan tersebut. Teknik ini telah digunakan sejak sebelum Masehi dan seiring perkembangan teknologi, steganografi telah bergeser ke arah media digital, seperti gambar, audio dan video digital. Ada banyak metode yang digunakan dalam bidang steganografi, salah satunya adalah modifikasi Least Significant Bit (LSB).

Least Significant Bit adalah metode penyisipan pesan dengan cara mengganti bit-bit LSB dari media penyamaran dengan bit-bit dari pesan rahasia. Metode ini menghasilkan perubahan yang tidak signifikan terhadap media penyamaran namun membutuhkan media yang besar[2].

Media penyamaran yang digunakan dalam penelitian ini adalah video digital dengan format AVI. Format AVI memiliki resolusi gambar yang tinggi dan mendukung jenis data yang tidak terkompresi. Dengan menggunakan media video diharapkan keamanan pesan dapat ditingkatkan dan memungkinkan kita untuk menyisipkan lebih banyak data. Dalam penelitian ini juga akan digunakan proses enkripsi dengan algoritma Blowfish untuk meningkatkan pengamanan pesan.

2 DASAR TEORI

2.1 *Steganografi*

Steganografi berasal dari bahasa Yunani yaitu *stegos* yang berarti penyamaran dan *graphia* yang berarti tulisan. Steganografi digunakan untuk menyembunyikan informasi rahasia ke dalam suatu media sehingga keberadaan pesan tersebut tidak diketahui oleh orang lain. Berbeda dengan kriptografi yang melakukan perubahan terhadap pesan, steganografi bertujuan untuk menghilangkan kecurigaan dengan cara menyamarkan pesan tersebut. Steganografi juga dapat dikombinasikan dengan kriptografi untuk menghasilkan perlindungan yang lebih baik bagi pesan[2].

2.2 *Kriteria Steganografi*

Dalam steganografi, penyembunyian data harus memperhatikan kriteria-kriteria tertentu[3], antara lain:

1. *Fidelity*

Kualitas media penampung setelah disisipkan pesan tidak boleh berbeda jauh dari kualitas aslinya. Perubahan yang terjadi tidak dapat dideteksi oleh indera manusia.

2. *Robustness*

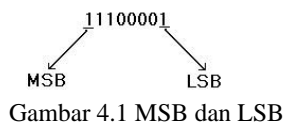
Data yang disembunyikan harus tahan terhadap perubahan yang dilakukan pada media penampung.

3. *Recovery*

Tidak hanya melakukan penyembunyian data saja, data yang telah disembunyikan juga harus dapat diambil kembali.

2.3 *Metode 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*.



Gambar 4.1 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. Penggunaan metode LSB umumnya tidak mengubah ukuran file dan bekerja dengan baik pada file gambar/audio yang memiliki resolusi/bit rate tinggi[3].

Dalam gambar digital, penggantian bit-bit LSB pada warna akan menyebabkan perubahan sebesar satu angka dari nilai sebelumnya. Perubahan ini tidak menimbulkan perubahan warna yang signifikan sehingga mata manusia sulit untuk mendeteksi perubahan yang terjadi. Metode LSB bekerja dengan memanfaatkan keterbatasan indera penglihatan manusia yang kurang peka terhadap perubahan warna tersebut.

Pada penyisipan pesan dalam berkas bitmap 24-bit, terdapat 3 bit LSB yang dapat kita manfaatkan 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. Misal: panjang pesan maksimal yang akan disisipkan adalah 255 karakter, maka kita membutuhkan *header* berukuran 1 byte atau 8 bit. Jadi, untuk menyisipkan huruf 'A' kita membutuhkan 3 piksel tambahan. Panjang pesan yaitu 1 karakter (biner: 00000001). Total piksel yang dibutuhkan adalah 6 piksel. Piksel asli:

00100101	11101000	11001001
00100011	11001010	11101011
11001010	00100110	11101001
00100111	11101001	11001000
00100111	11001000	11101001
11001000	00100111	11101001

Piksel setelah penyisipan:

(ket: delapan bit pertama merupakan data *header*)

00100100	11101000	11001000
00100010	11001010	11101010
11001010	00100111	11101000
00100111	11101000	11001000
00100110	11001000	11101000
11001001	00100111	11101001

Dari contoh terlihat bahwa tidak semua nilai warna dari pixel mengalami perubahan. Dan untuk nilai warna yang mengalami perubahan hanya memiliki perbedaan sebesar satu angka dari nilai aslinya. Namun, penggunaan metode LSB membutuhkan media penyamaran yang cukup besar dalam menyembunyikan pesan[2].

2.4 Kriptografi

Kriptografi berasal dari bahasa Yunani yaitu *cryptos* yang berarti rahasia dan *graphein* yang berarti tulisan. Menurut Bruce Schneier[4], kriptografi dapat diartikan sebagai ilmu pengetahuan dan seni untuk menjaga keamanan pesan. Dalam penelitian ini kriptografi akan digunakan sebagai pengganti kunci pada proses steganografi sekaligus memberikan perlindungan tambahan terhadap data yang disembunyikan.

Kriptografi terdiri dari dua proses utama yaitu enkripsi dan dekripsi. Berdasarkan kunci yang digunakan dalam proses enkripsi dan dekripsi, terdapat dua jenis algoritma kriptografi yaitu kunci asimetri dan simetri. Algoritma kunci simetri terbagi ke dalam dua kategori[5], yaitu:

1. Stream cipher

Algoritma ini mengolah data dalam bentuk bit-bit tunggal. Proses enkripsi/dekripsi akan dilakukan pada satu bit dalam satu waktu.

2. Block cipher

Algoritma ini memecah data yang masuk menjadi beberapa blok dengan panjang data tertentu. Proses enkripsi/dekripsi akan dilakukan pada satu blok data dalam satu waktu.

2.5 Blowfish

Algoritma Blowfish merupakan algoritma kunci simetri *block-cipher* yang dirancang oleh Bruce Schneier pada tahun 1993. Algoritma ini dipublikasikan dengan status *license free* dan ditujukan untuk komputer yang mempunyai microprosesor besar (32-bit keatas dengan cache data yang besar) [1].

Perancangan algoritma Blowfish diharapkan memiliki kriteria berikut[4]:

1. Cepat, Blowfish melakukan enkripsi data pada microprocessors 32-bit dengan rate 26 *clock cycles per byte*.
2. *Compact*, dapat dijalankan pada memori kurang dari 5K.

3. Sederhana, Blowfish hanya menggunakan operasi-operasi sederhana seperti: penambahan, XOR, dan lookup tabel pada operan 32-bit.
4. Memiliki tingkat keamanan yang bervariasi, panjang kunci yang digunakan oleh Blowfish dapat bervariasi (32 - 448 bit).

Algoritma Blowfish terdiri dari dua bagian, yaitu:

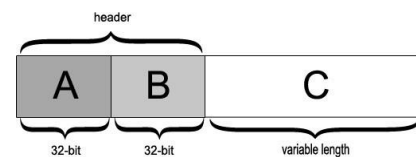
1. Ekspansi kunci
Ekspansi kunci berfungsi untuk merubah kunci (maksimum 448-bit) menjadi beberapa array subkunci dengan total 4168 *byte*.
2. Enkripsi dan dekripsi data
Terdiri dari 16 kali putaran, dengan masukan berupa 64-bit elemen data *x*. Operasi yang digunakan adalah operasi penambahan dan XOR pada variabel 32-bit serta empat penelusuran tabel *array* berindeks untuk setiap putaran.

2.6 AVI

AVI merupakan singkatan dari *Audio Video Interleave*. Format AVI diperkenalkan oleh Microsoft pada November 1992 sebagai bagian dari proyek *video for windows*. Format AVI menyesuaikan dengan spesifikasi *Resource Interchange File Format (RIFF)* yang dikeluarkan oleh *Microsoft*[6]. Data video/audio dalam file AVI dapat dikompresi ke dalam berbagai format kompresi, seperti Full Frame (Uncompressed), Intel Real Time (Indeo), Cinepak, Motion JPEG, VDOWave, ClearVideo/RealVideo dan sebagainya.

File AVI terdiri dari beberapa bagian utama yaitu *AVI Main Header*, *Stream Header*, *Stream Format*, *Stream Format* dan *Stream Data*. File AVI dimulai dengan *AVI Main Header* yang mengandung informasi umum mengenai file AVI, seperti: kecepatan maksimal data perdetik, penanda file AVI (misal: HASINDEX, ISINTERLEAVED, COPYRIGHTED), jumlah *stream* pada file, ukuran *buffer* yang digunakan untuk menyimpan data file AVI pada memory, tinggi dan lebar dari sekuensial file AVI, skala waktu yang digunakan pada keseluruhan file AVI dan durasi file AVI.

Stream Header berisi informasi mengenai *stream* dari file AVI seperti tipe *stream* (*stream video* atau *stream audio*), *handler* yang menangani kompresi file, penanda *stream*, prioritas, bahasa yang digunakan oleh *stream*, skala waktu yang digunakan, jumlah sampel *stream*, posisi *frame* awal, durasi *stream*, ukuran *buffer* yang digunakan untuk menyimpan data pada memori, kualitas data pada *stream*, ukuran sebuah sampel data pada *stream* serta



Gambar 3.1 Struktur penyimpanan bit pesan

Struktur penyimpanan pesan terdiri dari 3 blok yaitu blok A dan B yang masing-masing berukuran 32-bit (4 *byte*) dan blok C yang memiliki ukuran tidak tetap. Blok A dan B merupakan blok *header* yang masing-masing berisi informasi mengenai ukuran/panjang pesan dan informasi jenis/tipe pesan yang disisipkan. Sementara blok C merupakan bit-bit dari pesan itu sendiri.

3.2 Perhitungan Kapasitas Penyimpanan Maksimum

Besar ukuran pesan yang dapat disimpan pada video ditentukan dari resolusi video, jumlah frame dan jumlah elemen warna yang digunakan dari setiap piksel. Pada aplikasi ini hanya satu elemen warna dari setiap piksel yang digunakan sehingga setiap piksel akan menyimpan satu bit data. Kapasitas penyimpanan dari setiap video dihitung dengan menggunakan rumus:

$$\text{Kapasitas Video} = (\text{Resolusi Video} * \text{Jumlah Frame}) / 8 - \text{header}$$

(persamaan 1)

Keterangan:

Resolusi video = *frame width* * *frame height*

Jumlah frame = *frame rate per second* * durasi video
(dalam detik)

Header = panjang blok *header* (64-bit) = 8 *byte*

Sebagai contoh video berdurasi 30 detik dengan *frame rate* 25 *frame per second* dan memiliki resolusi 600 * 400 akan mempunyai daya tampung pesan sebesar:

Resolusi video = $600 * 400 = 240000$
 Jumlah frame = $25 * 30 = 750$
 Kapasitas video = $(240000 * 750) / 8 - 8$
 = $180000000 / 8 - 8$
 = $22500000 - 8$
 = 22499992 bytes

3.3 Use Case Diagram

Penggunaan *Use Case Diagram* dimaksudkan untuk mendeskripsikan fungsi/aktivitas yang terdapat pada sistem.

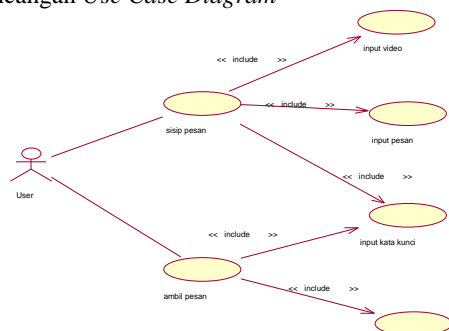
dimensi *frame*.

Stream Format berisi informasi mengenai format dari data yang terkandung pada *stream* sedangkan *Stream Data* merupakan data dari *stream* itu sendiri.

3 PERANCANGAN DAN PEMBAHASAN

3.1 Struktur Penyimpanan Bit-Bit Pesan

Agar pesan yang disisipkan ke dalam video dapat diambil kembali dengan benar, dibutuhkan sebuah ketentuan dalam struktur penyimpanan pesan tersebut. Dalam pembuatan aplikasi ini, digunakan struktur penyimpanan input video stage yang dapat dilihat pada Gambar 3.1. Gambar 3.2 Perancangan *Use Case Diagram*

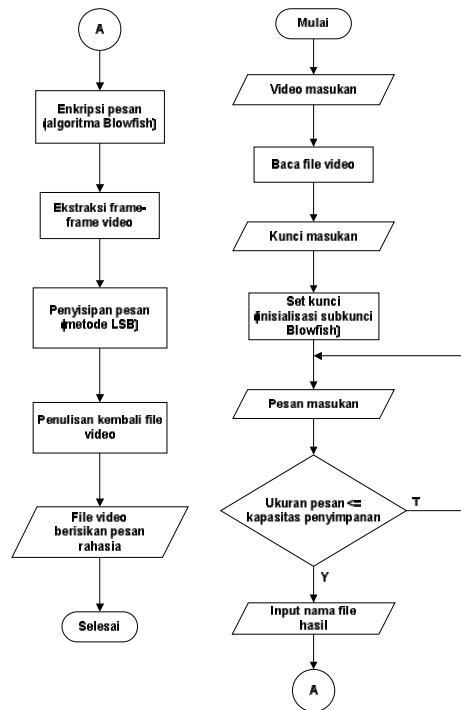


Dalam *Use Case Diagram* di atas dapat dilihat bahwa user dapat melakukan penyisipan pesan, memilih file video yang akan digunakan, kata kunci dan file pesan yang akan disisipkan. Pada proses pengambilan pesan, user dapat memasukkan kata kunci dan file video yang berisi pesan rahasia.

3.4 Flow Chart

Flow chart merupakan suatu metode yang digunakan untuk menggambarkan alur suatu program secara lebih mudah dan sederhana.

a. Flow Chart Penyisipan Pesan

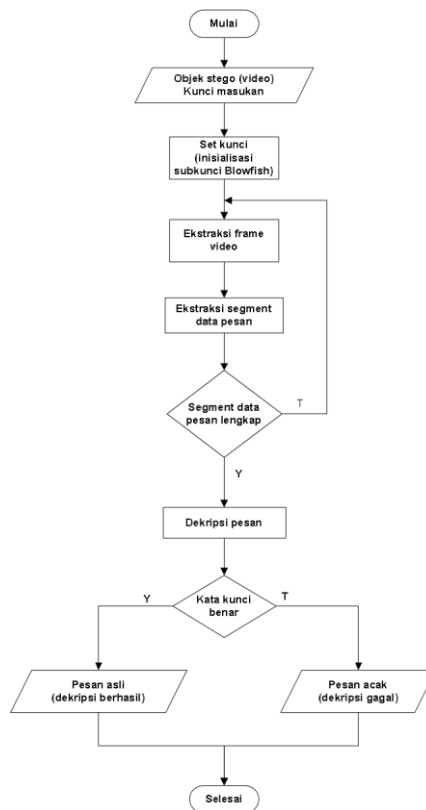


Gambar 3.3 *Flow chart* penyisipan pesan

Dalam proses ini dibutuhkan data masukan berupa file video, file pesan, dan kunci enkripsi yang akan digunakan. Sistem akan melakukan pengecekan kapasitas video dan ukuran pesan yang disisipkan. Jika ukuran pesan melebihi kapasitas video, maka user akan diminta untuk memilih file pesan baru. Jika kapasitas video mencukupi untuk penyisipan pesan, maka sistem akan melanjutkan dengan melakukan proses enkripsi pada pesan dan penyisipan pesan ke dalam frame-frame video.

b. *Flow Chart* Pengambilan Pesan

Dalam proses pengambilan pesan dibutuhkan data masukan berupa file video yang mengandung pesan rahasia dan kata kunci untuk dekripsi pesan. Pada proses ini sistem akan mengambil bagian-bagian pesan dari frame-frame video. Proses ekstraksi frame video tidak dilakukan untuk keseluruhan frame video, namun hanya pada frame video yang mengandung data pesan. Setelah pengambilan data pesan selesai, dilanjutkan dengan proses dekripsi pada pesan. Jika kata kunci yang digunakan benar maka didapatkan pesan asli. Sebaliknya, jika kata kunci yang digunakan salah, maka didapatkan pesan acak.



Gambar 3.4 *Flowchart* pengambilan pesan


4 HASIL DAN DISKUSI

Pengujian dilakukan dengan beberapa cara yaitu pengujian sistem, perbandingan kapasitas penyimpanan dan ukuran video, pengujian perubahan ukuran video, pengujian lama waktu komputasi, dan pengujian perubahan kualitas video.

a. Pengujian sistem


Pengujian dilakukan dengan menyisipkan pesan ke dalam video dan mengekstraksi kembali pesan tersebut. File pesan yang disisipkan dapat dilihat pada tabel 4.1

Tabel 4.1 Isi file pesan

Jenis File	Isi
txt	Tes penyesipan pesan dalam bentuk file txt
docx	Tes aplikasi steganografi™ 

Kemudian dilakukan pengambilan pesan dengan menggunakan kunci yang benar. Hasil pengambilan pesan dapat dilihat pada tabel 4.2.

Tabel 4.2 Pengambilan pesan dengan kunci benar

Jenis File	Isi
txt	Tes penyisipan pesan dalam bentuk file txt
docx	Tes aplikasi steganografi™ 

Dilakukan pengambilan pesan sekali lagi dengan menggunakan kunci yang berbeda. Hasil dapat dilihat pada tabel 4.3.

Tabel 4.3 Pengambilan pesan dengan kunci berbeda

Jenis File	Isi
txt	ðör;P 9éY8_È{7n_äþrY_r:öw~¼c_6úm_ w_ÿ_ØÑéeä£âRÍ ú
docx	File tidak dapat dibuka

Dari hasil pengujian terbukti bahwa aplikasi telah berjalan dengan baik. Proses pengambilan dengan kunci yang salah dapat ditangani, yaitu dengan menghasilkan pesan yang berbeda dari aslinya.

b. Perbandingan kapasitas dan ukuran video

Pengujian ini bertujuan untuk mengetahui rasio perbandingan antara kapasitas pesan yang dapat ditampung dan ukuran file video. Pengujian dilakukan dengan menghitung persentase kapasitas penyimpanan terhadap ukuran video.

Tabel 4.4 Perbandingan kapasitas dan ukuran video

Ukuran video (bytes)	Kapasitas video (bytes)	Persentase (%)
112.350.132	4.607.992	4,101456685
225.114.148	9.235.192	4,102448505
344.135.560	14.111.992	4,100707291
455.098.484	18.662.392	4,100737018
567.466.004	23.270.392	4,100755259

Dari hasil pengujian didapatkan bahwa jumlah pesan yang dapat disimpan bertambah sesuai dengan bertambahnya ukuran video dengan persentase yang relatif sama. Persentase ukuran pesan yang dapat disimpan pada video adalah sekitar 4,1% dari ukuran file video.

c. Pengujian perubahan ukuran video Pengujian dilakukan dengan melakukan penyisipan beberapa file pesan dengan ukuran berbeda ke dalam sebuah file video.

Tabel 4.5 Hasil pengujian ukuran video

Ukuran video (bytes)	Jenis Pesan	Ukuran file pesan (bytes)	Ukuran video hasil (bytes)
481.234.576	docx	1.151.544	481.234.576
	doc	2.303.032	481.234.576
	pdf	3.454.568	481.234.576
	docx	4.607.984	481.234.576

Dari pengujian terlihat bahwa ukuran file pesan yang disisipkan tidak mempengaruhi ukuran dari file video. Hal ini mendukung teori LSB bahwa bit-bit dari video hanya digantikan bukan ditambah atau dikurangi.

d. Pengujian waktu komputasi

Pengujian ini dilakukan dengan mengamati waktu yang dibutuhkan untuk proses penyisipan dan ekstraksi pesan.

Tabel 4.6 Pengujian waktu penyisipan pesan

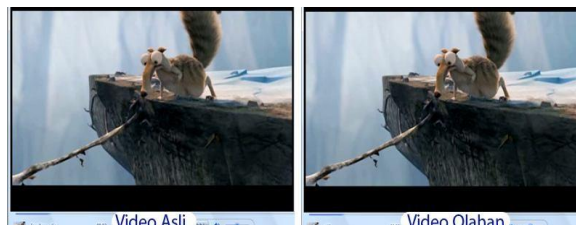
Ukuran video (bytes)	Jenis Pesan	Ukuran file pesan (bytes)	Waktu komputasi (ms)
112.350.132	docx	1.151.544	33220
	doc	2.303.032	52371
	pdf	3.454.568	69246
	docx	4.607.984	87652

Tabel 4.7 Pengujian waktu ekstraksi pesan

Ukuran video (bytes)	Jenis Pesan	Ukuran file pesan (bytes)	Waktu komputasi (ms)
112.350.132	docx	1.151.544	7752
	doc	2.303.032	15292
	pdf	3.454.568	22712
	docx	4.607.984	29628

Dari hasil pengujian didapatkan bahwa semakin besar ukuran file pesan yang disisipkan maka semakin lama waktu yang dibutuhkan untuk penyisipan dan ekstraksi pesan. Ukuran file video yang digunakan juga mempengaruhi lama waktu komputasi.

e. Pengujian perubahan kualitas video Pengujian kualitas dilakukan dengan cara subjektif dan objektif. Cara subjektif dilakukan dengan perbandingan secara visual.



Gambar 4.1 Perbandingan tampilan video

Sedangkan pengujian objektif dilakukan dengan perhitungan nilai PSNR. Peak Signal-to-Noise Ratio (PSNR) adalah pengukuran kualitas video dengan membandingkan perubahan pada video hasil dari video asli. Nilai PSNR ditentukan oleh besar atau kecilnya perubahan yang terjadi pada video. Satuan nilai dari PSNR adalah dB (deciBell). Perhitungan PSNR dilakukan dengan rumus:

$$\begin{aligned}
PSNR &= 10 \cdot \log_{10} \left(\frac{MAX_I^2}{MSE} \right) \\
&= 20 \cdot \log_{10} \left(\frac{MAX_I}{\sqrt{MSE}} \right) \\
&= 20 \cdot \log_{10}(MAX_I) - 10 \cdot \log_{10}(MSE) \text{ (persamaan 2) Dimana MSE didapat dari:}
\end{aligned}$$

$$MSE = \frac{1}{m \cdot n} \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} [f(i, j) - K(i, j)]^2 \text{ (persamaan 3)}$$

Semakin besar nilai PSNR, maka semakin baik pula kualitas dari video hasil. Sebaliknya, semakin kecil nilai PSNR, maka semakin buruk pula kualitas dari video hasil. Kualitas video yang baik memiliki nilai PSNR minimal 30 dB[7]. Dalam penelitian ini digunakan aplikasi *MSU Quality Measurement Tools* untuk mendapatkan perhitungan nilai PSNR. Hasil pengujian dapat dilihat pada tabel 4.8.

Tabel 4.8 Hasil pengujian PSNR video

Kapasitas video	Jenis pesan	Ukuran pesan	Nilai PSNR
4.607.992	docx	1.151.544	77,34
	doc	2.303.032	74,33
	pdf	3.454.568	72,57
	docx	4.607.984	71,32

Pada pengujian subjektif, tidak terlihat perbedaan yang signifikan pada video. Sedangkan pada objektif, nilai PSNR yang didapat berada di atas 30 dB. Hal ini menunjukkan kualitas video steganografi cukup baik. Perubahan nilai PSNR proporsional terhadap ukuran pesan yang disembunyikan.

5 KESIMPULAN DAN STUDI PENGEMBANGAN

Dalam pembuatan tugas akhir Implementasi Steganografi untuk Penyembunyian Pesan pada Video dengan Metode LSB ini didapat kesimpulan sebagai berikut :

1. Metode LSB dan algoritma Blowfish berhasil diimplementasikan untuk penyembunyian pesan dan pengamanan pesan.
2. Pesan yang telah disembunyikan dapat dibaca/diambil kembali dengan menggunakan kunci yang sama.
3. Daya tampung pesan dari setiap video adalah sekitar 4,1% dari ukuran file video.
4. Ukuran pesan yang disembunyikan mempengaruhi kualitas video hasil, dimana semakin besar ukuran pesan maka semakin rendah kualitas video. Berdasarkan perhitungan PSNR didapat nilai di atas 30 dB yang menunjukkan bahwa kualitas video cukup baik dan perubahan terjadi tidak signifikan secara kasat mata.
5. Ukuran file video dan pesan berpengaruh terhadap waktu yang diperlukan untuk penyisipan dan ekstraksi pesan, dimana semakin besar ukuran pesan/video maka semakin lama waktu komputasi yang diperlukan.

Adapun saran yang dapat diajukan sebagai bahan pertimbangan dalam pengembangan aplikasi yang jauh lebih sempurna lagi di masa mendatang adalah sebagai berikut:

1. Teknik penyisipan pesan dapat dikembangkan atau disempurnakan sehingga menjadi lebih cepat dan efisien.
2. Menambah dukungan terhadap jenis video lainnya sehingga aplikasi menjadi semakin mudah untuk digunakan.
3. Aplikasi dapat dikembangkan untuk juga memanfaatkan *stream audio* sebagai media steganografi.

DAFTAR REFERENSI

- [1] Randy, Adhitya. (2009). *Studi dan Perbandingan Algoritma Blowfish dan Twofish*. Bandung : Institut Teknologi Bandung.
- [2] Krenn, J. Robert. (2004). *Steganography and Steganalysis*. Diakses 8 November 2011, dari <http://www.krenn.nl>
- [3] Ria, Gemita. (2010). *Studi Perbandingan Steganografi pada Audio, Video dan Gambar*. Bandung : Institut Teknologi Bandung.
- [4] Schneier, Bruce. (1996). *Applied Cryptography* (2nd ed). New York : John Wiley & Son.
- [5] Menezes, Alfred, Paul C. Van Oorschot, Scott A. Vanstone. (1997). *Handbook of Applied Cryptography*. CRC Press.
- [6] Microsoft. (2005). *AVI RIFF File Reference*. Diakses 12 November 2011, dari <http://msdn.microsoft.com>
- [7] Qadarisman, Aditya Yuda. (2011). *Steganografi Video dengan Menggunakan Metode Discrete Cosine Transform*. Bogor : Institut Pertanian Bogor.