Penyisipan Teks Dengan Metode *Low Bit Coding* Pada Media Audio Menggunakan MATLAB 7.7.0

Hartana Wijaya[1], Karti Wilianti[2]

[1,2] *Magister Ilmu Komputer, Universitas Budi Luhur*

Jl. Ciledug Raya, Petukangan Utara, Jakarta Selatan, 12260.

DKI Jakarta, Indonesia. Telp: 021-5853753

[1]justinbodhi@yahoo.co.id

[2]kwilianti88@gmail.com

***Abstrak*—Steganografi adalah teknik menyamarkan atau menyembunyikan pesan ke dalam sebuah media pembawa (*carrier)*. Kelebihan steganografi terletak pada sifatnya yang tidak menarik perhatian atau kecurigaan orang lain. Salah satu media yang dapat digunakan sebagai *carrier* adalah berkas audio. Teknik steganografi pada berkas audio memanfaatkan kelemahan pendengaran manusia, karena kualitas suara antara berkas audio asli dengan berkas audio yang telah disisipkan pesan rahasia tidak jauh berbeda. Salah satu metode steganografi audio yang sering digunakan adalah *Low Bit Coding*. Metode ini diterapkan dengan mengganti bit-bit yang tidak terlalu berpengaruh dari berkas audio dengan bit-bit pesan. Merancang suatu aplikasi yang menerapkan steganografi pada berkas audio WAV dengan menggunakan metode *Low Bit Coding*. Metode *Low Bit Coding* ini diujicoba untuk melakukan proses penyisipan dan ekstraksi pesan. Dimana jenis pesan yang dapat disisipkan adalah pesan berupa teks (tulisan). Kesimpulan dari penulisan tentang aplikasi audio steganografi dengan metode *Low Bit Coding* yang dibuat ini adalah sebagai berikut : Aplikasi steganografi yang dibuat dapat menyisipkan karakter pesan text untuk digunakan sebagai media penyampaian pesan yang bersifat rahasia dengan menggunakan media audio. Aplikasi steganografi yang dibuat membutuhkan waktu proses yang relatif lama saat melakukan penyimpanan file, terutama pada file wav dengan ukuran yang sangat besar. Proses penyisipan text yang terjadi pada fiIe wav tidak menyebabkan perubahan yang berarti pada kualitas suara, sehingga suara yang terdengar tidak dapat dibedakan dengan file wav aslinya.**

***Kata Kunci* - Low Bit Coding, WAV,carrier.**

I. PENDAHULUAN

Perkembangan teknologi informasi dan telekomunikasi saat ini sangat pesat dan sangat berpengaruh bagi kehidupan manusia. Hal paling jelas yang dialami saat ini adalah perkembangan jaringan internet yang membuat manusia dapat bertukar data dan informasi dengan orang lain, misalnya mengirim *email, download* dan *upload* berkas tertentu di internet. Namun seiring dengan perkembangan tersebut, kejahatan dalam bidang teknologi informasi dan telekomunikasi semakin marak terjadi. Oleh karena itu, keamanan data dan informasi menjadi sebuah kebutuhan vital bagi para pengguna internet saat ini agar privasi mereka bisa tetap terjaga.Salah satu teknik pengamanan data yang sering digunakan adalah steganografi.

Steganografi adalah teknik menyamarkan atau menyembunyikan pesan ke dalam sebuah media pembawa (*carrier).* Kelebihan steganografi terletak pada sifatnya yang tidak menarik perhatian atau kecurigaan orang lain. Salah satu media yang dapat digunakan sebagai *carrier* adalah berkas audio.

Teknik steganografi pada berkas audio memanfaatkan kelemahan pendengaran manusia, karena kualitas suara antara berkas audio asli dengan berkas audio yang telah disisipkan pesan rahasia tidak jauh berbeda.Salah satu metode steganografi audio yang sering digunakan adalah *Low Bit Coding*.Metode ini diterapkan dengan mengganti bit-bit yang tidak terlalu berpengaruh dari berkas audio dengan bit-bit pesan.

II. LANDASAN TEORI

A. Sejarah Steganografi

Steganografi adalah teknik menyembunyikan atau menyamarkan keberadaan pesan rahasia dalam suatu media penampung sehingga orang lain tidak menyadari adanya pesan di dalam media tersebut. Kata steganografi pada awalnya berasal dari kata *steganos*, *steganos* sendiri sebenarnya merupakan kata dari bahasa Yunani. Lebih lengkapnya : *steganos* memiliki arti penyamaran atau penyembunyian dan *graphein* atau *graptos* memiliki arti tulisan. Pengertian steganografi yang cukup sering digunakan dalam pembelajaran dengan metodologi sejarah adalah "menulis tulisan yang tersembunyi atau terselubung".

Steganografi sudah digunakan sejak dahulu kala sekitar 2500 tahun yang lalu untuk kepentingan politik, militer, diplomatik, serta untuk kepentingan pribadi. Dan sesungguhnya prinsip dasar dalam

steganografi lebih dikonsentrasikan pada kerahasian komunikasinya bukan pada datanya.

Seiring perkembangan teknologi terutama teknologi komputasi juga bertambahnya kebutuhan dan keinginan dengan kontinuitas yang tinggi, steganografi merambah juga ke media digital. Ada dua proses utama dalam steganografi digital yaitu penyisipan (*embedding/encoding*) dan penguraian (*extraction/decoding*) pesan. Pesan dapat berupa *plaintext*, *chipertext*, citra, atau apapun yang dapat ditempelkan ke dalam *bit-stream*. *Embedding* merupakan proses menyisipkan pesan ke dalam *berkas* yang belum dimodifikasi, yang disebut media cover (*cover object*). Kemudian media cover dan pesan yang ditempelkan membuat media stego (*stego object*). *Extraction* adalah proses menguraikan pesan yang tersembunyi dalam media stego. Suatu password khusus (*stego key*) juga dapat digunakan secara tersembunyi, pada saat penguraian selanjutnya dari pesan. Ringkasnya, steganografi adalah teknik menanamkan *embeddedmessage* pada suatu *cover object*, dimana hasilnya berupa *stego object*.

Pihak yang terkait dengan steganografi antara lain *embeddor*, *extractor*, dan *stegoanalyst*. *Embeddor* adalah orang yang melakukan *embedding* dengan menggunakan aplikasi steganografi, *extractor* adalah orang yang melakukan *extract stego image* dengan menggunakan aplikasi steganografi. Sedangkan *stegoanalyst* adalah orang yang melakukan steganalisis. Steganalisis merupakan ilmu dan seni untuk mendeteksi pesan yag tersembunyi dalam steganografi[1].

B. Kriteria Steganografi

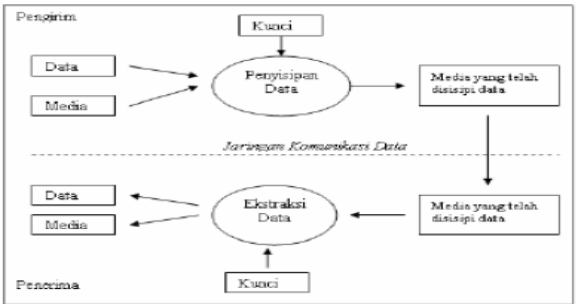
Steganografi yang dibahas di sini adalah penyembunyian data di dalam citra digital saja. Meskipun demikian, penyembunyian data dapat juga dilakukan pada wadah berupa suara digital, teks, ataupun video. Penyembunyian data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut. Kriteria yang harus diperhatikan dalam penyembunyian data adalah :

1) *Fidelity* : Mutu citra penampung tidak jauh berubah. Setelah penambahan data rahasia, citra hasil steganografi masih terlihat dengan baik. Pengamat tidak mengetahui kalau di dalam citra tersebut terdapat data rahasia.

2) *Robustness* : Data yang disembunyikan harus tahan terhadap manipulasi yang dilakukan pada citra penampung (seperti pengubahan kontras, penajaman, pemampatan, rotasi, perbesaran gambar, pemotongan (*cropping*), enkripsi, dan sebagainya). Bila pada citra dilakukan operasi pengolahan citra, maka data yang disembunyikan tidak rusak.

3) *Recovery* : Data yang disembunyikan harus dapat diungkapkan kembali (*recovery*). Karena tujuan steganografi adalah data hiding, maka sewaktu-waktu data rahasia di dalam citra penampung harus dapat diambil kembali untuk digunakan lebih lanjut.

4) *Imperceptible* : Keberadaan pesan rahasia tidak dapat dipersepsi [2].



Gambar 1 Diagram Sistem Steganografi [3]

C. Metode-Metode Steganografi

Metode-metode umum yang digunakan untuk menyembunyikan data dalam sebuah digital images antara lain:

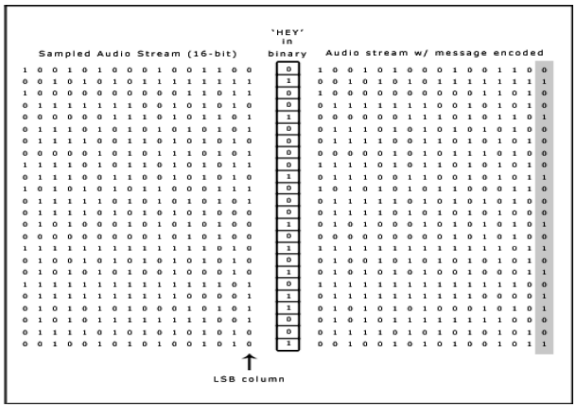
1) *Least Significant Bit Insertion (LSB)*

Dalam steganografi audio terdapat beberapa teknik dalam penyisipan pesan, salah satunya adalah metode *Low Bit Coding*. Metode *Low Bit Coding* adalah cara yang paling sederhana untuk menyimpan data kedalam file audio. Teknik ini diimplementasikan dengan mengganti bit yang paling tidak penting atau *low significantbit (LSB)* pada setiap titik *sampling* dengan string berkode biner (*coded binary string*), kita dapat menyisipkan sejumlah besar data ke dalam suara digital. Kelemahan metode ini adalah lemahnya kekebalan terhadap manipulasi. Pada prakteknya, metode ini hanya berguna pada lingkungan *digital-to-digital* yang tertutup.

Sistem Steganografi akan menyembunyikan sejumlah informasi dalam suatu berkas dan akan mengembalikan informasi tersebut kepada pengguna yang berhak. Terdapat dua langkah dalam sistem Steganografi yaitu proses penyembunyian dan *recovery* data dari berkas penampung. Penyembunyian data dilakukan dengan mengganti bit-bit data di dalam segmen citra dengan bit-bit data rahasia. Metode yang paling sederhana adalah metode modifikasi LSB (*Least Significant Bit Modification*) dari setiap sample pada audio (*Cvejic & Seppanen*). Pada susunan bit di dalam sebuah byte (1 byte = 8 bit), ada bit yang paling berarti (*Most Significant Bit* atau MSB) dan bit yang paling kurang berarti (*Least Significant Bit* atau LSB).

Contoh : 1 1 0 1 0 0 1 0

Bit pada digit pertama merupakan MSB, dan bit pada digit terakhir merupakan LSB. Bit yang cocok untuk diganti adalah bit LSB, sebab perubahan tersebut hanya mengubah nilai byte satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Karena perubahan satu-dua buah bit pada file audio tidak akan terdengar dengan jelas. Metode ini memanfaatkan keterbatasan dari *system* indera pendengaran manusia yaitu HAS (*Human Auditory System*). Batas pendengaran manusia berada pada frekuensi antara 20 Hz - 20000 Hz. Namun metode ini memiliki keterbatasan jumlah bit pesan yang dapat disisipkan atau disubstitusikan pada setiap byte-nya [4].



Gambar 2 *Least Significant Bit*

2) *Masking and Filtering*

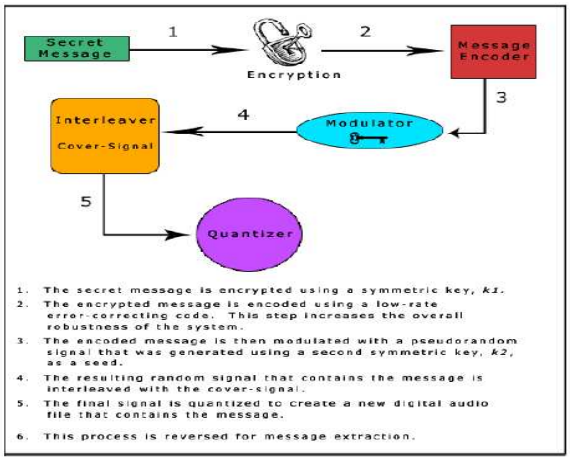
Kedua metode ini menyembunyikan informasi dengan cara mirip dengan penanda kertas. Hal ini dapat dilakukan, contohnya dengan memodifikasi *luminance* sebagian dari citra, tetapi apabila dilakukan dengan hati-hati distorsi baru dapat terlihat.

3) *Discrete Cosine Transformation (DCT)*

adalah salah satu metodetransformasi untuk mentransformasi 8*8 blok pixel dari sebuah citra secara berurutan kedalam masing-masing koefisien 64 DCT. Alat steganografi dapat menggunakan LSB dari koefisien DCT yang terbagi-bagi utnuk menyembunyikan informasi (metode JSteg).SebagaitambahanDCT, citra dapat diproses dengan FFT atau dengan *Wavelet Transformation*. Properti citra yang lain seperti luminance juga dapat dimanipulasi.

4) *Spread Spectrum Image Steganography*

Metode-metode yang didasari oleh teknologi ini menyandikan pesan yang diinginkan agar tersembunyi. Untuk menyandikan, digunakan sebuah *pseudorandom noise* generator yang lebar untuk membuat sebuah barisan yang tersebar. Kemudian, sebuah skema modulasi digunakan untuk memperluas spektrum yang sempit dari sebuah pesan dengan barisan yang tersebar, dengan demikian menyusun sinyal yang dibawa yang masuk ke dalam *interleave* dan ruang penyebar. *Inner leaver* juga dapat mempergunakan kunci untuk mendikte algoritma *interleaving*. Sinyal ini sekarang digabungkan dengan *cover* dari citra untuk menghasilkan citra stego, yang sudah dibagi-bagi dengan layak untuk memelihara *dynamic range* awal dari *cover* citra. Citra stego tersebut kemudian diteruskan kepada penerima pesan. Untuk lebih jelasnya, lihat gambar berikut [5] :



Gambar 3.*Spread Spectrum*

D. *Steganografi Audio*

Pada steganografi audio yang berbasis komputer, pesan rahasia ditanam pada suara digital. Pesan rahasia tersebut ditanam dengan cara merubah urutan biner dari file suara. Steganografi audio dapat diterapkan pada file suara WAV, AU dan bahkan MP3. Menanam pesan rahasia pada suara digital pada umumnya mempunyai proses yang lebih sulit dibandingkan menanam pesan pada media lainnya, seperti media gambar.

Untuk menanam pesan rahasia dengan sukses, beberapa metode untuk menanam informasi di dalam audio digital telah diperkenalkan. Mulai dari metode yang sederhana dengan menyisipkan informasi dalam bentuk gangguan sinyal sampai kepada metode yang lebih sulit dengan menggunakan teknik pengolahan sinyal yang rumit untuk menyembunyikan informasi.

E. *Waveform Data* (WAV)

Format suara WAV merupakan format standar dari RIFF (*Resource Interchage File Format*) yang berjalan pada Microsoft Windows. Format suara WAV diindikasikan dengan ekstensi *Waveform data* (.WAV). Format resmi suara digital WAV (format

RIFF) terdiri atas 3 *chunk*, yaitu *chunk header*, *chunk fmt*, dan *chunk data*. Isi dari ketiga *chunk* dapat dilihat padaTabel 1, Tabel 2, dan Tabel 3 [6].

TABEL 1
ISI CHUNK HEADER

Offset	Panjang	Isi
0	4 byte	'RIFF'
4	4 byte	<file length – 8>
8	8 byte	'WAVE'

TABEL II
ISI CHUNK FMT

Offset	Panjang	Isi	Keterangan
12	4 byte	'fmt '	
16	4 byte	0x00000010	Panjang *chunk* fmt (16 bytes)
20	2 byte	0x0001	Format tag: 1 = PCM
22	2 byte	<channels>	Channels: 1 = mono, 2 = stereo
24	4 byte	<sample rate>	Banyak sample per detik: contoh: 44100
28	4 byte	<bytes/second>	sample rate * block align
32	2 byte	<block align>	channels * bits/sample / 8
34	2 byte	<bits/sample>	8 or 16

TABEL III
ISI CHUNK DATA

Offset	Panjang	Isi
36	4 byte	'data'
40	4 byte	<length of the data block>
44	4 byte	<sample data>

Jumlah sampel suara adalah genap. Untuk suara dengan jumlah bit tiap sampel suara 8 disimpan sebagai bilangan desimal tidak bertanda, dengan rentang nilai dari 0 sampai 255.

III. PERANCANGAN

A. Perancangan Spesifikasi Aplikasi

Spesifikasi aplikasi meliputi input (masukan) dan output (keluaran). Masukan bergantung pada proses yang akan dilakukan (proses penyisipan atau ekstraksi). Pada proses penyisipan dibutuhkan dua buah masukan yaitu media *cover* dan pesan rahasia. Media *cover* yang digunakan adalah berkas audio WAV dan pesan rahasianya berupa teks. Sedangkan pada proses ekstraksi hanya dibutuhkan satu buah masukan. Masukan tersebut adalah berkas audio WAV yang telah disisipi pesan rahasia (*stego*). Keluaran yang dihasilkan juga bergantung pada proses yang akan dilakukan. Pada proses penyisipan, keluaran yang dihasilkan adalah berkas audio WAV yang telah disisipi pesan rahasia (*stego*). Sedangkan pada proses ekstraksi, keluarannya adalah pesan rahasia berupa teks.

B. PerancanganAlgoritma

Metode yang digunakan dalam pembuatan aplikasi steganografi ini adalah metode *Low Bit Coding*. Ditinjau dari segi fungsional, maka aplikasi yang akan dibangun memiliki 2 proses, yaitu proses *encoding* (penyisipan) dan proses *decoding* (ekstraksi). Algoritma proses penyisipan dan ekstraksi dengan metode *Low Bit Coding*akan dijelaskan pada bagian ini.

1) Algoritma Proses Penyisipan

Langkah-langkah proses penyisipan bit-bit pesan (teks) ke dalam berkas audio WAV adalah sebagai berikut :

a) Berkas audio WAV dibaca dan byte-byte hasil pembacaan disimpan ke dalam variable "fid1".

b) Informasi mengenai berkas audio disiapkan dengan cara mengambil 40 byte pertama dari "fid1" dan pindahkan ke dalam variable "header". Byte-byte ini berisi informasi mengenai chunk header, chunk fmt, chunk data dan tidak dapat disisipi bit pesan.

c) Informasi tentang ukuran berkas audio di mabil dari 41 byte sampai 43 byte.

d) Byte sampel data (byte ke 44 sampai byte terakhir) dipindahkan ke dalam variable "dta".

e) Pesan berupa text diinput.

f) 8 bit unik disiapkan dan disimpan ke dalam variable "identitas". Bit-bit ini dibutuhkan sebagai penanda apakah di dalam berkas audio terdapat pesan teks atau tidak. Identitas ini didefinisikan oleh programmer dengan seunik mungkin dan identitas yang digunakan adalah [1 0 1 0 1 0 1 0].

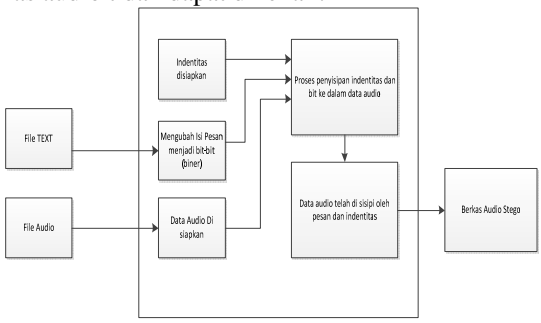
g) Pesan text dan ukurannya dibaca.

h) Sebelum dilakukan penyisipan, terlebih dahulu dicek apakah penyisipan dapat dilakukan atau tidak. Pengecekan dilakukan berdasarkan jumlah byte sampel data dan jumlah bit identitas + bit ukuran pesan + bit pesan. Jika jumlah bit identitas + bitukuran pesan + bit pesan lebih kecil dari jumlah byte sampel data, maka proses penyisipan dapat dilakukan.

i) Apabila pengecekan bernilai ya, maka langkah selanjutnya adalah pesan dan ukurannya diubah ke dalam bentuk bit. Kemudian bit identitas, bit ukuran pesan, dan bit pesan disisipkan ke dalam "dta". Penyisipan dilakukan dengan mengganti bit pertama (bit yang tidak terlalu berpengaruh) dari setiap byte sampel data. Identitas disisipkan pada byte pertama sampai byte keenambelas. Sedangkan bit ukuran pesan dan bit pesan disisipkan pada byte ketujuhbelas dan seterusnya.

Data audio yang telah disisipi pesan disimpan dengan mengikutsertakan informasi berkas audio dari variable "header".Informasi ini ditulis pada awal

berkas.Jika informasi ini tidak diikutsertakan, maka berkas audio tidak dapat dikenali.



Gambar. 4 Proses Penyisipan

Berikut ini adalah penggalan program untuk proses *encoding* yang dibuat dengan menggunakan MATLAB :

- Script mengambil *file Audio*

```
[filename, pathname] = uigetfile('*.wav','Select a file');
[y,fs,nbits,opts]=wavread([pathname filename],[1 2]);
fid1=fopen([pathname filename],'r');
```

Fungsi script di atas digunakan untuk menampilkan jendela pencarian file audio yang berekstensi *.wav. Nama file terpilih akan disimpan ke dalam variable file name dan pathnya akan disimpan ke dalam variable File audio kemudian dibuka dengan menggunakan fungsi fopen dan hasil pembacaannya disimpan ke dalam variable fid1.

- Fungsi fread digunakan untuk mengambil 40 byte pertama dari "fid1" dan dipindahkan ke dalam variable "*header*". Byte-byte ini berisi informasi mengenai berkas audio.

```
data_size=fread(fid1,1,'uint32');
```

- Fungsi fread berikutnya digunakan untuk membaca ukuran file audio.

```
data_size=fread(fid1,1,'uint32');
```

- Fungsi fread berikutnya digunakan untuk membaca dan memindahkan byte-byte sampel data ke dalam variable dta. Sedangkan variable count digunakan untuk membaca dan menyimpan jumlah byte sampel data.

```
[dta,count]=fread(fid1,inf,'uint16');
fclose(fid1);
lsb=1;
```

- Kemudian pesan di input dengan object textbox dan di simpan dalam variable msg. apabila isi pesan lebih besar dari berkas file audio maka muncul pesan "Pesan Terlalu besar,Mohon masukan isi pesan yang lebih kecil"

```
msg=get(handles.edit1,'string');        [ro,co]=size(msg);
if ( (ro*co*8+28) > count )
msgbox('Pesan terlalu besar, Mohon memasukan isi pesan yang lebih kecil','Empty');
```

- Identitas = [1 0 1 0 1 0 1 0]. Byte penanda disiapkan dan disimpanke dalam variable identity.

```
msg_double=double(msg);
msg_bin=de2bi(msg_double,8);
[m,n]=size(msg_bin);
msg_bin_re=reshape(msg_bin,m*n,1);
m_bin=de2bi(m,10)';
n_bin=de2bi(n,10)';
len=length(msg_bin_re);
len_bin=de2bi(len,20)';
identity=[1 0 1 0 1 0 1 0]';
dta(1:8)=bitset(dta(1:8),lsb,identity(1:8));
dta(9:18)=bitset(dta(9:18),lsb,m_bin(1:10));
dta(19:28)=bitset(dta(19:28),lsb,n_bin(1:10));
dta(29:28+len)=bitset(dta(29:28+len),lsb,msg_bin(1:len)')
    ;
```

Kemudian isi pesan yang ingin di sisipkan di ubah terlebih dahulu ke tipe data double dan pesan di ubah kedalam bentuk biner/bit dengan menggunakan fungsi de2bi. Kemudian bit indentitas di sisipkan dan bit-bit isi pesan di sisipkan ke dalam sampel data audio dengan fungsi bitset.

- Tahapan terakhir menyimpan berkas audio stego yang sudah di sisipi dengan pesan dan indentitas.

```
fid2=fopen(['new' randname '.wav'],'w');
fwrite(fid2,header,'uint8');
fwrite(fid2,data_size,'uint32');
fwrite(fid2,dta,'uint16');
fclose(fid2);
```

2) Algoritma Proses Ekstraksi

Langkah-langkah proses ekstraksi bit-bit pesan dari berkas audio stego adalah sebagai berikut ini :

a) Berkas audio stego dibaca dan byte-byte hasil pembacaan disimpan ke dalam variable "fid1".

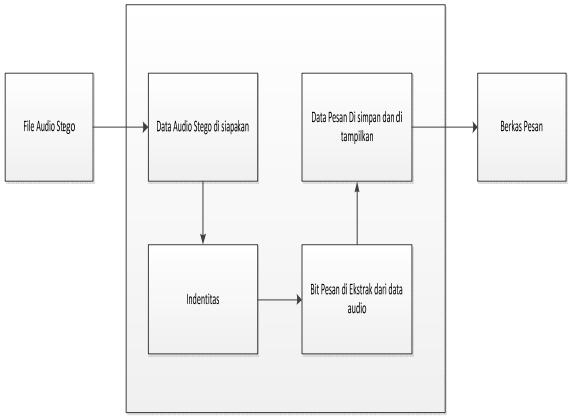
b) 40 byte pertama dari "fid1" yang berisi informasi mengenai berkas audio dipindahkan ke dalam variable "header".

c) Byte sampel data (byte ke 44 sampai byte terakhir) dipindahkan ke dalam variable "dta".

d) Bit pertama dari byte pertama sampai byte keenambelas diekstrak dari dta. Jika hasilnya sama dengan salah satu identitas pada saat penyisipan, maka di dalam berkas tersebut terdapat pesan rahasia dan proses ekstraksi dapat dilakukan.

e) Apabila pengecekan bernilai ya, maka langkah selanjutnya adalah bit ukuran pesan dan bit pesan diekstrak. Bit-bit pesan yang telah diekstrak dikembalikan ke bentuk semula berdasarkan ukuran pesan.

f) Pesan *text* hasil ditampilkan dalam textbox pada program.



Gambar. 5 Proses Ekstraksi

Berikut ini adalah penggalan program untuk proses *decoding* yang dibuat menggunakan MATLAB:

- Script mengambil *file Audiostego*

```
[filename, pathname] = uigetfile('*.wav','Select a
    file');
[y,fs,nbits,opts]=wavread([pathname
    filename],[1 2]);
fid1=fopen([pathname filename],'r');
```

Fungsi *script* di atas digunakan untuk menampilkan jendela pencarian file audio stego yang berekstensi *.wav. Nama file terpilih akan disimpan ke dalam variable *file name* dan pathnya akan disimpan ke dalam variable File audio kemudian dibuka dengan menggunakan fungsi fopen dan hasil pembacaannya disimpan ke dalam variable fid1.

- Fungsi fread digunakan untuk mengambil 40 byte pertama dari "fid1" dan dipindahkan ke dalam variable "header"

```
[dta,count]=fread(fid1,inf,'uint16');
 ans=fclose(fid1);
 lsb=1;
```

Byte-byte sampel data dipindahkan ke dalam variable dta dengan menggunakan fungsi fread.

- Bit pertama dari byte pertama sampai byte kedelapan diekstrak dari dta dengan menggunakan fungsi bitget untuk mendapatkan identitas.

```
identity=bitget(dta(1:8),lsb)';
```

- Kemudian di mulai validasi data pada file audio stego dengan menggunakan percabangan if , apakah bit yang berhasil di ekstrak sama dengan bit indentitas pada proses penyisipan. Jika sama, maka bit ukuran pesan dan pesan teks di ekstrak juga dengan menggunakan fungssi bitget.

```
if identity==[1 0 1 0 1 0 1 0]
      len_bin=zeros(20,1);
      m_bin=zeros(10,1);
      n_bin=zeros(10,1);

      m_bin(1:10)=bitget(dta(9:18),lsb);
      n_bin(1:10)=bitget(dta(19:28),lsb);
          m=bi2de(m_bin');
      n=bi2de(n_bin');
      len=m*n*8;
      secmsg_bin=zeros(len,1);
      secmsg_bin(1:len)=bitget(dta(29:28+len),lsb);
      secmsg_bin_re=reshape(secmsg_bin,len/8,8);
      secmsg_double=bi2de(secmsg_bin_re);
      secmsg=char(reshape(secmsg_double,m,n));
      set(handles.edit1,'string',secmsg);
 else
  msgbox('tidak ada pesan yang di sisipkan ','Empty');
end
```

Dan pesan yang di sisipkan di tampilkan pada textbox pada program. Jika hasil indentitas tidak sama maka muncul pesan "*tidak ada pesan yang di sisipkan*" dan proses ekstraksi berhasil.

IV. UJI COBA DAN ANALISA

A. Perancangan Pengujian

Pengujian dilakukan berdasarkan spesifikasi aplikasi dan juga terhadap ketahanan data. Pengujian berdasarkan spesifikasi aplikasi meliputi pengujian kesesuaian proses, kesesuaian data, dan kualitas suara. Sedangkan pengujian ketahanan data hanya dilakukan terhadap berkas suara WAV *stego*. Untuk melakukan pengujian, digunakan beberapa berkas audio WAV dengan spesifikasi seperti yang terdapat pada Tabel IV.

TABEL IV
SPESIFIKASI AUDIO WAV YANG DIUJI

No.	Nama File Audio WAV	Durasi	Ukuran Data (Byte)	Sampling Rate (Hz)	Jenis Kanal Suara
1	adios.wav	00:00:03	35100	11000	Mono
2	aribba.wav	00:00:03	40100	11000	Mono

B. Pelaksanaan Pengujian

Pada pelaksanaan pengujian dibutuhkan beberapa perangkat lunak pendukung, yaitu *Free WAV to MP3 Converter* 7.3.2 dan *Wave Editor 3.1.0.0*.Perangkat lunak *FreeWAV to MP3 Converter* digunakan untuk mengkompres audio WAV *stego*, yaitu dengan mengubah format audio WAV *stego* menjadi MP3 (dengan ekstensi .mp3) dan sebaliknya. Sedangkan *Wave Editor* digunakan untuk melakukan manipulasi amplitudo, pemotongan dan dapat juga menampilkan grafik sinyal dari audio WAV *stego*.

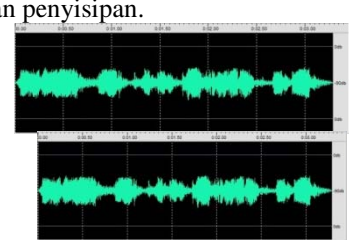
1) Pengujian Berdasarkan Spesifikasi Aplikasi

Hasil pengujian terhadap kesesuaian proses, kesesuaian data, dan kualitas suara dapat dilihat pada Tabel V.

TABEL V
HASIL PENGUJIAN BERDASARKAN SPESIFIKASI APLIKASI

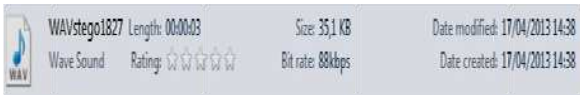
No	Nama Berkas Audio	Pesan Teks	Penyisipan	Nama berkas Audio WAV Stego	Ekstrasi	Kesesuaian Data	Kualitas Suara
1	adios.wav	Pesan Rahasia	Berhasil	WAVstego1827.wav	Berhasil	Sesuai	Sama
2	aribba.wav	Pesan Rahasia	Berhasil	WAVstego1265.wav	Berhasil	Sesuai	Sama

Hasil pengujian pada Tabel V menunjukkan bahwa perangkat lunak berhasil untuk ketiga faktor pengujian. Pada pengujian terhadap kesesuaian proses, perangkat lunak dapat melakukan proses penyisipan dan ekstraksi dengan baik. Walaupun pada proses penyisipan dapat terjadi kegagalan karena ukuran pesan yang terlalu besar. Pengujian terhadap kesesuaian data, menunjukkan bahwa data yang berhasil diekstrak dari audio WAV *stego* bersesuaian dengan data yang disisipkan. Dan pada pengujian terhadap kualitas suara, baik berdasarkan pendengaran maupun secara visual melalui grafik sinyal, dapat dikatakan bahwa kualitas suara antara berkas audio asli dengan berkas audio *stego* adalah sama. Gambar 6 berikut ini adalah salah satu gambar yang menunjukkan grafik sinyal audio dengan menggunakan aplikasi*Wave Editor 3.1.0.0,* sebelum dan sesudah dilakukan penyisipan.



Gambar 6. Grafik Sinyal adios.wav Sebelum dan Sesudah Penyisipan

Setelah proses penyisipan, hal lain yang dapat diperbandingkan adalah ukuran berkas audio sebelum dan setelah penyisipan.



Gambar 7. Ukuran Berkas Audio Sebelum dan Setelah Penyisipan

Tabel V menunjukkan bahwa seluruh uji coba penyisipan tidak menyebabkan perubahan pada ukuran berkas audio WAV atau dengan kata lain, ukuran berkas audio WAV sebelum dan setelah penyisipan adalah sama.

2) Pengujian Ketahanan Data

Pengujian ketahanan data dilakukan dengan teknik kompresi.Kompresi audio *stego* dari WAV ke MP3 dan sebaliknya dilakukan dengan menggunakan aplikasi *Free WAV to MP3 Converter* 7.3.2. Hasil pengujian proses ekstraksi setelah kompresi terhadap berkas audio *stego* dapat dilihat pada Tabel VI.

TABEL VI
HASIL PENGUJIAN EKSTRAKSI SETELAH KOMPRESI.

Nama berkas Audio WAV Stego	Pesan Teks	Nama berkas MP3 setelah kompresi	Nama berkas Audio WAV Setelah kompresi Ulang	Ekstrasi
WAVstego1265.wav	Pesan Rahasia	WAVstego1265(mp3).mp3	WAVstego1265(Wav).wav	Gagal

C. Analisis Hasil Pengujian

Dari pengujian yang telah dilakukan, dapat dilakukan beberapa analisis terhadap hasil pengujian tersebut. Berikut ini merupakan analisis terhadap hasil pengujian berdasarkan spesifikasi aplikasi dan ketahanan data.

1) Analisis Hasil Pengujian Berdasarkan Spesifikasi Aplikasi

Hasil pengujian berdasarkan spesifikasi aplikasi menunjukkan bahwa perangkat lunak steganografi ini berhasil untuk setiap faktor pengujian yang dilakukan. Faktor yang diuji meliputi faktor kesesuaian proses, kesesuaian data, dan kualitas suara sebelum dan sesudah file disisipkan teks.

Pada pengujian kesesuaian proses, perangkat lunak dapat melakukan proses penyisipan dan ekstraksi pesan teks dengan baik. Proses penyisipan berhasil jika ukuran data (jumlah bit identitas + bit ukuran pesan teks + bit pesan teks) tidak lebih besar dari jumlah byte sampel data audio yang akan disisipi pesan Teks. Pada pengujian kesesuaian data, teks yang berhasil diekstrak sesuai dengan teks yang disisipkan.Kesesuaian ditinjau dari teksnya.

Pengujian kualitas suara dilakukan untuk mengetahui sama tidaknya suara WAV asli dengan suara WAV stego. Pengujian dilakukan secara subjektif dan objektif. Pengujian dengan cara subjektif dilakukan dengan mendengarkan langsung suara WAV asli dan suara WAV stego, kemudian keduanya dibandingkan. Hal ini disebabkan karena perubahan pada bit pertama atau *least significant bit* sangat sulit dideteksi oleh pendengaran manusia.

Dan hal ini juga ditunjukkan oleh grafik sinyal audio sebelum dan sesudah penyisipan yang hampir tidak kelihatan perbedaannya. Selain ketiga faktor di atas, hasil uji coba juga menunjukkan bahwa ukuran berkas audio WAV sebelum dan setelah penyisipan adalah sama. Hal ini terjadi karena penyisipan pesan dilakukan dengan mengganti bit yang tidak terlalu berpengaruh (*least significant bit*) dengan bit-bit pesan teks, bukan dengan menambah bit baru ke dalam audio

2) Analisis Hasil Pengujian Ketahanan Data

Pada pengujian ketahanan data terhadap kompresi, seluruh hasil menunjukkan bahwa pesan teks tidak dapat diekstrak. Proses kompresi dan manipulasi amplitudo tidak mengubah ukuran berkas audio, akan tetapi menyebabkan perubahan pada bit pertama atau *least significant bit* dari sampel data audio. Dimana bit pertama merupakan tempat untuk menyisipkan bit header, bit ukuran pesan, bit indentitas dan bit pesan. Jika terjadi kerusakan pada bit identitas, maka otomatis bit ukuran pesan dan bit pesan dalam keadaan apapun (rusak/ tidak) tidak dapat dibaca dan diekstrak. Dan berdasarkan hasil pengujian ketahanan, maka dapat disimpulkan bahwa penyisipan pesan teks dengan metode *Low Bit Coding* tidak tahan terhadap kompresi.

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Kesimpulan dari penulisan tentang aplikasi audio steganografi dengan metode *Low Bit Coding* yang dibuat ini adalah sebagai berikut :

1) Aplikasi steganografi yang dibuat dapat menyisipkan karakter pesan text untuk digunakan sebagai media penyampaian pesan yang bersifat rahasia dengan menggunakan media audio.

2) Aplikasi steganografi yang dibuat membutuhkan waktu proses yang relatif lama saat melakukan penyimpanan file, terutama pada file wav dengan ukuran yang sangat besar.

3) Proses penyisipan text yang terjadi pada fiIe wav tidak menyebabkan perubahan yang berarti pada kualitas suara, sehingga suara yang terdengar tidak dapat dibedakan dengan file wav aslinya.

B. Saran

Dari aplikasi yang dibuat ini, masih terdapat banyak kekurangan-kekurangan, diantaranya diharapkan dapat ditambah dalam banyak ekstensi file audio dengan banyak tipe file audio tidak hanya wav saja. Selain itu juga, diharapkan pengembangan aplikasi kedepannya dapat ditambah jenis penyisipan data tidak hanya teks, tapi dapat mendukung juga berbagai tipe dokumen lainnya seperti gambar, audio, video, dll. Kemudian untuk secara aplikasi dapat dikembangkan kedalam versi Web. Agar kegunaan aplikasi ini dapat dirasakan oleh banyak orang.

REFERENSI

[1] Zebua, Hendrikus "*Implementation Steganography WAV Audio Files On Low Bit Using Coding*", Gunadarma University Library: http://library.gunadarma.ac.id

[2] Sejarah Steganografi http://infosteganografi.blogspot.com/ 2012/12/sejarah- steganografi.html. Diakses pada tanggal 04 April 2013.

[3] Keamanan Komputer http://indhoarief.wordpress.com/ category/ uncategorized/.Diakses pada tanggal 04 April 2013.

[4] LSB Coding http://www.snotmonkey.com/ work/school/405/ methods.html. Diakses pada tanggal 04 April 2013.

[5] Digital Audio., http://www.snotmonkey.com/ work/school/405/overview.html., Diakses pada tanggal 04 April 2013.

[6] File Structure., http://www.sonicspot.com/guide/ wavefiles.html., Diakses pada tanggal 04 April 2013.