Jurnal Mahajana Informasi, Vol.3 No. 1, 2018
e-ISSN: 2527-8290

Steganografi Pesan Teks ke dalam Citra Digital Menggunakan Kombinasi Algoritma Dijktra dan Algoritma Bubble Sort

Bambang T.J. Hutagalung

bambangtjh@gmail.com
Universitas Potensi Utama

ABSTRAK

Data dan informasi adalah sesuatu hal yang sangat penting jika mengandung suatu informasi yang penting dan rahasia. Oleh karena itu, data dan informasi sangat perlu untuk dilindungi dari berbagai tindak kejahatan seperti pencurian, modifikasi dan perusakan. Steganografi adalah salah satu teknik untuk melindungi data dan informasi dengan baik. Algoritma yang digunakan penulis adalah sesuatu yang baru, yaitu dengan mengkombinasikan antara algoritma Dijktra dan algoritma *Bubble Sort*. Data atau informasi yang disisipkan adalah data teks dan sebagai penampungnya adalah data berupa citra digital.

Kata Kunci – data, informasi, citra digital, steganografi, algoritma Dijktra, algoritma *Bubble Sort*.

I. PENDAHULUAN

A. Latar Belakang

Dalam sistem komputer, tidak akan pernah bisa terlepas dari penggunaan data dan informasi sebagai sumber masukan atau hasil keluaran. Data dan informasi tersebut biasanya disebut *file* dalam sistem komputer. *File* tersebut dapat berbentuk teks, citra digital, audio, video, dan audio video. Sebuah *file* akan sangat berharga dan bersifat vital jika memuat hal-hal yang penting dan rahasia, sehingga akan sangat rentan terhadap berbagai tindak kejahatan. Tindak kejahatan terhadap *file* tersebut dapat berupa pengrusakan, modifikasi, pencurian, dan sebagainya. Oleh karena itu, diperlukan media yang aman dan handal untuk mengirim, menyimpan, maupun menerima *file* tersebut. Selain melakukan pengamanan pada media transfer *file*, pengamanan dapat juga dilakukan pada media penyimpanan *file* sebelum dikirim melalui media transfer.

Bentuk pengamanan kerahasiaan isi atau konten *file* dapat menggunakan teknik kriptografi dan steganografi. Dalam teknik kriptografi, terdapat dua proses penting yaitu proses enkripsi untuk mengubah karakter dari *file*, dan proses dekripsi untuk mengembalikan bentuk karakter *file* ke bentuk yang semula. Bentuk karakter dari isi file hasil enkripsi yang telah berupa menjadi susunan karakter yang sulit dimengerti, tentunya akan menarik perhatian para codebreaker. Sehingga mereka akan berusaha untuk mendekripsi isi file tersebut menjadi bentuk karakter yang semula. Oleh karena itu, diperlukan sebuah teknik penyimpanan *file* yang aman, tidak terlalu mencolok dan tidak terlalu menarik perhatian para pelaku tindak kejahatan terhadap file tersebut. Teknik steganografi adalah salah satu teknik penyimpanan *file*, dengan cara menyisipkannya ke dalam media penampung. Proses ini terdapat dua bagian besar, yaitu proses penyisipan *file* ke media penampung dan ekstraksi *file* dari media penampung. Pada proses penyisipan *file* tersebut berhasil dilakukan, tentu harus tidak memberikan perubahan yang mencolok atau hampir tidak ada perubahan pada media penampungnya. Sehingga media penampung yang telah disisipi *file* tersebut tidak akan menarik perhatian perhatian para pelaku kejahatan.

Oleh karena itu, penulis tertarik membuat sebuah teknik steganografi dengan melakukan hybrid terhadap dua metode yang berbeda. Dan yang menarik dari penelitian ini adalah bahwa algoritma yang digunakan sangat jarang atau

mungkin belum pernah digunakan untuk teknik steganografi. Algoritma yang digunakan dalam penelitian ini adalah algoritma yang biasanya untuk mencari lintasan terpendek yaitu algoritma Dijktra, dan algoritma untuk pengurutan bilangan bulat yaitu algoritma Bubble Sort. Dalam penelitian ini, *file* yang akan diuji adalah *file* teks yang akan disisipkan dan *file* citra digital sebagai media penampungnya.

II. LANDASAN TEORI

A. Analisis Algoritma Dijktra

Misalkan G adalah graf berarah berlabel dengan titik-titik V(G) = {v1, v2,..., vn} dan path terpendek yang dicari adalah dari v1 ke vn. Algoritma *Dijkstra* dimulai dari titik v1. dalam iterasinya, algoritma akan mencari satu titik yang jumlah bobotnya dari titik 1 terkecil. Titik-titik yang terpiih dipisahkan dan titik-titik tersebut tidak diperhatikan lagi dalam iterasi berikutnya. Misalkan:

V (G) = {v1, v2,.., vn}
L = Himpunan titik-titik e V (G) yang sudah terpilih dalam jalurpath terpendek.
D(j) = Jumlah bobot path terkecil dari v1 ke vj.
w(i,j) = Bobot garis dari titik vi ke vj.
w*(1,j) = Jumlah bobot path terkecil dari v1 ke vj

Secara formal, algoritma *Dijkstra* untuk mencari *path*/rute terpendek adalah sebagai berikut:
L = { };
V = {v2, v3,..,vn}.
Untuk i = 2, ..., n, lakukan D(i) – w(1, i)
Selama v
n ∉ L lakukan:
a. Pilih titik vk ∈ V - L dengan D(k) terkecil.
L = L ∪ {vk}
b. Untuk setiap vj ∈ V - L lakukan:
Jika D(j) > D(k) + W(k,j) maka ganti D(j)
dengan D(k) + W(k,j) Untuk setiap vj ∈ V, w*(1, j) = D(j)

Menurut algoritma di atas, *path* terpendek dari titik v1 ke vn adalah melalui titik-titik dalam L secara berurutan, dan jumlah bobot path terkecilnya adalah(n).

B. Analisis Algoritma Bubble Sort

Pengurutan merupakan proses dasar yang ada dalam algoritma dan stuktur data. Terdapat banyak algoritma pengurutan yang sering digunakan, namun pada tulisan kali ini akan dibahas mengenai dasar algoritma Bubble Sort. Algortima ini merupakan algortima pengurutan sederhana dan biasanya dipelajari sebagai pokok bahasan seputar pengurutan.

Algoritma Bubble Sort ini merupakan proses pengurutan yang secara berangsur-angsur berpindah ke posisi yang tepat karena itulah dinamakan Bubble yang artinya gelembung. Algoritma ini akan mengurutkan data dari yang terbesar ke yang terkecil (*ascending*) atau sebaliknya (*descending*).

Secara sederhana, bisa didefenisikan algoritma Bubble Sort adalah pengurutan dengan cara pertukaran data dengan data disebelahnya secara terus menerus sampai dalam satu iterasi tertentu tidak ada lagi perubahan.

Untuk belajar algoritma Bubble Sort ini kita hanya perlu memahami cara yang digunakan untuk mengurutkan data, sederhananya algoritma ini menggunakan perbandingan dalam operasi antar elemennya. Di bawah ini merupakan gambaran dari algoritma Bubble Sort dengan array “3 1 4 2 8”.
Proses pertama
(3 1 4 2 8) menjadi (1 3 4 2 8)
(1 3 4 2 8) menjadi (1 3 4 2 8)
(1 3 4 2 8) menjadi (1 3 2 4 8)
(1 3 2 4 8) menjadi (1 3 2 4 8)
Proses kedua
(1 3 2 4 8) menjadi (1 3 2 4 8)
(1 3 2 4 8) menjadi (1 2 3 4 8)
(1 2 3 4 8) menjadi (1 2 3 4 8)
(1 2 3 4 8) menjadi (1 2 3 4 8)
Proses ketiga
(1 2 3 4 8) menjadi (1 2 3 4 8)

(1 2 3 4 8) menjadi (1 2 3 4 8)
(1 2 3 4 8) menjadi (1 2 3 4 8)
(1 2 3 4 8) menjadi (1 2 3 4 8)

Jika kita perhatikan proses diatas, para proses kedua data sudah terurut dengan benar. Tetapi algoritma Bubble Sort tetap berjalan hingga proses kedua berakhir. Proses ketiga masih terus berjalan karena pada algoritma Bubble Sort maksud terurut itu adalah tidak ada satupun penukaran pada suatu proses. Proses ketiga ini dilakukan untuk verifikasi data.

C. Algoritma Penyisipan dan Ekstraksi

1. Proses Penyisipan

Langkah-langkah proses penyisipan pesan.

1) Membaca pesan (R) yang akan disisipkan ke dalam citra digital (C).
2) Misalkan: pesan = RAHASIA
3) Membaca panjang karakter pesan (p), sehingga p=7.
4) Mengkonversi setiap karakter pesan menjadi bilangan decimal berdasarkan kode ASCII.
5) KR[p]={82,65,72,65,83,73,65}
6) Membuka citra digital (C) sebagai penampung pesan (R) dan membaca ukuran tinggi (m) dan lebarnya (n).

Misalkan gambar 2.1 adalah sebuah citra digital (C) dengan ukuran $m_x n$.

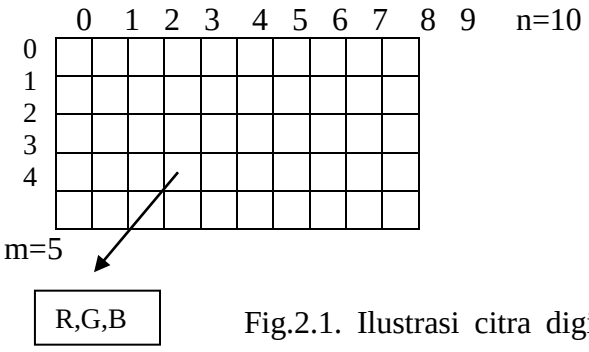


Fig.2.1. Ilustrasi citra digital (C)

R=Red; G=Green; B=Blue.

1.) Membaca nilai RGB piksel dari setiap kolom, dari kolom berindeks n=0 hingga n-1.

Misalkan diambil satu kolom piksel dari citra digital (C) yaitu kolom pertama atau kolom yang berindeks 0 beserta dengan nilai gradiasi warnanya. Gambarnya dapat ditunjukkan pada gambar 2.2.

	n=0
0	(60,75,100)
1	(65,77,101)
2	(63,74,99)
3	(64,64,98)
4	(61,80,102)

Fig. 2.2. Ilustrasi nilai RGB piksel dari kolom n=0.

2.) Membaca isi pesan dan panjang pesan yang akan disisipkan ke dalam citra digital.
Pesan = RAHASIA dan panjang teks p=7
3.) Kemudian mengubah karakter dari setiap isi pesan ke dalam kode ASCII.
Misalnya:
PR[p] = [82,65,72,65,83,73,65];
4.) Baca nilai RGB piksel citra digital pada kolom pertama, yaitu dari titik (m,n) =0,0 hingga titik (m,n) = 0,n-1.
Misalnya:
Nilai piksel dari kolom pertama dari citra digital

R=60;G=75;B=100
R=65;G=77;B=101
R=63;G=74;B=99
R=64;G=64;B=98
R=61;G=80;B=102

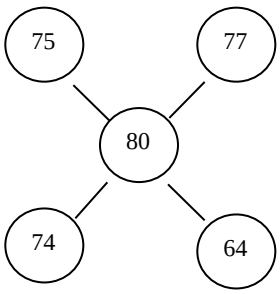
5.) Baca nilai Red (R) dari nilai piksel kolom pertama dan simpan nilainya dalam array ER[m]. Dan indeks elemenya dicatat dalam array IR[m]. Pengurutannya menggunakan algoritma Bubble Sort.

Misalnya:
ER[m]=[60,65,63,64,61]
IR[m]=[0,1,2,3,4,5]

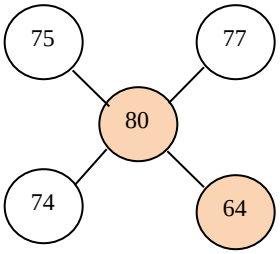
6.) Pengurutan elemen array ER[m] diikuti dengan pergeseran elemen indeksnya yang disimpan dalam array IR[m]
Misalnya:
ER[m]=[60,61,63,64,65]
IR[m]=[0,4,2,3,1]

7.) Dibaca elemen pertama dari array PR[] yaitu PR[0]=82. Kemudian berdasarkan elemen ER[m], dilakukan lompatan secara berulangan sebanyak 82 kali.
Misalnya:
ER[m]=[60,61,63,64,65], setelah dilakukan lompatan sebanyak 82 kali, posisi terakhir pada angka 61 yaitu elemen kedua dengan indeks IR[1]=4.

8.) Pada indeks IR[1]=4, dibaca nilai Green (G) pada posisi piksel tersebut. Kemudian jadikan nilainya sebagai titik pusatnya, untuk dibandingkan dengan nilai Green (G) pada titik-titik yang lain pada kolom pertama dari piksel citra digital.
Misalnya:



9.) Dengan menggunakan algoritma Dijktra, carilah jarak terdekat dari titik pusat ke titik yang terhubung dengannya. Dan diperoleh nilai yang terdekat adalah nilai 64.
Misalnya:



10.) Kemudian dicatat posisi indeks dari nilai Green (G) = 64 ke arrayi Kunci[] pada elemen pertama. Elemen ini berfungsi sebagai kunci untuk mengekstrak pesan dari citra digital.
Misalnya:
Kunci[0]=3

11.) Kemudian dibaca nilai Blue (B) piksel pada indek yang sama dengan nilai Green (G) = 64. Nilai Blue (B) nya adalah 98. Kemudian diubah nilainya sama dengan nilai PR[0]=82.
Misalnya:
Nilai awal piksel dari kolom pertama:

R=60;G=75;B=100
R=65;G=77;B=101
R=63;G=74;B=99
R=64;G=64;**B=98**
R=61;G=80;B=102

Nilai akhir piksel dari kolom pertama:

R=60;G=75;B=100
R=65;G=77;B=101
R=63;G=74;B=99
R=64;G=64;**B=82**
R=61;G=80;B=102

12.) Langkah proses nomor 8 hingga nomor 15 dilakukan hingga semua elemen PR[] telah tesisipkan ke dalam citra digital.

2. Proses Ekstraksi

Langkah-langkah proses ekstraksi pesan.

1.) Dengan menggunakan kunci[]=[3,0,1,2,2,4,3], maka dibaca elemenya satu persatu.
2.) Elemenya tersebut digunakan untuk membaca perkolom dari citra digital yang telah disisipi dengan pesan teks.

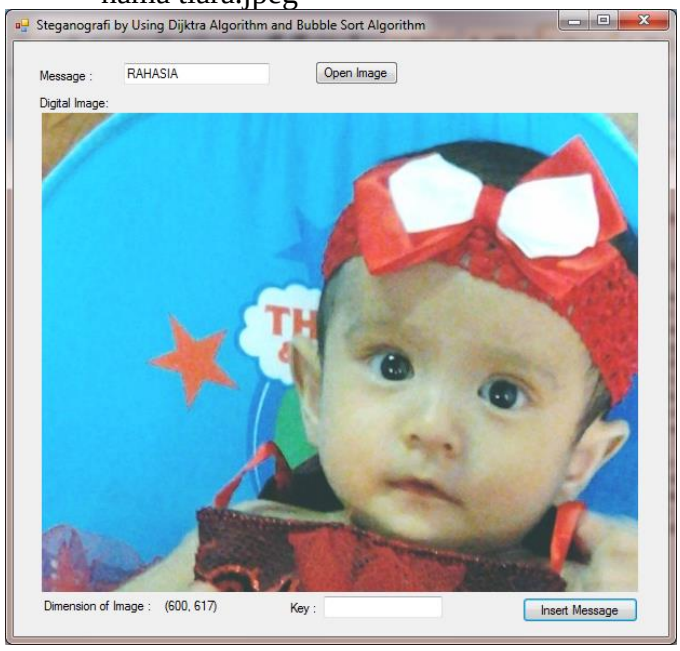
3.) Jumlah elemen kunci[], menunjukkan jumlah karakter teks yang tersimpan pada citra digital. Elemen pada kunci[] menunjukkan posisi baris dari nilai piksel Blue (B).

4.) Nilai piksel Blue (B) yang sesuai dengan elemen kunci[] adalah nilai decimal karakter yang sesuai dengan tabel ASCII. Kemudian semua elemennya tersebut simpan ke dalam array E[].
Misalnya:
E[]=[82,65,72,65,83,73,65]

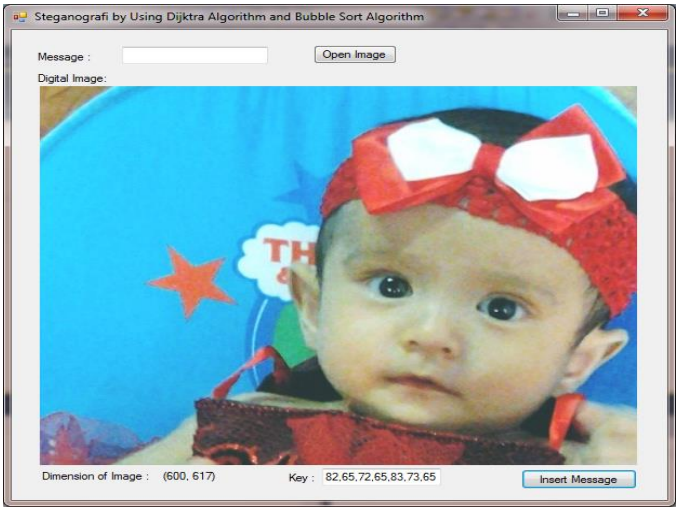
5.) Semua elemen array E[] diubah menjadi karakter yang sesuai dengan tabel ASCII. Hasil konversinya disimpan dalam variabel pesan.
Misalnya:
Pesan=RAHASIA

III. HASIL PENGUJIAN

1. Tampilan program sebelum proses penyisipan pesan yang berisi kata "RAHASIA" ke dalam citra digital dengan nama tiara.jpeg



2. Tampilan program setelah proses penyisipan pesan ke dalam citra tiara.jpeg. Kemudian akan menghasilkan key, yang akan digunakan untuk ekstraksi pesan dari citra tiara.jpeg yang telah disisipi pesan.



3. Sebelum proses ekstraksi, telebih dahulu diinputkan key yang telah dibangkitkan pada proses penyisipan dan membuka citra tiara.jpeg yang telah disisipi oleh pesan teks yang disisipkan.

4. Setelah proses ekstraksi, maka pesan ditampilkan yaitu "RAHASIA" yang sesuai dengan pesan yang disisipkan sebelumnya.

IV. KESIMPULAN DAN SARAN

A. Kesimpulan

Penelitian ini menghasilkan beberapa kesimpulan sebagai berikut:

1. Proses penyisipan teks ke dalam citra digital dan ekstraksi teks dari dalam citra digital dapat dilakukan dengan baik.
2. Waktu proses penyisipan teks ke dalam citra digital dan ekstraksi teks dari dalam citra digital dapat dilakukan dengan waktu yang singkat.

B. Saran

Saran penulis terhadap penelitian ini adalah agar data yang digunakan sebagai data untuk disisipkan dapat berupa gambar, audio, dan video, serta data sebagai penampungnya dapat berupa gambar, audio, dan video.

DAFTAR PUSTAKA

Alam, M. & Chugh, A., "Sorting Algorithm: An Empirical Analysis", International Journal of Engineering Science and Innovative Technology (IJESIT),Vol 3, Issue 2, March 2014

Fitria and Triansyah, A., Implementasi Algoritma Dijkstra Dalam Aplikasi Untuk Menentukan Lintasan Terpendek Jalan Darat Antar Kota di Sumatera Bagian Selatan, JSI, Vol. 5, No. 2, Oktober 2013

Kumalasari, D., Analisa Perbandingan Kompleksitas Algoritma Bubble Sort, Cocktail Sort dan Comb Sort dengan Bahasa Pemrograman C++, Speed, Vol. 9, No. 2, 2017

Mundra, J. and Pal, B.L., "Minimizing Execution Time of Bubble Sort Algorithm", International Journal of Computer Science and Mobile Computing, vol.4, pp 173-181, September 2015

Purnomo, D.M.J and et al, "Implementation of Serial and Parallel Bubble Sort of FPGA", Journal of Computer Science and Information, vol 2, pp 113-120, February 2016

Rohmanu, A., "Implementasi Kriptografi dan Steganografi Dengan Metode Algoritma DES dan Metode End Of File", SIMANTIK, Vol.1, No. 2, Maret 2017

Syawal, M.F., et al, Implementasi Teknik Steganografi Menggunakan Algoritma Vigenere Cipher Dan Metode LSB, TICOM, Vol. 4, No.3, Mei 2016

Trivedi, D & et al, " Min-Max Select Bubble Sorting Algorithm", International Journal of Applied Information System (IJAIS), 2013

Yusuf, M.A, et al, "Implementasi Algoritma Dijktra dalam Menemukan Jarak Terdekat dari Lokasi Pengguna ke Tanaman yang dituju Berbasis Android", Pengembangan Teknologi

Informasi dan Ilmu Komputer, Vol. 1 No. 12,
pp. 1779-1787, Desember 2017