ISSN 2085-4552

Rancang Bangun Aplikasi Steganografi dengan Metode Least Significant Bit di Audio pada Sistem Operasi Android

Lovebbi, Dodick Z. Sudirman

Program Studi Teknik Informatik, Universitas Multimedia Nusantara, Tangerang, Indonesia.
lovebbi@live.com

Diterima 10 Mei 2012
Disetujui 20 Mei 2012

***Abstract*—Easy access to information this day, has increase the importance of security, especially private and secret information. the big amount of data transaction via smartphone without a proper data security has become a problem that needs attention. This paper's goal is to implements steganography with least significant bit method for audio as alternative to data security. The implementations is done in Android operating system and produce an application for hiding message. This research proves that least significant bit in Android operating system fulfill the criteria of good steganography method.**

Keywords—*data security, steganography, least significant bit, android*.

I. PENDAHULUAN

Perkembangan teknologi kian pesat dalam setiap lini kehidupan yang secara disadari atau tidak telah mengubah gaya hidup masyarakat. Pengaruh arus globalisasi dan semakin majunya dunia teknologi informasi telah menciptakan kebutuhan baru bagi masyarakat terhadap komunikasi tanpa batas. Oleh karena itu, saat ini banyak perangkat keras yang dapat digunakan untuk mengirim dan menerima informasi.

Masalah keamanan dan kerahasiaan data merupakan hal yang sangat penting dalam suatu organisasi maupun pribadi. Dengan adanya Internet, pihak ketiga dapat memodifikasi, menggandakan, dan menyebarkan data *multimedia digital* melalui Internet. Untuk mengurangi atau mencegah terjadinya hal tersebut perlu dikembangkan suatu aplikasi yang mampu menyembunyikan pesan pada suatu media yang dapat diakses oleh setiap orang. Teknik ini disebut steganografi. Steganografi adalah seni dan ilmu menulis dan menyembunyikan pesan dengan suatu cara sehingga selain si pengirim dan si penerima, tidak ada seorang pun yang mengetahui atau menyadari bahwa ada suatu pesan rahasia [6].

Salah satu metode steganografi adalah *least significant bit*. Metode ini merupakan metode yang sederhana dan menggunakan algoritma yang tidak terlalu rumit sehingga tidak memerlukan *resource* yang besar untuk menggunakannya. Dengan demikian, metode ini cocok digunakan pada berbagai macam perangkat seperti komputer atau telepon genggam yang memiliki memori yang relatif kecil dan tidak dapat melakukan proses yang terlalu rumit dalam waktu cepat. Prinsip kerja dari metode *least significant bit* adalah mengubah nilai bit paling rendah dari suatu sampel audio dengan bit pesan yang akan disisipkan.

Penelitian yang berkaitan dengan steganografi dilakukan untuk menambah kemampuan pengamanan informasi sehingga tidak sembarang orang dapat mengakses informasi yang bersifat pribadi atau rahasia. Media yang digunakan adalah audio sebagai media data penampung pesan yang dapat disembunyikan. Informasi rahasia dapat dienkripsi terlebih dahulu sebelum disisipkan ke dalam media penampung.

Pertukaran informasi yang pesat didukung juga oleh perkembangan telepon genggam, khususnya *smartphone*. Sistem Operasi yang banyak digunakan pada *smartphone* adalah Android. Penjualan Android yang telah mencapai 33,3 juta perangkat pada kuartal 4 tahun 2010 keberhasilannya menjadi platform smartphone terbesar di dunia [17], menyebabkan android dipilih sebagai sistem operasi yang digunakan pada penelitian ini.

Berkas audio dipilih sebagai media penampung data karena memiliki kapasitas yang lebih besar dibandingkan dengan berkas teks maupun gambar. Selain itu berkas audio memiliki ukuran yang lebih kecil dibandingkan dengan berkas video. Audio digital memiliki ukuran yang proporsional sehingga memungkinkan untuk menyisipkan lebih banyak pesan rahasia.

Dari penelitian yang dilakukan oleh Agus Susanto [14], didapat kesimpulan bahwa kualitas berkas audio berformat *Musical Instrument Digital Interface* (MIDI) yang dihasilkan dari proses steganografi tergantung dari besarnya ukuran pesan yang disisipkan. Semakin besar pesan yang disisipkan maka semakin banyak *noise* yang terbentuk. *Noise* tersebut bisa dikurangi dengan cara membagi *region* secara merata ke seluruh data audio sehingga kualitas bisa tetap terjaga.

Menurut Andy Gunawan [7], penyimpanan *text* yang terjadi pada format audio *waveform* tidak menyebabkan perubahan yang berarti pada kualitas suara sehingga suara yang terdengar tidak dapat dibedakan dengan file audio berformat *waveform* (wav) asli, hal ini disebabkan karena tiap byte data audio hanya diubah 1 bit terakhirnya saja (*least significant bit*).

Berdasarkan latar belakang ini penulis melakukan penelitian yang berfokus pada pembangunan aplikasi tentang steganografi pada audio untuk pengamanan data pada ponsel bersistem operasi Android.

II. TELAAH LITERATUR

A. Steganografi

A.1. Pengertian Steganografi

Steganografi adalah seni dan ilmu menulis atau menyembunyikan pesan tersembunyi dengan suatu cara sehingga selain si pengirim dan si penerima, tidak ada seorang pun yang mengetahui atau menyadari bahwa ada suatu pesan rahasia. Keberadaan pesan steganografi adalah rahasia. Istilah Yunani ini berasal dari kata *Steganos*, yang berarti tertutup dan *Graphia*, yang berarti menulis [4].

Tujuan dari steganografi adalah merahasiakan atau menyembunyikan keberadaan dari sebuah pesan tersembunyi atau sebuah informasi. Dalam prakteknya kebanyakan diselesaikan dengan membuat perubahan tipis terhadap data digital lain yang isinya tidak akan menarik perhatian dari penyerang potensial. Kelebihan steganografi dari pada kriptografi adalah pesan-pesannya tidak menarik perhatian orang lain. Pesan-pesan berkode dalam kriptografi yang tidak disembunyikan, walaupun tidak dapat dipecahkan, akan menimbulkan kecurigaan. Seringkali, steganografi dan kriptografi digunakan secara bersamaan untuk menjamin keamanan pesan rahasianya [11].

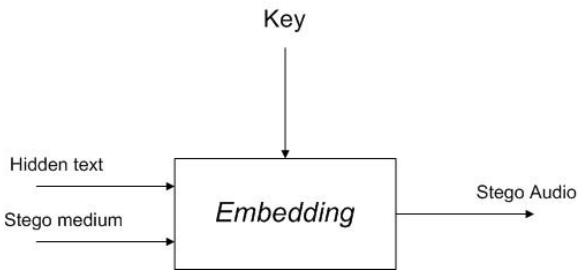
A.2 Kriteria Steganografi yang Baik

Menurut Munir, ada beberapa kriteria yang harus diperhatikan dalam steganografi [10], yaitu :

- *Imperceptibility*. Keberadaan pesan rahasia tidak dapat dipersepsi oleh inderawi. Misalnya, jika *cover text* berupa citra, maka penyisipan pesan membuat citra *stegotext* sulit dibedakan oleh mata dengan citra *cover text*-nya. Jika *cover text* berupa audio, maka indera telinga tidak dapat mendeteksi perubahan pada audio *stegotext*-nya.
- *Fidelity*. Mutu *stegomedium* tidak berubah banyak akibat penyisipan. Perubahan tersebut tidak dapat dipersepsi oleh inderawi. Misalnya, jika *cover text* berupa citra, maka penyisipan pesan membuat citra *stegotext* sulit dibedakan oleh mata dengan citra *cover text*-nya. Jika *cover text* berupa audio, maka audio stegotext tidak rusak dan indera telinga tidak dapat mendeteksi perubahan tersebut.
- *Recovery*. Pesan yang disembunyikan harus dapat diungkapkan kembali. Karena tujuan steganografi adalah *data hiding*, maka sewaktu- waktu pesan rahasia di dalam *stegotext* harus dapat diambil kembali untuk digunakan lebih lanjut.

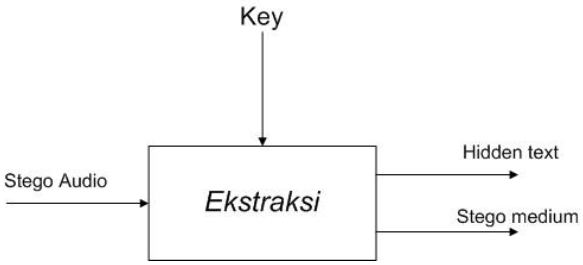
A.3. Proses Steganografi

Secara umum, terdapat dua proses di dalam steganografi. Yaitu proses penyisipan untuk menyembunyikan pesan dan prose ekstraksi untuk mengambil pesan tersembunyi. Proses-proses tersebut dapat dilihat pada gambar dibawah ini [3]:



Gambar 1. Penyisipan Audio [15]

Pada gambar 1, ditunjukkan proses penyembunyian pesan dimana pada bagian pertama, dilakukan proses *embedding hidden text* yang hendak disembunyikan secara rahasia ke dalam *stegomedium* sebagai media penyimpanan, dengan memasukkan kunci tertentu (*key*), sehingga dihasilkan media dengan data tersembunyi di dalamnya (*stegoaudio*).



Gambar 2. Ekstrasi Audio [15]

Gambar 2, dilakukan proses ekstraksi pada *stegoaudio* dengan memasukkan *key* yang sama sehingga didapatkan kembali *hidden text*. Kemudian dalam kebanyakan teknik steganografi, ekstraksi pesan tidak akan mengembalikan s*tegomedium* awal persis sama dengan *stegomedium* setelah dilakukan ekstraksi bahkan sebagian besar mengalami kehilangan. Karena saat penyimpanan pesan tidak dilakukan pencatatan kondisi awal dari *stegomedium* yang digunakan untuk menyimpan pesan [4].

A.4. Teknik Steganografi

Menurut Ariyus, ada tujuh teknik dasar yang digunakan dalam steganografi [2], yaitu :

1. *Injection*, merupakan suatu teknik menanamkan pesan rahasia secara langsung ke suatu media. Salah satu masalah dari teknik ini adalah ukuran media yang diinjeksi menjadi lebih besar dari ukuran normalnya sehingga mudah dideteksi. Teknik ini sering juga disebut *embedding*.
2. *Substitusi*, data normal digantikan dengan data rahasia. Biasanya, hasil teknik ini tidak terlalu mengubah ukuran data asli, tetapi tergantung pada *file* media dan data yang akan disembunyikan. Teknik substitusi bisa menurunkan kualitas media yang ditumpangi.
3. *Transform Domain*, teknik ini sangat efektif. Pada dasarnya, transformasi domain menyembunyikan data pada *transform space*. Akan sangat lebih efektif teknik ini diterapkan pada *file* berekstensi JPG.
4. *Spread Spectrum*, sebuah teknik pentransmisian menggunakan *pseudo-noise code*, yang independen terhadap data informasi sebagai modulator bentuk gelombang untuk menyebarkan energi sinyal dalam sebuah jalur komunikasi (*bandwidth*) yang lebih besar daripada sinyal jalur komunikasi informasi. Oleh penerima, sinyal dikumpulkan kembali menggunakan replika *pseudo-noise code* tersinkronisasi.
5. *Statistical Method*, teknik ini disebut juga skema *steganographic* 1-bit. Skema tersebut menanamkan satu bit informasi pada media tumpangan dan mengubah statistik walaupun hanya 1 bit. Perubahan statistik ditunjukkan dengan indikasi 1 dan jika tidak ada perubahan, terlihat indikasi 0. Sistem ini bekerja berdasarkan kemampuan penerima dalam membedakan antara informasi yang dimodifikasi dan yang belum.
6. *Distortion*, metode ini menciptakan perubahan atas benda yang ditumpangi oleh data rahasia.
7. *Cover Generation*, metode ini lebih unik daripada metode lainnya karena *cover object* dipilih untuk menyembunyikan pesan. Contoh dari metode ini adalah *Spam Mimic*.

B. Audio

B.1 Pengertian Audio

Audio adalah suara atau bunyi yang dihasilkan ketika molekul di udara berubah oleh suatu gerakan yang ditimbulkan sebuah objek yang menghasilkan sebuah getaran. . Jumlah getaran ini disebut sebagai frekuensi dari getaran tersebut. Satu kali gerakan maju dan mundur tersebut disebut *cycle* (putaran). Maka satuan untuk frekuensi adalah *cycle per second***,** atau cps. Satuan ini biasa dikenal dengan sebutan Hertz (Hz).

Menurut teori, agar sebuah suara dapat didengarkan oleh manusia, frekuensi minimal dari gerakan suara adalah 20 Hz dan frekuensi tertingginya adalah 20 kHz [18]. Pada kenyataan manusia dapat menangkap frekuensi tinggi yang mendekati antara 15 kHz dan 17 kHz. Binatang atau *microphone* mempunyai frekuensi yang berbeda.

B.2 Format Audio Populer

Ketika seseorang merekam suara, elektronik menyajikannya dalam bentuk gelombang. Agar bisa disimpan ke dalam komputer, gelombang itu diubah menjadi bentuk digital. Hal ini bisa dilakukan dengan mengambil *sample* sejumlah bagian gelombang per detiknya, lalu disimpan ke komputer dalam format *Waveform*.

Waveform digital *sample* disimpan dalam sebuah PC dengan format audio yaitu WAVE. *File* WAV adalah *file* audio standar yang digunakan oleh Windows [8]. Suara yang berupa digital audio dalam *file* WAV disimpan dalam bentuk gelombang, karena itulah file ini memiliki ekstensi .WAV (Wave). File WAV ini dapat dibuat dengan menggunakan berbagai program wave *editor* maupun wave *recorder*. Wave adalah standar format audio yang dikembangkan oleh

Microsoft dan IBM dengan ekstensi *file* (*.wav) yang mempunyai atribut Durasi Waktu, *Channels*, *Bit Deph* dan *Sample Rate*. Penjelasan dari masing – masing atribut tersebut adalah sebagai berikut [9]:

- Durasi Waktu
  Durasi waktu adalah panjang data suara yang terkonversi ke dalam format digital audio.

- *Channels*
  Channels adalah banyaknya suara yang di simpan dalam sebuah *file* dengan format Wave. Ada dua tipe *channels* pada format ini yaitu *mono* dan *stereo*. *Channels mono* mempunyai satu buah suara yang tersimpan pada format ini yaitu suara kiri atau suara kanan atau gabungan dari kedua buah suara tesebut. Sedangkan *stereo* mempunyai dua buah suara yang tersimpan yaitu suara kiri dan suara kanan.

- *Sample Rate*
  Wave dengan *sample rate* 44100Hz mempunyai ketepatan suara yang lebih dari 22050Hz. *Sample rate* 44100Hz adalah *rate* yang digunakan oleh Audio CD, sedangkan *rate* 22050Hz adalah *rate* yang digunakan oleh radio FM (*Frequency Modulation*). Semakin besar *rate* sebuah *file* wave, semakin besar pula kapasitas memori atau *hard disk space* yang digunakan oleh *file* tersebut.

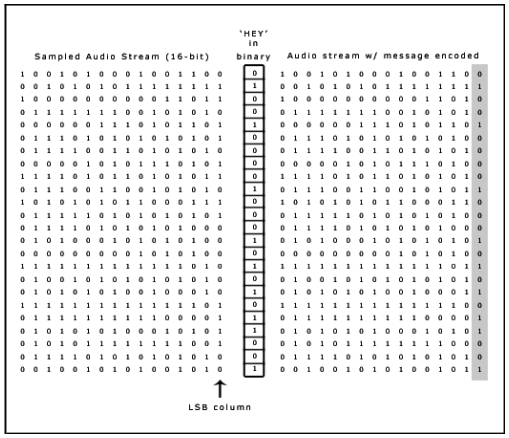
- *Bit Depth*
  Bit depth adalah nama lain dari sampling resolution. *Bit depth* merupakan banyaknya jumlah digit bilangan biner yang digunakan oleh *sample* suara. *Sample* dengan kedalaman 8 bit menggunakan sedikit memori atau *hard disk space* dari pada *sample* dengan kedalaman *bit* 16. Tapi semakin kecil jumlah *bit* suatu *sample* maka semakin sedikit jumlah data yang tersimpan pada PC.

C. Low Bit Coding / Least Significant Bit

Dalam audio metode *least significant bit disebut dengan low bit coding. Namun, teknik yang digunakan oleh low bit coding sama dengan least significant bit yaitu mengambil bit terakhir dan menggantinya dengan bit dari pesan yang akan disisipkan* [12].

Dalam penelitian ini LSB (*Least significant bit*) adalah algoritma terapan dari metode *substitusi. Substitusi* merupakan sebuah metode dimana data normal data normal digantikan dengan data rahasia. Teknik ini tidak terlalu mengubah ukuran data asli, tetapi tergantung pada *file* media dan data yang akan disembunyikan.

Dengan mengganti bit yang paling terakhir dari setiap bit pesan yang akan dimasukkan. Gambar 3 di bawah ini menjelaskan bagaimana pesan 'HEY' dikodekan dalam *sample* 16-bit dengan menggunakan metode *least significant bit.*



Gambar 3. Contoh penyisipan dengan metode *least significant bit* [5]

D. Advanced Encryption Standard (AES)

Dalam kriptografi, Anvanced Encryption Standard (AES), juga dikenal sebagai Rijndael, Rijndael adalah sebuah *block cipher* yang dijadikan standar enkripsi oleh pemerintah Amerika Serikat [16]. Algoritma Rijndael atau yang kerap disebut AES menggunakan substitusi, permutasi dan sejumlah putaran yang dikenakan pada tiap blok yang akan dienkripsi dan dekripsi [13].

Algoritma AES beroperasi dalam *byte*, bukan dalam *bit*. Algoritma ini mampu melakukan enkripsi terhadap *plain text* sebesar 16 *byte* atau 128 *bit*. Selain itu, algoritma ini juga menggunakan kunci sebanyak 16 *byte*. Dengan kunci sepanjang 128 *bit*, maka terdapat $2128 = 3{,}4 \times 10^{38}$ kemungkinan kunci [19]. Dengan demikian, waktu yang dibutuhkan untuk menebak kunci yang ada dengan komputer yang cepat pun membutuhkan 10^{18} tahun.

III. ANALISIS DAN PERANCANGAN SISTEM

A. Metode Penelitian

Pada bab sebelumnya telah dijabarkan perihal mengenai audio, metode dalam steganografi dan sistem operasi Android. Penelitian akan diawali dengan mengumpulkan materi yang mendukung fakta dalam penelitian seperti jurnal tentang steganografi, audio dan algoritma *least significant bit* (LSB) serta mencari buku-buku yang mendukung penelitian ini.

Setelah melakukan kajian teori, tiga komponen utama yang harus diperhatikan adalah media audio, teknik steganografi dan sistem operasi *mobile* yang dipergunakan untuk mengembangkan aplikasi. Penelitian ini akan menggunakan audio dengan format wav karena audio dengan format wav memiliki ukuran yang besar dan kualitas yang baik. Ukuran format wav yang besar mendukung metode steganografi yang akan digunakan dalam penelitian ini.

Metode steganografi yang digunakan dalam adalah metode *least significant bit*. Metode ini digunakan karena pesan hanya disisipkan dengan cara mengganti bit terakhir pada *file* audio [1]. Ini menunjukkan pada metode LSB ukuran pesan yang dimasukkan tidak akan merubah ukuran *file* audio aslinya. Dan juga metode ini cukup dapat menampung pesan tersembunyi yang besar. Meskipun dapat menampung pesan yang besar, LSB mudah ditemukan. Untuk menambah faktor keamanan maka sebelum dimasukan ke dalam audio pesan rahasia dienkripsi terlebih dahulu menggunakan AES (*Advanced Encryption Standard*).

Penelitian ini menggunakan metode prototyping untuk *software* development. Pengembangan aplikasi dimulai dengan pengumpulan kebutuhan yang diperlukan seperti menyiapkan seluruh *hardware* dan *software* untuk pengembangan aplikasi ini. Ketika seluruh kebutuhan telah terpenuhi, pengembangan dilanjutkan ke tahap analisis dan perancangan sistem. Pada tahap ini bentuk dan alur aplikasi dibuat dalam bentuk *flowchart*. Setelah perancangan sistem selesai, aplikasi mulai dibangun berdasarkan *flowchart* yang telah dibuat.

Proses pembangunan (coding) aplikasi juga disertai juga dengan uji coba aplikasi. Uji coba yang dilakukan melingkupi uji coba permodul dan uji coba secara keseluruhan. Proses pembangunan dan uji coba dilakukan secara berulang sampai aplikasi berjalan sesuai dengan rancangan yang telah dibuat. Aplikasi yang telah selesai dibuat dapat langsung di implementasikan ke dalam Android device.

B. *Alat yang Digunakan*

Dibawah ini adalah konfigurasi perangkat lunak dan perangkat keras yang digunakan selama proses pembuatan aplikasi.

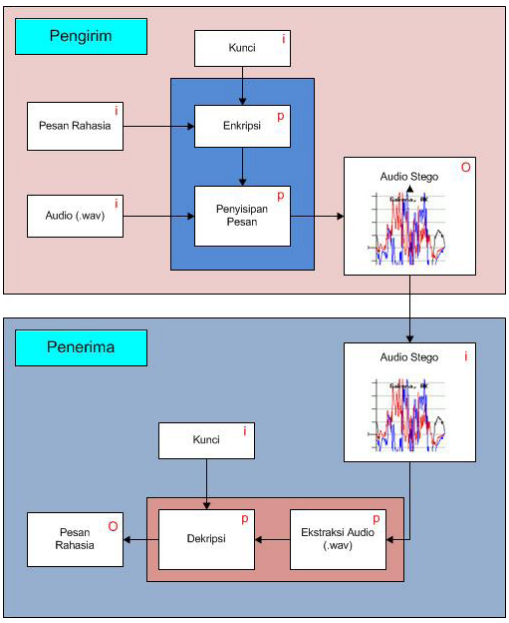
- Processor : Intel Core i5-480M
- Memori : DDR3 4GB
- Harddisk : 500GB
- Device : Samsung Galaxy Wonder i-850
- Sistem Operasi : Windows 7 Home Premium
- *System Type* : 32-bit Operating System
- Eclipse Indigo
- *Java Enterprise Edition* (J2EE) 6 versi 32-bit
- Android SDK
- Android *Development Tool*
- Android Virtual Device v 2.3 level 9

C. *Diagram Sistem*

Penelitian dimulai dari 1 Oktober 2011 sampai 24 Januari 2012. Gambar 4 mengilustrasikan aplikasi steganografi yang dirancang pada penelitian ini. Proses steganografi dibagi menjadi 2 bagian yaitu

1. *Embedded*
   Proses *embedded* dimulai dengan memilih *file* audio(.wav) yang akan menjadi media penyisipan pesan rahasia. Setelah *file* dipilih, pesan rahasia dan kunci dimasukkan. Ketika media penyimpanan dan pesan rahasia sudah ditentukan maka proses steganografi dapat dilakukan.

2. Ekstraksi
   Proses *embedded* dilakukan dengan memasukan audio yang mengandung pesan rahasia dan juga memasukan *password* yang telah disepakati sebelumnya. Sehingga proses ekstraksi dapat dilakukan dan *user* dapat mendapatkan pesan rahasia yang terkandung di dalam media penyimpanan.



Gambar 4. Rancangan Aplikasi Steganografi pada WAV

Keterangan:
i = *input*
o = *output*
p = proses

IV. METODE DAN PERANCANGAN APLIKASI

A. Implementasi Aplikasi

Aplikasi diimplementasikan dengan mengikuti rancangan yang telah dibuat pada proses sebelumnya. Pembuatan aplikasi dimulai dengan membuat *form* menu utama yang diikuti dengan pembuatan *form* lainnya yang merupakan konten dari menu utama.

Setiap *form* yang telah dibuat memiliki *class* tersendiri dan juga terdapat beberapa *class* yang merupakan *class utility* dimana *class* tersebut dapat dipanggil oleh *class* lainnya. *Class utility* dibuat untuk meningkatkan *reusability* dan juga menghindari *spaghetti code*.

AudioSteg menggunakan *class* bernama SteganografiLSB.java dalam implementasi *least significant bit.* Dibawah ini merupakan potongan *code* dari *class* tersebut.

```
public String Hide( String path, String msg, boolean b)
{
        String tmpfile = getFilenameSteg();
        FileInputStream in = null;
        FileOutputStream out = null;

        byte[] MsgN = null;

        try {
          MsgN = end.encrypt(msg);
          Log.d("hasil enkrip", new String(MsgN));
        } catch (Exception e) {
          e.printStackTrace();
        }

        try {
          in = new FileInputStream(path);
          out = new FileOutputStream(tmpfile);
        } catch (FileNotFoundException e) {
          e.printStackTrace();
        }

        int i = 0;
        int k = 0;
        int space= 20;
        byte[] buffer = new byte[44];

        try {
          in.read(buffer);
          out.write(buffer);
          buffer = new byte[32 * space];
          in.read(buffer);
          int len = MsgN.length;
          for (int l = 0 ; l < 32 * space ; l+= space ){
          buffer[l] = (byte) ((buffer[l]>>1)<<1);
          buffer[l] = (byte) ((buffer[l]) | ((len >> (31 - (l/
space))) & 1));
                out.write(buffer[l]);
                out.write(buffer, l+1, space-1);
          }
          buffer = new byte[space-1];
          while (k < (MsgN.length)){
                for(int j=0; j < 8; j++){
                        i = in.read();
                        i = (byte) ((i >> 1) << 1);
        i = (byte) ((byte) i ^ ((MsgN[k] >> j) & 1));
                        out.write(i);
                                in.read(buffer);
                                out.write(buffer);
                }
                k++;
          }

          buffer = new byte[1024];
          while ( (i = in.read(buffer,0,1024)) != -1){
                out.write(buffer,0,i);
          }
          out.close();
        } catch (IOException e) {
          e.printStackTrace();
        }
        return tmpfile;
}
```

Function *Hide* di atas berfungsi untuk menyembunyikan pesan rahasia ke dalam audio. Sebelum disembunyikan, pesan terlebih dahulu dienkripsi dan disimpan dalam *array* yang bertipe data *bytes*. Pesan terenkripsi tersebut lalu disisipkan ke dalam data audio tanpa merubah *header* audio sama sekali.Karena itu, sebelum memulai penyisipan pesan, *header* dari data audio sebesar 44 *bytes* disalin terlebih dahulu ke file *output*.

Setelah file header disalin, barulah pesan terenkripsi disisipkan ke dalam bit - bit file audio dan dituliskan ke file *output*. Ketika menyisipkan pesan, di awal pesan juga disisipkan besar dari pesan tersebut agar pesan dapat diekstraksi dengan benar saat diperlukan. Pesan disisipkan ke dalam data audio menggunakan metode LSB dimana bit terakhir dari 1 *byte* data audio diubah dengan 1 bit pesan rahasia.

```
public String Extract( String path, boolean b) throws
Exception{
FileInputStream in = null;
try {
        in = new FileInputStream(path);
} catch (FileNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
}
int i = 0;
byte[] bit = new byte[8];
byte tampung;
String Msg = "";
byte[] buffer = new byte[44];
int lengthPass = 0;
byte [] temp;
int space= 20;
try {
        in.read(buffer);
        lengthPass = in.read() & 1;
        buffer = new byte[space-1];
        for(int l = 1 ; l < 32 ; l++){
                in.read(buffer);
        lengthPass = (lengthPass << 1) | (in.read() &
1);
        }
        Log.d("Panjang text", lengthPass+"");
        temp = new byte[lengthPass];
        in.read(buffer);
        for( int a = 0; a < lengthPass; a++){
                for (int b1 = 0 ; b1 < 8 ; b1++){
                        i = in.read();
                        bit[7-b1] = (byte)(i & 1);
                        in.read(buffer);
                }
                tampung = bit[0];
                for(int b2 = 1 ; b2< 8 ; b2++){
        tampung = (byte) ((byte)(tampung << 1) |
bit[b2]) ;
                }
                temp[a] = tampung;
        }
        byte [] decryptMsg = null;
        decryptMsg = end.decrypt(temp);
        Msg = new String(decryptMsg);
} catch (IOException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
}
return Msg;
}
```

Sedangkan function Extract berfungsi untuk mengambil pesan yang sebelumnya telah disisipkan dengan *Hide*. karena pada saat penyembunyian *header* dari audio tidak diubah, maka pada saat ekstraksi pun *header file* sebanyak 44 byte sengaja dilewati.

Setelah *header* dilewati, bit - bit terakhir dari data audio yang merupakan bagian dari pesan tersembunyi diambil dan disusun kembali menjadi sebuah pesan terenkripsi dan ditampung ke dalam sebuah *array* bertipe *data bytes*. *Array of bytes* ini kemudian didekripsi untuk mendapatkan pesan asli yang disembunyikan.

B. Pengujian Aplikasi

Pada bab sebelumnya telah dijelaskan bahwa ada 3 kriteria yang perlu dipenuhi sebagai steganografi yang baik yaitu *imperceptibility, fidelity* dan *recovery*. Untuk membuktikan bahwa penelitian yang telah dibuat memenuhi tiga kriteria diatas maka dilakukan wawancara kepada 20 responden di Universitas Multimedia Nusantara. Teknik pengukuran yang digunakan dalam aplikasi ini adalah pertanyaan tertutup dengan jawaban ya/tidak.

Dari wawancara yang telah dilakukan, dapat dibuktikan bahwa metode yang digunakan dalam penelitian ini telah memenuhi kriteria dari steganografi yang baik. Bukti yang menunjukan penelitian ini telah memenuhi kriteria *imperceptibility* dapat dilihat dari tabel 1 dibawah ini.

Tabel 1. Hasil Wawancara Perbandingan Audio

	Audio dengan 160 karakter		Audio dengan 520 karakter		Audio dengan 6500 karakter	
Audio asli	Sama	Berbeda	Sama	Berbeda	Sama	Berbeda
	17	3	18	2	20	0

Dari tabel di atas dapat disimpulkan bahwa 85% dari responden tidak dapat mendeteksi adanya perbedaan antara audio asli dengan audio yang berisi pesan sebanyak 160 karakter. Sedangkan untuk audio yang berisi pesan sebanyak 520 karakter, 90% dari responden tidak dapat membedakannya dari audio asli. Pada audio yang diisikan pesan secara maksimal, seluruh responden tidak dapat membedakan audio tersebut dengan audio asli.

Dari data di atas dapat disimpulkan bahwa metode yang digunakan dalam penelitian ini telah memenuhi kriteria *imperceptability* dengan baik. Tampak dari data bahwa responden yang diwawancara memiliki kesulitan untuk mendeteksi adanya perbedaan antara audio asli dengan audio hasil steganografi.

Selain perbedaan audio ketika didengar oleh manusia, perbedaan secara fisik seperti perbedaan besar *file* antara *file* audio juga dapat menimbukan kecurigaan adanya data tersembunyi. Idealnya, implementasi metode LSB di audio tidak merubah besar *file* dari audio sehingga tidak menimbulkan kecurigaan. Besar audio tidak berubah karena metode ini hanya merubah data audio yang ada, bukan menambah atau mengurangi. Berikut merupakan tabel yang menunjukkan hasil steganografi yang menggunakan pesan dengan panjang bervariasi.

Tabel 2. Tabel Uji coba penyisipan pesan

File (.wav)	Ukuran (Mb)	Panjang Pesan	Ukuran akhir (Mb)
Audio1.wav	1.21 MB (1,276,300 *bytes*)	0 karakter	1.21 MB (1,276,300 *bytes*)
		160 karakter	1.21 MB (1,276,300 *bytes*)
		520 karakter	1.21 MB (1,276,300 *bytes*)
		6500 karakter	1.21 MB (1,276,300 *bytes*)

Dari tabel hasil uji coba di atas dapat dilihat bahwa panjang pesan yang berbeda-beda yang digunakan dalam uji coba dapat disisipkan ke dalam *file* wav. tanpa merubah ukuran *file*.

Dalam penelitian ini, penyimpanan 1 karakter (1 *byte*) pesan memerlukan 160 *byte* data audio. Hal ini dikarenakan pada 1 *byte* pesan terdapat 8 bit yang harus disimpan pada data audio. Setiap bit pesan disimpan ke dalam bit terakhir dari 1 *byte* data audio. Namun, penyimpanan setiap bit pesan selalu diselang 19 *byte* untuk meminimalisir munculnya noise pada audio hasil steganografi.

Selain menyimpan pesan, akan disimpan pula panjang dari pesan yang akan disembunyikan. Panjang dari pesan yang akan disembunyikan menggunakan tipe data integer. Tipe data integer memerlukan tempat sebanyak 4 *byte*, sehingga untuk disimpan ke dalam audio diperlukan 4 x 8 x 20 *byte* = 640 *byte* data audio.

Dari penjelasan diatas, maka dapat didapatkan perhitungan besar *file* minimum untuk proses steganografi sebagai berikut.

BFM = 44 byte header wav + 640 byte + Bp(byte) x 160

BFM = Besar *file* minimum
Bp = Besar pesan

Rumus di atas digunakan untuk menentukan besar *file* minimum dari audio yang akan disisipi pesan rahasia sehingga dengan adanya rumus ini dapat mencegah penyembunyian pesan yang terlalu besar ke dalam *file* audio.

Mutu dari suatu metode steganografi dapat diukur dengan melihat bagaimana responden dapat mengidentifikasi audio. Pada wawancara terdapat 4 audio yang perlu didengarkan oleh responden. Audio tersebut disisipi dengan pesan - pesan tersembunyi seperti yang telah dijabarkan pada tabel 2. Responden yang telah mendengar ke 4 audio, diminta untuk menetukan *file* audio mana yang asli. Hasil dari pilihan responden dapat dilihat pada tabel dibawah ini.

Tabel 3. Hasil Wawancara untuk Pemilihan Audio Asli

	Audio 1	Audio 2	Audio 3	Audio 4	Tidak bisa dibedakan
Audio asli	6	4	3	2	5
				Jumlah	20

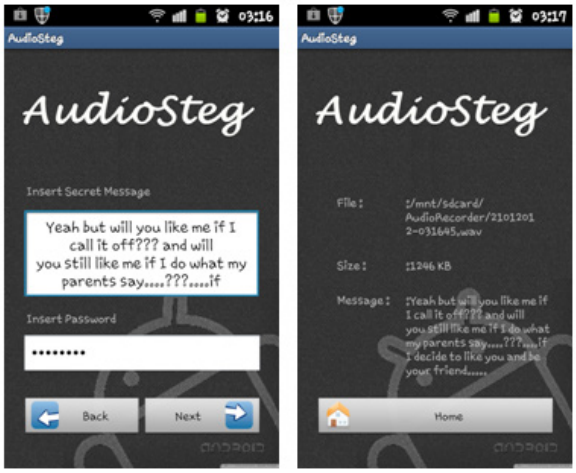
Dari hasil wawancara, dapat dilihat presentase responden yang memilih audio 1 sebagai audio yang asli sebesar 30%, audio 2 sebagai audio asli 20%, audio 3 sebagai audio asli 15%, audio 4 sebagai audio asli sebesar 10% dan responden yang tidak dapat membedakan mana audio yang asli sebanyak 20%.

Hasil wawancara menunjukkan jawaban yang diberikan oleh responden beragam. Dari 20 responden hanya 3 responden yang dapat menjawab dengan benar bahwa audio 3 merupakan audio yang asli. Oleh karena itu, ditarik kesimpulan bahwa audio belum dapat dibedakan dengan menggunakan indera pendengaran dan mutu dari audio hasil steganografi cukup mengimbangi mutu dari audio asli.

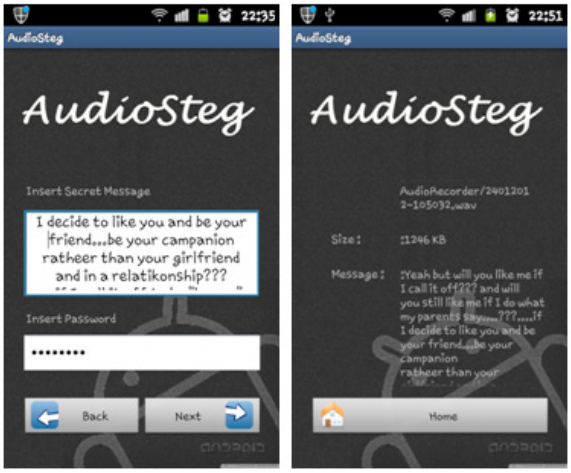
Kriteria steganografi yang terakhir adalah *recovery.* Pemenuhan kriteria ini dapat dibuktikan dengan menyisipkan 3 pesan dengan jumlah karakter yang berbeda - beda. Hasil ekstraksi harus sesuai dengan hasil penyisipan yang sudah dilakukan. Audio yang disisipkan berdurasi 7 detik dan memiliki ukuran 1,2MB. Pada percobaan ini ekstraksi dilakukan sebanyak 10 kali untuk setiap pesan yang disembunyikan dan hasil ekstraksi harus sama tiap pengujian.

Berikut merupakan tampilan langsung dari aplikasi AudioSteg untuk *hide* pesan dan ekstraksi pesan. Pada tampilan tersebut juga dapat dilihat bahwa pesan yang disembunyikan dan diekstrak tidak berubah. Dibawah

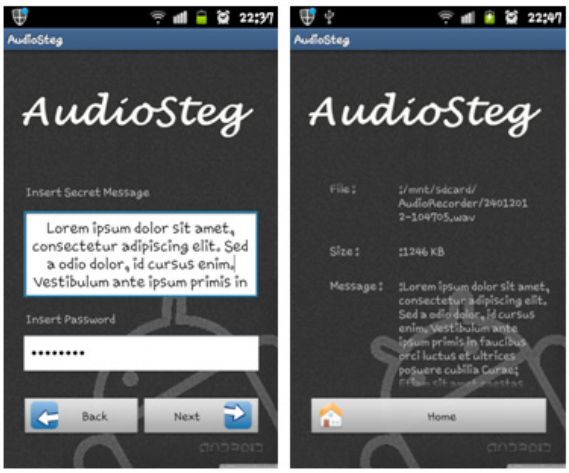
ini merupakan *screen shoot* dari hide dan ekstraksi pesan.



Gambar 5. Screen Hide dan hasil ekstraksi pesan untuk 160 karakter



Gambar 6. Screen Hide dan hasil ekstraksi pesan untuk 520 karakter



Gambar 7. Screen Hide dan hasil ekstraksi pesan untuk 6500 karakter

V. SIMPULAN DAN SARAN

A. *Simpulan*

Penelitian berhasil menunjukan implementasi steganografi dalam sistem operasi android dengan menggunakan metode *least significant bit* dan penambahan fitur enkripsi untuk meningkatkan keamanan. Kualitas yang dimiliki aplikasi AudioSteg baik karena telah memenuhi kriteria steganografi berdasarkan ukuran *imperceptibility*, *fidelity* dan *recovery.*

B. *Saran*

Pada penelitian ini informasi yang disisipkan berupa teks sehingga untuk penelitian selanjutnya informasi yang disisipkan dapat dikembangkan tidak selalu teks tetapi dapat berupa *file* ataupun gambar. Untuk format audio populer yang digunakan. Format dapat diganti dengan menggunakan MP3 yang memiliki size yang lebih kecil atau format OGG yang merupakan format terbuka dan bebas digunakan siapapun. Penelitian ini dapat dilanjutkan dengan menguji coba metode steganografi yang lain atau mencoba mengembangkan metode *least significant bit* dalam audio untuk mendapatkan hasil yang terbaik.

DAFTAR PUSTAKA

[1] Ardhyana, Alfebra Stavia dan Asep Juarna. 2008. Aplikasi Steganografi Pada Mp3 Menggunakan Teknik LSB.

[2] Ariyus, Dony. 2009. Keamanan Multimedia. Yogyakarta : Andi.

[3] Bender, W. – Gruhl, D. – Morimoto, N.,. 1995. Techniques for Data Hiding, Massachusetts Institute of Technology, Media Laboratory Cambridge, Massachusetts 02139 USA, From the Proceedings of the SPIE, 2420:40, San Jose CA, February.

[4] Cox, I. J., Miller, M. L., Bloom, J. A., Fridrich, J. dan Kalker, T. 2008. Digital Watermarking and Steganography. 2nd Edition. Burlington: Morgan Kaufmann.

[5] Gibson, Tyler. 2007. *Methods of Audio Steganography*. URL: http://www.snotmonkey.com/work/school/405/methods.html#eval. Tanggal akses: 15 Desember 2011.

[6] Gultom, Lipantri Mashur. 2011. Modifikasi Algoritma Arithmetic Coding Dalam Mengatasi Kelemahan Pada Kompresi File Teks.

[7] Gunawan, Andy. 2009. Penyembunyian Pesan Teks pada File WAV dengan Metode Least Significant Bit.

[8] Gunawan, Ibnu dan Kartika Gunadi. 2005. Pembuatan Perangkat Lunak Wave Manipulator Untuk Memanipulasi File Wav.

[9] Lu, Chun-Shien(2005), Multimedia Security : Steganography and Digital Watermarking Techniques for Protection of Intellectual Property, Institute of Information Science Academia Sinica, Taiwan, ROC

[10] Munir, Rinaldi. 2006. Kriptografi. Bandung : Informatika.

[11] Nasution, Atika Sari Alam. 2010. Teknik Penyembunyian Citra Digital Pada File Video Dengan Metode End Of File.

[12] Noto, Mark,. 2001. MP3Stego: Hiding Text in MP3 Files. SANS Institute.

[13] Satria, Eko. 2009. Algoritma Rijndael dalam Sistem Keamanan Data.

[14] Susanto, Agus (2004). Studi dan Implementasi Steganografi pada Berkas MIDI.Susatio, Yerri & Aulia Siti Aisyah. Identifikasi Kerusakan Mesin Berputar Berdasarkan Sinyal dengan Metode Adaptive Neuro Fuzzy Inference System.

[15] Utami, Ema. 2009. Pendekatan Metode Least Bit Modification Untuk Merancang Aplikasi Steganography Pada File Audio Digital Tidak Terkompresi. JURNAL DASI Vol. 10 No. 1

[16] Virgan R.Y., Agung B.P., Aghus Sofwan. 2008. Aplikasi Enkripsi Dan Dekripsi Menggunakan Algoritma Rijndael.

[17] Wahono, Tri. 2011. Android Kini Platform "Smartphone" Terbesar di Dunia. URL: http://tekno.kompas.com/read/2011/02/01/22572437/Android.Kini.Platform.Smartphone.Terbesar.di.Dunia. Tanggal akses: 24 Februari 2012.

[18] Waluyanti, Sri dkk. 2008. Teknik Audio Video.

[19] Panggabean, Igor Bonny Tua. Perbandingan Algoritma RC6 dengan Rijndael pada AES.