JURNAL TIMES
TECHNOLOGY INFORMATICS & COMPUTER SYSTEM

Volume X No 2, Desember 2021
*p*ISSN : 2337 – 3601
*e*ISSN : 2549 – 015X

Tersedia *online* di http://ejournal.stmik-time.ac.id

PERANCANGAN APLIKASI PENYISIPAN PESAN MELALUI METODE MIXING *(PARITY CODING* DAN ENKRIPSI *ADVANCED ENCRYPTION STANDARD/AES*) PADA FILE MP3

Raja Nasrul Fuad
Program Studi Sistem Komputer
Universitas Pembangunan Panca Budi Medan
e-mail : rajanasrulfuad@dosen.pancabudi.ac.id

Abstrak

One of the encoding techniques is steganography as a way of hiding secret messages/files on the media as a container in a certain way so that no one knows or realizes that there is a secret message/file, while in cryptographic techniques the original message you want to send (plaintext) is changed or encrypted. with a key into a meaningless random information (ciphertext), because it is random, a suspicion arises of the message sent. The advantages of Steganography and Cryptography in data security so that through this study, two data security techniques were combined using the Parity Coding method for steganography and the AES (Advanced Encryption Standard) algorithm. In this study, the carrier or cover file proposed is an mp3 file, while the secret message/file to be inserted in the carrier is in the form of various types of txt formats.
Keyword*: Steganografi, Kriptografi, Parity Coding, AES*

1. Pendahuluan

Latar Belakang

Perkembangan yang pesat dalam proses pengiriman data berdampak besar pada masalah keamanan data. Pengirim data melalui media-media secara polos (*plain*) dikhawatarkan dapat terjadi kebocoran untuk itu perlu dilakukan proses pengamanan data yang akan dikirim, salah satunya dengan melakukan enkripsi pada data tersebut. Dewasa ini metode ini dirasakan masih kurang dalam keamanan data, maka data yang sudah di-enkripsi akan disembunyikan ke dalam data lain yang disebut media pembawa (*carrier*). Metode ini disebut dengan *Steganography*.

Metode Steganography dan *cryptography* merupakan teknik pengamanan data agar tidak dapat diketahui oleh yang tidak berwenang ketika membuka data. *Cryptography* merupakan teknik pengacakan data, sedangkan *steganography* merupakan teknik penyembunyian data. *Steganografi* membutuhkan dua properti yaitu media pembawa dan pesan rahasia. Media pembawa yang umum digunakan adalah gambar, suara, video, atau teks. Pesan yang disembunyikan dapat berupa artikel, gambar, daftar barang, kode program, atau pesan lain.Media pembawa *audio* dipilih karena mempunyai kapasitas yang lebih besar dibanding berkas teks maupun gambar dan tidak terlalu rumit dibandingkan berkas video (Ariyus, 2007).

Dalam melakukan pengacakan dan penyembunyian pesan, penulis akan menggunakan metode AES (*Advanced Enkryption Standard*) dan metode parity coding. MP3 adalah media penampungnya sehingga tidak menimbulkan kecurigaan yang terlalu berlebih. Mengatasi permasalahan tersebut, maka penggunaan *MP3* sebagai media steganografi merupakan langkah yang baik. Lalu lintas pertukaran*MP3* di internet merupakan hal biasa sehingga steganografi menggunakan *MP3* adalah teknik yang baik untuk mengamankan pesan rahasia melalui media internet.

Perumusan Masalah

Berdasarkan latar belakang masalah yang telah diuraikan di atas, dapat dirumuskan masalah-masalah yang timbul sebagai berikut:

1. Bagaimana menyembunyikan pesan ke dalam *file MP3* dengan metode *Parity Coding*?
2. Bagaimana cara pengacakan pesan yang akan disisipkan dengan kriptografi yang menggunakan metode AES (*Advance Encryption Standard*)?

2. Metode Penelitian

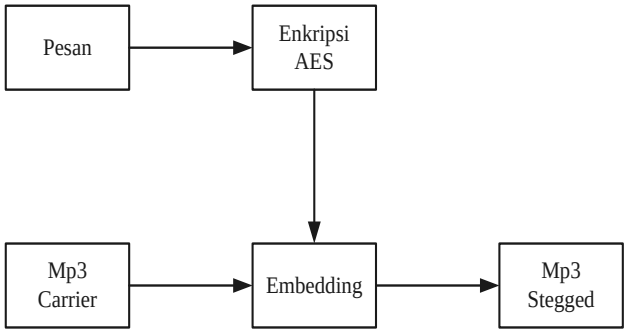
Analisis Sistem

Metode pemanfaatkan keterbatasan pada indera pendengaran manusia (*human auditory system*). Dalam steganografi, format *audio* memiliki kelebihan dibandingkan format citra maupun *video.*

Pada penelitian ini akan dirancang dan dibangun sebuah sistem steganografi, yaitu menyembunyikan sebuah pesan rahasia pada berkas MP3. Secara umum proses steganografi pada sistem ini ada dua, yaitu proses penyisipan pesan (*embedding*) dan proses pengungkapan pesan (*ekstraksi*).

1. Analisa Proses Embedding

Ada dua tahap yang dilakukan dalam proses embedding, yaitu enkripsi AES yang mentransformasikan pesan asli (*plaintext*) menjadi teks acak (*ciphertext*) , selanjutnya dilakukan penyisipan (*embedding*) dalam berkas MP3 pembawa (*carrier*). Ilustrasi dari proses embedding dapat dilihat pada Gambar 3.1.



Gambar 2.1 Proses Embedding

2. Proses Enkripsi AES

Dalam proses enkripsi AES ada 4 tahapan yaitu: (Ariyus, 2007)

a. Transformasi *SubByte*

Dalam operasi ini, setiap *byte* yang akan dienkripsi disubtitusikan dengan nilai *byte* lain dengan menggunakan S-box yang ada pada tabel 2.2.

Hasil yang didapat dari pemetaan dengan menggunakan tabel S-Box ini sebenarnya adalah hasil dari dua proses transformasi *byte*s, yaitu :

1) *Invers* perkalian dalam GF(28), adalah fungsi yang memetakan 8 bit ke 8 bit yang merupakan *invers* dari elemen *finite field* tersebut. Suatu *byte* a merupakan invers perkalian dari *byte* b bila a•b = 1, kecuali {00} dipetakan ke dirinya sendiri. Setiap elemen pada *state* akan dipetakan pada tabel *invers*. Sebagai contoh, elemen "01010011" atau {53} akan dipetakan ke {CA} atau "11001010". ? Transformasi *Affine* pada *state* yang telah dipetakan.
2) Transformasi *Affine* ini apabila dipetakan dalam bentuk matriks adalah seperti pada persamaan 2.4.
Misal input yang akan dienkripsi:

95	95	08	19
4f	6b	5c	6e
c8	89	80	26
fc	75	4e	6c

Maka setelah dilakukan operasi *SubByte* hasilnya adalah:

2a	2a	30	d4
84	7f	4a	9f
e8	a7	cd	f7
b0	9d	2f	50

a. Transformasi *ShiftRow*

Transformasi *ShiftRow* pada dasarnya adalah pergeseran sebagai contoh dengan menggunakan hasil output dari operasi *SubByte* maka setelah dilakukan operasi *ShiftRow* hasilnya adalah:

Penggeseran 1 byte pada row (baris) kedua	→ 2a 2a 30 D4 / 7f 4a 9f 84 / E8 A7 Cd F7 / B0 9d 2f 50
Penggeseran 2 byte pada row (baris) ketiga	2a 2a 30 D4

	7f 4a 9f 84 Cd F7 E8 A7 B0 9d 2f 50
Penggeseran 3 byte pada row (baris) keempat	2a 2a 30 D4 7f 4a 9f 84 Cd F7 38 A7 50 B0 9d 2f

Dan hasil transfarmasi Shiftrow adalah:

2a	2a	30	d4
7f	4a	9f	84
cd	f7	e8	a7
50	b0	9d	2f

Transformasi *MixColumn*

Mix Column mengoperasikan setiap elemen yang berada dalam satu kolom pada *state*. Elemen pada kolom dikalikan dengan suatu polinomial tetap.

Pada gambar 2.6 state S adalah masukan dari tranformasi ShiftRow dan State S' adalah keluaran dari tranformasi MixColumn. Representasikan biner elemen S sebagai polynomial lalu lakukan perkalian vektor antara baris ke-i pada kostan dan kolom ke-j pada S. Perkalian dan penjumlahan menggunakan GF(2^8). Untuk kolom ke-j j={0,1,2,3} pada keluaran transformasi MixColumn lakukan:

$$\begin{bmatrix} 02 & 03 & 1 & 1 \\ 1 & 02 & 03 & 1 \\ 1 & 1 & 02 & 03 \\ 03 & 1 & 1 & 02 \end{bmatrix} \begin{bmatrix} S0,0 & S0,1 & S0,2 & S0,3 \\ S1,0 & S1,1 & S2,1 & S1,3 \\ S2,0 & S2,1 & S2,2 & S2,3 \\ S3,0 & S3,1 & S3,2 & S3,3 \end{bmatrix} = \begin{bmatrix} S'0.0 & S'0,1 & S'0,2 & S'0,3 \\ S'1,0 & S'1,1 & S'1,2 & S'1,3 \\ S'2,0 & S'2,1 & S'2,2 & S'2,3 \\ S'3,0 & S'3,1 & S'3,2 & S'3,3 \end{bmatrix}$$

$S'_{1,0} = S_{0,0} * (x)$ xor $S_{1,0} * (x+1)$ xor $S_{2,0}$ xor $S_{3,0}$

$S'_{1,1} = S_{0,1}$ xor $S_{1,1} * (x)$ xor $S_{2,1} (x+1)$ xor $S_{3,1}$

$S'_{1,2} = S_{0,2}$ xor $S_{1,2}$ xor $S_{2,2} *(x)$ xor $S_{3,2} * (x+1)$

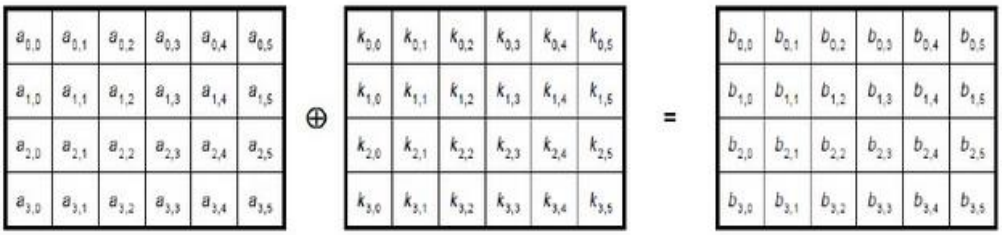
$S'_{0,j} = S_{0,3} * (x+1)$ xor $S_{1,3}$ xor $S_{2,3}$ xor $S_{3,3} *(x)$

Sebagai contoh dengan menggunakan hasil output dari operasi *ShiftRow* maka setelah dilakukan operasi *MixColumn* hasilnya adalah:

48	cd	af	ac
c8	0c	ab	1a
24	5e	d8	74
6c	b8	06	fa

Transformasi *AddRoundKey*

Pada proses *AddRoundKey*, sebuah round key ditambahkan pada *state* dengan operasi bitwise XOR. Setiap round key terdiri dari Nb *word* dimana tiap *word* tersebut akan dijumlahkan dengan *word* atau kolom yang bersesuaian dari *state.*



48	cd	af	ac
c8	0c	ab	1a
24	5e	d8	74
6c	b8	06	fa

Hasil dari transformasi MixColumn RoundKey 1

Cara diatas hanya berlaku untuk ekspanci kunci word pada elemen ketiga dan apabila state lebih dari 3 elemen adapun cara memcari ekspanci kunci pada elemen ke 4 adalah W[4] = w[0] xor SubWord(RotWOrd(w[3])) xor RC[1] Sebagai contoh dengan menggunakan hasil output dari operasi hasilnya adalah:

a3	c5	08	08
78	a4	ff	d3
00	ff	36	36
28	5f	01	02

Ab	08	A7	A4
B0	A8	54	C9
24	A1	C6	22
44	E7	07	08

Analisis Proses Embedding dengan Parity Coding

Proses pertama yang dilakukan dalam penyembunyian pesan adalah membagi media steganografi yang dalam hal ini adala berkas MP3 kedalam *region-region* . Banyaknya *region* ini ditentukan oleh panjang isi berkas yang dijadikan objek stegano. Berikut adalah spesifikasi faktor-faktor penyembunyian :

Nama objek : pesan.txt

Pesan *cipher* : 100010011100010110010110 (dalam biner)

Nama media : musik.mp3

Hasil *encode* media :

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

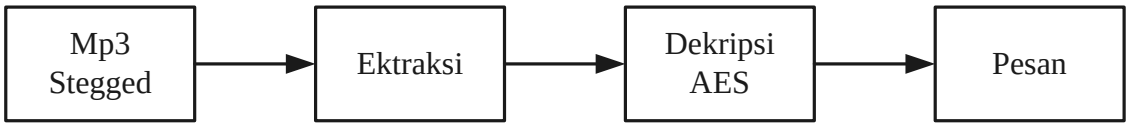
Dari spesifikasi diatas diketahui panjang dari isi pesan adalah 24 bit, sedangkan panjang bit dari media stegano adalah 120 bit yang terdiri dari 10 *frame*. Sebelum proses dilanjutkan ke pembagian *region* , informasi mengenai spesifikasi akan disimpan di *headerframe* awal media steganografi. Hal ini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

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

Dalam proses pembagian *region*, media steganografi akan dibagi menjadi *region -region* dengan banyak yang sesuai panjang isi pesan.

Analisis Proses Ekstraksi Pesan

Dalam proses ekstraksi ada dua tahap yang dilakukan yaitu seperti pada Gambar 2.2 berikut:



Gambar 2.2 Proses Ekstraksi Pesan

Proses Ekstraksi dengan Teknik Parity Coding

Ekstraksi yang dilakukan adalah kebalikan proses dari proses penyembunyian data seperti pada Gambar 2.2. Pesan rahasia akan diekstrak dari media stegano dengan kunci yang sama dengan yang dimiliki oleh pengirim. Kunci ini dipakai untuk membangkitkan kembali angka-angka *pseudorandom* yang dipakai untuk mengacak pesan terenkripsi sebelumnya sehingga bisa diurutkan kembali ke posisi semula. Setelah proses di atas, Proses yang dilakukan untuk mengekstraksi adalah dengan menyusun kembali nilai-nilai parity bit yang ada pada setiap *region* . Untuk menghasilkan *parity bit region* yang benar, tentu dibutuhkan pembagian *region* yang sama juga dengan proses penyembunyian. Oleh karena itu, dibutuhkan informasi spesifikasi penyembunyian yang disimpan pada *byte* pertama setiap *frame* media steganografi. Setelah dibaca barulah proses dilanjutkan ke pembagian *region* - *region* yang kemudian dilanjutkan dengan penghitungan parity bit*region* dimana nilai parity bit inilah yang disusun menjadi isi pesan setelah dienkripsi.

Proses Dekripsi AES

Karena AES merupakan algoritma simetris, transformasi *Cipher* dapat dibalikkan dan diimplementasikan dalam arah yang berlawanan untuk menghasilkan *inverse cipher*. Transformasi *byte* yang digunakan pada *invers cipher* adalah Inv*ShiftRow*s, Inv*SubByte* s, InvMixColumns, dan *AddRoundKey*. Berikut adalah proses dekripsinya:

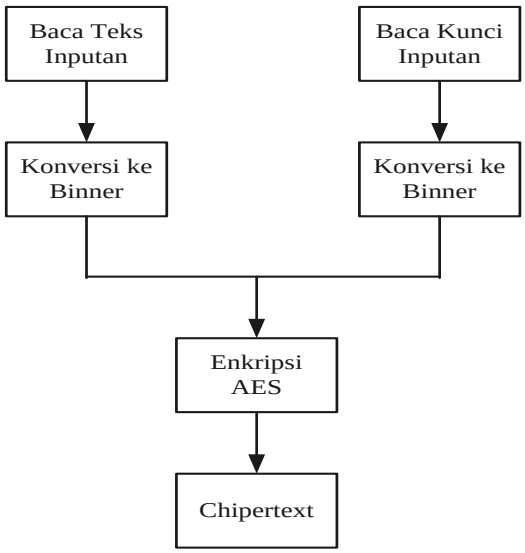
1. Inv*SubByte* s
   Inv*SubByte* s juga merupakan transformasi *byte*s yang berkebalikan dengan transformasi *SubByte* s. Pada Inv*SubByte* s, tiap elemen pada *state* dipetakan dengan menggunakan tabel 2.3. Langkah-langkah pemetaan dengan tabel inverse S-Box sama dengan proses pemetaan dengan tabel S-Box pada proses *SubByte.*
2. Inv*ShiftRow*s
   Inv*ShiftRow*s adalah transformasi *byte* kebalikan dari transformasi *ShiftRow*s. Transformasi yang terjadi Inv*ShiftRow*s, dilakukan dengan melakukan pergeseran bit ke kanan sedangkan pada *ShiftRow*s dilakukan pergeseran bit ke kiri. Pada baris kedua, pergeseran bit dilakukan sebanyak 3 kali, sedangkan pada baris ketiga dan baris keempat, dilakukan pergeseran bit sebanyak dua kali dan satu kali.
3. InvMixColumns
   Pada InvMixColumns, kolom-kolom pada tiap *state* (*word*) akan dipandang sebagai polinom atas GF(28) dan mengalikan modulo x4 + 1 dengan polinom tetap a-1(x) yang diperoleh dari (2.9) atau dalam bentuk matriks (2.11).
4. Inverse AddRoundKey
   Transformasi Inverse *AddRoundKey* tidak mempunyai perbedaan dengan transformasi *AddRoundKey* karena pada transformasi ini hanya dilakukan operasi penambahan sederhana dengan menggunakan operasi *bitwise* XOR.

Perancangan Proses Embedding

Ada dua tahap yang dilakukan dalam proses embedding, meliputi proses enkripsi AES yaitu mentransformasikan pesan teks yang diinputkan menjadi karakter acak (*ciphertext*) dengan menggunakan algoritma AES 128 bit dan tahap embedding yaitu menyisipkan (*ciphertext*) ke dalam berkas MP3 dengan teknik Parity coding.

Perancangan Proses Enkripsi AES

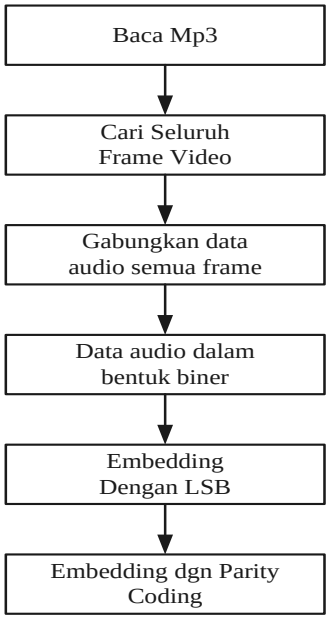
Proses enkripsi AES dapat dilihat dalam diagram alir pada gambar 2.4.



Gambar 2.4 Proses Enkripsi AES

Perancangan Proses Embedding Parity Coding

Embedding Dengan Teknik Parity Coding Proses embedding dapat dilihat dalam diagram alir sebagai berikut:



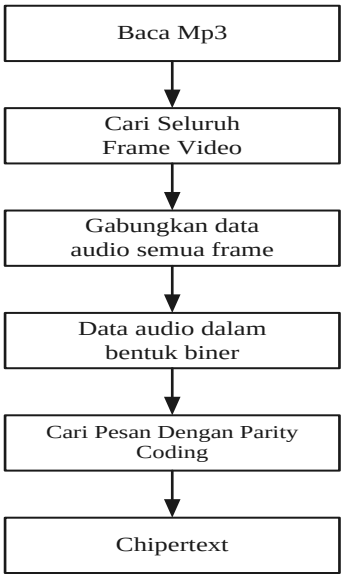
Gambar 2.5 Peranangan Proses Embedding

Perancangan Proses Ekstraksi Pesan

Proses ekstraksi pesan dilakukan proses ekstraksi atau pengambilan bit-bit pesan dari dalam MP3 carrier dengan menggunakan teknik Parity Coding , sehingga didapat *ciphertext* yang kemudian dilakukan proses dekripsi dengan algoritma AES 128 bit untuk mentrasformasikan *ciphertext* tersebut menjadi sebuah pesan asli.

Perancangan Proses Ekstraksi dengan Teknik Parity Coding

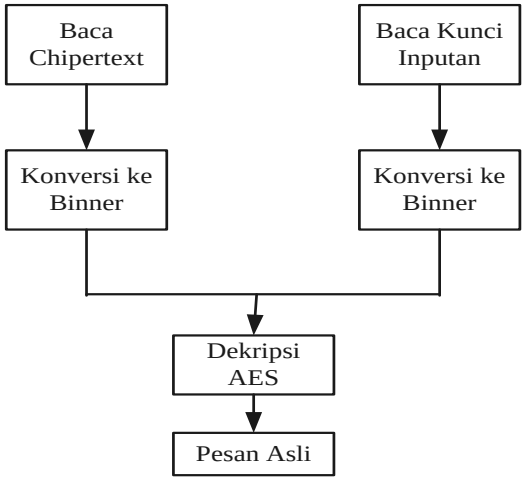
Proses ekstraksi pesan dapat dilihat dalam diagram alir sebagai berikut:



Gambar 2.6 Perancangan Proses Ekstraksi Pesan

Perancangan Proses Dekripsi AES

Proses dekripsi AES dapat dilihat dalam diagram alir sebagai berikut:



Gambar 2.7 Perancangan Proses Dekripsi AES

3. Hasil Penelitian Dan Pembahasan

Hasil Penelitian

Adapun hasil perancangan Penggabungan Metode Parity Coding dan Metode AES (*Advanced Encryption Standard*) Melalui Teknik Penyisipan Pesan Pada MP3. yang sudah dibuat, dapat dilihat dibawah ini.

Implementasi Sistem

Implementasi sistem merupakan prosedur yang dilakukan sebagai penyelesaian desain yang ada dalam desain sistem yang disetujui dan menguji, menginstal, memulai, serta menggunakan sistem yang baru atau sistem yang diperbaiki.

Komputer sebagai pendukung keputusan untuk memecahkan permasalahan membutuhkan suatu sistem yang baik, sehingga memungkinkan berhasilnya komputer dalam melaksanakan tugasnya, yaitu mengolah data menjadi informasi. Langkah implementasi yang dilakukan dalam Penggabungan Metode Parity Coding dan Metode AES (Advanced Encryption Standard) Melalui Teknik Penyisipan Pesan Pada MP3, Menyediakan perangkat keras (*Hardware*) dan perangkat lunak (*Software*). Dalam tahap ini disediakan perangkat keras.

Perangkat lunak yang dibutuhkan adalah Sistem Operasi Windows 7 dan bahasa pemograman yang digunakan untuk menulis program ke dalam komputer. Menguji sistem menjelaskan mengenai hasil pengujian sistem yang dilakukan pada Penggabungan Metode Parity Coding dan Metode AES (Advanced Encryption Standard) Melalui Teknik Penyisipan Pesan Pada MP3.

Metode pengujian sistem yang digunakan adalah *black-box testing*. *Black-box testing* adalah metode pengujian yang dilakukan terhadap sebuah aplikasi bukan terletak pada spesikasi logika/fungsi aplikasi tersebut, tapi masukan (*input)* dan keluaran (*output)*. Dengan berbagai masukan (*input)* yang diberikan akan dievaluasi apakah suatu sistem/aplikasi dapat memberikan keluaran (*output*) yang sesuai dengan harapan penguji. Pengujian sistem dilakukan dengan cara sebagai berikut:

1. Hasil pengujian sistem disajikan dalam bentuk tabel.
2. Pengujian ditargetkan pada setiap proses yang dimiliki Penggabungan Metode Parity Coding dan Metode AES (*Advanced Encryption Standard*) Melalui Teknik Penyisipan Pesan Pada MP3

Pembahasan

Tampilan Menu Utama

Sewaktu menjalankan program, *form* ini akan tampil sewaktu pertama sekali, pada form interface ini terbagi 2 *tab* yaitu :

a. *Tab* 1 HIDE MESSAGE yaitu proses untuk menyembunyikan pesan teks yang sebelumnya pesan teks tersebut di enkripsi dengan menggunakkan metode AES 128 Bit, untuk menyembunyikan pesan option yang dipilih adalah Record Audio on the fly, kemudian *Click Button* Browse untuk mencari *file* mp3 sebagai *file* kunci.
b. *Tab* 2 EXTRACT MESSAGE yaitu proses untuk mengambil pesan teks yang ada di dalam *file* .mp3 yang disimpan sebelumnya. Dan mendekripsikan pesan teks tersebut dengan menggunakkan metode AES 128 Bit untuk dikembalikan ke pesan aslinya (*plainteks*) , untuk mengambil pesan dari *file* mp3 option yang dipilih adalah Audio Mp3 dan kuncinya pada option Record Audio on the fly, kemudian Click Button Browse untuk mencari *file* .mp3 sebagai *file* kunci.

Pengujian Aplikasi

Untuk membuktikan aplikasi berjalan atau tidak maka diperlukan suatu pengujian pada aplikasi yang dibangun yaitu Penggabungan Metode Parity Coding dan Metode AES (*Advanced Encryption Standard*) Melalui Teknik Penyisipan Pesan Pada MP3.

a. Pengujian Penyembunyian Pesan

Tab yang digunakan untuk penyembunyian pesan adalah *tab* HIDE MESSAGE, adapun langkah-langkahnya:

1. Melakukan pemilihan *file* mp3 pada option *Record Audio on the fly*. Pada pengujian ini *file* mp3 yang diuji adalah ” 9 BATAK.mp3”
2. Pemilihan option save message as untuk memilih lokasi penyimpanan dan nama *file* mp3 adalah “test_stegano1.mp3”
3. Pada option display dimasukkan pesan teks yang akan disembuyikan contoh pesan yang disembunyikan adalah “Selamat datang di aplikasi saya”.
4. Langkah berikutnya adalam pengisian *password* AES *Key* pada pesan yang akan di enkripsi, AES *Key* “123”.
5. Penekan tombol Encrypt untuk mengubah pesan teks (*plainteks)* ke *chiperteks*

```
private void btnHide_Click(object sender, System.EventArgs e) {
                WaveUtility utility = new WaveUtility(audioStream, destinationStream);
                                utility.Hide(messageStream, keyStream);
                        }
                        catch(Exception ex) {
                                this.Cursor = Cursors.Default;
                                MessageBox.Show(ex.Message);
                        }
                        finally{
                                if(keyStream != null){ keyStream.Close(); }
                                if(messageStream != null){
messageStream.Close(); }
                                if(audioStream != null){ audioStream.Close(); }
                                if(sourceStream != null){ sourceStream.Close(); }
                                if(destinationStream != null){
destinationStream.Close(); }
                                this.Cursor = Cursors.Default;
                        }
                }
        }
```

6. Pada posisi ini dilakukan penekanan tombol Hide Message.

Listing program penyembunyian pesan pada mp3

Untuk membuktikan apakah *key file* .mp3 masuk atau tidak sewaktu *file key* mp3 di save diatas maka harus dilakukan pengecekan. Nah berikut hasilny bahwa *key file* mp3 berhasil tersimpan .

b. Pengujian Pengungkapan Pesan

Pada pengujian pengungkapan pesan yang disembunyikan ke *file* mp3 adapun langkah-langkahnya :

1. Pemilihan *tab* HIDE MESSAGE, pada option Audio Mp3 browse path .mp3 yang disimpan pada waktu penyembunyian pesan yaitu “stegano1.mp3”.
2. Pada *key file* pilih *key file* mp3 dengan nama “9 BATAK.mp3” pada penyembunyian sebelumnya.
3. Click button EXTRACT MESSAGE sewaktu button ini diclick makan pesan akan tampil tetapi pesan yang tampil masih berbentuk chiperteks.
4. Masukkan kunci ke dalam AES *Key* pada waktu penyembunyian *Key* dimasukkan adalah “123” kemudian Click button Decrypt maka pesan asli akan ditampilkan

Listing program ekstraksi pesan dari mp3

```
private void btnExtract_Click(object sender, System.EventArgs e)
{
utility.Extract(messageStream, keyStream);
                        messageStream.Seek(0,
SeekOrigin.Begin);
                        if(rdoMessageDstFile.Checked){
FileStream fs = new FileStream(txtMessageDstFile.Text,
FileMode.Create);
                        byte[] buffer = new
byte[messageStream.Length];
                  messageStream.Read(buffer, 0, buffer.Length);
                                    fs.Write(buffer, 0,
buffer.Length);
                                    fs.Close();
                                }else{ //display result
                                    txtExtractedMessage.Text =
new StreamReader(messageStream).ReadToEnd();
                                }
                        }
```

4. Kesimpulan

Berdasarkan hasil penelitian dan hasil yang ada, maka dapat dijadikan simpulan sebagai berikut:

1. Steganografi audio dengan teknik *Parity Coding* dan enkripsi *AES* (*Advanced Encryption Standard*) bias diterapkanpadaberkasaudio*MP3*.
2. Pengungkapan pesan hanya dapat dilakukan jika pengguna memberikan kunci dekripsi yang sesuai, jika tidak maka pesan tidak akan dapat diungkap.
3. Kapasitas yang dapat ditampung oleh sebuah MP3 bergantung pada banyaknya *frame* yang mengandung byte homogen, dan bukan bergantung pada ukuran MP3. Oleh karena itu tidak semua MP3 dapat dijadikan sebagai wadah penampung pesan. Hanya MP3 dengan *frame* yang mengandung byte- byte homogen saja yang dapat digunakan sebagai penampung pesan.
4. Aplikasi ini tidak dapat mendukung aspek *robustness*, karena ketika dilakukan perubahan pada sinyal MP3 maka struktur byte pada MP3 tersebut juga akan berubah, sehingga akan mengganggu susunan byte pesan rahasia yang ada di dalamnya.
5. Proses enkripsi pada pesan yang akan disisipkan menggunakan metode *Advanced Encryption Standard* (AES) dibuat menggunakan visual basic 2008.
6. Perancangan aplikasi steganografi dengan metode *parity coding* pada dokumen audio dengan format MP3 dibuat mengguanakan Visual Basic 2008 4.

5. Daftar Pustaka

[1] Alatas, Putri. (2009).Implementasi Teknik Steganografi dengan Metode LSB pada Citra Digital. Universitas Gunadarma : Jakarta

[2] Ariyus, Dony. 2007. Keamanan Multimedia. Yogyakarta : Penerbit Andi
Cahyadi, Tri.(2012).Steganografi LSB dengan Enkripsi Vigenere Chiper pada Citra JPEG.Universitas Dipenogoro.

[3] Finna Monica, Ahmadi Surahman, Aplikasi Steganografi Citra Digital Menggunakan Metode LSB (Least Significant Bit), UIN Sunan Gunung Djati

[4] Setiawan, Wawan.(2012)Aplikasi Keamanan Pesan Menggunakan Algoritma Steganografi dan Kriptografi.Teknik Informatika UPN: Yogyakarta.