Prosiding Seminar Nasional Ilmu Komputer dan Teknologi Informasi
e-ISSN 2540-7902 dan p-ISSN 2541-366X
Vol. 3, No. 2, Desember 2018

Implementasi Steganografi Pesan Text ke Dalam Audio Dengan Metode Spread Spectrum

1st* Muhammad Maulana Assyahid Syaifullah
Teknologi Informasi
Politeknik Negeri Samarinda
Samarinda, Indonesia
assyahidm97@gmail.com

2nd Rihartanto
Teknologi Informasi
Politeknik Negeri Samarinda
Samarinda, Indonesia
rihart.c@gmail.com

3rd Didi Susilo Budi Utomo
Teknologi Informasi
Politeknik Negeri Samarinda
Samarinda, Indonesia

***Abstract*— Proses Steganografi memerlukan dua media yang berbeda, yaitu media pesan dan media cover. Pesan berupa teks dan cover berupa audio. Teks disisipkan menggunakan metode Spread Spectrum dan Least Significant Bit (LSB). Metode Spread Spectrum digunakan untuk memodulasi pesan yang akan disisipkan dan metode Least Significant Bit untuk menyisipkan pesan. Hasil penyisipan berupa stego-audio diukur nilai MSE dan PSNR nya. Apabila nilai MSE dari stego-audio semakin rendah maka nilai PSNR dari stego-audio tersebut akan semakin besar. Tujuan penelitian ini menghasilkan stego-audio dengan nilai PSNR minimal 40 dB. Hasil tiga percobaan didapatkan nilai rata-rata MSE sebesar 2,2692e-06 dan nilai rata-rata PSNR sebesar 111,9842 dB maka hasil stego-audio mempunyai kualitas yang cukup baik.**

Kata kunci— lsb, mse, psnr, spread spectrum, steganografi.

I. PENDAHULUAN

Steganografi adalah seni dan ilmu menulis atau menyembunyikan pesan dengan suatu cara tertentu sehingga selain si pengirim dan si penerima, tidak ada orang lain yang mengetahui atau menyadari adanya suatu pesan rahasia. Kata steganografi berasal dari bahasa Yunani steganos, yang artinya tersembunyi atau terselubung, dan graphein artinya menulis.

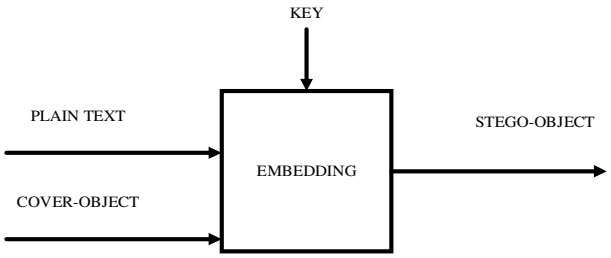
Pada umumnya, pesan steganografi muncul dalam bentuk lain seperti gambar, artikel, daftar belanjaan, atau bentuk-bentuk lainnya yang sering disebut sebagai cover. Cover ini merupakan bentuk atau tulisan yang menyelubungi atau menutupi pesan asli. Contohnya, suatu pesan bisa disembunyikan dengan menggunakan tinta yang tidak terlihat diantara garis-garis yang kelihatan [1].

Ada banyak metode yang dapat digunakan dalam steganografi, diantaranya adalah Least Significant Bit (LSB), Spread Spectrum, Algorithms and Tranformation, dan Redundant Pattern Encoding. Dalam penelitian ini steganografi dilakukan dengan menggunakan metode Spread Spectrum. Metode spread spectrum diilhami dari sistem komunikasi spread spectrum, yang melakukan penyebaran spektrum sinyal informasi ke dalam bandwidth yang jauh lebih lebar dibanding bandwidth sinyal informasi pada umumnya. Tujuan dari penyebaran sinyal ini adalah untuk mengurangi kemungkinan terjadinya penyadapan karena data yang dikirimkan bersifat acak dan memiliki kecenderungan sifat seperti noise.

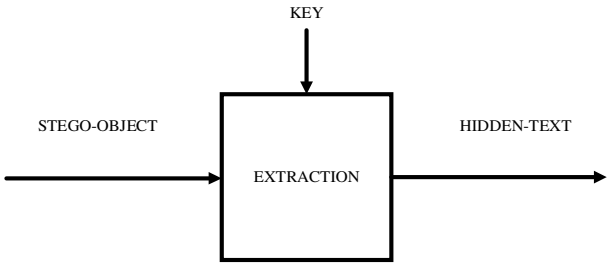
Media yang dijadikan sebagai cover dalam penelitian ini adalah data audio dengan format WAV, sementara pesan yang sisipkan adalah data dalam bentuk teks. Data teks akan disebar ke dalam data audio untuk menghasilkan stego-audio yang diharapkan memiliki kualitas audio yang tidak jauh berbeda dibandingkan dengan audio aslinya. Kualitas stego-audio diukur menggunakan PNSR, dimana kualitas audio masih dianggap baik jika nilai PNSR lebih besar dari 60dB.

II. METODOLOGI DAN BAHAN PENELITIAN

Secara umum, terdapat dua proses di dalam steganografi, yaitu proses embedding untuk menyembunyikan atau menyisipkan pesan dan proses extraction untuk mengambil pesan tersembunyi. Kedua proses tersebut ditunjukkan pada Gambar 1 dan Gambar 2.



Gambar 1. Proses Penyisipan



Gambar 2. Proses Ekstraksi

Proses embedding melibatkan pesan (plaintext) dan cover-object serta sebuah kunci atau password (key) untuk menghasilkan stego-object. Sementara proses ekstraksi

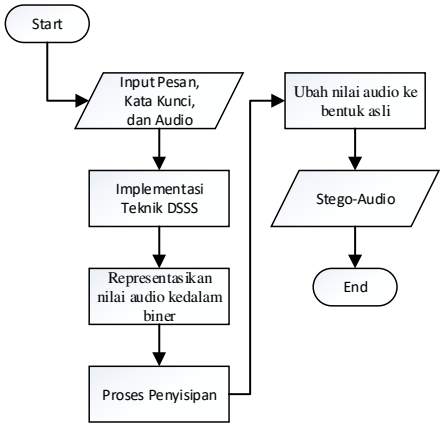
*Corresponding Author

melibatkan stego-object dan key untuk memperoleh pesan tersembunyi.

Kunci yang digunakan pada proses penyisipan dan ekstraksi adalah kunci simetris, dalam arti kunci yang digunakan pada proses penyisipan sama persis dengan kunci untuk melakukan ekstraksi. Proses ekstraksi pesan dari stego-object hanya bertujuan untuk mengambil pesan tersembunyi saja, dan mengabaikan kondisi objek yang digunakan sebagai cover. Sehingga proses ekstraksi tidak berupaya untuk mengembalikan cover yang digunakan menjadi cover aslinya kembali. Hal ini diantaranya dikarenakan pada proses penyisipan pesan tidak dilakukan pencatatan kondisi awal dari cover-object yang digunakan untuk penyembunyian pesan [2].

A. Proses Penyisipan

Metode *Spread Spectrum* merupakan metode untuk menyisipkan data dengan cara menyebarkan data rahasia sepanjang sinyal *audio cover*. Data rahasia yang disisipkan pada sinyal *audio cover* disebar menggunakan *Direct Sequence Spread Spectrum* (DSSS) [3]. Alur proses penyisipan pesan menggunakan metode *Spread Spectrum* ditunjukkan pada Gambar 3.



Gambar 3. Proses Penyisipan Pesan

Penyebaran pesan ke dalam data audio menggunakan teknik teknik *Direct-Sequence Spread Spectrum* (DSSS) yang merupakan salah satu teknik dalam metode *Spread Spectrum* yang menggunakan modulasi Pseudo Noise Sequence (PN Sequence) [4].

Sebagai contoh misalkan terdapat pesan "coba" yang akan disisipkan ke cover-object. Langkah pertama yaitu mengubah pesan tersebut ke dalam bentuk biner sehingga diperoleh urutan bit 01100011 01101111 01100010 01100001. Selanjutnya pesan dalam bentuk biner ini disebar menggunakan besaran skalar empat sehingga dihasilkan segmen baru berikut:

00001111	11110000	00000000	11111111
00001111	11110000	11111111	11111111
00001111	11111111	00000000	11110000
00001111	11110000	00000000	00001111

Pseudonoise dibangkitkan dari kata kunci yang diberikan. Dalam contoh ini kata kunci yang digunakan adalah "lana". Operasi XOR dilakukan pada kata kunci ini untuk memperoleh nilai yang nantinya digunakan sebagai bibit (seed) pembangkit bilangan acak.

l	01101100	
a	01100001	(*XOR*)
	00001101	
n	01101110	(*XOR*)
	01100011	
a	01100001	(*XOR*)
	00000010	⟶ 2 (Desimal)

Nilai 2 yang diperoleh terebut selanjutnya digunakan sebagai seedm random. Metode pembangikatn bilangan acak yang digunakan disini adalah *Linear Congruential Generator* (LCG) yang menggunakan rumus:

$$X_{n+1} = (aX_n + c)\ mod\ m \quad \text{.....(1)}$$

X_{n+1} adalah bilangan acak ke-(n+1), X_n = bilangan acak ke-n, a adalah konstanta pengali, dan c adalah konstanta kenaikan, serta m adalah konstanta modulus.

Misalkan a = 21, c = 3, dan m = 16, maka perhitungannya adalah sebagai berikut.

$X_1 = (21 \times 2 + 3) \bmod 16 = 13$
$X_2 = (21 \times 13 + 3) \bmod 16 = 4$
$X_3 = (21 \times 4 + 3) \bmod 16 = 7$
$X_4 = (21 \times 7 + 3) \bmod 16 = 6$
$X_5 = (21 \times 6 + 3) \bmod 16 = 1$
Demikian seterusnya untuk $X_6, X_7, X_8, X_9, \ldots. X_n$

Dari perhitungan tersebut diambil 16 angka random pertama, berturut–turut adalah "13, 4, 7, 6, 1, 8, 11, 10, 5, 12, 15, 14, 9, 0, 3, 2" jika diubah dalam bentuk biner menjadi "00001101 00000100 00000111 00000110 00000001 00001000 00001011 00001010 00000101 00001100 00001111 00001110 00001001 00000000 00000011 00000010"

Untuk mendapatkan hasil modulasi, segmen pesan akan dimodulasikan dengan pseudonoise signal menggunakan fungsi XOR.
Segmen Pesan:

00001111	11110000	00000000	11111111
00001111	11110000	11111111	11111111
00001111	11111111	00000000	11110000
00001111	11110000	00000000	00001111

Pseudonoise Signal:

00001101000001000000011100000110000000010000100000
00101100001010000001010000110000001111000011100000
10010000000000000001100000010

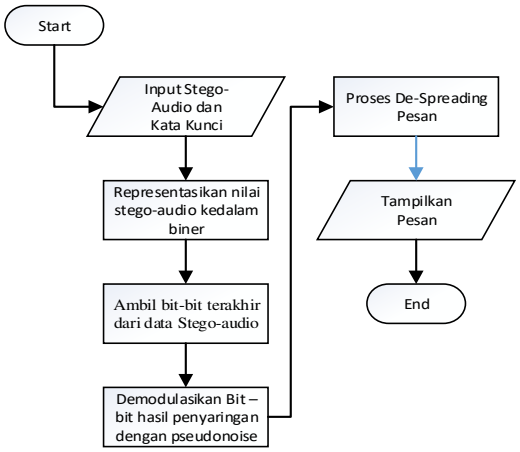
Maka hasil proses modulasi antara segmen pesan dengan pseudonoise signal menggunakan fungsi XOR adalah

00000010	11110100	00000111	11111001
00001110	11111000	11110100	11110101
00001010	11111100	00001111	11111110
00000110	11110000	00000011	00001101

Hasil dari proses modulasi inilah yang akan disisipkan ke bit – bit audio dengan menggunakan teknik *Least Significant Bit* (LSB).

B. Proses Ekstraksi

Tahapan penelitian dalam melakukan proses Ekstraksi atau pengambilan pesan ditunjukkan dalam Gambar 4.



Gambar 4. Proses Ekstraksi Pesan

Pada proses ektraksi prosesnya adalah kebalikan dari proses embedding. Pilih audio yang akan diekstrak, gunakan kata kunci yang sama seperti saat proses embedding yaitu "lana". Langkah awal adalah merepresentasikan nilai audio kedalam biner. Selanjutnya dilakukan proses pengambilan bit–bit terakhir hasil modulasi dari data audio yang telah direpresentasikan dalam bentuk biner. Hasil dari pengambilan bit – bit terakhir dari data audio adalah sebagai berikut:

00000010	11110100	00000111	11111001
00001110	11111000	11110100	11110101
00001010	11111100	00001111	11111110
00000110	11110000	00000011	00001101

Setelah semua bit–bit hasil modulasi diperoleh kemudian dilakukan proses demodulasi dengan *pseudonoise signal* dari kata kunci yang sama pada proses modulasi sehingga memperoleh bit – bit yang berkorelasi. Hasil penyaringan:

00000010	11110100	00000111	11111001
00001110	11111000	11110100	11110101
00001010	11111100	00001111	11111110
00000110	11110000	00000011	00001101

Pseudonoise Signal:

00001101000001000000011100000110000000010000100000
00101100001010000001010000110000001111000011100000
10010000000000000001100000010

Hasil Demodulasi:

00001111	11110000	00000000	11111111
00001111	11110000	11111111	11111111
00001111	11111111	00000000	11110000
00001111	11110000	00000000	00001111

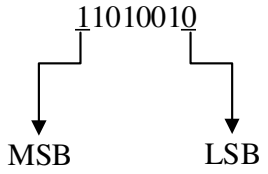
Proses berikutnya yaitu membagi empat hasil demodulasi, yang berguna untuk menyusutkan hasil demodulasi menjadi isi pesan yang sebenarnya. Proses penyusutan (*de-spreading*) segmen tersebut menjadi:

01100011 01101111 01100010 01100001

Hasil akhir "01100011 01101111 01100010 01100001" merupakan segmen pesan yang sama ketika disembunyikan pada proses embedding. Hasil tersebut kemudian diubah ke bentuk karakter akan menjadi "coba".

C. Metode Least Significant Bit (LSB)

Metode *Steganografi* yang paling umum pada format suara adalah modifikasi *Least Significant Bit*. Metode ini banyak digunakan karena komputasinya tidak terlalu kompleks dan pesan yang disembunyikan cukup aman. Strategi penyembunyian data pesan yang digunakan adalah mengganti bit-bit paling rendah dalam data audio dengan bit-bit pesan. [5].



Gambar 5. MSB dan LSB

Gambar 5 menunjukkan posisi MSB dan LSB dalam satu byte. Bit pertama adalah bit yang paling signifikan sementara bit terakhir adalah bit yang paling tidak signifikan. Sebagai contoh jika terdapat pesan 8 bit yang akan disisipkan pada data audio, maka diperlukan 8 byte data audio untuk menampung pesan

tersebut. Jika pesan yang disisipkan adalah 0100 0001 dan data audio tempat penyisipan adalah

00110011 01011001 10100010 01101110
10100011 10110101 00100110 00010101

Maka hasil pentisipan pada bit LSB adalah:

00110010 01011001 10100010 01101110
10100010 10110100 00100110 00010101

Secara teoritis, ukuran ukuran file cover tidak akan mengalami perubahan yang signifikan, sehingga perubahan tersebut akan sulit terdeteksi oleh indra manusia.

D. Peak Signal to Noise Ratio (PSNR)

Perhitungan PNSR digunakan untuk mengetahui perbandingan kualitas audio sebelum dan sesudah disisipkan pesan. Untuk menentukan PSNR, terlebih dahulu harus mencari nilai rata-rata kuadrat dari error (*Mean Square Error*-MSE). MSE adalah nilai rata-rata kuadrat error antara audio asli (*cover-object*) dengan audio hasil penyisipan (*stego-audio)*

Perhitungan MSE dan PSNR menggunakan persamaan sebagai berikut:

$$MSE = \frac{\sum_1^n (x-y)^2}{M+N} \quad \text{.....(2)}$$

$$PSNR = 10 * log_{10} \frac{R^2}{MSE} \quad \text{.....(3)}$$

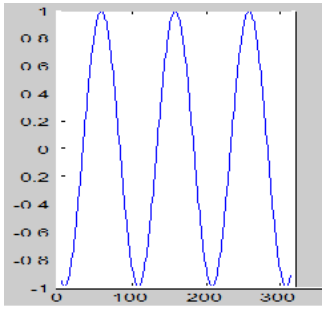
Dimana : R^2 = Nilai maksimum sinyal, x = Sinyal Asli, y = Sinyal Stegano, M = Baris input Sinyal dan N = Kolom Input sinyal

PSNR sering dinyatakan dalam skala logaritmik, dalam *decibel* (dB). Apabila nilai PSNR dibawah 30 dB mengindikasikan kualitas *stego-audio* yang relatif rendah, sedangkan kualitas *stego-audio* yang tinggi berada pada nilai 40 dB dan diatasnya [6].

E. Bahan Penelitian

Pada penelitian ini, data yang digunakan adalah audio sebagai media cover dan text sebagai pesan yang disembunyikan. Sehingga data awal harus direpresentasikan menjadi bentuk data biner. Proses representasi dijelaskan seperti berikut

A. Representasi Tipe Data Audio Ke Biner



Gambar 6. Contoh Sinyal Audio

Pada sinyal audio diatas memiliki keterangan sebagai berikut

Panjang Data Audio : 300 x 1

Tipe Data Audio : Double

Data Audio : [-0,234344; -0,433887 ; ……..…; 0,632957]

Pada data audio di atas diambil beberapa contoh data audio untuk proses representasi tipe data double ke binary. Dalam data audio terdapat nilai data audio < 0, sebelum melakukan proses representasi tipe data ke binary terlebih dulu melakukan proses normalisasi dengan menggunakan rumus berikut:

$$Normalisasi = Xa - \min(Xa) \quad \text{.....(4)}$$

Dimana : Xa = Nilai Data Audio, min(Xa) = Nilai Terkecil Data Audio.

Setelah melakukan proses normalisasi maka diperoleh hasil data audio sebagai berikut

Data Audio : [0 ; -0,668287 ; ……; 0,398557]

Pada data audio diatas penulis mengambil satu data audio sebagai contoh proses representasi tipe data audio ke binary yang dijelaskan seperti berikut

2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0
128	64	32	16	8	4	2	1

1 => 1	16 => 0.0625
2 => 0.5	32 => 0.03125
4 => 0.25	64 => 0.015625
8 => 0.125	128 => 0.0078125

Dalam merepresentasikan tipe data pada audio dapat dilakukan dengan cara mengurangkan nilai variabel audio dengan data decimal pada bit binary seperti contoh berikut:

0.398557 => 0.398557
0.25 => 4
0.148557
0.125 => 8
0.023557
0.015625 => 64
0.007932
0.0078125 => 128
0.0001195

[1100 1100]

B. Representasi Tipe Data Pesan Text Ke Biner

Karakter pesan text direpresentasikan dalam bentuk biner dengan pengkodean ASCII sehingga menghasilkan bilangan binary seperti table berikut :

TABEL I. REPRESENTASI KARAKTER TEXT KE BINARY

Karakter	ASCII	Biner	Karakter	ASCII	Biner
A	65	01000001	a	97	01100001
B	66	01000010	b	98	01100010
C	67	01000011	c	99	01100011
D	68	01000100	d	100	01100100
E	69	01000101	e	101	01100101
F	70	01000110	f	102	01100110
G	71	01000111	g	103	01100111
H	72	01001000	h	104	01101000
I	73	01001001	i	105	01101001
J	74	01001010	j	106	01101010
K	75	01001011	k	107	01101011
L	76	01001100	l	108	01101100
M	77	01001101	m	109	01101101
N	78	01001110	n	110	01101110
O	79	01001111	o	111	01101111
P	80	01010000	p	112	01110000
Q	81	01010001	q	113	01110001
R	82	01010010	r	114	01110010
S	83	01010011	s	115	01110011
T	84	01010100	t	116	01110100
U	85	01010101	u	117	01110101
V	86	01010110	v	118	01110110
W	87	01010111	w	119	01110111
X	88	01011000	x	120	01111000
Y	89	01011001	y	121	01111001
Z	90	01011010	z	122	01111010

III. HASIL DAN PEMBAHASAN

A. Data Penelitian

Dalam proses Steganografi menggunakan audio sebagai media cover dan text sebagai media pesan kemudian penulis melakukan proses konversi tipe data pada media cover dan media pesan bisa melakukan proses metode *Spread Spectrum*. Berikut proses konversi tipe data

File Data Audio (Cover)

Nama File Audio : Data.wav
Ukuran File Audio : 436 kb
Panjang Data : 111.201

Data Cover-Audio 1 sampai 10									
0	-0,0000	0	-0,0000	0	-0.0000	-0.0000	0.0002	0.0069	0.0077
Data Cover-Audio 11 sampai 20									
-0.0151	-0.0355	-0.0194	0.0148	0.0044	-0.0464	-0.0497	0.0256	0.1299	0.1982

Gambar 7. Nilai Data Cover-Audio ke 1 sampai 20

Data *cover-audio* yang telah diperoleh kemudian dinormalisasi dan direpresentasikan dalam bentuk biner 16 bit. Data *cover-audio* hasil normalisasi yang direpresentasikan dalam bentuk biner ditunjukkan pada Tabel III.

TABEL II. REPRESENTASI DATA COVER-AUDIO KE-1 SAMPAI 20 DALAM DESIMAL DAN BINER

No	Data Audio	Biner
1	0,9944	0001001011111110
2	0,9944	1110001011111110
3	0,9944	0001001011111110
4	0,9944	1110001011111110
5	0,9944	0001001011111110
6	0,9944	1110001011111110
7	0,9944	1110001011111110
8	0,9946	0000101011111110
9	1,0013	1101010000000001
10	1,0021	1010001000000001
11	0,9793	0101101010111110
12	0,9589	0011110101011110
13	0,9749	1101001100111110
14	1,0092	0111010010000001
15	0,9987	1110101111111110
16	0,9480	1001101010011110
17	0,9447	1101011100011110
18	1,0200	0000100101000001
19	1,1243	1001011111110001
20	1,1926	0110010100011001

File Data Text (Message)

Pesan Text : Jurusan Teknologi Informasi Politeknik Negeri Samarinda
Panjang Pesan: 55 Karakter

Dengan melakukan proses representasi tipe data pesan ke biner maka menghasilkan data pesan biner yang ditunjukkan pada Gambar 8.

J	01001010	o	01101111	P	01010000	e	01100101
u	01110101	g	01100111	o	01101111	r	01110010
r	01110010	i	01101001	l	01101100	i	01101001
u	01110101	Spasi	00100000	i	01101001	Spasi	00100000
s	01110011	I	01001001	t	01110100	S	01010011
a	01100001	n	01101110	e	01100101	a	01100001
n	01101110	f	01100110	k	01101011	m	01101101
Spasi	00100000	o	01101111	n	01101110	a	01100001
T	01010100	r	01110010	i	01101001	r	01110010
e	01100101	m	01101101	k	01101011	i	01101001
k	01101011	a	01100001	Spasi	00100000	n	01101110
n	01101110	s	01110011	N	01001110	d	01100100
o	01101111	i	01101001	e	01100101	a	01100001
l	01101100	Spasi	00100000	g	01100111		

Gambar 8. Data Biner Pesan Text

Hasil penyisipan pesan

Sebelum melakukan penyisipan pesan, langkah pertama adalah mengimplementasikan teknik *Direct Sequence Spread Spectrum* pada data pesan yang akan disisipkan. Pesan disebar dengan besaran skalar pengalinya empat atau disebut juga dengan proses *Spreading*. Setelah pesan disebar, langkah selanjutnya adalah memodulasi pesan yang akan disisipkan dengan bilangan *pseudorandom* (bilangan acak semu) yang dibangkitkan dengan menggunakan algoritma *Linear Congruential Generator* (LCG) dengan kata kunci "jurti". Setelah melakukan modulasi pesan dengan bilangan *pseudorandom*, langkah selanjutnya adalah menyisipkan pesan kedalam audio. Hasil proses penyisipan pesan ke dalam audio dapat dilihat pada Gambar 9.

Data Audio Sebelum Disisipi Pesan			
0001001011111110	1110001011111110	0001001011111110	1110001011111110
0001001011111110	1110001011111110	1110001011111110	0000101011111110
1101010000000001	1010001000000001	0101101010111110	0011110101011110
1101001100111110	0111010010000001	1110101111111110	1001101010011110
1101011100011110	0000100101000001	1001011111110001	0110010100011001
Data Audio Setelah Disisipi Pesan			
0001001011111110	0110001011111110	0001001011111110	0110001011111110
1001001011111110	1110001011111110	0110001011111110	0000101011111110
0101010000000001	0010001000000001	0101101010111110	0011110101011110
0101001100111110	0111010010000001	1110101111111110	0001101010011110
1101011100011110	1000100101000001	1001011111110001	1110010100011001

Gambar 9. Hasil Penyisipan Pesan

Setelah pesan disisipkan maka langkah selanjutnya adalah mengubah audio ke bentuk desimal. Nilai desimal dari audio yang telah disisipi pesan ditunjukkan pada Gambar 10.

Data Cover-Audio 1 sampai 10									
0.9944	0.9943	0.9944	0.9943	0.9944	0.9944	0.9943	0.9946	1.0013	1.0021
Data Cover-Audio 11 sampai 20									
0.9793	0.9589	0.9749	1.0092	0.9987	0.9480	0.9447	1.0201	1.1243	1.1926

Gambar 10. Nilai Data Audio ke-1 sampai 20 yang Telah Disisipi Pesan

Setelah mendapat nilai data audio desimal yang telah disisipi pesan maka data audio tersebut diubah ke bentuk asli sehinga audio yang telah disispi pesan berubah menjadi *stego-audio.* Nilai data *stego-audio* ditunjukkan pada Gambar 11.

Data Stego-Audio 1 sampai 10									
0	-0.0001	0	-0.0001	0.0000	-0.0000	-0.0001	0.0002	0.0069	0.0077
Data Stego-Audio 11 sampai 20									
-0.0151	-0.0355	-0.0195	0.0148	0.0044	-0.0464	-0.0497	0.0257	0.1299	0.1982

Gambar 11. Nilai Stego Audio

Hasil ekstraksi pesan

Pada ektraksi pesan, langkah pertama membaca data dari *stego-audio.* Nilai data dari *stego-audio* ditunjukkan pada Gambar 12.

Data Stego-Audio 1 sampai 10									
0	-0.0001	0	-0.0001	0.0000	-0.0000	-0.0001	0.0002	0.0069	0.0077
Data Stego-Audio 11 sampai 20									
-0.0151	-0.0355	-0.0195	0.0148	0.0044	-0.0464	-0.0497	0.0257	0.1299	0.1982

Gambar 12. Nilai Data Stego-Audio ke-1 sampai 20

Setelah mendapat nilai data dari *stego-audio*, langkah selanjutnya adalah melakukan normalisasi dan merepresentasikan nilai *stego-audio* ke dalam bentuk biner. Hasil representasi data *stego-audio* hasil normalisasi dalam desimal dan biner ditunjukkan pada Tabel III.

TABEL III. REPRESENTASI DATA STEGO-AUDIO KE-1 SAMPAI 20 DALAM DESIMAL DAN BINER

No.	Data Stego-Audio	Biner
1	0.9944	0001001011111110
2	0.9943	0110001011111110
3	0.9944	0001001011111110
4	0.9943	0110001011111110
5	0.9944	1001001011111110
6	0.9944	1110001011111110
7	0.9943	0110001011111110
8	0.9946	0000101011111110
9	1.0013	0101010000000001
10	1.0021	0010001000000001
11	0.9793	0101101010111110
12	0.9589	0011110101011110
13	0.9749	0101001100111110
14	1.0092	0111010010000001
15	0.9987	1110101111111110
16	0.9480	0001101010011110
17	0.9447	1101011100011110
18	1.0201	1000100101000001
19	1.1243	1001011111110001
20	1.1926	1110010100011001

Setelah mendapatkan biner dari *stego-audio*, selanjutnya adalah mencari pesan yang mengandung pesan rahasia. Hasil *stego-audio* yang mengandung pesan rahasia ditunjukkan pada Gambar 13.

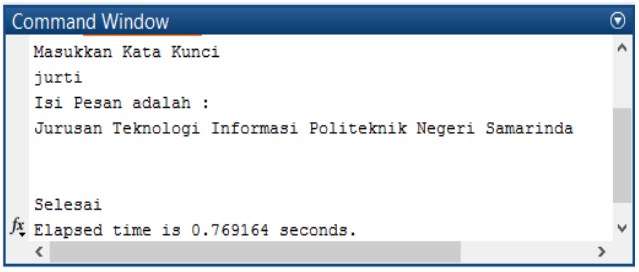
0001001011111110	0110001011111110	0001001011111110	0110001011111110
1001001011111110	1110001011111110	0110001011111110	0000101011111110
0101010000000001	0010001000000001	0101101010111110	0011110101011110
0101001100111110	0111010010000001	1110101111111110	0001101010011110
1101011100011110	1000100101000001	1001011111110001	1110010100011001

Gambar 13. Data Biner Stego-Audio ke-1 sampai 20 yang Mengandung Pesan Rahasia

Setelah mendapatkan pesan rahasia dari *stego-audio* kemudian pesan tersebut dimodulasikan dengan bilangan *pseudorandom* dengan kata kunci yang sama saat proses penyisipan dengan menggunakan fungsi *Exclusive OR (XOR)*. Setelah pesan dimodulasi, kemudian dilakukan proses *de-spreading* pesan dan direpresentasikan menjadi huruf atau karakter dengan menggunakan pengkodean *ASCII*. Hasil dari proses *de-spreading* ditunjukkan pada Gambar 14.

01001010	01101111	01010000	01100101
01110101	01100111	01101111	01110010
01110010	01101001	01101100	01101001
01110101	00100000	01101001	00100000
01110011	01001001	01110100	01010011
01100001	01101110	01100101	01100001
01101110	01100110	01101011	01101101
00100000	01101111	01101110	01100001
01010100	01110010	01101001	01110010
01100101	01101101	01101011	01101001
01101011	01100001	00100000	01101110
01101110	01110011	01001110	01100100
01101111	01101001	01100101	01100001
01101100	00100000	01100111	

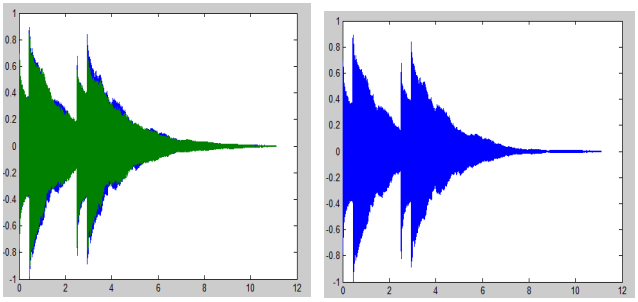
Gambar 14. Hasil De-Spreading Pesan



Gambar 15. Hasil Ekstraksi Pesan

Uji Performansi

Pengukuran Performansi digunakan untuk mengetahui kualitas audio sebelum dan sesudah disisipkan pesan dengan menganalisa hasil sinyal audio sebelum dan sesudah disisipkan pesan dan menentukan nilai PSNR pada audio.



Gambar 16. Sinyal Audio Sebelum dan Sesudah Disisip Pesan

TABEL IV. HASIL PENGUJIAN MSE DAN PSNR

Stego-Audio	Pengujian Performansi		
	Pesan	Nilai *MSE*	Nilai *PSNR*
tes1.wav	Jurusan Teknologi Informasi Politeknik Negeri Samarinda	2,7378e-06	110,2535 dB
tes2.wav	Politeknik Negeri Samarinda, Kaltim	2,1676e-06	112,2821 dB
tes3.wav	Politeknik Negeri Samarinda	1,9021e-06	113,4169 dB

Berdasarkan hasil pengujian pada Tabel IV, maka dapat dilihat bahwa semakin rendah nilai *MSE* maka semakin baik, dan semakin besar nilai *PSNR* maka kualitas *stego-audio* juga semakin baik.

IV. KESIMPULAN

Berdasarkan percobaan yang telah dilakukan maka dapat disimpulkan sebagai berikut:

1. Proses penyisipan pesan menggunakan metode *Spread Spectrum* menjadikan pesan yang disisipkan menjadi lebar dan acak.
2. Untuk bisa mengekstraksi pesan hasil penyisipan, maka kata kunci yang digunakan untuk membangkitkan bilangan *Pseudorandom* untuk proses ekstraksi harus sama seperti kata kunci yang digunakan untuk membangkitkan bilangan *Pseudorandom* di proses penyisipan.
3. Hasil pengujian performasi *MSE* dan *PSNR* menunjukkan bahwa dengan data audio yang memiliki durasi 2 detik dan 3 pesan yang memiliki jumlah karakter yang berbeda, dengan pengujian sebanyak 3 kali diperoleh nilai rata-rata *Mean Squared Error* (*MSE*) sebesar 2,2692e-06 dengan nilai rata-rata *Peak Signal to-Noise Ratio* (*PSNR*) sebesar 111,9842 dB.

DAFTAR PUSTAKA

[1] Y. Aditya, A. Pratama and A. Nurlifa, "Studi Pustaka Untuk Steganografi Dengan Beberapa Metode," *Seminar Nasional Aplikasi Teknologi Informasi 2010 (SNATI 2010), ISSN: 1907-5022,* pp. 32-35, 2010.

[2] Lovebbi and D. Z. Sudirman, "Rancang Bangun Aplikasi Steganografi dengan Metode Least Significant Bit di Audio pada Sistem Operasi Android," *ULTIMATICS, No. 1, ISSN 2085-4552,* vol. IV, pp. 7-16, 2012.

[3] R. A. Saragih, "Metode Parity Coding Versus Metode Spread Spectrum Pada Audio Steganography," *Seminar Nasional Aplikasi Teknologi Informasi (SNATI), ISSN: 1907-5022,* pp. 71-76, 2006.

[4] R. M. Nugraha, "Impementation of Direct Sequence Spread Spectrum Steganography on Audio Data," *International Conference on Electrical Engineering and Informatics, 978-1-4577-0752-0,* 2011.

[5] J. V. Purba, M. Situmorang and D. Arisandi, "Implementasi Steganografi Pesan Text Ke Dalam File Sound (.Wav) Dengan Modifikasi Jarak Byte Pada Algoritma Least Significant Bit (LSB)," *Jurnal Dunia Teknologi Informasi Vol.1, No.1,* vol. I, pp. 50-55, 2012.

[6] E. R. Djuwitaningrum and M. Apriyani, "Teknik Steganografi Pesan Teks Menggunakan Metode Least Significant Bit dan Algoritma Linear Congruential Generator," *Jurnal Informatika, ISSN: 2086-9398 Nomor 2,* vol. IV, pp. 79-85, 2016.