Pembangunan Perangkat Lunak Steganografi Audio MP3 dengan Teknik *Parity Coding* pada Perangkat *Mobile Phone*

Herianto (13504077)
Teknik Informatika, Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Bandung
e-mail: jonkscelebess@gmail.com

Abstrak *- Steganografi merupakan ilmu dan seni yang mempelajari cara penyembunyian informasi rahasia ke dalam suatu media sedemikian sehingga manusia tidak dapat menyadari keberadaan pesan tersebut. Pada makalah ini, dilakukan studi mengenai penerapan steganografi dengan teknik Parity Coding pada media audio MP3 yang diimplementasikan diatas perangkat mobile phone. Implementasi steganografi akan disertai dengan penerapan kriptografi berupa enkripsi dan dekripsi. Pesan yang sudah dienkripsi terlebih dahulu akan disembunyikan secara merata pada setiap region pada MP3 yang sudah dibagi.*

Pembagian ini akan disesuaikan dengan panjang bit pesan beserta struktur dan jumlah frame yang ada. Pesan yang nantinya diekstraksi dari region harus didekripsi lagi agar mendapatkan pesan asli. Objek steganografi yang dihasilkan mengandung noise yang terlihat dari penurunan nilai kekuatan sinyal sehingga nilai PSNR cenderung menurun jika kapasitas pesan yang disembunyikan semakin besar dan sebaliknya. Penggunaan kunci juga dilakukan untuk memperkuat keamanan, dimana hanya kunci yang digunakan pada saat penyembunyian saja yang dapat mengekstraksi pesan tersebut. Perangkat lunak ini dibangun pada perangkat mobile phone yang mendukung aplikasi Java dengan konfigurasi CLDC 1.1 dan MIDP 2.0. Kakas pembangun yang digunakan adalah Java 2 Micro Edition, dengan IDE NetBeans, dan emulator Sun Java Wireless Toolkit.

Kata kunci*: steganografi, Parity Coding, MP3, mobile phone, PSNR*

1. PENDAHULUAN

segala aspek kegiatan dan pekerjaan manusia sangat bergantung terhadap kualitas pengiriman pesan atau data sebagai komponen utama informasi. Kualitas tersebut tentu saja dipandang melalui segi ketersampaian dari pihak pengirim kepada pihak penerima dengan keadaan pesan yang tidak mengalami perubahan, kecepatan ketersampaian pesan maupun dari segi keamanannya. Segi keamanan inilah yang akan ditonjolkan dalam pengerjaan makalah ini.

Kriptografi pun akhirnya diperkenalkan sebagai suatu sistem pengamanan dengan ide mengenkripsi data yang ada sedemikian rupa. Namun sistem ini dianggap terlalu *public* karena setiap orang mempunyai kesadaran bahwa pesan yang terlihat memang mengandung suatu kerahasiaan sehingga usaha untuk memecahkan kode enkripsi atau yang lebih dikenal dengan kriptanalisis tidak dapat dihindarkan.

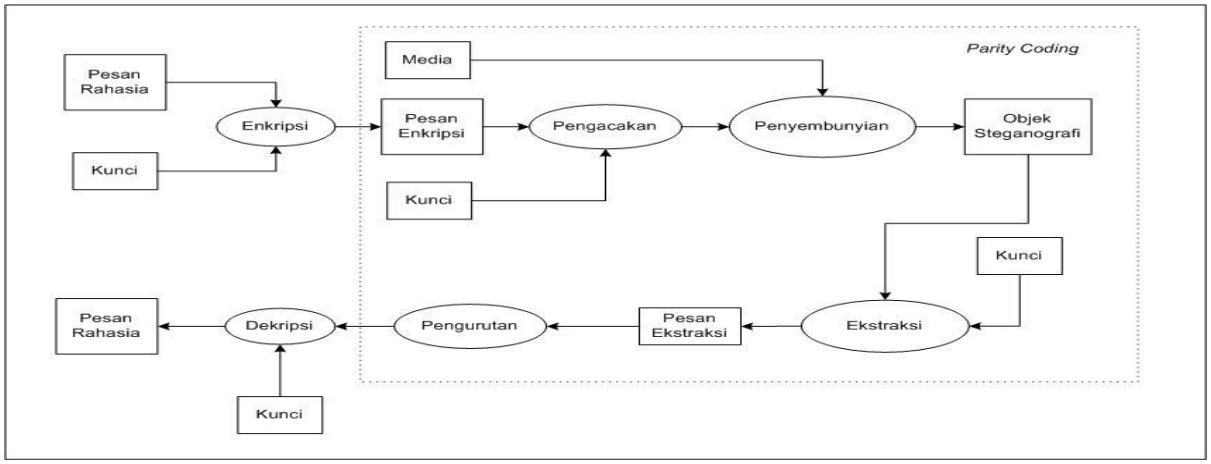
Steganografi merupakan sistem pengamanan yang muncul dari kekurangan yang dirasakan ada pada kriptografi. Idenya adalah meyembunyikan pesan rahasia kedalam pesan lain dengan harapan bahwa pihak luar tidak menyadari atau curiga adanya pesan rahasia yang terkandung didalamnya [1]. Dalam makalah ini akan steganografi audio yang diimplementasikan.

Berkas audio dipilih karena mempunyai kapasitas yang lebih besar dibanding berkas teks maupun gambar dan tidak terlalu rumit dibandingkan berkas video. Untuk MP3 sendiri, dilatarbelakangi karena segi kepopuleran yang lebih dibanding format berkas yang lain. Untuk *platform*, penulis lebih memilih perangkat *mobile phone* karena pemakai perangkat lunak ini nantinya bisa mendapatkan nilai fleksbilitas dan mobilitas.

Steganografi audio mengenal sebuah teknik yang dinamakan teknik parity coding. Pada teknik parity coding sinyal dari berkas audio dipecah menjadi beberapa región berbeda dan mengenkripsi setiap bit dari pesan rahasia yang ingin disisipkan pada sebuah sampel región yang berisi parity bit [2]. Jika parity bit dalam región yang ditentukan tidak sesuai dengan bit pesan rahasia yang akan dienkripsi, maka proses diganti dengan menukarkan LSB pada salah satu sampel región, sehingga si pengirim mempunyai pilihan lebih dalam mengenkripsi bit pesan rahasia tersebut dan sinyal audio dapat diubah dengan cara yang lebih halus.

2. ANALISIS

Berikut ini adalah analisis dari proses penyembunyian pesan dan proses ekstraksinya. Terdapat juga analisis mengenai penggunaan kunci, pengukuran kualitas video, serta pengaruh perangkat *mobile phone* pada implementasi perangkat lunak.



Gambar 1 Sistem Penyembunyian dan Ekstraksi Pesan

2.1. Analisis Penyembunyian Pesan

Proses penyembunyian pesan ke dalam suatu media ditunjukkan pada Gambar 1. Proses ini dimulai dari pengenkripsian yang dilakukan terhadap pesan rahasia dengan menggunakan kunci hash yang dibangkitkan.

Enkripsi ini sendiri hanya mengambil beberapa bit paling kiri dari kunci hash sesuai dengan kebutuhan blok *cipher* apakah 4 bit, 8 bit maupun kelipatan 8 berikutnya. Setelah pengenkripsian, pesan terenkripsi diacak terlebih dahulu dengan memanfaatkan *random number* yang degenerate terhadap kunci sebelumnya, barulah disembunyikan ke dalam berkas MP3 (media) yang sudah tersedia. Disinilah teknik *parity coding* diimplementasikan dalam penyembunyian pesan menggunakan kunci yang sama dengan enkripsi sebelumnya.

Teknik ini juga diterapkan pada ekstraksi pesan sehingga bisa mengambil nilai bit yang benar dari objek steganografi. Setelah semua bit terambil, posisi bit-bit tersebut haruslah disusun kembali dengan *random number* yang sama dari kunci yang sama pula. Bit-bit hasil penyusunan inilah yang didekripsi untuk menghasilkan pesan yang asli.

2.1.1 Enkripsi Pesan

Misalkan pesan yang ingin disisipkan berbentuk plainteks (P_1) adalah
10100010001110101001 1100
Bagi plainteks menjadi blok-blok yang berukuran 8 bit :
10100010 00111010 10011100
atau dalam notasi HEX adalah A23A9C.

Misalkan kunci (K) yang digunakan adalah (panjangnya juga 8 bit)
10110001
atau dalam notasi HEX adalah B1.

Misalkan fungsi enkripsi Eyang dipakai adalah fungsi sederhana (tetapi lemah) yaitu dengan meng-XOR-kan blok plainteks Pi dengan K, kemudian geser secara wrapping bit-bit hasil XOR satu posisi ke kanan. Proses enkripsi untuk setiap blok digambarkan sebagai berikut :

10100010	00111010	10011100
10110001	10110001	10110001+
Hasil XOR		
00010011	10001011	00101101
Geser 1 bit ke kanan		
10001001	11000101	10010110
Dalam notasi HEX		
89	C5	96

Jadi, hasil enkripsi plainteks
101000100011101010011100 (A23A9 dalam notasi HEX)
adalah
100010011100010110010110 (89C596 dalam notasi HEX)

2.1.2 Penyembunyian pesan dengan Parity Coding

Sinyal media yang sudah diencode akan dibagi menjadi beberapa region terpisah dengan ukuran statis. *Parity* bit dari setiap region akan dihitung terlebih dahulu untuk disimpan nilainya. Bit dari pesan rahasia akan disisipkan secara merata kedalam region yang ada. Jika bit yang akan dimasukkan ke dalam region nilainya berbeda, maka susunan dari bit-bit LSB harus diubah sedimikian rupa sehingga *parity* bit region nilainya sama dengan bit pesan rahasia yang akan disisipkan namun jika nilainya sama region tidak diperlu diubah. Kondisi dalam penghitungan *parity* bit adalah *even parity*. Gambar 2 adalah contoh penyembunyian suatu pesan rahasia “010” kedalam suat media stegano.

Proses pertama yang dilakukan dalam penyembunyian pesan adalah membagi media

stegano yang dalam hal ini adala berkas MP3 kedalam region-region. Banyaknya region ini ditentukan oleh panjang isi berkas yang dijadikan objek stegano. Misal contoh diambil dari plainteks yang sudah dienkripsi dari proses yang dijelaskan sebelumnya.

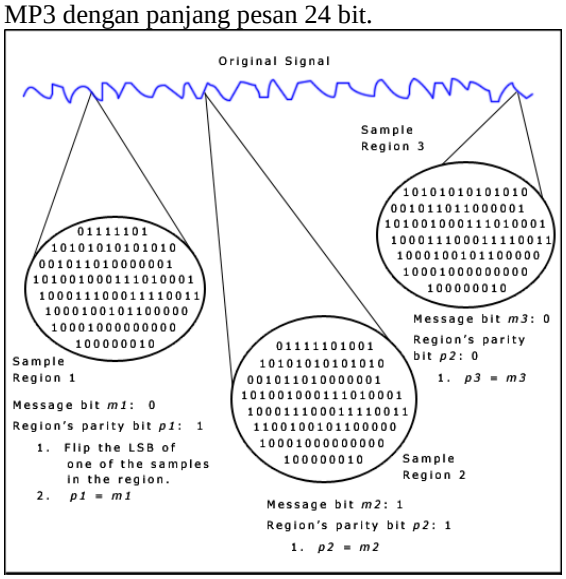
Berikut adalah spesifikasi faktor-faktor penyembunyian :
Nama objek : pesan.txt
Pesan *cipher* : 100010011100010110010110 (dalam biner)
Nama media : musik.mp3
Hasil encode media :

1010001100-0100100100-1010001100
0100100101-1010001100-0100100101
1010001100 0100100101 1010001100
0100100101-1010001100 0100100101

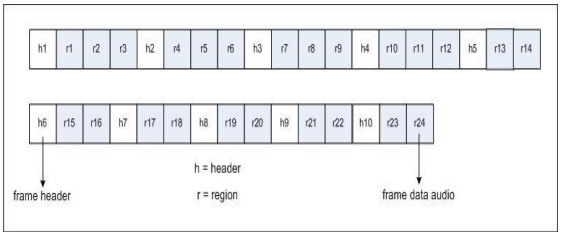
Dari spesifikasi diatas diketahui panjang dari isi pesan adalah 24 bit, sedangkan panjang bit dari media stegano adalah 120 bit yang terdiri dari 10 *frame*. Sebelum proses dilanjutkan ke pembagian region, informasi mengenai spesifikasi akan disimpan di header *frame* awal media steganografi. Hal ini berguna nantinya untuk melakukan ekstraksi agar pembagian region yang serupa bisa dilakukan saat ekstraksi.

Sebelum masuk ke proses pembagian region, pesan terenkripsi tersebut diacak terlebih dahulu dengan memanfaatkan angka-angka *pseudorandom* yang dibangkitkan oleh *pseudorandom generator* memakai kunci masukan.

Dalam proses pembagian region, media stegano akan dibagi menjadi region-region dengan banyak yang sesuai panjang isi pesan. Gambar 3 menunjukkan contoh pembagian region pada media MP3 dengan panjang pesan 24 bit.



Gambar 2 Konsep Parity Coding



Gambar 3 Pembagian Region

Sebelum bit pesan *cipher* diproses secara parity, bit-bit pesan *cipher* tersebut terlebih dahulu diacak urutannya dengan menggunakan *random number* yang dibangkitkan dari kunci. Karena bit pesan *cipher* berjumlah 24 maka *random number* yang terbentuk berkisar dari 0 sampai 23. Sebagai contoh, *random number* yang terbentuk diasumsikan sebagai berikut :
16,3,8,6,4,10,14,9,12,23,1,20,13,18,11,2,5,22,15,17,7,19,0,21
maka bit pesan *cipher* akan diacak sesuai angka pengacakan yang berarti bit pertama pesan *cipher* akan dipindahkan ke posisi 16 (urutan *array*) atau menjadi bit ke-17 dalam susunan bit *cipher* yang baru dan seterusnya sehingga menjadi :

101011000100100011110001

Setelah media dibagi menjadi region-region yang dibutuhkan, bit-bit hasil pengacakan tersebut diproses secara *parity* dalam region masing-masing. Berikut adalah proses penyembunyian bit-bit pesan ke dalam region yang telah diasumsikan:

bit 1 : 1
region 1 : 10100011
parity bit region 1 → 1 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 1 = 0

karena nilai bit pertama pesan tidak sama dengan *parity* bit region 1, maka bit-bit pada region 1 haruslah dimanipulasi sedemikian rupa agar nilai *parity* bit nya sama dengan bit pesan. Manipulasi yang dilakukan dalam tugas akhir ini adalah menukar LSB dari region.

bit 2 : 0
region 2 : 00010010
parity bit region 2 → 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ 0 = 0

karena nilai bit kedua pesan sama dengan *parity* bit region 2, maka region tidak perlu dimodifikasi.

Dari pertukaran region 1 yang dilakukan maka diperoleh region 1 baru dengan bit-bit sebagai berikut:

region 1 baru : 1010001-**1**
parity bit region 1 baru → 1⊕ 0 ⊕ 1 ⊕ 0 ⊕ 0 ⊕ 0 ⊕ 1 ⊕ **1** = 0

Proses ini dilakukan keseluruh bit-bit yang pada pesan dalam region-region yang telah disediakan.

2.2. Analisis Ekstraksi Pesan

Untuk mengekstraksi isi dari pesan ada dilakukan proses yang berkebalikan dengan proses penyembunyian, yaitu melakukan ekstraksi dengan *parity* coding, setelah itu medekripsikan kembali agar bisa mendapatkan isi yang sebenarnya.

2.2.1 Ekstraksi Pesan

Ekstraksi yang dilakukan adalah kebalikan proses dari proses penyembunyian data seperti pada Gambar 1. Pesan rahasia akan diekstrak dari media stegano dengan kunci yang sama dengan yang dimiliki oleh pengirim. Kunci ini dipakai untuk membangkitkan kembali angka-angka pseudorandom yang dipakai untuk mengacak pesan terenkripsi sebelumnya sehingga bisa diurutkan kembali ke posisi semula. Setelah proses ini, pesan harus didekripsi lagi menggunakan bit kunci *hash* sesuai ukuran blok enkripsi agar pihak penerima bisa mengetahui orisinalitas pesan tersebut.

Proses yang dilakukan untuk mengekstraksi adalah dengan menyusun kembali nilai-nilai *parity* bit yang ada pada setiap region. Untuk menghasilkan *parity* bit region yang benar, tentu dibutuhkan pembagian region yang sama juga dengan proses penyembunyian. Oleh karena itu, dibutuhkan informasi spesifikasi penyembunyian yang disimpan pada byte pertama setiap *frame* media steganografi. Setelah dibaca barulah proses dilanjutkan ke pembagian region-region yang kemudian dilanjutkan dengan penghitungan *parity* bit region dimana nilai *parity* bit inilah yang disusun menjadi isi pesan setelah dienkripsi.

2.2.1 Dekripsi Pesan

Proses yang dilakukan untuk mendekripsi pesan adalah dengan melakukan proses kebalikan dari fungsi enkripsi tersebut. Jika pada proses enkripsi pesan di-XOR dengan kunci lalu digeser sejauh 1 bit ke kanan maka dalam dekripsi pesan harus digeser terlebih dahulu sejauh 1 bit ke kiri lalu di-XOR dengan kunci.

Pesan yang didapat dari ektraksi : 101011000100100011110001 disusun terlebih dahulu dengan *random number* yang dibangkitkan oleh kunci yang sama pada pengacakan yaitu :

16,3,8,6,4,10,14,9,12,23,1,20,13,18,11,2,5,22,15,17,7,19,0,21.

Angka 16 yang didapat berarti bit pesan 16(urutan *array*) atau bit ke-17 dipindahkan urutannya ke posisi pertama susunan bit yang baru dan seterusnya sehingga menjadi :

100010011100010110010110

Bit-bit ini kemudian dibagi kembali menjadi blok blok yang dalam kasus ini 8 bit yaitu

10001001 11000101 10010110

digeser satu bit ke kiri menjadi

00010011 10001011 00101101

lalu di-XOR dengan kunci yang sama

00010011	10001011	00101101
10110001	10110001	10110001+
10100010	00111010	10011100

(A23A9 dalam notasi HEX)

dan hasil diatas sama persis dengan pesan yang pertama kali sebelum dienkripsi.

2.3. Pengukuran Kualitas Audio

Metode proses pengukuran kualitas audio akan dilakukan secara subjektif dan objektif. Cara subjektif yaitu dengan melakukan pengamatan langsung terhadap audio hasil penyembunyian dan audio yang asli.

Sedangkan cara objektif akan memakai perhitungan nilai PSNR (*Peak Signal to Noise Ratio*) dengan nilai minimal 30 DB. Perhitungan PSNR ini dilakukan dengan memakai rumus persamaan :

$$\mathrm{PSNR} = 10\log_{10}\left(\frac{P_1^{\,2}}{P_1^{\,2} + P_0^{\,2} - 2P_1P_0}\right)$$

Dimana P_1 adalah kekuatan sinyal berkas audio setelah proses penyembunyian pesan dan P_0 adalah kekuatan sinyal awal.

2.4. Implementasi Pada Mobile Phone

Pada tugas akhir ini, perangkat lunak akan diimplementasikan diatas *platform mobile phone* yang pada kenyataannya memiliki beberapa perbedaan dengan implementasi berbasis *Desktop*. Salah satunya adalah keterbatasan jumlah memori yang dapat digunakan yang mengakibatkan pembacaan berkas audio maupun pesan harus dilakukan secara bagian per bagian. Akibatnya pengacakan pesan tidak dapat dilakukan langsung secara utuh, namun hanya sejumlah 1024 bit atau 128 byte.

Perbedaan lainya adalah pembangunan pada perangkat *mobile phone* biasanya lebih lambat dibandingkan pada Desktop. Selain itu, tidak seragamnya spesifikasi antar *mobile phone* membuat implementasi pada *platform* ini membutuhkan penanganan yang berbeda pula. Sebagai contoh, apabila ingin membuat perangkat

lunak yang dapat mendukung MIDP 1.0 maka perangkat lunak tersebut harus dikembangkan ulang khusus pada perangkat yang mendukung MIDP 1.0 karena beberapa fungsi tidak kompatibel pada perangkat berbasis MIDP 2.0.

3. HASIL DAN PEMBAHASAN

Berdasarkan hasil analisis, telah berhasil dikembangkan perangkat lunak yang memiliki fungsi untuk menyisipkan dan mengekstraksi pesan. Kemudian dilakukan pengujian untuk memeriksa kebenaran dari perangkat lunak tersebut, beserta kinerja perangkat lunak tersebut.

Hasil penyembunyian ditunjukkan pada Tabel 1

Tabel 1 Hasil Pengujian Penyembunyian Pesan

Masukan audio	Masukan pesan	Masukan kunci	Keluaran audio
Zoe.mp3	heri.txt	heri	zoe-T-heri.mp3
	sindentosca.txt	heri	zoe-T-sindentosca.mp3
	a.jpg	heri	zoe-G-a.mp3
	images.jpeg	heri	Masukan tidak dapat diproses karena berkas audio tidak mencukupi
	nada.mp3	heri	Masukan tidak dapat diproses karena berkas audio tidak mencukupi
Kekkaishi.mp3	heri.txt	heri	kekka-T-heri.mp3
	sindentosca.txt	heri	kekka-T-sindentosca.mp3
	a.jpg	heri	kekka-G-a.mp3
	images.jpeg	heri	kekka-G-images.mp3
	nada.mp3	heri	kekka-S-nada.mp3

Hasil ektraksi pesan ditunjukkan pada Tabel 2.

Tabel 2 Hasil Pengujian Ekstraksi

Keluaran audio	Masukan kunci	Pesan keluaran	Kesimpulan
zoe-T-heri.mp3	hero	heri.txt tapi dengan isi yang tidak valid	Tidak diterima
zoe-T-sinde.mp3	heri	sindestosca.txt	Diterima
zoe-G-a.mp3	heri	a.jpg	Diterima
kekka-T-heri.mp3	heri	heri.txt	Diterima
kekka-T-sinde.mp3	heri	sindentosca.txt	Diterima
kekka-G-a.mp3	heri	a.jpg	Diterima
kekka-G-images.mp3	heri	images.jpeg	Diterima
kekka-S-nada.mp3	heri	nada.mp3	diterima

Untuk pengujian secara objektif, hasil pengukuran PSNR terlihat pada tabel 3.

Tabel 3 Hasil pengujian nilai PSNR

Berkas audio asli	Objek Steganografi	Hasil Pengujian	
		Subjektif	PSNR
zoe.mp3 P0 = -14.68dB	zoe-T-heri.mp3 P1 = -14.49dB	baik	37.6dB
	zoe-T-sinde.mp3 P1 = -10.92dB	buruk	9.3dB
	zoe-G-a.mp3 P1 = -11.41dB	buruk	10.9dB
Kekkai.mp3 P0 = -19.32dB	Kekkai-T-heri.mp3 P1 = -19.45dB	baik	43.5dB
	Kekkai-T-sinde.mp3 P1 = -19.39dB	baik	48.9dB
	Kekkai-G-a.mp3 P1 = -19.44	baik	44.2dB
	Kekkai-G-image.mp3 P1 = -19.42dB	baik	45.8dB
	Kekkai-S-nada.mp3 P1 = -19.34dB	baik	59.7dB

Pengujian kinerja perangkat lunak menunjukkan hasil yang memuaskan. Kecepatan proses penyembunyian dan ekstraksi tergantung pada besarnya media steganografi yang digunakan dan besarnya pesan yang disembunyikan. Akan tetapi, ketergantungan ini lebih dipengaruhi oleh besar media.

Hal ini dikarenakan proses penyembunyian membutuhkan pembacaan media secara menyeluruh terlebih dahulu untuk menganalisis struktur berkas audio tersebut sedangkan pada proses utamanya dibutuhkan pembacaan media sekali lagi untuk pembagian region dan penulisan bit *parity* region.

Dari hasil pengujian kasus 2 dapat dianalisis bahwa ada keterkaitan nilai objektif PSNR dengan nilai subjektif. Semakin besar nilai PSNR media maka semakin baik pula kualitas audio tersebut secara subjektif. Jika nilai PSNR yang didapat lebih kecil dari 30 dB maka akan terdengar *noise* yang sangat jelas terdengar oleh telinga manusia. Nilai PSNR ini sendiri dipengaruhi oleh dua hal yaitu besarnya pesan yang disembunyikan dan struktur media steganografi.

Semakin besar pesan yang disembunyikan pada media yang sama, maka semakin besar kecil pula nilai PSNR yang didapat. Tapi terkadang nilai ini bisa berubah tergantung dari penyebaran *frame* dan kecocokannya dengan panjang bit pesan. Jika bit pesan bisa tersebar merata sesuai jumlah *frame* maka nilai PSNR yang didapat cenderung naik.

4. KESIMPULAN

Beberapa hal yang bisa disimpulkan dari pelaksanaan Tugas Akhir ini adalah :

1. Steganografi audio dengan teknik Parity Coding bisa diterapkan pada berkas audio MP3.
2. Perangkat lunak yang mengimplementasikan steganografi audio dengan teknik Parity Coding pada berkas MP3 berhasil dibangun. Kebutuhan fungsional dari perangkat lunak seperti proses penyembunyian dan ekstraksi pesan serta penggunaan kunci sudah dapat dilakukan dengan benar.
3. Kualitas berkas audio yang dihasilkan tergantung dari besarnya ukuran pesan. Semakin besar pesan yang dimasukkan maka noise yang terbentuk. Noise tersebut bisa dikurangi dengan cara membagi region secara merata keseluruh data audio sehingga kualitas bisa tetap terjaga.
4. Pengujian nilai PSNR menunjukkan bahwa nilai PSNR cenderung menurun seiring dengan bertambahnya ukuran pesan yang disembunyikan. Jika ukuran pesan yang disembunyikan semakin besar maka nilai PSNR semakin kecil yang berarti kualitas berkas audio yang disisipkan semakin buruk. Namun kecenderungan tersebut bias berubah tergantung kepada kesesuaian jumlah bit pesan denga jumlah frame media.

DAFTAR REFERENSI

[1] Munir, Rinaldi. 2004. *Kriptografi.* Program Studi Teknik Informatika, Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung.

[2] Gibson, Tyler. 2002. *Methods of Audio Steganography.* http://www.snotmonkey.com/work/school/405/methods.html#eval Tanggal akses 1 April 2008.