Perancangan Aplikasi Penyisipan Pesan Teks Dalam Audio Dengan Mengunakan Metode Low Bit Coding

Muhammad Ramadani
STMIK Budi Darma Medan, Jl. Sisingamangaraja No. 338 Sp. Limun, Sumatera Utara, Indonesia
E-Mail : muhmmadramadani36@gmail.com

ABSTRACT
The development of steganography began to penetrate into the digital world which in this study was devoted to mp3 audio files. The techniques for developing a steganography process are briefly discussed. This study also tried to present the steganography process in audio file comparisons based on the formulas in the existing literature. A software called MP3 Stego is used to assist in the process of observing problems and completing the steganography process. MP3 Stego is tried, observed, and hypothesized for future development suggestions. After looking more deeply at audio steganography, especially in MP3 audio files, it is quite prospective in the next development. Well studied from psychological and technical factors, it turns out that MP3 media is a very appropriate media to become a message hiding media. In addition, it turns out that steganography can also be used in more in-depth and broader targets such as campaign suggestions, notifications, organizational communication, and so on..

Kata kunci : MP3, Steganografi, Capacity MP3 Stego

PENDAHULUAN

Pesan rahasia merupakan hal penting yang butuh untuk dilindungi dan dijaga kerahasiaannya. Pesan rahasia merupakan sebuah harta karun dimana banyak orang yang ingin berusaha untuk mencari terlebih mengetahui isinya[1]. Oleh karena itu maka tidak jarang muncul kejahatan-kejahatan yang dengan sengaja dilakukan oleh orang yang tidak bertanggung jawab. Dengan semakin banyaknya orang yang melakukan tindakan kriminal yang dengan sengaja melakukan pencurian data rahasia dan merusak data rahasia sehingga bisa merugikan pihak tertentu[2], [3].

Steganografi adalah ilmu menyembunyikan teks pada media lain yang telah ada sehingga teks yang tersembunyi menyatu dengan media pembawa (*carrier*). Media tempat penyembunyian pesan tersembunyi dapat berupa media teks, gambar, audio atau video[4].

Metode *Low Bit Coding* adalah cara untuk menyimpan data kedalam *file audio* yang diimplementasikan dengan mengganti *bit-bit* yang paling akhir atau *low significant bit (LSB)* pada setiap titik *sampling* dengan string berkode biner (*coded binary string*), dapat menyisipkan sejumlah besar data ke dalam suara digital. Setiap aplikasi yang dibuat dapat menyisipkan karakter pesan teks untuk digunakan sebagai media penyisipan, aplikasi steganografi membutuhkan waktu proses yang relatif lama saat melakukan penyisipan teks dan proses penyisipan teks yang terjadi pada *file audio* tidak menyebabkan perubahan suara, sehingga suara yang terdengar tidak dapat dibedakan dengan *file audio* aslinya[5].

Pada penelitian ini, penulis merumuskan masalah sebagai berikut:

1. Bagaimana proses penyisipan teks ke dalam *audio*?
2. Bagaimana menerapkan metode *low bit coding* dalam menyisipkan pesan teks*?*
3. Bagaimana merancang aplikasi penyisipan pesan teks dalam audio?

Agar masalah tidak menyimpang dari rumusan masalah maka penulis membatasi pembahasan penelitian ini. Adapun batasan masalah yaitu sebagai berikut:

1. *File* yang akan dibahas adalah *audio* dengan format *MP3,* yang berukuran maksimal 500 *KB.*
2. Pesan yang dapat disisipkan adalah pesan dalam bentuk teks dan angka tidak berbentuk gambar.
3. Pesan teks yang mau disisipkan diketik secara *manual*.

LANDASAN TEORI

2.1 *Audio MP3*

MP3 adalah salah satu format berkas pengodean suara yang memiliki kompresi yang baik sehingga ukuran berkas bisa memungkinkan menjadi lebih kecil. *MP3* dikembangkan oleh seorang insinyur yang

berasal dari jerman Karlheinz Brandenburg. *MP3* memakai pengodean *PCM (Pulse Code Modulation).* Berkas yang mempunyai nama lengkap *MPEG-1 Audio Layer* ini mengurangi jumlah bit yang diperlukan dengan menggunakan model psychoacoustic untuk menghilangkan komponen-komponen suara yang tidak terdengar oleh manusia[5].

MP3 memakai sebuah transformasi *hybrid* untuk mentransformasikan sinyal pada ranah waktu ke sinyal pada ranah frekuensi:

1. *Filter polyphase quadrature 32-band*
2. 36 atau 12 *MDCT (modified discrete cosine transform)*, dengan ukuran dapat dipilih secara independen untuk sub-band 0…1 dan 2…31
3. *Postproses aliasing reduction*

Standar *MPEG-1* tidak menspesifikasikan secara spesifik cara melakukan encode *MP3*. Sebaliknya, algoritma *decode* serta *format file* didefinisikan secara spesifik. Yang ingin mengimplementasikan *encoder MP3* harus membuat sendiri algorima untuk menghilangkan bagian dari informasi pada *file audio* asal (atau pada representasi *MDCT* pada ranah frekuensi).

Karena itu, maka cara *encode* setiap *encoder MP3* berlainan dan menghasilkan kualitas hasil yang berlainan juga. Hal yang harus diperhatikan adalah dari semua *encoder* yang ada, terdapat encoder yang bagus untuk *bit rate* tinggi maupun *encoder* yang bagus untuk *bit rate* rendah.

MP3 mempunyai beberapa batasan/limit:

1. Bit rate terbatas, maksimum 320 *kbit/s* (beberapa *encoder* dapat menghasilkan *bit rate* yang lebih tinggi, tetapi sangat sedikit dukungan untuk *mp3-mp3* tersebut yang memiliki *bit rate* tinggi)
2. Resolusi waktu yang digunakan *mp3* dapat menjadi terlalu rendah untuk sinyal-sinyal suara yang sangat transient, sehingga dapat menyebabkan *noise*.
3. Resolusi frekuensi terbatasi oleh ukuran *window* yang panjang kecil, mengurangi efisiensi *coding*
4. Tidak ada *scale factor* band untuk frekuensi di atas 15,5 atau 15,8 kHz
5. Mode *jointstereo* dilakukan pada basis per *frame*
6. Delay bagi *encoder/decoder* tidak didefinisikan, sehingga tidak ada dorongan untuk *gapless playback* (pemutaran *audio* tanpa *gap*). Tetapi, beberapa encoder seperti *LAME* dapat menambahkan metadata tambahan yang memberikan informasi kepada *MP3* player untuk mengatasi hal itu.

2.2 Pengertian Bit Rate

Rate dalam bahasa Indonesia adalah suatu kecepatan. Sedangkan *Bit* (kepanjangan dari *Binary* Digital) dalam bahasa Indonesia adalah kepingan atau potongan..

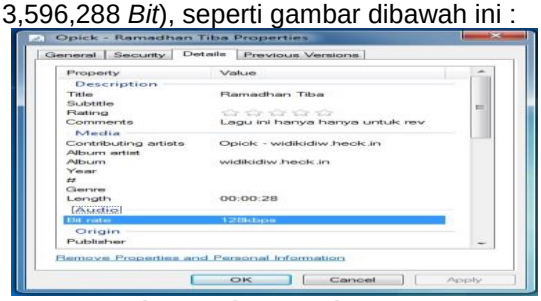
Dari definisi tersebut, definisi *Bit* agak rancu bila digabungkan dengan definisi *Rate*. Namun akan lebih mudah bila mendefinisikan *Bit* menjadi suatu digit antara 1 atau 0 yang merupakan awal pembentukan sebuah *char* (huruf) atau bilangan. Diketahui bahwa setiap huruf atau bilangan itu dibentuk dari kumpulan bit-bit. Pembentuk kata “Komputer” terdiri dari beberapa *bit* pembentuk huruf “K”,”o”,”m”,”p”,”u”,”t”,”e”, dan “r”.

Bit dan *Byte* keduanya sama-sama membentuk suatu karakter. Namun 1 *Byte* itu berisikan 8 buah *Bit* yang merepresentasikan sebuah karakter pada penyimpanan data di *memory system* perkomputeran[6]. Ada hal penting yang harus dipahami bahwa *Byte* biasanya digunakan untuk mengetahui kapasitas ukuran data pada suatu media penyimpanan seperti *harddisk*, *flashdisk*, dan lain-lain. Ukurannya pun dibagi-bagi menjadi *Kilo Byte (KB), Mega Byte (MB), Giga Byte (GB)*, dll.

Sedangkan Bit digunakan untuk mengetahui ukuran kecepatan suatu transfer data dari tempat satu (terminal) ke tempat lain. Jadi kalau sedang mendownload data, lalu lihat kecepatan transfernya yaitu 60 *Kbps*, maka artinya adalah kecepatan data masuk adalah 60 *Kilo Bit per Second* (detik) bukan 60 *Kilo Byte per Second*.

Dari uraian di atas, dapat didefinisikan bahwa bit rate adalah suatu ukuran kecepatan bit suatu data dari tempat satu ke tempat lain yang biasanya diukur dengan waktu seperti *Kbps (Kilobit per second), Mbps (Megabit per second)* dan seterusnya.

Sebagai contoh penggunaan *bit rate* pada *mp3* dengan judul Opick – Ramadhan Tiba, dengan ukuran 438 *Kb* (449,536 *Byte* / 3,596,288 *Bit*), seperti gambar dibawah ini :



Gambar 1 Ukuran *Bit rate MP3*

Dari gambar, dapat kita lihat bahwa *file mp3* ini memiliki bit rate sebesar 128 *kbps*. Maksudnya adalah saat mendengarkan lagu ini, tiap suara yang keluar, diukur dari *bit rate*-

nya. Karena seperti penjelasan di atas bahwa *bit* adalah pembentuk dari suatu *byte* dimana *byte* itu merepresentasikan sebuah karakter. Jadi semakin besar *bit rate*-nya maka semakin halus juga lagu yang didengarkan. Sama seperti hal nya dengan video, semakin besar *bit rate*-nya semakin halus pula tampilan video nya.

2.3 Teknik Penyisipan Pesan dalam Audio

Pada *audio* terdapat beberapa teknik dalam penyisipan pesan (informasi), diantaranya adalah *Low Bit Coding*, *Phase Coding*, *Spread Spectrum*, dan *Echo Data Hiding*[4].

1. *Low Bit Coding*
   Adalah cara untuk menyimpan data kedalam *file audio*, mengganti *bit* yang paling tidak penting atau *low significant bit (LSB)* pada setiap titik *sampling* dengan string berkode biner (*coded binary string*), Kelemahan metode ini adalah lemahnya kekebalan terhadap manipulasi. Pada prakteknya, metode ini hanya berguna pada lingkungan *digital-to-digital* yang tertutup.
2. *Phase Coding*
   Adalah merekayasa fasa dari sinyal masukan. dengan mensubstitusi awal fasa dari tiap awal segmen dengan fasa yang telah dibuat sedemikian rupa dan merepresentasikan pesan yang disembunyikan. Hal ini menghasilkan keluaran yang lebih baik namun dikompensasikan dengan kerumitan dalam realisasinya.
3. *Spread Spectrum*
   Adalah penyebaran *spektrum.*, pesan dikodekan dan disebar ke setiap spektrum frekuensi yang memungkinkan. Maka dari itu akan sangat sulit bagi yang akan mencoba memecahkannya kecuali ia memiliki akses terhadap data tersebut atau dapat merekonstruksi sinyal *random* yang digunakan untuk menyebarkan pesan pada range frekuensi.
4. *Echo Data Hiding*
   Adalah menyembunyikan pesan melalui teknik *echo*, menyamarkan pesan ke dalam sinyal yang membentuk *echo*, pesan disembunyikan dengan menvariasikan tiga parameter dalam *echo* yaitu besar *amplitudo* awal, tingkat penurunan *atenuasi* dan *offset*.

2.4 *Low Bit Coding*

Pada dasarnya, metode steganografi *low bit coding* pada *audio* sama saja dengan metode steganografi *least signifikan bit (LSB)*[4][2] pada *image(citra)* . pada metode ini sebagian bit pada *file audio* diubah menjadi nilai lain dalam representasi *biner*. Perubahan dapat dilakukan dengan berbagai cara dan algoritma, misalnya mengubah nilai biner 0 menjadi 1 atau sebaliknya, melakukan operasi *XOR* antara nilai biner pada *file* dengan nilai biner pada kunci, karena dalam representasi biner, maka perubahan yang mungkin terjadi adalah nilai 1 menjadi 0, atau nilai biiner 0 menjadi 1.

Suatu *file audio* dapat memiliki satu channel (*mono)* atau dua *channel* (*stereo*). Secara umum, kapasitas satu channel adalah *kbps per kilohertz*, karena ukuran channel dapat mencapai 44000 *byte*, maka kapasitas maksimal yang dapat ditampung oleh satu channel adalah 44 *kbps per kilohertz.*

Perancangan Aplikasi Penyisipan Data Pada *Audio* Dengan Mengunakan Metode *Low Bit Coding.*

Metode steganografi yang paling umum pada tipe berkas *audio* dan gambar adalah low bit coding atau disebut juga least significant bit. Metode ini berasal dari angka yang paling kurang signifikan dari jumlah bit dalam 1 byte. Bit yang memiliki signifikan paling tinggi adalah numerik yang memiliki nilai tertinggi artinya yang paling tidak signifikannya adalah yang memiliki nilai terrendah[4].

Sebagai contoh menyisipkan karakter ‘R’ pada 8 byte berkas carrier atau bit dari media pembawa. low bit coding ditandai dengan garis bawah dan tebal.

00010101 00101100 00001001 00110100
01101001 01001001 10101001 10001100

Dimana 01010010 adalah bentuk huruf “R” dalam biner. kedelapan bit ini bisa dituliskan kedalam low bit coding dari tiap-tiap byte pada kedelapan carrier seperti berikut ini :

00010100 00101101 00001000 00110101
01101000 01001000 10101001 10001101

Pada contoh di atas, hanya sebagian dari low bit coding yang berubah. Dengan mengubah low bit coding, nilai dari byte tidak akan berubah banyak, sehingga akan sulit dideteksi oleh telinga manusia.

Untuk melakukan proses ekstraksi pesan stego sehingga pesan kembali ke nilai awal yaitu dengan mengambil setiap bit akhir pada file stego. Berikut langkah-langkah atau rumus untuk proses ekstraksi pada suatu pesan :

1. Membuat blok-blok data ke dalam 8 byte per blok. Untuk setiap blok dikerjakan langkah “2” sampai dengan langkah “3” untuk i = 0,1,2,3,..,7.

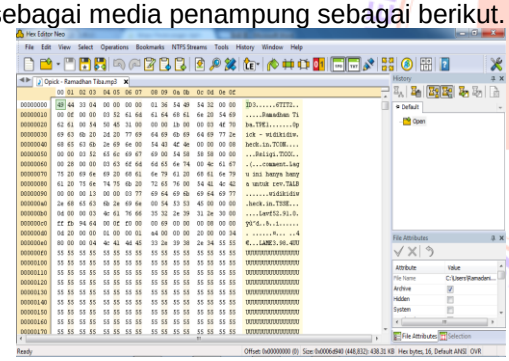
2. Mengambil nilai bit terakhir byte pesan ke-i dengan meng-and-kan dengan 1.
3. Menyimpan hasil setelah di-and-kan dengan 1, dan mengalikan dengan nilai posisi bit, yaitu : (2(7-i)).
4. Menjumlahkan semua hasil perhitungan untuk i=0 sampai dengan i=7.
5. Menentukan karakter ASCII yang bersesuaian dengan hasil perhitungan.

PEMBAHASAN

3.1. Analisa Media Audio

Metode yang dipakai dalam proses penyisipan *bit-bit* data ke dalam *byte-byte MP3* adalah menggunakan teknik penyisipan pada *Low Bit Coding*. *Low Bit Coding* adalah cara yang paling sederhana untuk menyimpan data ke dalam *file audio*. Teknik ini diimplementasikan dengan mengganti bit yang paling tidak penting atau *low significant bit (LSB)* pada setiap titik *sampling* dengan string berkode biner (*coded binary string*), kita dapat menyisipkan sejumlah besar data ke dalam suara digital.

File MP3 yang digunakan sebagai media yaitu "**Opick - Ramadhan Tiba.mp3**" berukuran 439 *Kb* (449,536 *bytes*). Adapun contoh *hexadecimal audio* yang digunakan sebagai media penampung sebagai berikut.



Gambar 2 *Audio* dalam *Hexa* Menggunakan *Hex Editor Neo*

3.2 Analisa Teks (Pesan)

Untuk melakukan analisa terhadap teks maka proses pertama yang akan dilakukan yaitu dengan mengubah nilai-nilai karakter menjadi bentuk biner. Operasi dilakukan per *byte* dimana data teks diubah kedalam bentuk biner.

Plaintext : RAMADANI

Jumlah karakter pada plainteks untuk pesan rahasia yang disembunyikan yaitu 16 karakter. Berikut tabel yang digunakan sebagai *plaintext*, dimana nilai-nilai karakter diubah menjadi nilai biner.

Tabel 1 Plainteks

Carakter	R	A	M	A	D	A	N	I
Hexa	52	41	4D	41	44	41	4E	49
Decimal	82	65	77	65	68	65	78	73
Biner	01010010	01000001	01001101	01000001	01000100	01000001	01001110	01001001

3.3 Penerapan Metode Low Bit Coding

Proses penyisipan pesan rahasia yaitu bagaimana pesan rahasia disisipkan pada sebuah *file audio* sehingga *file* tersebut tidak diketahui keberadaannya. Pada proses ini. Proses penyisipan membutuhkan dua buah masukan yaitu media *cover* sebagai tempat penyisipan pesan rahasia. Media *Cover* yang digunakan adalah *file audio* dan pesan rahasia yang dapat disisipkan berupa teks. *File audio* yang digunakan adalah *file audio MP3* sedangkan pada proses penguraian hanya dibutuhkan satu buah masukan. Masukan tersebut adalah *file audio MP3* yang telah disisipi pesan rahasia.

Untuk melanjutkan cara kerja dalam penyisipan pesan rahasia ini, cara kerja yang dilakukan adalah dengan menginputkan *file audio* dan menyimpan *file audio* yang telah disisipi pesan dan simpan *file audio* yang telah disisipkan pesan.

Output yang dihasilkan juga bergantung pada proses yang akan dilakukan pada proses penyisipan, keluaran yang dihasilkan adalah *audio MP3* yang telah disisipi pesan rahasia. Sedangkan pada proses ekstraksi, keluarannya adalah pesan rahasia yang dapat berupa teks.

Berikut proses *encoding* dan *decoding* dalam penyisipan pesan teks dalam audio.

3.3.1 *Encoding*

Adapun prosedur *encoding* pesan rahasia ini adalah :

1. Menyediakan *file audio* sebagai media penampungan pesan.
2. Meng-*input*-kan pesan rahasia kedalam media *cover*.
3. Memproses pesan rahasia yang akan disisipkan kedalam media *cover*.
4. Pesan rahasia sudah tersembunyi.

Adapun contoh studi kasus proses penyisipan pesan rahasia tersebut yaitu sebagai berikut.

1. Misalnya media suara yang akan disisipkan mempunyai panjang bit 8, dengan nilai seperti berikut. Berikut adalah *file audio* sebagai media penampung pesan.

Tabel 2 Media *file audio* dalam bentuk Biner

Baris 0	01001001	01000100	00110011	00000100	00000000	00000000	00000000	00000000
	00000001	00110110	01010100	01001001	01010100	00110010	00000000	00000000
Baris 1	00000000	00001111	00000000	00000000	00000011	01010010	01100001	01101101
	01100001	01100100	01101000	01100001	01101110	00100000	01010100	01101001
Baris 2	01100010	01100001	00000000	01010100	01010000	01000101	00110001	00000000
	00000000	00000000	00011011	00000000	00000000	00000011	01001111	01110000
Baris 3	01101001	01100011	01101011	00100000	00101101	00100000	01110111	01101001
	01100100	01101001	01101011	01101001	01100100	01101001	01110111	00101110
Baris 4	01101000	01100101	01100011	01101011	00101110	01101001	01101110	00000000
	01010100	01000011	01001111	01001110	00000000	00000000	00000000	00001000
Baris 5	00000000	00000000	00000011	01010010	01100101	01101100	01101001	01100111
	01101001	00000000	01010100	01011000	01011000	01011000	00000000	00000000
Baris 6	00000000	00100110	00000000	00000000	00000011	01100011	01101111	01101101
	01101101	01100101	01101110	01110100	00000000	01001100	01100001	01100111
Baris 7	01110101	00100000	01101001	01101110	01101001	00100000	01101000	01100001
	01101110	01111001	01100001	00100000	01101000	01100001	01101110	01111001
Baris 8	01100001	00100000	01110101	01101110	01110100	01110101	01101011	00100000
	01110010	01100101	01110110	00000000	01010100	01000001	01001100	01000010
Baris 9	00000000	00000000	00000000	00010011	00000000	00000000	00000011	01110111
	01101001	01100100	01101001	01101011	01101001	01100100	01101001	01110111
Baris 10	00101110	01101000	01100101	01100011	01101011	00101110	01101001	01101110
	00000000	01010100	01010011	01010011	01000101	00000000	00000000	00000000
Baris 11	00001101	00000000	00000000	00000011	01001100	01100001	01110110	01100110
	00110101	00110010	00101110	00111001	00110001	00101110	00110000	00000000
Baris 12	11111111	11111011	10010100	01100100	00000000	00001111	11110000	00000000
	00000000	01101001	00000000	00000000	00000000	00001000	00000000	00000000

2. Meng-*input*-kan pesan rahasia kedalam media *cover*

Pesan Teks yang akan disisipkan yaitu RAMADANI

Tabel 3 Pesan Yang Ingin Disisipkan Dalam Bentuk Biner

R	0	1	0	1	0	0	1	0
A	0	1	0	0	0	0	0	1
M	0	1	0	0	1	1	0	1
A	0	1	0	0	0	0	0	1
D	0	1	0	0	0	1	0	0
A	0	1	0	0	0	0	0	1
N	0	1	0	0	1	1	1	0
I	0	1	0	0	1	0	0	1

3. Berikut proses penyisipan pesan terhadap media suara yang dijadikan sebagai penampung.

Proses penyisipan pesan rahasia yaitu dengan mengganti *bit-bit* akhir. Setiap *bit* pesan rahasia disisipkan kedalam *bit* media yaitu pada setiap *bit* akhir media. Adapun proses penyisipan dilakukan dengan operasi *And* dan *Or*. Operasi logika *And* digunakan apabila *bit* pesan yang disisipkan yaitu bernilai 0 dan operasi logika *Or* digunakan apabila *bit* pesan yang disisipkan yaitu bernilai 1. Maka *stego* yang dihasilkan adalah sebagai berikut :

1	01001001	*And*	0	=	01001000	2	00110110	*And*	0	=	00110110
	01000100	*Or*	1	=	01000101		01010100	*Or*	1	=	01010101
	00110011	*And*	0	=	00110010		01001001	*And*	0	=	01001000
	00000100	*Or*	1	=	00000101		01010100	*And*	0	=	01010100
	00000000	*And*	0	=	00000000		00110010	*And*	0	=	00110010
	00000000	*And*	0	=	00000000		00000000	*And*	0	=	00000000
	00000000	*Or*	1	=	00000001		00000000	*And*	0	=	00000000
	00000001	*And*	0	=	00000000		00000000	*Or*	1	=	00000001
3	00001111	*And*	0	=	00001110	4	01100100	*And*	0	=	01100100
	00000000	*Or*	1	=	00000001		01101000	*Or*	1	=	01101001
	00000000	*And*	0	=	00000000		01100001	*And*	0	=	01100000
	00000011	*And*	0	=	00000010		01101110	*And*	0	=	01101110
	01010010	*Or*	1	=	01010011		00100000	*And*	0	=	00100000
	01100001	*Or*	1	=	01100001		01010100	*And*	0	=	01010100
	01101101	*And*	0	=	01101100		01101001	*And*	0	=	01101000
	01100001	*Or*	1	=	01100001		01100010	*Or*	1	=	01100011
5	01100001	*And*	0	=	01100000	6	00000000	*And*	0	=	00000000
	00000000	*Or*	1	=	00000001		00011011	*Or*	1	=	00011011
	01010100	*And*	0	=	01010100		00000000	*And*	0	=	00000000
	01010000	*And*	0	=	01010000		00000000	*And*	0	=	00000000
	01000101	*And*	0	=	01000100		00000011	*And*	0	=	00000010
	00110001	*Or*	1	=	00110001		01001111	*And*	0	=	01001110
	00000000	*And*	0	=	00000000		01110000	*And*	0	=	01110000
	00000000	*And*	0	=	00000000		01101001	*Or*	1	=	01101001
7	01100011	*And*	0	=	01100010	8	01101001	*And*	0	=	01101000
	01101011	*Or*	1	=	01101011		01101011	*Or*	1	=	01101011
	00100000	*And*	0	=	00100000		01101001	*And*	0	=	01101000
	00101101	*And*	0	=	00101100		01100100	*And*	0	=	01100100
	00100000	*Or*	1	=	00100001		01101001	*Or*	1	=	01101001
	01110111	*Or*	1	=	01110111		01110111	*And*	0	=	01110110
	01101001	*Or*	1	=	01101001		00101110	*And*	0	=	00101110
	01100100	*And*	0	=	01100100		01101000	*Or*	1	=	01101001

4. Pesan rahasia sudah disisipkan pada sebuah *media audio*.

Tabel 4 Hasil Penyisipan Pesan Rahasia dalam Biner

Baris 0	01001000	01000101	00110010	00000101	00000000	00000000	00000001	00000000
	00000000	00110111	01010100	01001000	01010100	00110010	00000000	00000001
Baris 1	00000000	00001111	00000000	00000000	00000011	01010011	01100000	01101101
	01100000	01100101	01101000	01100000	01101110	00100000	01010100	01101001
Baris 2	01100010	01100001	00000000	01010100	01010000	01000101	00110000	00000000
	00000000	00000001	00011010	00000000	00000000	00000010	01001110	01110001
Baris 3	01101000	01100011	01101010	00100000	00101101	00100001	01110111	01101000
	01100100	01101001	01101010	01101000	01100101	01101000	01110110	00101111
Baris 4	01101000	01100101	01100011	01101011	00101110	01101001	01101110	00000000
	01010100	01000011	01001111	01001110	00000000	00000000	00000000	00001000
Baris 5	00000000	00000000	00000011	01010010	01100101	01101100	01101001	01100111
	01101001	00000000	01010100	01011000	01011000	01011000	00000000	00000000
Baris 6	00000000	00100110	00000000	00000000	00000011	01100011	01101111	01101101
	01101101	01100101	01101110	01110100	00000000	01001100	01100001	01100111
Baris 7	01110101	00100000	01101001	01101110	01101001	00100000	01101000	01100001
	01101110	01111001	01100001	00100000	01101000	01100001	01101110	01111001
Baris 8	01100001	00100000	01110101	01101110	01110100	01110101	01101011	00100000
	01110010	01100101	01110110	00000000	01010100	01000001	01001100	01000010
Baris 9	00000000	00000000	00000000	00010011	00000000	00000000	00000011	01110111
	01101001	01100100	01101001	01101011	01101001	01100100	01101001	01110111
Baris 10	00101110	01101000	01100101	01100011	01101011	00101110	01101001	01101110
	00000000	01010100	01010011	01010011	01000101	00000000	00000000	00000000
Baris 11	00001101	00000000	00000000	00000011	01001100	01100001	01110110	01100110
	00110101	00110010	00101110	00111001	00110001	00101110	00110000	00000000
Baris 12	11111111	11111011	10010100	01100100	00000000	00001111	11110000	00000000
	00000000	01101001	00000000	00000000	00000000	00001000	00000000	00000000

3.3.2 *Decoding*

Untuk menampilkan pesan rahasia ke nilai awal ataupun pesan asli yaitu pada proses ini pesan asli yang telah disisipi *file audio* diekstraksi. Proses ekstraksi dimulai dengan menginputkan pesan yang sudah disisipi *file audio* kemudian diekstrak dan simpan nama pesan yang sudah diekstrak. Pada proses ini maka *bit-bit* yang sudah diganti pada proses *encoding* akan diubah kembali ke nilai awal.

Adapun langkah-langkah proses *decoding* pesan rahasia ini adalah :

1. Mengambil *file audio* yang sudah disisipkan pesan rahasia.

Tabel 5 Audio dalam Biner yang telah disipkan Pesan Rahasia

Baris 0	01001000	01000101	00110010	00000101	00000000	00000000	00000001	00000000
	00000000	00110111	01010100	01001000	01010100	00110010	00000000	00000001
Baris 1	00000000	00001111	00000000	00000000	00000011	01010011	01100000	01101101
	01100000	01100101	01101000	01100000	01101110	00100000	01010100	01101001
Baris 2	01100010	01100001	00000000	01010100	01010000	01000101	00110000	00000000
	00000000	00000001	00011010	00000000	00000000	00000010	01001110	01110001
Baris 3	01101000	01100011	01101010	00100000	00101101	00100001	01110111	01101000
	01100100	01101001	01101010	01101000	01100101	01101000	01110110	00101111
Baris 4	01101000	01100101	01100011	01101011	00101110	01101001	01101110	00000000
	01010100	01000011	01001111	01001110	00000000	00000000	00000000	00001000
Baris 5	00000000	00000000	00000011	01010010	01100101	01101100	01101001	01100111
	01101001	00000000	01010100	01011000	01011000	01011000	00000000	00000000
Baris 6	00000000	00100110	00000000	00000000	00000011	01100011	01101111	01101101
	01101101	01100101	01101110	01110100	00000000	01001100	01100001	01100111
Baris 7	01110101	00100000	01101001	01101110	01101001	00100000	01101000	01100001
	01101110	01111001	01100001	00100000	01101000	01100001	01101110	01111001
Baris 8	01100001	00100000	01110101	01101110	01110100	01110101	01101011	00100000
	01110010	01100101	01110110	00000000	01010100	01000001	01001100	01000010
Baris 9	00000000	00000000	00000000	00010011	00000000	00000000	00000011	01110111
	01101001	01100100	01101001	01101011	01101001	01100100	01101001	01110111
Baris 10	00101110	01101000	01100101	01100011	01101011	00101110	01101001	01101110
	00000000	01010100	01010011	01010011	01000101	00000000	00000000	00000000
Baris 11	00001101	00000000	00000000	00000011	01001100	01100001	01110110	01100110
	00110101	00110010	00101110	00111001	00110001	00101110	00110000	00000000
Baris 12	11111111	11111011	10010100	01100100	00000000	00001111	11110000	00000000
	00000000	01101001	00000000	00000000	00000000	00001000	00000000	00000000

2. Mengekstrak pesan rahasia tersebut dengan mengambil setiap *bit* akhir pada *file audio.*

a. Membuat blok-blok data ke dalam 8 *byte* per blok. Untuk setiap blok dikerjakan langkah “b” samapi dengan langkah “c” untuk i = 0,1,2,3,…,7.

b. Mengambil nilai bit terakhir *byte* pesan ke-I dengan meng-*and*-kan dengan 1.

c. Menyimpan hasil setelah di-*and*-kan dengan 1, dan mengalihkan dengan nilai posisi bit, yaitu : $(2^{(7-i)})$.

d. Menjumlahkan semua hasil perhitungan untuk i=0 sampai dengan i=7.

e. Menentukan karakter *ASCII* yang bersesuaian dengan hasil perhitungan. Sebagai contoh akan dilakukan *decoding* untuk membaca informasi yang disisipkan dengan mengambil nilai bit dari media penampung *file MP3* seperti berikut ini :

01001001	=	01001000	*And*	1=0	nilai =0x2^7	=0
01000100	=	01000101	*And*	1=1	nilai =1x2^6	=64
00110011	=	00110010	*And*	1=0	nilai =0x2^5	=0
00000100	=	00000101	*And*	1=1	nilai =1x2^4	=16
00000000	=	00000000	*And*	1=0	nilai =0x2^3	=0
00000000	=	00000000	*And*	1=0	nilai =0x2^2	=0
00000000	=	00000001	*And*	1=1	nilai =1x2^1	=2
00000001	=	00000000	*And*	1=0	nilai =0x2^0	=0 +
						=82 Kode *ASCII* yaitu “R”
00110110	=	00110110	*And*	1=0	nilai =0x2^7	=0
01010100	=	01010101	*And*	1=1	nilai =1x2^6	=64
01001001	=	01001000	*And*	1=0	nilai =0x2^5	=0
01010100	=	01010100	*And*	1=0	nilai =0x2^4	=0
00110010	=	00110010	*And*	1=0	nilai =0x2^3	=0
00000000	=	00000000	*And*	1=0	nilai =0x2^2	=0
00000000	=	00000000	*And*	1=0	nilai =0x2^1	=0
00000000	=	00000001	*And*	1=1	nilai =0x2^0	=1 +
						=65 Kode *ASCII* yaitu “A”
00001111	=	00001110	*And*	1=0	nilai =0x2^7	=0
00000000	=	00000001	*And*	1=1	nilai =1x2^6	=64
00000000	=	00000000	*And*	1=0	nilai =0x2^5	=0
00000011	=	00000010	*And*	1=0	nilai =0x2^4	=0
01010010	=	01010011	*And*	1=1	nilai =1x2^3	=8
01100001	=	01100001	*And*	1=1	nilai =1x2^2	=4
01101101	=	01101100	*And*	1=0	nilai =0x2^1	=0
01100001	=	01100001	*And*	1=1	nilai =1x2^0	=1 +
						=77 Kode *ASCII* yaitu “M”
01100100	=	01100100	*And*	1=0	nilai =0x2^7	=0
01101000	=	01101001	*And*	1=1	nilai =1x2^6	=64
01100001	=	01100000	*And*	1=0	nilai =0x2^5	=0
01101110	=	01101110	*And*	1=0	nilai =0x2^4	=0
00100000	=	00100000	*And*	1=0	nilai =0x2^3	=0
01010100	=	01010100	*And*	1=0	nilai =0x2^2	=0
01101001	=	01101000	*And*	1=0	nilai =0x2^1	=0
01100010	=	01100011	*And*	1=1	nilai =1x2^0	=1 +
						=65 Kode *ASCII* yaitu “A”
01100001	=	01100000	*And*	1=0	nilai =0x2^7	=0
00000000	=	00000001	*And*	1=1	nilai =1x2^6	=64
01010100	=	01010100	*And*	1=0	nilai =0x2^5	=0
01010000	=	01010000	*And*	1=0	nilai =0x2^4	=0
01000101	=	01000100	*And*	1=0	nilai =0x2^3	=0
00110001	=	00110001	*And*	1=1	nilai =1x2^2	=4
00000000	=	00000000	*And*	1=0	nilai =0x2^1	=0
00000000	=	00000000	*And*	1=0	nilai =0x2^0	=0 +
						=68 Kode *ASCII* yaitu “D”
00000000	=	00000000	*And*	1=0	nilai =0x2^7	=0
00011011	=	00011011	*And*	1=1	nilai =1x2^6	=64
00000000	=	00000000	*And*	1=0	nilai =0x2^5	=0
00000000	=	00000000	*And*	1=0	nilai =0x2^4	=0
00000011	=	00000010	*And*	1=0	nilai =0x2^3	=0
01001111	=	01001110	*And*	1=0	nilai =0x2^2	=0
01110000	=	01110000	*And*	1=0	nilai =0x2^1	=0
01101001	=	01101001	*And*	1=1	nilai =1x2^0	=1 +
						=65 Kode *ASCII* yaitu “A”
01100011	=	01100010	*And*	1=0	nilai =0x2^7	=0
01101011	=	01101011	*And*	1=1	nilai =1x2^6	=64
00100000	=	00100000	*And*	1=0	nilai =0x2^5	=0
00101101	=	00101100	*And*	1=0	nilai =0x2^4	=0
00100000	=	00100001	*And*	1=1	nilai =1x2^3	=8
01110111	=	01110111	*And*	1=1	nilai =1x2^2	=4

01101001	=	01101001	And	1=1	nilai =1x2^1	=2
01100100	=	01100100	And	1=0	nilai =0x2^0	=0 +
					=78 Kode *ASCII* yaitu "N"	
01101001	=	01101000	And	1=0	nilai =0x2^7	=0
01101011	=	01101011	And	1=1	nilai =1x2^6	=64
01101001	=	01101000	And	1=0	nilai =0x2^5	=0
01100100	=	01100100	And	1=0	nilai =0x2^4	=0
01101001	=	01101001	And	1=1	nilai =1x2^3	=8
01110111	=	01110110	And	1=0	nilai =0x2^2	=0
00101110	=	00101110	And	1=0	nilai =0x2^1	=0
01101000	=	01101001	And	1=1	nilai =1x2^0	=1 +
					=73 Kode *ASCII* yaitu "I"	

3. Pesan rahasia terbaca dan di tampilkan.
4. *file audio* akan disimpan dan dikembalikan nilai *bit*-nya ke bentuk semula.

KESIMPULAN

Setelah melakukan pembahasan baik itu secara teoritis, penganalisaan masalah, metode, serta pembahasan dalam penyelesaian penelitian ini, maka dapat ditarik beberapa kesimpulan sebagai berikut:

1. Pada proses penyisipan pesan membutuhkan dua buah properti pendukung yaitu *audio* sebagai media penampung pesan dan juga pesan rahasia yang akan disisipkan sedangkan untuk proses ekstraksi pesan hanya membutuhkan pesan *stego*.
2. Metode ini diterapkan dengan mengganti *bit-bit* terakhir dari *file audio* dengan *bit-bit* pesan atau data.
3. Perancangan steganografi menggunakan alat bantu perancangan *use case* untuk membangun form *encoding* dan *decoding* dan telah di implementasikan untuk penggunaan data pada *file audio*.

DAFTAR PUSTAKA

[1] T. Limbong and P. D. P. Silitonga, "Testing the Classic Caesar Cipher Cryptography using of Matlab," *Int. J. Eng. Res. Technol.*, vol. 6, no. 2, pp. 175–178, 2017.

[2] R. Munir, "Kriptografi," *Inform. Bandung*, 2006.

[3] D. Ariyus, "Kriptografi keamanan data dan komunikasi," *Yogyakarta Graha Ilmu*, 2006.

[4] D. Novianto and Y. Setiawan, "Aplikasi Pengamanan Informasi Menggunakan Metode Least Significant Bit (LSB) dan Algoritma Kriptografi Advanced Encryption Standard (AES)," *J. Ilm. Inform. Glob.*, vol. 09, no. 2, pp. 83–89, 2018.

[5] H. Santoso and M. Fakhriza, "PERANCANGAN APLIKASI KEAMANAN FILE AUDIO FORMAT WAV (WAVE FORM) MENGGUNAKAN ALGORITMA RSA," *J. Ilmu Komput. dan Inform.*, vol. 2, no. 1, pp. 47–54, 2018.

[6] C. Paper, A. Solichin, and U. Budi, "Implementasi Steganografi Dengan Metode Bit Plane Complexity Segmentation Untuk Menyembunyikan," no. March, pp. 2–3, 2016.