G-Tech : Jurnal Teknologi Terapan
Volume 7, No. 2, April 2023, hal. 543-553
E-ISSN: 2623-064x | P-ISSN: 2580-8737

Implementasi Metode *Least Significant Bit* Dalam Teknik Steganografi pada Berkas Audio Dengan Stego Citra Digital

Rifqi Alwanul Akmal, Mhd. Furqan, Rakhmat Kurniawan R[3]

[1,2,3] Program Studi Ilmu Komputer, Fakultas Sains dan Teknologi, Universitas Islam Negeri Sumatera Utara, Medan, Indonesia

Informasi Artikel

Riwayat Artikel

Diserahkan : 12-02-2023
Direvisi : 21-02-2023
Diterima : 02-03-2023

ABSTRAK

Steganografi adalah teknik menyembunyikannya pesan rahasia ke dalam sebuah media yang tidaklah dapat diketahui oleh seseorang yang lain maka pesan rahasia tidaklah bisa dideteksi. Metode steganografi yang dipergunakan dalam riset maupun penelitian ini yakni metode *Least Significant Bit*. Metode tersebut adalah teknik menyembunyikan pesan yang dilakukannya dengan digantikannya bit-bit di dalam segmentasi citra dengan bit-bit pesan rahasia. Steganografi dengan mempergunakan metode LSB bakal menghasilkannya *file* yang nyaris sama/mirip dengan *cover image* yang dipergunakan. Penelitian ini ditujukannya teruntuk memanfaatkan penggunaan steganografi dengan metode LSB perihal mengamankan berkas audio yang berformat .AAC ke dalam citra digital yang berformat .BMP serta .PNG. Ukuran dari citra digital dan durasi dari berkas audio berbeda-beda, melalui perbedaan tersebut dihasilkannya proses penyisipan serta pengekstraksian yang berbeda. Keseluruhan pengujian dilakukannya dengan cara melaluinya tahapan penyisipan dengan ukuran piksel citra digital yang berbeda. Maka daripada itu jika semakin besar ukuran dari citra digital yang dipakai, maka akan makin lama juga tahapan dalam waktu pengekstraksiannya dan juga sebaliknya.

Kata Kunci:

Audio, Citra, *Least Significant Bit*, Pesan Rahasia, Steganografi

ABSTRACT

Steganography is a technique of hiding secret messages in a medium that cannot be known by someone else, so secret messages cannot be detected. The steganographic method used in this research and research is the Least Significant Bit method. This method is a technique of hiding messages by replacing the bits in the image segmentation with secret message bits. Steganography using the LSB method will produce files that are almost the same/similar to the cover image used. This research aims to take advantage of the use of steganography with the LSB method regarding securing audio files in .AAC format into digital images in .BMP and .PNG formats. The size of the digital image and the duration of the audio file are different, through these differences a different insertion and extraction process results. The entire test is carried out by going through the insertion stages with different digital image pixel sizes. Therefore, if the larger the size of the digital image used, the longer the extraction time, and vice versa.

Keywords **:**

Audio, Image, Least Significant Bit, Secret Message, Steganography

Corresponding Author **:**
Rifqi Alwanul Akmal
Program Studi Ilmu Komputer Fakultas Sains dan Teknologi, Universitas Islam Negeri Sumatera Utara
Jln. Lapangan Golf, Desa Durian Jangak, Pancur Batu, Deli Serdang, Sumatera Utara
alwanulakmal@gmail.com

PENDAHULUAN

Memasukinya era revolusi industri 4.0 yang ada pada saat yang sekarang ini, informasi menjadi begitu sangat berkembang dengan cepatnya serta gampang untuk bisa memasukinya wilayah antar kota, negara, juga antar benua yang bukanlah jadi suatu kendala teruntuk sistem pertukaran data yang ada pada era yang sekarang ini (Sriani, dkk, 2017). Makin berkembang pesatnya teknologi yang terdapat pada saat yang sekarang ini, membuat komunikasi maupun pertukaran informasi bisa dilakukannya oleh para manusia dengan cara sangatlah begitu praktis dan efisien dengan tidaklah adanya halangan oleh adanya suatu jarak maupun waktu (Syawal, dkk, 2016). Di zaman dengan perkembangan serta kemajuan dari teknologi serta data global yang begitu berubah serta cepat dalam perihal perkembangan akan perangkat lunak yang terus mengalami perkembangan, baik dari sisi keamanan dari sebuah program aplikasi serta juga kerahasiaan dari sebuah data yang adalah sebuah hal yang sangat penting teruntuk dijaga dikarenakan adalah sebuah aspek yang cukup terbilang penting yang ada pada sebuah sistem informasi (Setiawan, 2018).

Berbagai macam mesin pencari (*search engine*) konsisten untuk bisa dikembangkan serta diciptakannya dan ditambahnya juga lagi juga dengan adanya penyerangan virus, *spam*, penyadap, maupun *hacker* yang sangat cukup anyak yang dapat mengambil berbagai macam data yang tergolong sebagai suatu data yang dirahasiakan (Anwar, 2017). Untuk bisa mengatasinya perihal tersebut, pengamanan yang dibuat pada suatu data dan juga informasi perlu adanya peningkatan guna bisa menjamin sebuah keamanan (Aldo dan Hakim, 2018). Terdapat 3 macam cara pada teknik pengamanan data yang bisa diimplementasikannya seperti halnya kriptografi, steganografi, serta juga *watermarking* (Anti, dkk, 2017). Namun pada penulisan ini penulis hanyalah mendalami tentang steganografi saja. Steganografi pada dasarnya yakni sebuah ilmu yang meneliti, mempelajari, serta juga mengembangkannya seni yang mengacu pada ilmu komunikasi yang *invisible* ataupun yang tidaklah terlihat (Kuniadi, dkk, 2017). Teknik steganografi adalah salah satu teknik yang dipergunakan teruntuk mengamankannya suatu data dengan mempergunakan metode menyisipkan maupun disembunyikannya sebuah data tersebut dalam objek dengan tidak mengubah bentuk dari objeknya tersebut. Hingga membuat data-data yang diamankan tidaklah bisa untuk dicurigai oleh individu yang yang tidaklah bertanggung jawab (Sinaga dan Sitorus, 2017). Tujuan dari dilakukannya teknik penyisipan ini adalah teruntuk bisa melakukan penjagaan privasi dari pesan yang bakal dikirimkan (Santa dan Situmorang, 2018).

Keamanan privasi adalah problematika yang jadi skala prioritas utama dalam dunia digital. Privasi artinya sesuatu hal yang berkaitan milik pribadi yang dalam bentuk berupa foto, video, lokasi dan lainnya yang memiliki urgensi dengan individu yang memiliki hal tersebut. Teruntuk memberikannya kemanan privasi terhadap sebuat data, maka daripada itu bisa dilakukannya dengan pemanfaatan dari teknik steganografi (Darwis, 2017). Algoritma Steganografi yang dipergunakan pada riset maupun penelitian ini adalah LSB (*Least Significant Bit).* LSB adalah suatu teknik yang umum atau biasa dipergunakan dalam dekripsi maupun enkripsi terhadap informasi yang dirahasiakan (Nasution, dkk, 2020). Metode yang dapat dikatakan cukup sederhana dalam penerapan steganografi. Selain daripada demikian, proses penyisipan serta ekstraksi dengan menggunakan metode ini juga relatif efisien. Metode ini adalah metode yang memanfaatkan bit pangkatnya terkecil di tiap-tiap byte sumber. Selanjutnya, dimasukkannya pada sub-sub byte yang ditujukan. Metode LSB yakni menyisipkannya sebuah pesan ke dalam cover image terhadap bit yang dianggapnya kurang berarti (Lutfi dan Rosihan, 2018).

METODE PENELITIAN

Teknik Pengumpulan Data

Teknik pengumpulan data yang dipergunakan oleh peneliti yakni mempergunakan teknik studi literatur. Studi Literatur adalah suatu teknik yang dilakukan untuk mengumpulkan berbagai macam data yang dilakukannya dengan cara melakukan pencarian terhadap informasi serta pengetahuan yang didapat dari berbagai macam jurnal ilmiah, buku, internet, serta beberapa sumber yang lain yang relevan pada fokus kajian yang ditelitikan tentang steganografi

mempergunakan metode LSB. Sesudah berbagai macam sumber tersebut sudah ditemukannya, maka dengan begitu seluruh sumber tersebut bakal dikritisi dengan cara internal ataupun eksternal, hingga dari keseluruhan hasil terkait pada pengumpulan data dan juga informasi yang didapatkan atas cakupan yang luas bisa dijadikannya yakni sebagai landasan teori dan juga tuntunan perihal menjawab masalah yang ada pada tahapan maupun proses penelitian.

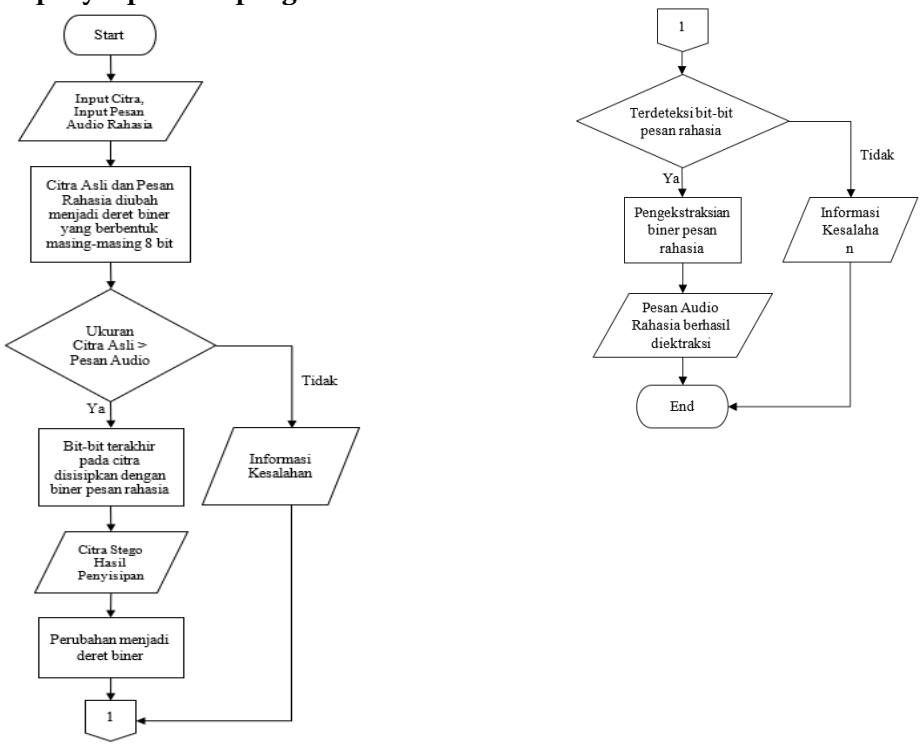
Analisis

Proses sistem yang ada di dalam riset maupun penelitian ini dirancakannya dengan cara mempergunakan metode LSB. Metode ini memiliki tujuan teruntuk melakukan manipulan terhadap citra digital yang ada di dalam berkas audio sehingga tidaklah bosa diketahuinya akan keberadaan dari citra digital tersebut serta secara kasat mata tidaklah ada terjadinya suatu perubahan terhadap citra yang sudah dimanipulasikan.

Perancangan

Di tahapan berikut ini dilakukannya suatu perancangan sistem agar mempermudah *user* perihal memahaminya cara kerja dalam menggunakan sistem. Perancangan terdirinya atas perancangan *flowchart* metode LSB, perancangan *flowchart* sistem steganografi pada berkas audio dengan berkas gambar. Dengan melaluinya perancangan yang sudah dibuatnya ini, maka daripada demikian pengimplementasian metode pada sistem bakal jauh lebih gampang teruntuk dilakukannya.

1. *Flowchart* penyisipan dan pengekstrasian metode LSB



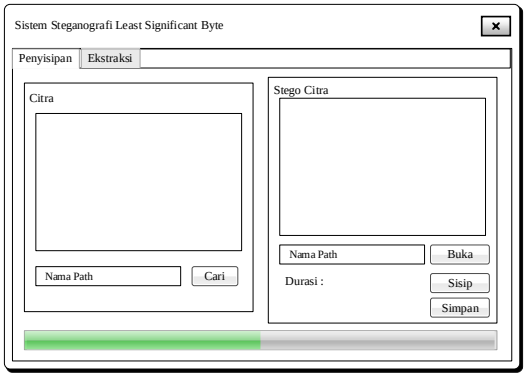
Gambar 1. *Flowchart* Penyisipan dan Ekstraksi LSB *(Least Significant Bit)*

2. Perancangan Antar Muka

Perancangan yang ada pada sistem ini yakni mempergunakan bahasa pemrograman C# (C *Sharp*). Perancangan tersebut mempunyai target maupun tujuan supaya mempermudah para user maupun pengguna teruntuk menggunakan sistem yang sudah dirancangkan tersebut. Perancangan guna tampilan antar mukanya terdirinya atas *form* penyisipan, ekstraksi, serta informasi.

Form Penyisipan

Pada *form* penyisipan bakal muncul pada saat user baru akan mau membukakan aplikasinya. Kemudian teruntuk langkah pertama dari penggunaan yang akan dilakukan, para pengguna dapat melakukan langkah yakni mengambilnya file citra digital yang memiliki format .PNG dan .BMP, kemudian memilih pesan rahasia yang mau disembunyikannya ke dalam wujud berkas audio yang memiliki format.AAC yang bakal dimasukkan ke dalam citra digital.



Gambar 2. Tampilan *Form* Penyisipan

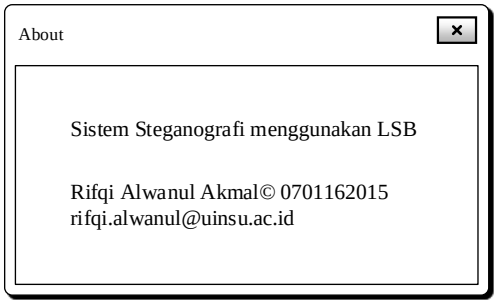
Form Ekstraksi

Pada *form* ekstraksi para user maupun pengguna melakukannya suatu tahapan pengekstraksian yang bakal menampilkannya pesan tersembunyi yang terdapat di dalam *stego image*.

Gambar 3. Tampilan *Form* Ekstraksi

Form Informasi

Pada *form* informasi bakal memperlihatkan judul dan juga informasi dari si pembuat programnya tersebut.



Gambar 4. Tampilan *Form* Informasi

HASIL DAN PEMBAHASAN

1. Menghitung Nilai RGB Citra

Pada citra warna 24 bit (*true color*) nilai RGB secara langsung diuraikannya ke dalam data *bitmap* dengan wujud bilangan biner. Teruntuk membacakan nilai dari RGB-nya, yakni dilakukannya dengan cara membacakan data *bitmap* yang memiliki panjang 3 *byte* yang mana komponen dari R, G, setya B dinyatakannya ke dalam tiap-tiap dari *byte.* Sebuah citra 8 bit dipresentasikannya ke dalam tiap-tiap dari *byte* yang ada pada citra warna hingga kandungan dari warnanya jadi 3 *byte* x 8 bit = 24 bit. Contoh citra warna 24 bit bisa dilihatnya pada Gambar yang ada dibawah ini.

Gambar 5. Gambar Citra Warna

Pada citra warna Gambar 4.1 yang ada di atas, setiap pikselnya memiliki kandungan yang berisikan 8 bit untuk tiap-tiap dari warna dasar R, G serta B maupun memiliki kandungan 24 bit warna dengan nilai rentang warnanya yang berkisar dari 0 (00000000) sampai dengan 255 (11111111) teruntuk setiap warna yang ada. Untuk mendapatkan nilai biner pada gambar tersebut dan mengetahui nilai RGB pada gambar, diawali dengan mengkonversi gambar ke hexadecimal. Caranya yaitu dengan menjalankan aplikasi *xxd.exe* dan jalankan perintah melalui *Command Prompt* seperti gambar dibawah ini.

```
D:\aplikasi\xxd-1.11_win32>xxd -i 600x600.bmp 600x600.h
```

Gambar 6. Perintah Menjalankan Aplikasi xxd.exe

Setelah itu format file ‘600x600.bmp’ akan dirubah kedalam format file ‘600x600.h’ dan untuk melihat nilai hexadecimal buka file ‘600x600.h’ menggunakan *text editor* misalkan, *sublime text 3.* Nilai hexadecimal dapat dilihatnya dari gambar yang terdapat di bawah.

```
unsigned char d600x600_bmp[] = {
  0x42, 0x4d, 0xf6, 0x7a, 0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x36, 0x00,
  0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x58, 0x02, 0x00, 0x00, 0x58, 0x02,
  0x00, 0x00, 0x01, 0x00, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0x7a,
  0x10, 0x00, 0x13, 0x0b, 0x00, 0x00, 0x13, 0x0b, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x1b, 0x18, 0x1c, 0x17, 0x14,
  0x1c, 0x17, 0x14, 0x1e, 0x19, 0x16, 0x1d, 0x18, 0x15, 0x1f, 0x1a, 0x17,
  0x1d, 0x18, 0x15, 0x1c, 0x17, 0x15, 0x1b, 0x16, 0x14, 0x1e, 0x19, 0x16,
  0x1f, 0x19, 0x15, 0x23, 0x1b, 0x1a, 0x1f, 0x18, 0x16, 0x21, 0x1a, 0x18,
  0x21, 0x19, 0x19, 0x1f, 0x17, 0x17, 0x21, 0x19, 0x19, 0x1f, 0x17, 0x17,
  0x23, 0x1b, 0x1b, 0x23, 0x1b, 0x1b, 0x21, 0x19, 0x19, 0x22, 0x1a, 0x1a,
  0x22, 0x1a, 0x1a, 0x25, 0x1d, 0x1d, 0x24, 0x1c, 0x1c, 0x22, 0x1a, 0x1a,
  0x22, 0x1a, 0x1b, 0x23, 0x1b, 0x1b, 0x23, 0x1b, 0x1b, 0x1f, 0x17, 0x18,
```

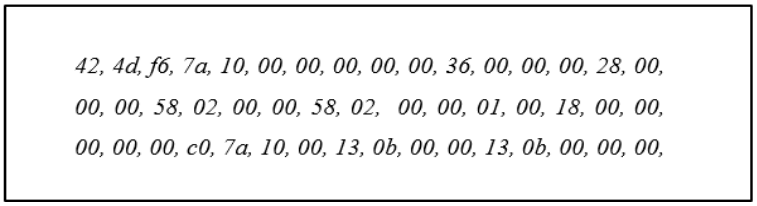
Gambar 7. Hasil Hexadesimal File 600x600.bmp

Dikarenakan nilai hexadecimal yang dikonversi terlalu banyak. Untuk mempermudah dalam proses perhitungan dilakukan pengambilan sampel sebanyak jumlah hexadecimal yang diblock pada gambar dibawah.

```
unsigned char d600x600_bmp[] = {
  0x42, 0x4d, 0xf6, 0x7a, 0x10, 0x00, 0x00, 0x00, 0x00, 0x00, 0x36, 0x00,
  0x00, 0x00, 0x28, 0x00, 0x00, 0x00, 0x58, 0x02, 0x00, 0x00, 0x58, 0x02,
  0x00, 0x00, 0x01, 0x00, 0x18, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0, 0x7a,
  0x10, 0x00, 0x13, 0x0b, 0x00, 0x00, 0x13, 0x0b, 0x00, 0x00, 0x00, 0x00,
  0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x1b, 0x18, 0x1c, 0x17, 0x14,
  0x1c, 0x17, 0x14, 0x1e, 0x19, 0x16, 0x1d, 0x18, 0x15, 0x1f, 0x1a, 0x17,
  0x1d, 0x18, 0x15, 0x1c, 0x17, 0x15, 0x1b, 0x16, 0x14, 0x1e, 0x19, 0x16,
  0x1f, 0x19, 0x15, 0x23, 0x1b, 0x1a, 0x1f, 0x18, 0x16, 0x21, 0x1a, 0x18,
```

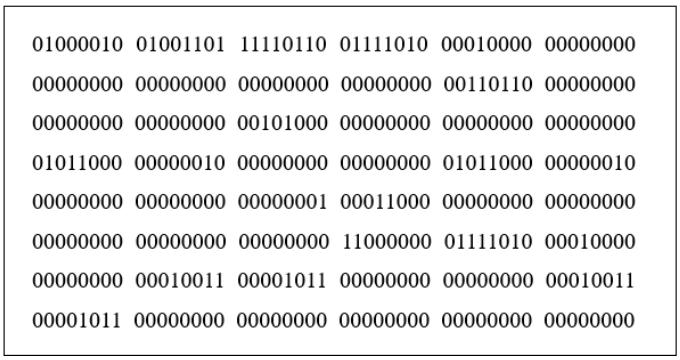
Gambar 8. Pengambilan Sampel Hexadesimal File 600x600.bmp

Sehingga sampel hexadecimal yang diambil adalah sebagai berikut:



Gambar 9. Pemisahan Sampel Hexadesimal File 600x600.bmp

Karena untuk mengetahui nilai RGB diperlukan biner. Jadi nilai hexadecimal diatas dikonversi kembali kedalam bentuk biner. Hasilnya sebagai berikut:



Gambar 10. Sampel Biner File 600x600.bmp

Pada biner di atas, kelompokan setiap biner menjadi 24 bit. karena panjang biner dari kandungan warna RGB adalah 24 bit maka hasil pengelompokan pertama yaitu, **010000100100110111110110.** Teruntuk memperoleh nilai RGB, maka dengan demikian tiap-tiap dari *pixel* yang ada pada citra dihitungkan dengan mempergunakan rumus seperti berikut:

$$Nilai\ R\ =\ c\ Mod\ 256 \quad (1.1)$$

$$Nilai\ G\ =\ (c\ and\ 65280)\ /\ 256 \quad (1.2)$$

$$Nilai\ B\ =\ (c\ and\ 16711680)\ /\ 65280 \quad (1.3)$$

Dimana c adalah nilai piksel citra.

Nilai piksel (0,0) dari hasil pengelompokan 1 adalah 010000100100110111110110.Nilai RGB dihitungnya dengan persamaan (1.1), (1.2) dan (1.3) seperti berikut:

Nilai komponen R dengan persamaan (1.1):

$$Nilai\ R\ =\ 010000100100110111110110\ mod\ 100000000\ =\ 11110110\ =\ 246\ (desimal)$$

Nilai komponen G dihitung dengan persamaan (1.2):

$$Nilai\ G\ =\ \frac{(010000100100110111110110\ and\ 1111111100000000)}{100000000}\ =\ 01001101\ =\ 77\ (Desimal)$$

Nilai komponen B dihitung dengan persamaan (1.3) :

$$Nilai\ B\ =\ \frac{(010000100100110111110110\ and\ 111111110000000000000000)}{10000000000000000}\ =\ 01000010\ =66(desimal)$$

Sehingga diperoleh nilai piksel (0,0) 11110110 01001101 01000010

$$R\ =\ \ 11110110\ =\ 246\ (desimal)$$

$$G\ =\ \ 01001101\ =\ 77\ (desimal)$$

$$B\ =\ \ 01000010\ =\ 66\ (desimal)$$

Terkait dengan analisa ini jumlah piksel yang dihitungkan berjumlah 16 *pixel* saja serta teruntuk memperoleh nilai RGB piksel berikutnya, dilakukannya sama dengan cara yang ada diatas serta kemudian nilai RGB dari seluruh nilai piksel yang ada pada citra dimasukkannya ke dalam matriks layaknya yang terdapat pada Tabel yang ada di bawah ini.

Tabel 1. Nilai RGB Citra

RGB	RGB	RGB	RGB
246, 77, 66	0, 16, 122	0,0,0	0, 54,0
40, 0, 0	0, 0, 0	0, 2, 88	2, 88, 0
1, 0, 0	0, 0, 24	0, 0, 0	61, 122,192
11, 19, 0	19, 0, 0	0, 0, 11	0, 0, 0

2. Menghitung Nilai Audio

Pada tahapan penyisipan yang dilakukan dengan algoritma LSB dilakukannya yakni dengan cara mengganti setiap dari bit penyisip ke bit nomor 1 terakhir dari *byte cover* citra yang disebutkan dengan sebutan metode LSB. Akan dilakukan penyisipan file 'suara1.aac' namun terlebih dahulu agar file bisa disisipkan harus dalam bentuk biner sehingga diawali dengan mengkonversi audio ke hexadecimal. Caranya yaitu dengan menjalankan aplikasi *xxd.exe* dan jalankan perintah melalui *Command Prompt* seperti gambar dibawah.

```
D:\aplikasi\xxd-1.11_win32>xxd -i suara1.aac suara1.h
```

Gambar 11. Perintah Menjalankan Aplikasi xxd.exe

Setelah itu format file 'suara1.aac' akan dirubah kedalam format file 'suara1.h' dan untuk melihat nilai hexadecimal buka file 'suara1.h' menggunakan *text editor* misalkan, *sublime text 3.* Nilai hexadecimal bisa dilihatnya dari gambar yang ada di bawah ini.

```
unsigned char suara1_aac[] = {
  0xff, 0xf1, 0x4c, 0x40, 0x15, 0x9f, 0xfc, 0xde, 0x02, 0x00, 0x4c, 0x61,
  0x76, 0x63, 0x35, 0x38, 0x2e, 0x35, 0x34, 0x2e, 0x31, 0x30, 0x30, 0x00,
  0x01, 0xc8, 0x60, 0x44, 0x19, 0xc2, 0x25, 0x40, 0x00, 0x00, 0x00, 0x07,
  0x84, 0xf0, 0xa6, 0x26, 0x91, 0x09, 0xbf, 0x50, 0xce, 0x3a, 0x9c, 0x0a,
  0x16, 0x9c, 0x39, 0x5b, 0xf3, 0xab, 0x6f, 0xd4, 0x11, 0x2e, 0x74, 0xda,
  0xee, 0xe2, 0x5e, 0xf5, 0x1b, 0x7b, 0x86, 0x69, 0x56, 0x47, 0xc4, 0xbb,
  0x48, 0xdb, 0x6b, 0x9c, 0xa2, 0xec, 0xa7, 0x3d, 0xf3, 0xdd, 0x3d, 0xd3,
  0x8c, 0x59, 0x6d, 0x19, 0xa8, 0xab, 0xdf, 0x5d, 0x15, 0x51, 0xa7, 0x67,
  0xf5, 0xbf, 0x5e, 0xee, 0x2b, 0x64, 0x93, 0xd8, 0x48, 0x37, 0x5f, 0x9d,
```

Gambar 12. Hasil Hexadesimal File suara1.aac

Dikarenakan nilai hexadecimal yang dikonversi dari file 'suara1.aac' terlalu banyak. Untuk mempermudah dalam proses perhitungan dilakukan pengambilan sampel sebanyak jumlah hexadecimal yang diblock pada gambar dibawah.

```
unsigned char suara1_aac[] = {
  0xff, 0xf1, 0x4c, 0x40, 0x15, 0x9f, 0xfc, 0xde, 0x02, 0x00, 0x4c, 0x61,
  0x76, 0x63, 0x35, 0x38, 0x2e, 0x35, 0x34, 0x2e, 0x31, 0x30, 0x30, 0x00,
```

Gambar 13. Pengambulan Sampel Hexadesimal File suara1.aac

Sehingga sampel hexadecimal yang diambil adalah sebagai berikut:

ff, f1, 4c

Gambar 14. Pemisahan Sampel Biner File suara1.aac

3. Proses Penyisipan Algoritma LSB

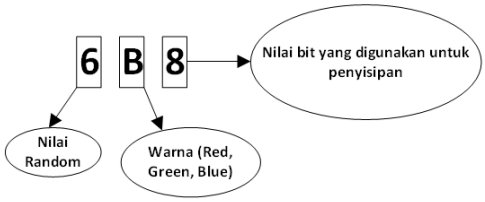
Untuk tahap penyisipan dilakukannya dengan cara acak, contohnya kalau terdapat 16 *byte* dan 3 *byte* dari sampel file 'suara1.aac' (24 bit= 11111111 11110001 01001100) yang bakal disembunyikan, maka dengan demikian *byte* yang digantikan pada bit yang ada di LSB dipilihnya dengan cara yang acak misal pada *byte* file 'suara1.aac' dibuat kode sebagai berikut:

6B8,10B8,9B8,15B8,1B8,2B8,3B8,4B8,12B8,13B8,8B8,11B8,14B8,1
6B8,5B8,7B8, 1G8,4G8,5G8,6G8,2G8,3G8,7G8,8G8

Gambar 15. Kode Acak File suara1.aac

Kode tersebut dibuat dari nilai acak yang dibangkitkan secara random untuk posisi nilai

byte pixel yang digabungkan dengan warna dan posisi nilai bit. Untuk lebih jelasnya lihat contoh kode pertama pada gambar dibawah ini.



Gambar 16. Penjelasan Kode

Sehingga nilai pixel yang akan menjadi tempat penyisipan adalah nilai pixel yang dibold dapat dilihatnya dari tabel yang terdapat di bawah ini.

Tabel 2. Penyisipan Pada Nilai Pixel

RGB	RGB	RGB	RGB
246, **77**, **66**	0, **16**, **122**	0,**0**,**0**	0, **54**,**0**
40, **0**, **0**	0, **0**, **0**	0, **2**, **88**	2, **88**, **0**
1, 0, **0**	0, 0, **24**	0, 0, **0**	61, 122,**192**
11, 19, **0**	19, 0, **0**	0, 0, **11**	0, 0, **0**

File ‘suara1.aac’ akan disisipkan dengan mengambil sampel nilai biner 24 bit terdepan yaitu 111111111111000101001100

- 6B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 10B8 = 24 dengan biner 00011000 disisip dengan 1 menjadi 00011001 maka nilai berubah menjadi 25
- 9B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 15B8 = 11 dengan biner 00001011 disisip dengan 1 menjadi 00001011 maka nilai berubah menjadi 11
- 1B8 = 66 dengan biner 01000010 disisip dengan 1 menjadi 01000011 maka nilai berubah menjadi 67
- 2B8 =122 dengan biner 01111010 disisip dengan 1 menjadi 01111011 maka nilai berubah menjadi 123
- 3B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 4B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 12B8 = 192 dengan biner 11000000 disisip dengan 1 menjadi 11000001 maka nilai berubah menjadi 193
- 13B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 8B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 11B8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 14B8 = 0 dengan biner 00000000 disisip dengan 0 menjadi 00000000 maka nilai berubah menjadi 0
- 16B8 = 0 dengan biner 00000000 disisip dengan 0 menjadi 00000000 maka nilai berubah menjadi 0
- 5B8 = 0 dengan biner 00000000 disisip dengan 0 menjadi 00000000 maka nilai berubah menjadi 0
- 7B8 = 88,dengan biner 01011000 disisip dengan 1 menjadi 01011001 maka nilai berubah menjadi 89
- 1G8 = 77 dengan biner 01001101 disisip dengan 0 menjadi 01001101 maka nilai berubah menjadi 77
- 4G8 = 54 dengan biner 00110110 disisip dengan 1 menjadi 00110111 maka nilai berubah menjadi 55
- 5G8 = 0 dengan biner 00000000 disisip dengan 0 menjadi 00000000 maka nilai berubah menjadi 0
- 6G8 = 0 dengan biner 00000000 disisip dengan 0 menjadi 00000000 maka nilai berubah menjadi 0
- 2G8 = 16 dengan biner 00010000 disisip dengan 1 menjadi 00010001 maka nilai berubah menjadi 17
- 3G8 = 0 dengan biner 00000000 disisip dengan 1 menjadi 00000001 maka nilai berubah menjadi 1
- 7G8 = 2 dengan biner 00000010 disisip dengan 0 menjadi 00000011 maka nilai berubah menjadi 2
- 8G8 = 88,dengan biner 01011000 disisip dengan 0 menjadi 01011001 maka nilai berubah menjadi 88

Hasil penyisipan algoritma LSB bisa dilihatnya dari Tabel yang ada dibawah.

Tabel 3. Penyisipan Pada Nilai Pixel

RGB	RGB	RGB	RGB
246, **77**, **67**	0, **16**, **123**	0, **1**, **1**	0, **55**,**1**
40, **0**, **0**	0, **0**, **0**	0, **2**, **89**	2, **88**, **1**
1, 0, **1**	0, 0, **25**	0, 0, **1**	61, 122, **193**
11, 19, **1**	19, 0, **0**	0, 0, **11**	0, 0, **0**

4. Proses Penyisipan Algoritma LSB

Pada tahapan ekstraksi teks dari file *stego image* dilakukannya dengan cara mempergunakan algoritma LSB yakni dengan membacakan bit nomor 8, dari tiap-tiap *byte pixel* citra *stego image* yang disesuaikan pada kunci kode yang dibuat yakni:

```
6B8,10B8,9B8,15B8,1B8,2B8,3B8,4B8,12B8,13B8,8B8,11B8,14B8,1
6B8,5B8,7B8, 1G8,4G8,5G8,6G8,2G8,3G8,7G8,8G8
```

Gambar 17. Kunci Kode

- 6B8, yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 10B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 9B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 15B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 1B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 2B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 3B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 4B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 12B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 13B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 8B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 11B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 14B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 0
- 16B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 0
- 5B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 0
- 7B8,yaitu *byte pixel* ke 6, warna *blue* dan bit ke 8,memiliki biner 1
- 1G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 0
- 4G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 1
- 5G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 0
- 6G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 0
- 2G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 1
- 3G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 1
- 7G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 0
- 8G8,yaitu *byte pixel* ke 6, warna *green* dan bit ke 8,memiliki biner 0

Hasil ekstraksi 8,bit diatas adalah 111111111111000101001100, 24 bit biner tersebut sesuai dengan sampel dari file ‘suara1.aac’

5. Hasil Pengujian Sistem Terhadap Penerapan Steganografi Pada Berkas Audio dengan Stego Citra Digital Menggunakan Metode LSB

Pada Gambar 18 yang ada di bawah ini memperlihatkannya hasil dari suatu pengujian yang ada pada tiap-tiap data berkas audio yang disisipkannya ke dalam citra digital dengan cara mempergunakan metode LSB. Dengan dilakukan pengujian ini maka daripada itu bajak mendapatkan suatu hasil dari dilakukannya penerapan steganografi dengan cara mempergunakan metode LSB yang tujuan ataupun target dari riset ini supaya bisa mengetahuinya kemampuan dari metode LSB perihal melakukan penyisipan berkas audio ke dalam citra digital.

Didasarkannya dari hasil evaluasi pada Gambar 18, semua pengujian yang dilakukan melaluinya tahapan penyisipan dengan *size* maupun ukuran piksel citra digital yang beda-beda. Makin besar ukuran dari citra digital yang digunakan, makin lama juga tahapan yang diperlukan dalam waktu pengekstraksiannya. Lalu jika makin kecil ukuran dari citra digital yang digunakannya, maka dengan demikian makin cepat juga tahapan yang diperlukan dalam waktu ekstraksinya.

Berkas Audio	*Cover Image*	Stego Image	*Insertion Time*	*Extraction Time*
suara1.aac 15 detik 137 KB	citra1.bmp 600 x 600 1,02 MB	sisip1.bmp 600 x 600 1,02 MB	697 Milidetik Rata-rata waktu = 695 milidetik	29 Milidetik Rata-rata waktu = 31 milidetik
suara1.aac 15 detik 137 KB	citra2.bmp 800 x 800 1,83 MB	sisip2.bmp 800 x 800 1,83 MB	423 Milidetik Rata-rata waktu = 420 milidetik	44 Milidetik Rata-rata waktu = 45 milidetik
suara2.aac 30 detik 267 KB	citra3.bmp 1000 x 1000 2,86 MB	sisip3.bmp 1000 x 1000 2,86 MB	534 Milidetik Rata-rata waktu = 538 milidetik	113 Milidetik Rata-rata waktu = 117 milidetik
suara2.aac 30 detik 267 KB	citra4.png 600 x 600 566 KB	sisip4.bmp 600 x 600 829 KB	931 Milidetik Rata-rata waktu = 925 milidetik	68 Milidetik Rata-rata waktu = 62 milidetik
suara3.aac 45 detik 399 KB	citra5.png 800 x 800 0,98 MB	sisip5.bmp 800 x 800 1,40 MB	840 Milidetik Rata-rata waktu = 842 milidetik	627 Milidetik Rata-rata waktu = 618 milidetik
suara3.aac 45 detik 399 KB	citra6.png 1200 x 1200 2,23 MB	sisip6.bmp 1200 x 1200 2,98 MB	164 Milidetik Rata-rata waktu = 165 milidetik	840 Milidetik Rata-rata waktu = 835 milidetik
suara1.aac 15 detik 137 KB	citra7.bmp 600 x 600 1,02 MB	sisip7.bmp 600 x 600 1,02 MB	265 Milidetik Rata-rata waktu = 262 milidetik	932 Milidetik Rata-rata waktu = 933 milidetik
suara2.aac 30 detik 267 KB	citra8.png 800 x 800 1,06 MB	sisip8.bmp 800 x 800 1,55 MB	93 Milidetik Rata-rata waktu = 99 milidetik	861 Milidetik Rata-rata waktu = 865 milidetik
suara3.aac 45 detik 399 KB	citra9.bmp 1000 x 1000 2,86 MB	sisip9.bmp 1000 x 1000 2,86 MB	747 Milidetik Rata-rata waktu = 755 milidetik	782 Milidetik Rata-rata waktu = 779 milidetik
suara4.aac 60 detik 530 KB	citra10.png 1200 x 1200 2,26 MB	sisip10.bmp 1200 x 1200 3,45 MB	923 Milidetik Rata-rata waktu = 932 milidetik	558 Milidetik Rata-rata waktu = 555 milidetik

Gambar 18. Hasil Pengujian Sistem

KESIMPULAN DAN SARAN

Kesimpulan

Didasarkannya pada riset maupun penelitian yang sudah dilakukannya oleh peneliti melalui serangkaian proses diawalinya dari penyisipan sampai dengan pada tahapan ekstraksi pada teknik yang digunakan steganografi terhadap file audio dengan berkas audio stego citra digital dan diakhiri rangkaian metode *Least Significant Bit* (LSB). Dapat disimpulkan bahwa hasil dari implementasi metode LSB yang diterapkan pada keamanan audio ke dalam citra digital sangat memiliki efisien dan tepat. Hal ini dikarenakan hasil dari teknik yang digunakan tidak menimbulkan kecurigaan serta tidak terlihat dalam kasat mata. Jika mempergunakan citra yang formatnya itu .BMP teruntuk dijadikannya suatu wadah pada tahapan penyisipan, maka dengan demikian tidaklah mengubah ukuran dari citra tersebut. Namun jika yang dipergunakan memiliki format .PNG teruntuk dijadikannya suatu wadah pada tahapan penyisipan, maka daripada itu bakal mengalaminya perubahan terkait dengan ukurannya, dikarenakan stego citra digital yang dihasilkan berubah jadi format .BMP. Walaupun begitu, citra yang dihasilkannya secara kasat mata tetaplah tidak mengalaminya suatu perubahan yang cukup signifikan maupun mencurigakan. Keseluruhan dari pengujian yang dilakukan dengan melalui tahapan penyisipan dengan berbagai macam ukuran piksel citra digital yang berbeda-beda. Maka daripada itu dapat diambil suatu kesimpulan bahwasanya jika semakin besar ukuran dari citra digital yang dipakai, maka akan makin lama juga tahapan dalam waktu pengekstraksiannya dan juga sebaliknya.

Saran

Didasarkannya pada riset maupun penelitian ini, teruntuk mengembangkan dan memanfaatkan penelitian adalah dengan cara memberikan penambahan terkait dengan opsi data rahasia yang bakal dilakukan dalam perihal penyisipan serta pengekstraksiannya, contohnya seperti data dokumen maupun video. Sistem ini bisa jauh lebih dikembangkannya lagi dengan cara yakni dibuatnya opsi *file* yang mempunyai berbagai macam format lainnya seperti halnya format

.JPEG, .GIF, .TIFF, serta juga .JPG. Pada penelitian selanjutnya disarankan untuk memberikan penambahan terkait dengan penggunaannya metode yang jauh lebih modern lagi serta yang pasti juga harus lebih mengamankan.

Ucapan Terima kasih

Terima kasih kepada Orang tua penulis, lalu kepada Bapak Dr. Mhd. Furqan, S.Si., M.Comp.Sc serta juga Bapak Rakhmat Kurniawan R, S.T., M.Kom, selaku Dosen Pembimbing I dan II saya yang sudah memberikan saya bimbingan dengan sangat baik, dan memberikan saya motivasi maupun arahan kepada penulis selama mengerjakan penulisan ini.

REFERENSI

Sriani, Triase, & Khairuna. (2017). Pendekomposisian Citra Digital dengan Algoritma DWT. *Algoritma*, 35-39.

M. F. Syawal, D. C. Fikriansyah, and N. Agani. 2016. "Implementasi Teknik Steganografi Menggunakan Algoritma Vigenere Cipher Dan Metode LSB," *J. TICOM*, vol. 4, no. 3, pp. 91–99.

D. Setiawan. 2018. "Dampak Perkembangan Teknologi Informasi dan Komunikasi Terhadap Budaya," *J. SIMBOLIKA Res. Learn. Commun. Study*, vol. 4, no. 1, p. 62, doi: 10.31289/simbollika.v4i1.1474.

S. Anwar. 2017. "Implementasi Pengamanan Data Dan Informasi Dengan Metode Steganografi LSB Dan Algoritma Kriptografi AES," *Jurnal*, vol. 6, pp. 2089–5615.

Aldo and L. Hakim. 2018. "Implementasi Steganografi Pada Citra Digital dan Kriptografi Algoritma Hill Cipher Untuk Pengamanan Informasi Berupa Text," *J. Ilm. Teknol. Inf. Terap.*, vol. V, no. 1, pp. 6–17.

Anti, U., Kridalaksana, A., & Khairina, D. (2017). Steganografi pada Video Menggunakan Metode Least Significant Bit (LSB) dan End Of File (EOF). *Jurnal Informatika Mulawarman*, 104-111.

B. Kuniadi, D. Puspitaningrum, and F. F. Coastera. 2017. "Perancangan Dan Pembuatan Aplikasi Steganografi Pesan Teks Pada Audio Digital Dengan Metode Least Significant Bit," *J. Rekursif*, vol. 5, no. 3, pp. 285–297.

Y. H. A. Sinaga and L. Sitorus. 2017. "Pengamanan File Citra Digital Dengan Menggunakan Metode Least Significant Bit Dan End Of File," *Jtiust*, vol. 02, no. 02, pp. 33–41.

E. Santa and Z. Situmorang. 2018. "Algoritma Hill Cipher Untuk Pengamanan Pengiriman File Via E-Mail," *Publ. Ilm. Teknol. Inf. Neumann*, pp. 46–50.

D. Darwis, 2017. "Teknik Steganografi untuk Penyembunyian Pesan Teks Menggunakan Algoritma GIFSHUFFLE," *J. Teknoinfo*, vol. 11, no. 1, p. 19, doi: 10.33365/jti.v11i1.6.

Nasution, Y., Furqan, M., & Sinaga, M. (2020). "Implementasi Steganografi Menggunakan Metode Spread Spectrum Dalam Pengamanan Data Teks Pada Citra Digital". *J-SAKTI*, 351-358.

S. Lutfi and R. Rosihan, 2018. "Perbandingan Metode Steganografi Lsb (Least Significant Bit) Dan Msb (Most Significant Bit) Untuk Menyembunyikan Informasi Rahasia Kedalam Citra Digital," *JIKO (Jurnal Inform. dan Komputer)*, vol. 1, no. 1, pp. 34–42, , doi: 10.33387/jiko.v1i1.1169.