Teknik Steganografi Pesan Teks Menggunakan Metode Least Significant Bit dan Algoritma Linear Congruential Generator *(Text Message Steganography Using Least Significant Bit Method and Linear Congruential Generator Algorithm)*

Endang Ratnawati Djuwitaningrum[1], Melisa Apriyani[2]

[1,2]*Program Studi Informatika – Institut Teknologi Indonesia*
Jl. Raya Puspiptek, Serpong, Tangerang Selatan
[1]endangrd@iti.ac.id
[2]melisa.apriyani@ymail.com

Abstrak– Makalah ini merepresentasikan suatu teknik steganografi untuk menyisipkan pesan rahasia berupa teks ke dalam citra warna 24 bit. Metode *Least Significant Bit* (LSB) digunakan untuk menyisipkan pesan ke dalam media penyisipan citra warna 24 bit (*cover image*) pada setiap 2 bit yang paling signifikan dari setiap warna citra (*Red, Green*, dan *Blue*), sehingga setiap *pixel* citra warna dapat menampung 6 bit pesan teks. Dalam menentukan letak *pixel* citra warna yang akan disisipi pesan, digunakan algoritma pembangkit bilangan acak semu, yaitu *Linear Congruential Generator* (LCG). Hasil eksperimen menunjukkan perhitungan *Peak Signal to Noise ratio* (PSNR) untuk setiap citra yang disisipkan dengan ukuran pesan maksimal (*stego image*), menghasilkan nilai diatas 40 dB.

Kata kunci: *cover image,* LCG, LSB, pesan teks, PSNR, steganografi, *stego image*

Abstract– Represented at this paper is a steganography technique to insert a secret text message into 24-bit color images. Least Significant Bit (LSB) method is used to insert a text message into 24 bit color image media (cover image) at each two most significant bit of each primary color (Red, Green, and Blue). So each pixel at color image media can accommodates 6 bit of text message. To determine the location of the color image pixel where the text message inserted, Linear Congruential Generator (LCG) which is pseudorandom number generator algorithm was used. The experimental results shows the Peak Signal to Noise Ratio (PSNR) for each image pasted with a maximum size message (stego image), produces a value above 40 dB.

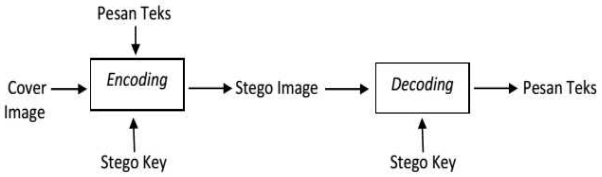
Keywords: cover image, LCG, LSB, PSNR, steganography, stego image, text messages

I. PENDAHULUAN

Saat ini pertukaran data melalui media internet sudah mudah dilakukan. Pertukaran data melalui internet harus juga diikuti dengan aspek keamanan, agar tidak disalahgunakan oleh pihak lain. Salah satu cara pengamanan tersebut adalah menggunakan steganografi. Steganografi merupakan teknik menyembunyikan pesan di dalam suatu media penyisipan pesan atau *cover image*, sehingga keberadaan pesan rahasia yang disisipkan tidak dapat dilihat secara langsung. Steganografi berasal dari dua kata Yunani: *steganos* berarti tersembunyi dan *graphein* berarti menulis.

Salah satu *cover image* yang dapat digunakan untuk menyembunyikan pesan adalah citra digital warna 24 bit. Setiap *pixel* pada citra warna 24 bit memiliki warna yang merupakan kombinasi dari tiga warna dasar *Red, Green, Blue* (RGB*)*. Sedangkan satu *pixel* citra warna 24 bit diwakili oleh tiga *byte,* dimana masing-masing *byte* merepresentasikan warna *Red, Green* dan *Blue*. Penyisipan pesan ke dalam *cover image* dinamakan *encoding*, sedangkan ekstraksi pesan dari *stego image* dinamakan *decoding.* Kedua proses memerlukan kunci rahasia (*stego key*), agar hanya pihak yang mempunyai kunci rahasia saja yang dapat melakukan penyisipan dan ekstraksi pesan. Proses *encoding* dan *decoding*

diberikan dalam bentuk diagram pada gambar 1 berikut



Gambar 1. Diagram penyisipan dan ekstraksi pesan

Metode *Least Signification Bit (LSB)* merupakan metode yang cukup sederhana dalam melakukan proses steganografi. Selain itu, proses penyisipan dan ekstraksi dari metode ini juga relatif cukup cepat. Metode LSB menyisipkan pesan ke dalam *cover image* pada bit yang paling kurang berarti. Untuk LSB 1 bit, bit yang disisipi adalah bit ke-8 untuk setiap *byte*, perubahan nilai desimal dari satu *byte* menjadi satu nilai lebih tinggi, atau satu nilai lebih rendah, atau sama dari nilai desimal dari satu *byte* sebelum terjadi penyisipan. Sedangkan untuk metode LSB 2 bit, bit yang disisipi adalah bit ke-7 dan bit ke-8 untuk setiap *byte*. Sehingga perubahan nilai pada 2 bit terakhir berkisar antara nol sampai dengan tiga dari nilai *byte* sebelum terjadi penyisipan. Untuk menyisipkan pesan ke dalam *pixel* yang ditentukan secara acak, dapat digunakan algoritma pembangkit bilangan acak semu, salah satunya *Linear Congruential Generator* (LCG).

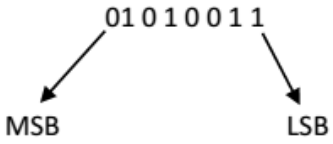
Sudah ada penelitian yang membahas teknik steganografi menggunakan metode LSB. Peneliti [1] dalam eksperimennya, mengimplementasikan penggunaan steganografi untuk menyisipkan pesan pada gambar dengan metode LSB 1 bit. Letak *pixel* yang akan disisipi pesan ditentukan menggunakan algoritma LCG. Peneliti [2] menerapkan metode LSB dalam merancang perangkat keras steganografi. Rancangannya diimplementasikan menggunakan prosesor ARM7TDMI dan GSM 900. Peneliti [3] menggabungkan kriptografi dengan Steganografi. Pesan yang akan disisipkan terlebih dahulu dienkripsi. Selanjutnya pesan yang sudah terenkripsi disisipkan ke dalam *cover image* menggunakan teknik LSB. Peneliti [4] dalam eksperimennya menerapkan teknik steganografi menggunakan metode LSB. Pesan yang disembunyikan dan media penyisipan, keduanya berupa citra warna 24 bit. Makalah ini menyajikan suatu teknik steganografi untuk menyisipkan pesan berupa teks pada media penyisipan berupa citra warna 24 bit.

II. METODE

Metode yang digunakan untuk penyisipan, menggunakan metode LSB 2 bit. Untuk menyebar isi pesan teks pada citra berwarna, digunakan algoritma LCG. Algoritma ini memberikan sebuah nilai yang dapat menunjukkan letak *pixel* yang akan disisipi pesan. Analisis hasil dilakukan untuk melihat keberhasilan dalam proses ekstraksi pesan, dan melihat kualitas *stego image* berdasarkan nilai PSNR.

1. Least Significant Bit (LSB)

Pada susunan bit di dalam sebuah *byte*, ada bit yang paling berarti *most significant bit* atau MSB dan bit yang paling kurang berarti *least significant bit* atau LSB. Gambar 2 menjelaskan posisi MSB dan LSB dalam susunan bilangan biner pada 1 *byte* atau 8 bit.



Gambar 2. Posisi MSB dan LSB pada bilangan biner 8 bit

Sebuah citra merupakan kumpulan dari titik-titik yang disebut *pixel*. Pada citra warna 24 bit, setiap *pixel* berukuran 3 *byte* dimana setiap *byte* mewakili warna dari setiap komponen *Red, Green*, dan *Blue*. Misalkan terdapat 2 *pixel*, dimana nilai intensitas setiap warna pada setiap *pixel* setelah dikonversikan kedalam biner memberikan nilai biner sebagai berikut :

(00100111 11101001 11001000)

(00100111 11001000 11101001)

Untuk menyisipkan sebuah karakter “C” dengan bilangan biner 01000011 (kode ASCII 67) kedalam 2 *pixel* citra warna tersebut, setiap 2 bit dari pesan yang dimulai dari MSB disisipkan kedalam 2 bit LSB dari setiap *byte* citra warna. Hasil penyisipannya memberikan nilai *pixel* baru sebagai berikut:

(00100**01** 111010**00** 110010**00**)

(00100**11** 11001000 11101001)

2. Algoritma Linear Congruential Generator (LCG)

Algoritma *Linear Congruential Generator* (LCG) [5], digunakan untuk menghasilkan bilangan acak semu, yang diberikan dalam bentuk persamaan sebagai berikut:

$$X_{n+1} = (aX_n + b) \bmod m \quad \ldots\ldots\ldots (1)$$

di mana :

X_{n+1} = bilangan acak ke-(n+1) dari barisan bilangan acak

X_n = bilangan acak ke-n

a = konstanta pengali

b = konstanta kenaikan

m = konstanta modulus

Persamaan 1 memiliki nilai awal X_0 sebagai kunci pembangkit atau sering juga disebut umpan (*seed*). X_0 merupakan bilangan bulat lebih besar atau sama dengan nol dan lebih kecil dari m. LCG mempunyai periode tidak lebih besar dari m dan akan mempunyai periode penuh jika memenuhi syarat sebagai berikut :

a. b relatif prima terhadap m
b. (a - 1) dapat dibagi dengan semua faktor prima dari m.
c. (a – 1) adalah kelipatan 4 jika m adalah kelipatan 4
d. m > maks (a,b, X_0)
e. a > 0, b > 0

Sebagai contoh, misalkan ditentukan nilai konstanta m = 64, a = 17, b = 23 yang memenuhi syarat periode penuh. Dengan mensubstitusi nilai-nilai konstanta m, a, dan b ke dalam persamaan 1, akan didapat periode penuh dan menghasilkan bilangan acak sebanyak modulus (m). Pada Tabel I dapat dilihat bilangan acak yang dihasilkan untuk setiap n, dimana n = 0,1,2,… Pada saat n = m, bilangan acak X_{64} yang didapat akan bernilai sama dengan bilangan acak X_0 yaitu 1.

TABEL I.
NILAI n DAN X_n UNTUK a=17, b=23, DAN m=64

n	X_n	n	X_n	n	X_n
0	1	22	43	44	21
1	40	23	50	45	60
2	63	24	41	46	19
3	6	25	16	47	26
4	61	26	39	48	17
5	36	27	46	49	56
6	59	28	37	50	15
7	2	29	12	51	22
8	57	30	35	52	13
9	32	31	42	53	52
10	55	32	33	54	11
11	62	33	8	55	18
12	53	34	31	56	9
13	28	35	38	57	48
14	51	36	29	58	7
15	58	37	4	59	14
16	49	38	27	60	5
17	24	39	34	61	44
18	47	40	25	62	3
19	54	41	0	63	10
20	45	42	23	64	1
21	20	43	30	65	40

Dari Tabel I, tampak bahwa barisan bilangan acak berulang lagi pada n = 64. Algoritma LCG mempunyai periode yang tidak lebih besar dari *modulus* (m), *modulus* ini merupakan ambang batas maksimum dalam pengacakan bilangan. Semakin besar ukuran citra yang akan disisipi pesan, semakin besar pula periodik LCG yang dapat dibangkitkan, sehingga semakin banyak pula pesan yang bisa disisipkan.

3. Proses Encoding dan Decoding Pesan

Dalam melakukan proses *encoding* dan *decoding* digunakan *stego key* yang sama yaitu nilai kostanta a, b, dan m.

a. Proses *Encoding*

Pada proses *encoding*, dilakukan penyisipan pesan rahasia ke dalam *cover image*, hasilnya adalah *stego image*.

Masukan : *Cover image*, pesan teks, nilai konstanta a,b dan m

Keluaran : *Stego image*

Proses:

1) Transformasi pesan teks kedalam bentuk biner.
2) Ubah susunan biner dari pesan menjadi rangkaian bit yang disimpan dalam suatu *array* satu dimensi.

3) Bangkitkan m bilangan acak menggunakan algoritma LCG untuk menentukan letak *pixel* pada *cover image* yang akan disisipi pesan.
4) Transformasi nilai setiap *pixel* dari *cover image* kedalam nilai *Red, Green*, dan *Blue*.
5) Transformasi nilai *Red, Green*, dan *Blue* pada setiap letak piksel yang didapat dari hasil langkah 2 kedalam bentuk biner
6) Sisipkan setiap 2 bit pesan ke dalam bit 7 dan 8 pada setiap nilai *Red, Green*, dan *Blue* hasil langkah 4.
7) Transformasi nilai biner hasil langkah 5 kedalam nilai *Red, Green*, dan *Blue*.
8) Simpan *cover image* yag sudah disisipi pesan menjadi file *stego image*.

b. Proses *Decoding*

Pada proses *decoding* dilakukan ekstrasi pesan dari *stego image*, hasilnya adalah pesan teks.
Masukan : *Stego image*, nilai konstanta a,b dan m.
Keluaran : Pesan teks.
Proses:
1) Bangkitkan m bilangan acak menggunakan algoritma LCG untuk mendapatkan letak *pixel* pada *stego image* yang disisipi pesan.
2) Transformasi nilai setiap *pixel* dari *stego image* kedalam nilai *Red, Green*, dan *Blue*.
3) Transformasi nilai *Red, Green*, dan *Blue* pada setiap letak piksel yang disisipi pesan kedalam bentuk biner.
4) Salin setiap 2 bit pesan pada bit 7 dan 8 dari setiap nilai *Red, Green*, dan *Blue* hasil langkah 3 ke dalam suatu *array* satu dimensi.
5) Transformasikan setiap 8 bit hasil langkah 4 menjadi bentuk karakter.
6) Tampilkan pesan teks.

III. HASIL DAN PEMBAHASAN

A. Eksperimen

Tujuan dari eksperimen ini adalah mengimplementasikan teknik steganografi dengan metode LSB 2 bit dan algoritma LCG untuk menyisipkan pesan kedalam *pixel* yang ditentukan secara acak pada *cover-image*. Sebelum dilakukan proses *encoding*, dilakukan perhitungan ukuran maksimal pesan yang dapat disisipkan pada setiap ukuran pada *cover image*. Untuk mengetahui kualitas citra hasil *encoding* dilakukan perhitungan nilai *Peak Signal to Noise Ratio* (PSNR) dari *stego-image*. Setelah dilakukan perhitungan PSNR, selanjutnya dilakukan proses *decoding*. Jika proses *decoding* menghasilkan pesan yang sesuai dengan pesan asli maka implementasi aplikasi steganografi dinyatakan berhasil. Eksperimen dilakukan menggunakan *tools Matlab*.

Cover image yang digunakan dalam eksperimen adalah berupa citra bitmap berukuran 40x40, 64x64, 160x160, 200x200, 250x250. Konstanta m sama dengan banyaknya *pixel* pada setiap *cover image*. Untuk konstanta a, dan b dihitung sehingga memberikan periode penuh dalam menghasilkan bilangan acak semu sebanyak modulus (m). Untuk *cover image* berukuran 40x40, banyaknya *pixel* pada *cover image* ada 1.600 *pixel*, dan dapat menyisipkan pesan sebanyak 1.600 x 6 bit = 9.600 bit atau 9.600/8 = 1.200 *byte*. Berarti ukuran maksimal pesan adalah 1.200 karakter. Dalam eksperimen yang dilakukan, 5 byte digunakan untuk *flag*. Jadi ukuran maksimal pesan teks yang bisa disipkan adalah 1.195 karakter. Nilai konstanta m dan hasil perhitungan nilai konstanta a, dan b, untuk mendapatkan periode penuh pada setiap masukan citra dapat dilihat pada Tabel II.

TABEL II.
NILAI KONSTANTA a, b, DAN m

No	Ukuran *Cover Image* (*pixel*)	Nilai Konstanta LCG
1	40 x 40	a = 101, b = 991, m = 1.600
2	64 x 64	a = 1.221, b = 2.351, m = 4.096
3	160 x 160	a = 5.001, b = 11.213, m = 25.600
4	200 x 200	a = 10.001, b = 28.411, m = 40.000
5	250 x 250	a = 30.201, b = 29.573, m = 62.500

Perhitungan PSNR digunakan untuk mengetahui perbandingan kualitas citra sebelum dan sesudah disisipkan pesan. Untuk menentukan PSNR, terlebih dahulu harus ditentukan nilai rata-rata kuadrat dari *error* (*Mean Square Error*-MSE). MSE adalah nilai rata-rata kuadrat *error* antara citra asli (*cover image*) dengan citra hasil penyisipan (*stego image*). Perhitungan MSE menggunakan persamaan berikut:

$$MSE = \frac{1}{mn}\sum_{i=1}^{m}\sum_{j=1}^{n}|I(i,j) - K(i,j)|^2 \text{.....(2)}$$

dimana :
MSE = Nilai *Mean Square Error* antara *cover image* dengan s*tego image*
m = panjang citra tersebut (dalam *pixel*)
n = lebar citra tersebut (dalam *pixel*)
(i,j) = koordinat *pixel*
I = *cover image*
K = *stego image*

Selanjutnya nilai PSNR dihitung dari kuadrat maksimum nilai *pixel* citra dibagi dengan MSE, yang diberikan dalam bentuk persamaan berikut:

$$PSNR = 10.\log\left(\frac{MAX_I^2}{MSE}\right) = 20.log\left(\frac{MAX_I}{\sqrt{MSE}}\right) \text{.....(3)}$$

dimana :
PSNR = nilai PSNR *stego image* (dalam dB)
MAX_I = maksimum nilai *pixel*
MSE = nilai MSE

PSNR sering dinyatakan dalam skala logaritmik, dalam *decibel* (dB). Menurut [6], nilai PSNR dibawah 30 dB mengindikasikan kualitas yang relatif rendah, dimana distorsi yang disebabkan penyisipan terlihat jelas. Sedangkan kualitas *stego image* yang tinggi berada pada nilai 40dB dan diatasnya.

B. Pembahasan

Hasil eksperimen proses *encoding* dan *decoding* diberikan pada Tabel III.

TABEL III.
HASIL EKSPERIMEN PROSES *ENCODING* DAN *DECODING*

Ukuran Citra (*pixel*)	**Ukuran File *Cover Image* (KB)**	**Ukuran File *Stego Image* (KB)**	**Panjang (Byte) Pesan (Encoding)**	**Panjang (Byte) Pesan (Decoding)**	**PSNR (dB)**
40 x 40	4,74	4,74	1.195	1.195	43,56
64 x 64	12,0	12,0	3.067	3.067	45,76
160 x 160	75,0	75,0	19.195	19.195	45,59
200 x 200	117	117	29.995	29.995	45,57
250 x 250	183	183	46.870	46.870	46,78

1. Analisis Hasil Ekstraksi

Analisis ini dilakukan untuk melihat apakah nilai konstanta a, dan b memberikan periode penuh dalam membangkitkan bilangan acak semu sebanyak m, sehingga proses ekstraksi menghasilkan pesan yang sama dengan pesan asli sebelum proses *encoding*. Dari tabel 3 dapat dilihat semua eksperimen berhasil melakukan proses penyisipan pesan dan ekstraksi pesan untuk ukuran pesan yang maksimal. Artinya perhitungan nilai konstanta a dan b memberikan periode penuh dalam membangkitkan bilangan acak semu sebanyak m.

2. Analisis Kualitas *Stego Image*

Analisis ini dilakukan dengan membandingkan antara citra asli (*Cover Image*) dengan citra hasil proses *encoding* (*Stego Image*) untuk mengetahui perubahan kualitas gambar yang dihasilkan. Kualitas gambar dari *stego image* bisa dianalisis dari hasil perhitungan nilai PSNR. Dalam eksperimen ini nilai PSNR didapat dari rata-rata nilai PSNR warna *Red, Green*, dan *Blue*. Untuk penyisipan pesan menggunakan metode LSB 2 bit, akan didapat maksimum perbedaan nilai *pixel* antara *cover image* dengan *stego image* untuk warna *Red, Green*, dan *Blue* masing-masing adalah 3, sehingga menggunakan persamaan (1) didapat nilai MSE sebesar 9. Untuk warna *Red, Green*, dan *Blue*, maksimum nilai *pixel*nya adalah 255, sehingga menggunakan persamaan (2) didapat nilai PSNR yang sama untuk warna Red, Green, dan Blue, yaitu sebesar 38,59 dB. Akibatnya nilai PSNR terburuk untuk *stego image* yang didapat menggunakan metode LSB 2 bit adalah rata-rata nilai PSNR warna *Red, Green*, dan *Blue,* yaitu sebesar 38,59 dB. Dari tabel 3 dapat dilihat semua eksperimen memberikan nilai PSNR di atas 40 dB. Berarti menurut [6], kualitas *stego image* dalam eksperimen ini termasuk tinggi.

3. Analisis Perbandingan *Cover image* dan *Stego image*

Bentuk citra yang belum disisipi pesan teks (*cover image*) dengan citra yang sudah disisipi

pesan teks (*stego image*) dapat dilihat pada Gambar 3 sampai Gambar 7. Secara kasat mata tidak terlihat adanya perbedaan kualitas citra antara *cover image* dengan *stego image* untuk semua ukuran citra yang dijadikan eksperimen. Hal ini menunjukkan bahwa keberadaan pesan rahasia di dalam *stego image* tidak dapat langsung diketahui.

(a)

(b)

Gambar 3. Citra ukuran 40x40, (a) *cover image*, (b) *stego image*

(c)

(d)

Gambar 4. Citra ukuran 64x64, (a) *cover image*, (b) *stego image*

(e)

(f)

Gambar 5. Citra ukuran 160x160, (a) *cover image*, (b) *stego image*

(g)

(h)

Gambar 6. Citra ukuran 200x200, (a) *cover image*, (b) *stego image*

(i) **(j)**

Gambar 7. Citra ukuran 250x250, (a) *cover image*, (b) *stego image*

IV. PENUTUP

Dalam makalah ini telah direpresentasikan penyisipan pesan teks ke dalam sebuah citra warna 24 bit menggunakan metode LSB 2 bit dan algoritma LCG. Secara kasat mata kualitas *stego image* tidak berbeda dengan *cover image*. Proses ekstrasi pesan dari *stego image* menghasilkan pesan teks yang sama dengan pesan teks asli. Untuk ukuran panjang pesan rahasia yang maksimal, penyisipan pesan menggunakan metode LSB 2 bit memberikan kualitas stego image yang termasuk tinggi, yaitu di atas 40 dB.

Dalam implementasi algoritma LCG untuk membangkitkan bilangan acak semu, nilai konstanta m sebaiknya tidak sama dengan jumlah *pixel cover image*, sehingga nilai ini yang juga merupakan nilai *stego key* akan lebih tahan terhadap serangan *brute-force attack.*

Dalam pengembangan selanjutnya dapat dilakukan eksperimen menyisipkan pesan berbentuk citra ke dalam citra dengan metode LSB dan algoritma LCG.

DAFTAR PUSTAKA

[1] Gabriel Macharia Kamau, Stephen Kimani, Waweru Mwangi, *An enhanced Least Significant Bit Steganographic Method for Information Hiding*, Journal of Information Engineering and Applications, Vol 2, No.9, 2012

[2] Champakamala .B.S, Padmini.K, Radhika .D. K, *Least Significant Bit algorithm for image steganography*, International Journal of Advance Computer Technology, volume 3, number 4, August 2014

[3] Vikas Tyagi, Atul kumar , Roshan Patel , Sachin Tyagi , Saurabh Singh Gangwar, *Image Steganography using Least Significant Bit with Cryptography*, Journal of Global Research in Computer Science, Volume 3, No. 3, March 2012

[4] Deepesh Rawat, Vijaya Bhandari, 2013, A *Steganography Technique for Hiding Image in an Image using LSB Method for 24 Bit Color Image,* International Journal of Computer Applications, Volume 64, No.20, Feb 2013

[5] Park, S.K, and Miller, K.W. (1988). *Random Number Generators: Good Ones Are Hard To Find*, Communications of the ACM Vol. 31 No. 10: 1192–1201, Oktober 1988.

[6] Cheddad, A., Condell, J., Curran, K., Kevitt, P. Mc. , 2010, *Digital Image Steganography Survey and Analysis of Current Methods*. Signal Processing, Elsevier. Northern Ireland, UK, 2010.