KLIK: Kajian Ilmiah Informatika dan Komputer
ISSN 2723-3898 (Media Online)
Vol 2, No 1, Agustus 2021
Hal 20-27
https://djournals.com/klik

Implementasi Metode Enhanced Audio Steganogafi (EAS) Untuk Penyembunyian Text Terenkripsi Algoritma Gost

Selamat G M Siregar

Program Studi Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: Selamatsiregar700@gmail.com

Abstrak− File audio dengan format Mp3 digunakan sebagai media penampungan (cover text). Untuk dapat menyisipkan pesan rahasia ke dalam media penampung membutuhkan sebuah metode yang dapat memodifikasikan objek menjadi objek yang baru dengan informasi rahasia di dalamnya tanpa terjadi perubahan yang mencolok dari objek awal. Metode yang digunakan dalam penelitian ini adalah metode EAS (Enhanced Audio Steganography) dan untuk mengenkripsi pesan digunakan algoritma GOST. EAS (Enhanced Audio Steganography) yang merupakan suatu metode untuk menyisipkan suatu pesan yang di enkripsi terlebih dahulu untuk memperkuat pesan rahasia dan kemudian disisipkan ke dalam file audio. Enkripsi yang dilakukan pada pesan menggunakan algoritma GOST. Metode GOST merupakan suatu algoritma block cipher yang dikembangkan oleh seorang berkebangsaan Uni Soviet. GOST merupakan singkatan dari "Gosudarstvennyi Standard" atau "Government Standard". Algoritma ini merupakan suatu algoritma enkripsi sederhana yang memiliki jumlah proses sebanyak 32 round dan menggunakan 64 bit block cipher dengan 256 bit key. Metode GOST juga menggunakan 8 buah S-Box yang berbeda – beda dan operasi XOR serta Left Circular Shift. Namun, memodifikasi bit file audio secara berurutan dapat membuat noise yang besar dan mengundang kecurigaan bagi orang lain bahwa ada pesan yang tersisipkan didalamnya. Metode yang mampu mengurangi terjadinya noise pada file audio setelah disisipkan pesan adalah metode EAS (Enhanced Audio Steganography). Metode EAS merupakan modifikasi dari metode LSB. Dimana pada metode EAS, byte yang digunakan sebagai penampung hanya selective byte saja, yaitu penyisipan bit pada media penampung hanya dilakukan pada blok yang bernilai 254 atau 255 byte saja, sehingga media penampung tidak akan mengalami kerusakan yang signifikan. Pertama input pesan yang berformat *.doc atau *.pdf, akan dienkripsi menggunakan algoritma Gost, proses enkripsi diawali dengan menginputkan plaintext dan kunci. Setelah pesan terenkripsi selanjutnya proses embedding menggunakan metode EAS. Input file audio yang akan digunakan sebagai penampung, setelah itu file yang sudah terenkripsi akan disisipkan pada file audio. Pada proses embedding dilakukan hitung besar lokasi yang tersedia pada file audio. File audio yang digunakan sebagai media penampung harus memiliki jumlah byte bernilai 254 atau 255 yang cukup untuk menampung file pesan..

Kata Kunci: Kriptografi, Steganografi, EAS, Text, Audio, GOST.

Abstract− Audio files in the Mp3 format are used as cover media. To be able to insert a secret message into the container media requires a method that can modify the object into a new object with confidential information in it without any noticeable changes from the initial object. The method used in this study is the EAS (Enhanced Audio Steganography) method and to encrypt the message using the GOST algorithm. EAS (Enhanced Audio Steganography) which is a method for inserting an encrypted message first to strengthen a secret message and then inserting it into an audio file. Encryption is done on the message using the GOST algorithm. The GOST method is a block cipher algorithm developed by a Soviet national. GOST is an abbreviation of "Gosudarstvennyi Standard" or "Government Standard". This algorithm is a simple encryption algorithm that has 32 rounds of processes and uses 64 bit block cipher with 256 bit keys. The GOST method also uses 8 different S-Boxes and XOR and Left Circular Shift operations. However, modifying audio file bits sequentially can create huge noise and invite suspicion for others that there is a message embedded in it. The method that is able to reduce the occurrence of noise in audio files after inserting a message is the EAS (Enhanced Audio Steganography) method. The EAS method is a modification of the LSB method. Where in the EAS method, the bytes that are used as containers only selective bytes, that is, the insertion of bits in the storage media is only done in blocks that are 254 or 255 bytes only, so that the storage media will not experience significant damage. The first input message format *. Doc or *. Pdf, will be encrypted using the Gost algorithm, the encryption process begins with input plaintext and key. After the message is encrypted then the embedding process uses the EAS method. Input audio file that will be used as a container, after that the encrypted file will be inserted in the audio file. In the embedding process, a large number of locations are available on the audio file. Audio files that are used as storage media must have a number of bytes worth 254 or 255 which are sufficient to hold the message file.

Keywords: Cryptography, Steganography, EAS, Text, Audio, GOST.

1. PENDAHULUAN

Masalah keamanan pesan teks merupakan salah satu masalah yang terdapat dalam sebuah komunikasi yang berlangsung baik melalui surat, media elektronik, internet, sehingga data dapat diakses dengan mudah dan cepat hal ini membuat banyaknya data yang disalahgunakan orang yang tidak bertanggungjawab. Perubahan yang terjadi dalam pesan teks sangat merugikan sipemilik pesan teks hal ini diakibatkan pihak yang tidak bertanngungjawab dengan mudahnya mengakses pesan teks secara umun sehinnga pesan teks tersebut dapat di ragukan keasliannya.

Data terdiri dari berbagai macam jenis data yaitu, gambar, teks, audio dan video. Berkat kemajuan jaman semua itu dapat dinikmati secara bebas dan di sebaruaskan secara global. Namun tidak bisa dipungkiri banyaknya data yang di sebarluaskan secara bebas rentan membuat data tersebut dapat diubah, dihapus dan dimanipulasi orang yang tidak berkepentinggan membuat pesan teks tersebut diragukan keasliannya dari orang yang tidak bertanggungjawab [1]. Salah satu jenis data yang sering dimanipulasi keasliannya adalah data teks. Kriptografi secara umum merupakan salah satu teknik pengaman data atau informasi yang bersifat rahasia atau pribadi yang dilakukan dengan cara mengolah *plaintext* dengan menggunakan suatu proses enkripsi sehingga menghasilkan *chipertext* tanpa orang lain pahami. *Chipertext*

tersebut dikembalikan menjadi plainteks setelah melalui proses deskripsi pada algoritma GOST. Dalam hal ini kriptografi harus mencapai pengamanan diantaranya adalah kerahasiaan (menjaga informasi dari pihak yang tidak bertanggungjawab), integritas (agar tidak terjadi perubahan data teks dari pihak tidak bertanggungjawab), autentikasi (identifikasi data teks terhadap orang yang tidak berkepentingan dengan cara menyesuaikan data dari sistem itu sendiri), dan ketiadaan penyangkal agar tercegah dari suatu aksi yang dilakukan pelaku sistem informasi [2]. Steganografi dapat dipandang sebagai lanjutan dari kriftografi jika kriftografi merahsiakan maka steganografi menyembunyikan atau menutupi data yang di rahasiakan dalam prakteknya steganografi merupakan penyisipan data teks yang telah di enkripsikan kemudian *chipertext* disisipkan ke audio sehingga orang tidak dapat mengetahuinya. Steganografi merupakan dua media yang berbeda salah satunya adalah penyembunyian hal ini dapat dijalankan dengan menggunakan metode *enhanced audio steganografi* (*EAS*) metode ini dapat menyembunyikan data teks yang telah di*chipertextkan* kedalam *audio layer 3* (MP3) supaya keamanan data teks tersebut dapat terjaga dari pihak orang lain yang tidak berkepentinggan[3].

2. METODOLOGI PENELITIAN

2.1 Enhanched Audio Steganografi

Metode yang mampu mengurangi terjadinya noise pada file audio setelah disisipkan pesan adalah metode EAS (EnhancedAudio Steganography). Metode EAS merupakan modifikasi dari metode LSB. Dimana pada metode EAS, byte yang digunakan sebagai penampung hanya selective byte saja, yaitu penyisipan bit pada media penampung hanya dilakukan pada blok yang bernilai 254 atau 255 byte saja, sehingga media penampung tidak akan mengalami kerusakan yang signifikan File audio dengan format MP3 yang digunakan sebagai media penampung karena file audio dengan format MP3 tidak terkompresi dan memiliki kualitas suara yang baik, jadi ketika terjadi perubahan pada file audio tersebut tidak akan menimbulkan derau sehingga tidak mengundang kecurigaan. EAS merupakan algoritma enkripsi paling kuat yang sangat kompleks untuk dipecahkan. Menggunakan algoritma LSB (Least Significant Bit) yang dimodifikasi untuk mengkodekan pesan ke dalam audio. Dilakukan manipulasi tingkat bit untuk mengkodekan pesan. Ide dibalik penelitian ini adalah memberikan metode yang baik dan efisien untuk menyembunyikan data dari hacker dan dikirim ke tempat tujuan secara lebih aman. Kualitas suara tergantung pada ukuran audio yang pengguna pilih dan panjang pesannya. Meskipun menunjukkan penyimpangan tingkat bit pada bagan frekuensi, secara keseluruhan perubahan audio tidak dapatdi bedaka. Proses decode yang merupakan proses untuk membaca pesan di dalam file audio. Ketika file audio dimasukkan, sistem akan membaca apakah ada data yang disisipkan atau tidak. Sistem akan membaca panjang data yang disisipkan di dalam byte file audio yang bernilai 254 atau 255[4] [5].

2.2 Algoritma Gosudarsvennyi Standard

Algoritma *Gosudarsvennyi Standard* (GOST) atau standarad pemerintah adalah salah satu algoritma enkripsi dari Negara *uni soviet* yang dikembangkan pada tahun 1970. GOST beroperasi pada ukuran *block* pesan yang berukuran panjang 64 bit, sedangkan panjang ukuran kunci 256 bit, jumlah putaran pada GOST adalah sebanyak 32 kali putaran, setiap putaran mempunyai kunci internal. Untuk putaran pertama-tama *plainteks* dengan ukuran 64 bit dipecahkan menjadi 32 bit, bagian kiri L dan 32 bit bagian kana R. Kunci GOST menggnakan 8 buah S-box yang berbeda-beda, GOST juga menggunakan operasi XOR dan *Lift circular shift* 11 bit atau *rotate left shift* dan menggunakan modulo 2^{32}, subkunci untuk putaran i adalah Ki [6]. Pada suatu putaran ke-i operasinya adalah sebagai berikut :

1. Proses Pembangkitan Kunci
   Kunci internal pada algoritma GOST dibangkitkan dari kunci internal yang diberikan pengguna. Pembangkitan kunci internal dilakukan dengan membagi kunci eksternal 256 bit (k1, k2, k3, k4, k5, k6, k7, ...k256) kedalam delapan bagian yang masing masing panjangnya 32 bit yang dimana pembagiannya adalah sebagai berikut:
   K0 = (K32, ... ,K1)
   K1 = (K64, ... ,K33)
   K2 = (K96,..... ,K65)
   K3 = (K128, ... ,K97)
   K4 = (K160,..... ,K129)
   K5 = (K192, ... ,K161)
   K6 = (K224,..... ,K193)
   K7 = (K256, ... ,K225)
2. Proses Enkripsi
   Proses Enkripsi pada algoritma GOST untuk suatu putaran (iterasi), adalah sebagai berikut :
   a. 64 plainteks dibagi menjadi 2 bagian 32 bit, yaitu Li dan Ri dengan cara sebagai berikut :
      Input a1(0), a2(0), ... ,a32(0) ; b1(0), b2(0), ... ,b32(0).
      Ro = a32(0), a31(0), ... ,a1(0).
      L0 = b32(0), b31(0), ... ,b1(0).
   b. (Ri + Ki) mod 2^{32} hasil dari penjumlahan 2^{32}berubah menjadi 32 bit

c. Hasil dari penjumlahan modulo 2^{32} dibagi menjadi 8 bagian dimana masing masing dibagi menjadi 4 bit, setiap pembagian dimasukkan kedalam tabel S-Box yang berbeda. 4 bit pertama menjadi input dari S-Box 0, 4 kedua menjadi S-box 1 dan seterusnya. dan dapat dilihat dari contoh tabel S-Box berikut ini:

Tabel 1. S-Box Algoritma GOST

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	15	15
0x	4	10	9	2	13	8	0	14	6	11	1	12	7	15	5	3
1x	14	11	4	12	6	13	15	10	2	3	8	1	0	7	5	9
2x	5	8	1	13	10	3	4	2	14	15	12	7	6	0	9	11
3x	7	13	10	1	0	8	9	15	14	4	6	12	11	2	5	3
4x	6	12	7	1	5	15	13	8	4	10	9	14	0	3	11	2
5x	4	11	10	0	7	2	1	13	3	6	8	5	9	12	15	14
6x	13	11	4	1	3	15	5	9	0	10	14	7	6	8	2	12
7x	1	15	13	0	5	7	10	4	9	2	3	14	6	11	8	12

Sumber: Irfan anas, 2018 [8]

2.3 Teks

Teks adalah satuan lingual yang dimediakan secara tulis atau lisan dengan tata organisasi tertentu untuk mengungkapkan makna secara kontektual. Istilah teks dan wancana dianggap sama dan hanya dibedakan dalam hal wacana lebih bersifat abstrak dan merupakan realisasi makna dari teks dan tersusun sesuai dengan jenisnya. Teks memiliki beberapa jenis format yang sering digunakan, antara lain [7], yaitu:

1. *Portable Documen Format* (PDF)
2. *Electronic Publication* (Epub)
3. *DjVu*
4. *Mobipocket*

2.4 Audio

Seiring berkembangnya gejet saat ini audio merupakan alat peraga yang sifatnya dapat didengar, yang memiliki arti yaitu suara yang keluar dan dapat didengarkan oleh telinga. Audio memiliki beberapa jenis format yang biasanya dapat ditemukan di berbagai media elektronik, salah satunya adalah audio format audio Player3 (MP3). Audio merupakan audio dengan ukuran sangat kecil dibandingkan dengan format lain. Untuk menentukan nilai bit dari audio maka dibutuhkan alat bantu yaitu HxD dimana HxD dapat mengubah audio kedalam bentuk hexsadesimal hal ini bertujuan agar penyisipan lebih mudah dilakukan Format audio juga memiliki berbagai macam format selain MP3[8]. Audio Player3 yang digunakan sebagai penampung data yang disembunyikan memiliki struktur data yang ditunjukkan secara umum terdiri dari tiga bagian dasar, yaitu (1) deskripsi chunk RIFF, (2) format subchunk dan (3) data sub-chunk.

3. HASIL DAN PEMBAHASAN

Penerapan algoritma GOST dalam penyembunyian text terenkripsi yamg digunakan untuk terlebih dahulu proses yang dilakukan di algoritma GOST adalah penentuan kunci. Kunci internal pada algoritma GOST dibangkitkan dari kunci internal yang diberikan pengguna. Pembangkitan kunci internal dilakukan dengan membagi kunci eksternal 256 bit (k1, k2, k3, k4, k5, k6, k7, ...k256) kedalam delapan bagian yang masing masing panjangnya 32 bit yang dimana pembagiannya adalah sebagai berikut:

1. Konversi plainteks dan kunci kebiner
2. Klompokkan biner kunci menjadi 8 klompok dengan 32 bit perkelompok.
   K0 = (K32, ... ,K1)
   K1 = (K64, ... ,K33)
   K2 = (K96,..... ,K65)
   K3 = (K128, ... ,K97)
   K4 = (K160,..... ,K129)
   K5 = (K192, ... ,K161)
   K6 = (K224,..... ,K193)
   K7 = (K256, ... ,K225)
3. Kelompokkan *plainteks* men jadi 2 bagian yaitu R[0] dan L[0] dengan jumlah tiap kelompok adalah 32 bit. 32 bit bagian kiri menjadi R[0] dan bit ke 33 sampai bit ke 64 sebelah kanan L[0]. Proses penulisan dilakukan secara terbalik
   R[0] = bit[32], bit[31], ... bit[1]
   L[0] = bit[64], bit[63], ... bit[33]
   Plainteks = SELAMATS
   Kunci = Rahasia

CHAR	DEC	BINER
R	82	01010010
a	97	01100001

h	104	01101000
a	97	01100001
s	115	01110011
i	105	01101001
a	97	01100001
R	82	01010010
a	97	01100001
h	104	01101000
a	97	01100001
s	115	01110011
i	105	01101001
a	97	01100001
R	82	01010010
a	97	01100001

Gabungkan Biner Plainteks

01010010011000001011010000110000101110011011010010110000101010010011000010110100001100001011101101001011000010101001001100001011010000110000101110011011010010110000101010010011000010101000011000010111001101101001011000010101001001100001011010000110000101110011011010010110000101010010011000010110100001100001

Kunci rahasia merupakan salah satu kunci yang digunakan untuk enkripsi yang digunakan, hal ini membuat kunci harus terdiri dari 8 kelompok dengan 32 bit tiap kelompok dan setiap penulisan kunci yang digunakan ditulis dari sebelah kanan ke kiri

Jadi hasil Proses pengelompokan biner kunci diatas dengan mengambil 32 bit pertama dan ditulis binernya dari bit 32 hingga bit 1(ditulis terbalik) sampai seterusnya.

Tabel 1. Tabel Pembangkitan Kunci

Kode Kunci	Penulisan Kunci	Biner
K[0] =	32-1	10000110000101101000011001001010
K[1] =	64-33	01001010100001101001011011001110
K[2] =	96-65	11001110100001100001011010000110
K[3] =	128-97	10000110010010101000011010010110
K[4] =	160-129	10010110110011101000011000010110
K[5] =	192-161	00010110100001100100101010000110
K[6] =	224-193	10000110100101101100111010000110
K[7] =	256-255	10000110000101101000011001001010

Proses enkripsi adalah proses pengkodean yang dilakukan dengan mengubah pesan teks kedalam biner dengan menggunakan algoritma GOST, banyak putaran yang dilakukan algoritma GOST untuk mendapatkan kode pesan teks (*Chipherteks*) sebanyak 32 putaran dimana kunci yang digunakan mulai K[0] samapai K[7] sampai putaran 32 namun pada putaran ke 24 kunci yang digunakan terbalik mulai dari K[7] sampai K[0] hal ini bertujuan untuk mendapatkan cipherteks dari pesan teks yang akan disisipkan. Sebelum dilakukan penyisipan *cipherteks* kedalam audio maka audio terlebih dahulu diubah kedalam bentuk nilai bit dengan menggunakan HxD , HxD merupakan suatu aplikasi yang dapat mencari nilai bit dari suatu *cover* yang akan digunakan sebagai penampung. Setelah nilai-nilai bit dari audio ditemukan maka akan dilakukan sleksi nilai bit yang akan digunakan sebagai penampung, karena tidak semua nilai bit audio dapat digunakan sebagai penampung dari biner *cipherteks* yang akan disisipkan, nilai audio yang digunakan hanya bit yang bernilai 254 dan 255 hal ini bertujuan agar menjaga keaman dan keaslian audio dari kerusakan yang tidak diinginkan. Algoritma EAS hanya menyisipkan biner kedalam bit audio yang bernilai 254 dan 255, dalam penyisipan *cipherteks* nilai audio 254 dan 255 harus lebih banyak dari jumlah biner yang akan disisipkan , jika nilai bit yang akan disisipkan lebih banyak dari bit audio yang bernilai 254 dan 255 makan penyisipan tidak dilakukan.

1. Proses Enkripsi

   Proses enkripsi merupakan proses pengkodean yang dilakukan terhadap data teks yang akan disisipkan kedalam audio, hal ini bertujuan supaya pesan teks yang akan dikirim lebih aman dari pihak yang tidak bertanggungjawab dan mengubah teks asli menjadi teks yang lain.

   Proses Enkripsi :

   Tahap 1

   Ro = 10000010001100101010001011001010

   Lo = 11001010001010101000001010110010

   Tahap 2 $R(0) + K(0) \text{ MOD } 2^{32}$

   R[0] = 2184356554

   K[0] = 2249623114

   Hasil = 4433979668 Mod 2^{32}

   = 139012372

Biner = 00001000010010010010100100010100

Tahap 3 Pecah Hasil Tahap 2 menjadi 8 Kelompok

Biner	Des	S-Box	Des	Biner
0000	0	S-Box(0)	4	0100
1000	8	S-Box(1)	2	0010
0100	4	S-Box(2)	10	1010
1001	9	S-Box(3)	4	0100
0010	2	S-Box(4)	7	0111
1001	9	S-Box(5)	6	0110
0001	1	S-Box(6)	11	1011
0100	4	S-Box(7)	5	0101

Tahap 4 Hasil kemudian dilakukan Rotate Shif Left 11 kali

gabungan : 01000010101001000111011010110101

Rotate 11 00100011101101011010101000010101

Tahap 5 R(1) R(0) XOR L(0) --> R(0) adalah setelah digeser / hasil LRS

R(0)= 00100011101101011010101000010101

L(0)= 11001010001010101000001010110010

R(1)= 11101001100111110010100010100111

Tahap 6 L(1) = R(0) sebelum proses

L(1) = 10000010001100101010001011001010

HASIL PUTARAN KE-0

L(1) = 10000010001100101010001011001010

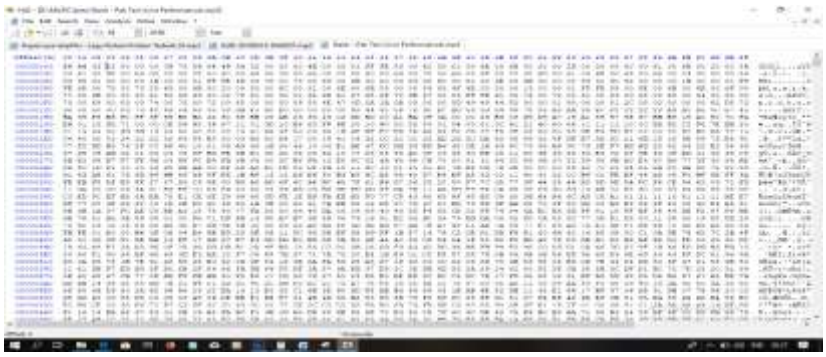
R(1) = 11101001100111110010100010100111

2. Mengubah audio kedalam hexsadesimal

Proses pengambilan hexsadesimal *audio* menggunakan HxD merupakan salah satu langkah untuk menyeleksi *audio* yang akan dijadikan sebagai penampung dari biner *cipherteks* yang akan disisipkan hal ini bertujuan agar jumlah nilai bit audio 254 dan 255 dapat diketahui jika nilai bit audio yang bernilai 254 dan 255 lebih banyak dari biner yang akan disisipkan maka penyisipan dilakukan, hal ini bertujuan agar audio yang digunakan sebagai penampung biner tampa mengalami kerusakan yang parah terhadap *audio*, berikut *audio* yang akan digunakan sebagai penampung:

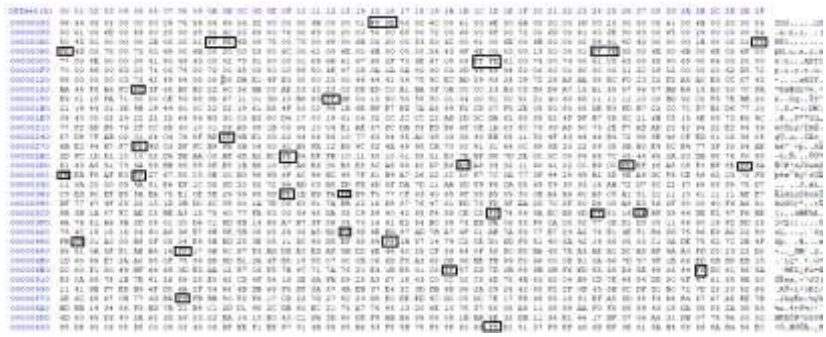
Nama *audio* : Slankkk - Pak Tani.mp3

Ukuran audio : 8.657 KB



Gambar 1. *Audio* dalam Hexadesimal Menggunakan HxD**.**

diatas makan nilai bit audio yang bernilai 254 dan 255 dapat diketahui dan lebih banyak dari jumlah biner yang akan disisipkan berikut gambar hexsadesimal dari *audio* yang akan disisipkan biner *cipherteks*



Gambar 2. Hexsadesimal yang akan disisipkan biner cipher**.**

3. Penyisipan Pesan Kedalam Audio dengan EAS

Proses penyisipan atau sering disebut *embeding* adalah menggabunkan dua obek yang berbeda dalam satu *cover* hal ini biasanya menyisipkan pesan-pesan rahasia yang tidak boleh orang lain tau selain sipenerima dengan menggunakan alghoritma *Enhanched audio steganografi (EAS)*, EAS merupakan metode turunan dari LSB dimana embeding dilakukan pada *byte* yang bernilai 254 dan 255. EAS mempunyai keunggulan dalam *embeding* dibanding LSB karena EAS hanya menggunakan selective byte saja maka media penampung yang digunakan akan mengalami kerusakan kecil saja ketika data teks yang disisipkan kedalam audio dengan ukuran audio 8.657 kb. Dari hasil gambar 3.5 diatas maka jumlah nilai bit 254 dan 255 dalam bentuk hexsa dapat ditemukan dan lebih besar dari jumlah biner yang akan disisipkan sehingga lebih mudah untuk melakukan *embedding* berikut adalah hasil dari nilai audio yang telah ditentukan. Agar sebuah *audio* dapat dibedakan apakah didalam *audio* tersebut tersisipkan pesan atau tidak, maka pada penelitian ini *audio* yang disisipkan pesan akan diberi penanda. Nilai penanda yang digunakan adalah 1, nilai penanda akan diletakkan menjadi nilai *audio* terakhir (nilai akhir *audio* dimodifikasi menjadi 1).

Tabel 3. Pembentukan Hexadesimal kedalam biner

Hex	Des	Biner	Hex	Des	Biner	Hex	Des	Biner
FF	255	11111111	**FF**	255	11111111	**FF**	255	11111111
FE	254	11111110	**FF**	255	11111111	**FE**	254	11111110
FF	255	11111111	**FF**			8D		
FE	254	11111110	**FF**	255	11111111	C6		
FF	255	11111111	**FF**	255	11111111	**FE**	254	11111110
FE	254	11111110	**FF**			76		
FF	255	11111111	**FE**	255	11111111	36		
FE	254	11111110	**FF**	255	11111111	FD		
FF	255	11111111	**FE**			**FE**	254	11111110
FE	254	11111110	**FE**	254	11111110	8D		
FF	255	11111111	FD	255	11111111	68		
FF	255	11111111	**FE**	254	11111110	**FE**	254	11111110
FE	254	11111110	2E	254	11111110	CE		
FE	254	11111110	D2	254	11111110	7D	255	11111111
FE	254	11111110	**FF**	255	11111110	**FF**	254	11111110
FE	254	11111110	B9	254	11111111	A8 E1		

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

Tabel 4. Biner *Ciphertext* yanga akan disisipkan

Hexadesimal	Biner	Desiamal	Krakter
1	00110010	50	2
9	10101101	173	-
17	10101010	170	ª
25	10111110	190	¾
33	00110101	53	5
41	10011000	152	˜
49	00011011	27	
57	01000011	67	C

Pesan rahasia diubah kedalam betuk biner lalu kedalam bentuk *hexadecimal* agar sesuai dengan *byte* yang akan dimasukkan kedalam audio, penyisipan biner kedalam *byte* audio yang hanya bernilai 254 dan 255. Biner atau data teks yang akan dimasukkan kedalam *byte* audio setelah *byte* audio diubah kedalam biner dan bit yang akan disisipkan dari bit teks adalah bit terakhir dari biner audio, hal ini bertujuan untuk menjaga keamanan audio dari kerusakan yang parah, berikut adalah pengubahan ciperteks kedalam biner 8 bit dan jumlah krakter yang digunakan utuk penyisipan adalah bit yang bernilai 254 dan 255.

5. Proses Deskripsi

Tahap 1 menentukan Lo dan Ro dari pembagian biner diatas

Tahap 1 L(0) = 11000010110110000001100110101100

R(0) = 01111101010101011011010101001100

Tahap 2 = R(0) + K(0) MOD 2^{32}

R[0] = 2102768972

K[0] = 4130252466

Hasil = 6233021438 Mod 2^{32}

Hasil Mod = 1938054142

Biner = 01110011100001000101101111111110

Tahap 3 Pecah Hasil Tahap 2 menjadi 8 Kelompok

Biner	Desimal	S-B0x	Desimal	Biner
0111	7	S-Box(0)	14	1110
0011	3	S-Box(1)	12	1100
1000	8	S-Box(2)	14	1110
0100	4	S-Box(3)	0	0000
0101	5	S-Box(4)	15	1111
1011	11	S-Box(5)	5	0101
1111	15	S-Box(6)	12	1100
1110	14	S-Box(7)	8	1000

Tahap 4 Gabungkan Hasil Tahap 3, kemudian dilakukan Rotate Shif Left 11 kali

gabungan : 11101100111000001111010111001000

Rotate 11 (RSL R[0]) 00000111101011100100011101100111

R(1)

Tahap 5 = RSL R[0] XOR L[0]--> R[0] adalah setelah digeser / hasil RSL <-- hasil

R[0]= 00000111101011100100011101100111 RLS

L[0]= 11000010110110000001100110101100 □ <-- L(0) awal

R[1]= 11000101011101100101111011001011

Tahap 6 L[1] = R(0) sebelum proses

L[1] = 01111101010101011011010101001100

HASIL PUTARAN KE-0

L[1]

= 01111101010101011011010101001100

R[1]

= 11000101011101100101111011001011

Putaran i=31

Tahap 1 Ambil L (i-1) dan R(i-1)

L[31] = 11101001100111110010100010100111

R[31] = 10000010001100101010001011001010

Tahap 2 = R[31] + K[0] MOD 2^{32}

R[31] = 2184356554

K[0] = 2249623114

Hasil = 4433979668 Mod 2^{32}

= 139012372

Biner = 00001000010010010010100100010100

Tahap 3 Pecah Hasil Tahap 2 menjadi 8 Kelompok

Biner	Desimal	S-Box	Desimal	Biner
0000	0	S-Box(0)	4	0100
1000	8	S-Box(1)	2	0010
0100	4	S-Box(2)	10	1010
1001	9	S-Box(3)	4	0100
0010	2	S-Box(4)	7	0111
1001	9	S-Box(5)	6	0110
0001	1	S-Box(6)	11	1011
0100	4	S-Box(7)	5	0101

Tahap 4 Hasil kemudian dilakukan Rotate Shif Left 11 kali

gabungan : 01000010101001000111011010110101

Rotate 11 00100011101101011010101000010101

Tahap 5 R[32] = R[31] sebelum proses

R[32]= 10000010001100101010001011001010

Tahap 6 L[32] = R[31] XOR L[31]

R[31] = 00100011101101011010101000010101

L[31] = 11101001100111110010100010100111

L[32] = 11001010001010101000001010110010

Tahap 7 Lakukan pembalikkan string biner L[32] dan R[32]

L[32] = 01001101010000010101010001010011

R[32] = 01010011010001010100110001000001

Tahap 8 Gabungkan nilai R[32] dan L[32]

Mulai dari R(1),R(2)...,R(32), L(1), L(2),...L(32)

Hasil :
0101001101000101010011000100000101001101010000010101010001010011

Tahap 9 Kelompokkan Hasil Penggabungan menjadi 8 bit per kelompok :

Biner	Desimal	Krakter
01010011	83	S
01000101	69	E
01001100	76	L
01000001	65	A
01001101	77	M
01000001	65	A
01010100	84	T
01010011	83	S

Sehingga dari hasil dekskripsi teks dapat dikembalikan seperti semula:
Plainteks: **SELAMATS**

4. KESIMPULAN

Dari hasil penelitian penyisipan pesan kedalam audio dengan algoritma Gosudarsvennyi Standard dan metode EAS, maka dapat diambil kesimpulan dimana proses enkripsi dan dekripsi dapat dilakukan dengan mpada pesan teks dengan melakukan pembangkitan kunci dengan menerpkan metode EAS dapat melakukan penyisipkan pesan teks kedalam audio dengan menyeleksi nilai 254 dan 255 pada audio sebagai penampung pesan teks, namun metode EAS kurang efektif dalam penyisipan pesan kedalam audio karena harus membutuhkan audio yang berukuran besar dan memiliki jumlah nilai audio 254 dan 255 minimal sama dengan jumlah biner cipherteks.

REFERENCES

[1] T. Zebua, “Pengamanan Data Teks Dengan Kombinasi Cipher Block Chainning dan LSB-1,” , Seminar Nasional Inovasi Dan Informasi .979-458-808-3, 2015.
[2] Fresly,Indah,Awang, "Implementasi Kriptografi Pengamanan Data Pada Pesan Teks, Isi File Dokumen, Dan File Dokmen Mengunakan Algoritma AES", Jurnal Informatika Mulawarman, 2015.
[3] Budiman, "Aplikasi Steganografi Pada Audio Dengan metode LSB", Bandung, Unikom, 2010.
[4] Akabid, Tarigan, "Implementasi Kebijakan Apa, Mengapa, Dimana", Jurnal administrasi Public, 2010.
[5] S. Aripin dan M. Syahrizal, “Pengaman File Video Menggunakan Algoritma Merkle Hellman Knapsack,” J. MEDIA Inform. BUDIDARMA, vol. 4, no. April, hal. 461–465, 2020, doi: 10.30865/mib.v4i2.2039.
[6] Deni Mahdiana, "analisa Dan Perancangan sitem Informasi Pegadaian Barang dengan Metodologi Berorientasi Objek", jurnal Telematika MKom 2011, sep.2.
[7] Dony Ariyus, & Rum Andri K.R, “Komunikasi Data” Ed.I, Yogyakarta : Andi, 2008, pp. 447–448.
[8] M. K. Emy Setyaningsih, S.Si., “Kriptografi & Implementasinya Menggunakan Matlab,” Jakarta, 2015, pp. 71–132.
[9] irfananas, putraarya, abdul, "Implementasi Algoritma Vigeneere Cipher Dan GOST Dalam Keaman Data", sinkron, 2018, april.
[10] Oos M.Anwas, "Model Buku teks pembelajaran Berbasis Teknologi Informasi dan komunikasi", balitang Kemendikbud, 2016, ap.17.
[11] A. Pulung Nurtantio, " Pengolahan citra Digital", Steganografi, Yogyakarta, 2017 .
[12] Noviyanti, " Pengembangan Aplikasi Live Chat Dengan Menggunakan Webrtc Sebagai Pemanfaatan Teknologi Informasi Dan Komunikasi Untuk Media Pembelajaran Jarak Jauh (E-Learning)"; 2014.
[13] I B Adisimakrisna Peling, N Putra Sastra, " Enhanced Audio Steganografi dengan Algoritma Advanced Encryption Standard Untuk Pengamanan Data Pada File Audio", Majalah Ilmiah dan Teknologi Elektro, 2018, jan-april.