PENYEMBUNYIAN DATA SECARA AMAN DI DALAM CITRA BERWARNA DENGAN METODE *LSB* JAMAK BERBASIS *CHAOS*

Rinaldi Munir
Sekolah Teknik Elektro dan Informatika ITB
rinaldi@informatika.org

Abstrak

Makalah ini mempresentasikan metode penyembunyian data di dalam citra berwarna berbasis chaos. Penyisipan data dilakukan pada bit LSB dari pixel-pixel citra. Untuk meningkatkan kapasitas data yang bisa disembunyikan, jumlah bit LSB yang dimanipulasi dapat lebih besar dari 1 asalkan kualitas citra yang dihasilkan masih relatif baik (metode LSB jamak). Untuk meningkatkan keamanan data, enkripsi sederhana dengan XOR dilakukan terhadap data sebelum disisipkan ke dalam citra. Fungsi chaos digunakan dua kali, pertama untuk menentukan posisi penyisipan data di di dalam citra, dan kedua untuk enkripsi data. Eksperimen menunjukkan bahwa ekstraksi data dari citra sensitif terjadap perubahan nilai awal chaos. Jumlah bit LSB yang dimanipulasi mempengaruhi kualitas citra yang dihasilkan.

Kata Kunci*: penyembunyian data, citra, LSB jamak, chaos, enkripsi.*

1. PENDAHULUAN

Menjaga kerahasiaan pesan atau data merupakan salah satu aspek penting di dalam komunikasi. Solusi yang umum digunakan untuk menjaga kerahasiaan pesan adalah dengan cara mengenkripsinya. Tetapi, pesan terenkripsi dapat menimbulkan kecurigaan bagi pihak lawan yang menduga pesan tersebut merupakan pesan rahasia [1]. Selain itu, eksistensi pesan tetap tidak bisa dihilangkan. Steganografi atau penyembunyian data (*data hiding*) dapat mengatasi masalah ini. Pesan disembunyikan di dalam media lain (*cover*) sehingga pengiriman pesan tidak menimbulkan kecurigaan. Media *cover* yang umum digunakan adalah citra, video, dan audio. Tujuan steganografi adalah menyisipkan sebanyak mungkin data ke dalam *cover* sehingga keberadaannya tidak dapat dipersepi (secara visual jika medianya gambar atau video, atau secara auditori jika medianya adalah sinyal audio) [2]. Hal ini memungkinkan pesan terenkripsi dapat disembunyikan keberadaannya sehingga tidak menarik perhatian pihak ketiga. Dengan kata lain, kriptografi dan steganografi dapat dilakukan secara serial.

Satu metode sederhana yang digunakan untuk menyembunyikan data adalah memanipulasi *least significant bit* (*LSB*) *cover* [1]. Manipulasi bit dilakukan dengan banyak cara, misalnya mengganti bit *cover* dengan bit data atau operasi aritmetika (biasanya operasi *XOR*) antara kedua bit tersebut [2]. Pada media *cover* berupa citra, biasanya hanya 1 bit LSB *pixel* yang dimanipulasi. Jika 1 *pixel* berukuran 1 *byte* (= 8 bit), maka bit paling kanan yang dimanipulasi. Untuk memperbesar ukuran data yang disembunyikan, kita dapat memanipulasi tidak hanya 1 bit LSB, tetapi bisa lebih banyak lagi asalkan kualitas *cover* tidak rusak secara berarti. Inilah yang kita namakan metode LSB jamak.

Untuk membuat penyembunyian data aman, maka data disembunyikan secara acak di dalam cover. Sembarang pembangkit bilangan acak dapat digunakan. Di dalam makalah ini digunakan fungsi *chaos* sebagai pembangkit bilangan acak. *Chaos* mempunyai karaktersitik yang penting yaitu peka terhadap perubahan nilai awal. Jika nilai awal chaos diubah sedikit, maka barisan nilai *chaos* yang dihasilkannya berubah secara signifikan. Sifat ini bersesuaian dengan prinsip *diffusion* dari Shannon [1]. Karena barisan nilai *chaos* yang dibangkitkan menyatakan posisi penyisipan acak di dalam *cover*, maka posisi penyisipan bit data di dalam *cover* berubah secara signifikan jika nilai awal *chaos* diubah hanya sedikit saja.

Makalah ini memaparkan eksperimen menyembunyikan data di dalam citra berwarna dengan metode *LSB* jamak dan menggunakan *chaos* untuk membangkitkan nilai-nilai acak. Untuk membuat pesan tersembunyi lebih aman, pesan tersebut dienkripsi terlebih dahulu dengan barisan *chaos*. Jadi, barisan *chaos* dibangkitkan dua kali: barisan pertama untuk mengenkripsi data dan barisan kedua untuk menentukan posisi penyisipan data di dalam *pixel-pixel* citra.

2. METODE *LSB* JAMAK

Tinjau 3 buah *byte* pada *pixel* sebuah citra:

11010100 10111101 10010110

dan pesan yang disembunyikan:

101101011

Dengan metode modifikasi *LSB* 1-bit, maka hanya 3 bit pertama data yang bisa disembunyikan pada setiap bit *LSB* dari setiap byte *pixel* tadi:

1101010**1** 10111100**1** 1001011**1**

Kita dapat meningkatkan jumlah bit data yang disembunyikan dengan memanipulasi 2-bit *LSB* sehingga kita dapat menyembunyikan 6 bit data:

1101011**10** 1011111**11** 1001010**01**

atau memanipulasi 3bit *LSB* sehingga kita dapat menyembunyikan 9 bit data:

1101**0101** 1011**1101** 1001**0011**

Semakin banyak bit *LSB* yang dimodifikasi dapat mempengaruhi mutu citra *cover*, oleh karena itu ada hubungan timbal balik antara jumlah bit LSB dan *fidelity* citra penampung. Untuk ukuran data yang relatif sangat kecil dibandingkan ukuran citra, peningkatan jumlah bit *LSB* yang dimanipulasi tidak terlihat pengaruhnya secara visual sebab jumlah *pixel* yang dimanipulasi relatif sedikit. Sebaliknya untuk ukuran data yang besar, semakin banyak pixel yang dimanipulasi dan penyembunyian pada bit-bit LSB yang lebih banyak dapat menyebabkan degradasi pada citra penampung.

3. TEORI *CHAOS*

Teori *chaos* berasal dari teori sistem yang memperlihatkan kemunculan yang tidak teratur, meskipun sebenarnya teori ini digunakan untuk menjelaskan kemunculan data acak [4,5]. Karakteristik yang umum di dalam teori *chaos* adalah kepekaannya terhadap perubahan kecil nilai awal (*sensitive dependence on initial condition*). Kepekaan ini berarti bahwa perbedaan kecil pada nilai awal fungsi, setelah fungsi diiterasi sejumlah kali, akan menghasilkan perbedaan yang sangat besar pada nilai fungsinya.

Salah satu fungsi *chaos* sederhana adalah persamaan logistik (*logistic map*). Persamaan logistik dinyatakan sebagai

$$x_{i+1} = r\, x_i\, (1 - x_i) \qquad (1)$$

dengan x_0 sebagai nilai awal iterasi. Daerah asal x adalah dari 0 sampai 1. Konstanta r menyatakan laju pertumbuhan fungsi, yang dalam hal ini $0 \le r \le 4$..

Sebagai contoh, jika kita menggunakan r = 4.0 dan nilai awal x_0 = 0.456, maka kita memperoleh barisan bilangan acak:

$$x_1 = 4.0x_0(1 - x_0) = 0.992256$$
$$x_2 = 4.0x_1(1 - x_1) = 0.030736$$
$$\ldots$$
$$x_{99} = 4.0x_{98}(1 - x_{98}) = 0.914379$$
$$x_{100} = 4.0x_{99}(1 - x_{99}) = 0.313162$$

Nilai-nilai *chaos* tersebut teletak di antara 0 dan 1 dan tersebar secara merata serta tidak ada dua nilai yang sama.

Sifat *chaos* yang peka terhadap perubahan kecil nilai awal diilustrasikan sebagai berikut. Jika nilai awal diubah sebesar 0.00001 menjadi x_0 = 0.45601, maka setelah iterasi ke-14 nilai *chaos* mulai lebih besar dari 0.0625 dan semakin lama nilainya mengalami divergensi dari nilai-nilai *chaos* bila x_0 = 0.456.

4. METODE YANG DIUSULKAN

Citra yang digunakan sebagai *cover* adalah citra berwarna yang berukuran m x n dengan kedalaman warna 24-bit. Setiap *pixel* panjangnya 3 *byte* (masing-masing bersesuaian dengan komponen R (*red*), G (*green*), dan B (*blue*). Jadi, setiap *pixel* dapat menyimpan minimal 3 bit data. Nyatakan *pixel-pixel* citra dalam sebuah larik *I* yang berukuran mn.

Misalkan *D* adalah larik sepanjang yang berukuran *ab* yang berisi bit-bit data yang akan disembunyikan ke dalam I. Sembarang data harus dinyatakan dalam biner dan setiap bit data disimpan di dalam matriks D tersebut.

Mula-mula bangkitkan barisan *chaos* C_1 dengan *logistic map* dengan nilai awal i_1 untuk menentukan posisi penyisipan bit data di dalam citra (posisi *pixel* adalah dari 1 sampai nm). Karena nilai-nilai *chaos* adalah bilangan riil sedangkan lokasi *pixel* adalah *integer*, maka setiap $x \in C_1$ dikonversi ke *integer* dengan mengalikannya dengan nm. Bilangan *integer* yang dihasilkan menyatakan nomor indeks *pixel* dalam rentang nilai 1 sampai nm. Hasil konversi ke *integer* disimpan di dalam larik *K*. Kita harus menjamin tidak boleh nilai-nilai *integer* yang dihasilkan ada yang sama.

Selanjutnya bangkitkan barisan *chaos* C_2 sepanjang *ab* dengan nilai awal i_2 untuk enkripsi data. Setiap $x \in C_2$ dikonversi ke {0, 1} dengan menggunakan operasi pengambangan (*thresholdding*) berikut:

$$T(x) = \begin{cases} 1 & , x \le u \\ 0 & , x > u \end{cases} \quad (2)$$

yang dalam hal ini *u* adalah nilai ambang yang dispesifikasikan pengguna. Hasil pengambangan disimpan di dalam matriks *P*. Data *D* yang akan disembunyikan dienkripsi terlebih dahulu dengan barisan *P* sebagai berikut:

$$E = D \oplus P \quad (3)$$

Selanjutnya *E* disisipkan ke dalam *I* pada posisi *pixel* yang ditentukan oleh *K* dengan memanipulasi bit *LSB*-nya. Jumlah bit *LSB* pada setiap *byte* yang dimanipulasi dispesifikasikan di awal, tetapi jumlahnya harus dalam rentang 1 sampai 7. Citra baru yang dihasilkan setelah penyembunyian data dinamakan *stego-image*.

Kedua nilai awal barisan *chaos* (yaitu i_1 dan i_2) berlaku sebagai kunci penyisipan dan keduanya digunakan kembali pada saat ekstraksi data dari *stego-image*. Bit-bit data diekstraksi dari *stego-image* pada posisi *pixel* yang ditentukan oleh K. Bit-bit hasil ekstraksi harus didekripsi terlebih dulu dengan barisan P untuk mendapatkan data semula dengan persamaan:

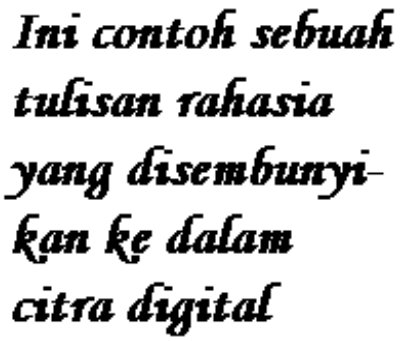
$$D = E \oplus P \quad (4)$$

Selama parameter kunci yang dimasukkan pada waktu ekstraksi sama dengan kunci pada waktu penyisipan, data yang diekstraksi dari *stego-image* tepat sama dengan data semula.

5. EKSPERIMEN DAN ANALISIS HASIL

Metode ini telah diuji dengan menggunakan kakas MATLAB 7. Citra yang digunakan sebagai penampung adalah citra *peppers* yang berukuran 256 x 256 dengan kedalaman warna 24 bit. Data yang disembunyikan adalah sebuah citra hitam-putih (biner) yang berisi sebuah tulisan (berukuran 18 KB). Keduanya diperlihatkan pada Gambar 1(a) dan 1(b). Nilai i_1 yang digunakan adalah 0.35 dan i_2 = 0.2785. Kualitas citra hasil diukur dari *PSNR* nya (selain dari pengamatan secara visual).

(a)

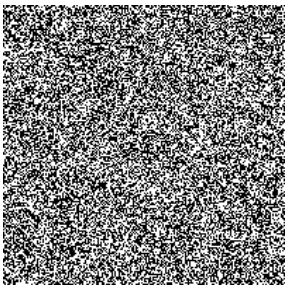


(b)

Gambar 1. (a) Citra *cover*; (b) data yang disembunyikan

Stego-image yang dihasilkan dengan jumlah bit *LSB* yang dimanipulasi adalah 1 bit diperlihatkan pada Gambar 2(a). PSNR nya = 53,29. Kualitas *stego-image* ini sangat baik dan tidak berbeda dengan citra semula.

(a)



(b)

Gambar 2. (a) *Stego-image* (bit_LSB = 1 dan *PSNR* = 53,29); (b) Data yang diekstraksi salah jika nilai i_1 diubah dari 0.35 menjadi 0.35001

Percobaan lain dilakukan dengan mengubah nilai i_1 pada saat ekstraksi data sebesar 10^{-5} menjadi 0.35001. Karena barisan *chaos* dengan nilai awal i_1 menentukan posisi penyisipan data di dalam citra, maka sifat *chaos* yang peka terhadap perubahan kecil nilai awal menyebabkan posisi penyisipan data menjadi kacau, akibatnya bit data yang diekstraksi salah. Data salah yang diekstraksi karena perubahan kecil pada i_1 diperlihatkan pada Gambar 1(d). Percobaan ini menunjukkan efek *diffusion* pada *chaos* dan ini berguna untuk meningkakan sensivitas keamanan data.

Percobaan dilanjutkan dengan memanipulasi bit *LSB* lebih banyak (2 bit, 3 bit, hingga 6 bit). *Stego-image* yang dihasilkan untuk masing-masing jumlah bit *LSB* diperlihatkan pada Gambar 3. Hasil percobaan menunjukkan kualitas citra yang dihasilkan dengan memanipulasi berbagai jumlah bit *LSB* hingga bit *LSB* = 6 secara visual masih relatif bagus meskipun *PSNR* nya terus menurun. Hal ini disebabkan ukuran data yang disisipkan masih relatif kecil dibanding jumlah pixel yang ada.

(a) (a) bit *LSB* = 1, *PSNR* =53,3

(b) bit *LSB* = 2, *PSNR* =49,3

(c) bit *LSB* = 3, *PSNR* =44,8

(d) bit *LSB* = 4, *PSNR* =39,9

(e) bit *LSB* = 5, *PSNR* =34,9

(f) bit *LSB* = 6, *PSNR* =29,6

Gambar 3. *Stego-image* dengan berbagai percobaan jumlah bit *LSB* yang dimanipulasi

Tetapi ketika jumlah bit *LSB* yang dimanipulasi = 7 bit, terlihat bintik-bintik putih pada *stego-image* yang mengindikasikan timbulnya kerusakan gambar karena nilai-nilai *pixel* yang dimanipulasi berubah secara signifikan dibandingkan nilai semula (Gambar 4).

Gambar 4. *Stego-image* yang cacat ketika jumlah bit *LSB* yang dimanipulasi = 7 bit (*PSNR* = 24,9)

6. KESIMPULAN

Penyembunyian data pada citra berwarna dengan metode *LSB* jamak dapat meningkatkan kapasitas data yang disisipkan, tetapi ada timbal baliknya dengan penurunan kualitas *stego-image*. Penggunaan *chaos* di dalam metode ini berhasil meningkatkan keamanan keamanan data terhadap perubahan kecil nilai kunci.

7. DAFTAR PUSTAKA

[1] Munir, Rinaldi, "*Kriptografi*", Penerbit Informatika, Bandung, 2006.

[2] Marvel, Lisa M., "*Image Steganography for Hidden Communication*", Disertasi di University of Delaware, 1999.

[3] Schneier, Bruce, *Aplied Cryptography 2nd*, John Wiley & Sons, 1996

[4] www.yahoo.com , *Chaos Theory: A Brief Introduction*, diakses pada bulan November 2005

[5] James Lampton, *Chaos Cryptography: Protecting Data Using Chaos*, Mississippi School for Mathematics and Science.