Penerapan Metode End Of File Pada Steganografi Citra Gambar dengan Memanfaatkan Algoritma Affine Cipher sebagai Keamanan Pesan

[1)]Achmad Fauzi
STMIK KAPUTAMA , Jl. Veteran No. 4A-9A, Binjai, Sumatera Utara
http://www.kaputama.ac.id // E-mail : fauzyrivai88@gmail.com

[2)] Rizka Putri Rahayu
UNIVERSITAS NEGERI MEDAN, Jalan Willem Iskandar Pasar V Medan Estate, Medan, 20221
Email: rizkaputrirahayu2@gmail.com

ABSTRAK
The development of the world of information technology is very fast lately influential in all aspects of life. Kriprografi is a science based on mathematical techniques to deal with information security such as confidentiality, data integrity, and entity authentication. Affine Cipher or Affine password is a substitution cipher that is specifically for encryption or decryption per character. Therefore, Affine's password includes the stream cipher cryptographic algorithm. Steganography is a way of hiding the contents of a data in a cover of media or other digital data that cannot be predicted by ordinary people so as not to arouse suspicion from those who see it. The steganography image file has the same number of pixels as the original image file and there is no color difference between the two. The size of the steganography image file will increase, according to the length of the hidden message plus 6 bytes, in the form of 1 byte EOF and 5 byte message length. The End Of File (EOF) method is a technique for inserting data at the end of a file. When opened with a photo editor application, both images have the same pixel size, namely 240 x 238, and do not change color. In this test, what changes is the size of the image file, ie the initial size of the image is 49.132 bytes, changes to 49.145 bytes, or there is an addition of 13 bytes.

Kata Kunci: Kriptografi, Steganografi, Metode End Of File, Affine Cipher, Citra

PENDAHULUAN

Perkembangan dunia teknologi informasi yang sangat pesat akhir-akhir ini berpengaruh dalam segala aspek kehidupan. Salah satunya yaitu dalam pengamanan informasi yang bersifat rahasia. Kerahasiaan informasi merupakan suatu aspek yang penting. Informasi yang sifatnya rahasia perlu disembunyikan agar tidak diketahui oleh orang yang tidak berhak.

Untuk mengamankan informasi, biasanya digunakan ilmu kriptografi. Kriptografi merupakan ilmu komputer yang memiliki fokus dalam bidang pengamanan data (*security*) [2]. Pihak luar juga dapat merusak pesan teracak dengan tujuan agar pihak yang dituju tidak menerima pesan dengan utuh. Untuk mengatasi masalah ini, maka dapat digunakan ilmu Steganografi. Steganografi merupakan cara menyembunyikan pesan di dalam suatu sampul media yang tidak dapat diduga oleh orang biasa sehingga tidak menimbulkan kecurigaan kepada orang yang melihatnya. Citra menjadi salah satu media sampul (*cover media*) yang sering digunakan dalam ilmu steganografi untuk menyembunyikan informasi rahasia. Citra digital merupakan gambar yang diolah oleh komputer yang disimpan pada komputer menjadi angka-angka yang menunjukkan besar intensitas pada masing-masing warna piksel.

Citra digital dibentuk oleh kumpulan titik yang dinamakan piksel (*pixel* atau *picture element*). Salah satu metode steganografi yang dapat digunakan adalah metode *End of File* (EOF) [3].

Metode EOF merupakan salah satu teknik untuk menyisipkan pesan pada akhir *file*. Prinsip kerja EOF adalah menyisipkan karakter EOF pada akhir *file* sebelum menambahkan pesan tersembunyi di belakang *file* tersebut [5]. Untuk memperkuat aspek keamanan pada pesan yang akan disembunyikan, maka pesan akan diacak atau dienkripsi dengan menggunakan algoritma *Affine Cipher* (sandi *Affine*) sebelum pesan disembunyikan dengan metode EOF.

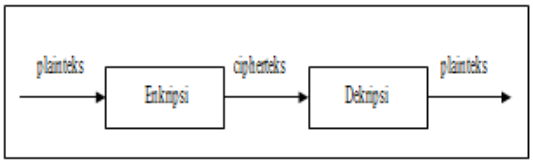
LANDASAN TEORI

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089

2.1 Defenisi Kriptografi

Kriprografi adalah ilmu yang bersandarkan pada teknik matematika untuk berurusan dengan keamanan informasi seperti kerahasiaan, keutuhan data, dan otentikasi entitas [7]. Dalam arti lain, *cryptografhy* adalah seni dan ilmu dalam mengamankan pesan. Dalam dunia kriptografi, pesan disebut *plaintext* atau *cleartext*. Proses untuk menyamarkan pesan dengan cara sedemikian rupa untuk menyembunyikan isi aslinya disebut enkripsi.

Pesan yang telah dienkripsi disebut chipertext. Proses pengembalian sebuah *ciphertext* menjadi *plaintext* disebut dekripsi. Pesan asli yang dirahasiakan dinamakan plainteks (*plaintext*, artinya teks jelas yang dapat dimengerti), sedangkan pesan hasil penyandian disebut cipherteks (*ciphertext*, artinya teks tersandi). Pesan yang telah disandikan dapat dikembalikan lagi ke pesan aslinya hanya oleh orang yang berhak (orang yang mengetahui metode penyandian dan memiliki kunci penyandian). Proses menyandikan plainteks menjadi cipherteks disebut enkripsi (*encryption*), sedangkan proses mengembalikan cipherteks menjadi plainteks disebut dekripsi (*decryption*).



Gambar 1. Skema Proses Enkripsi dan Dekripsi

2.2 Algoritma Affine Cipher

Affine *Cipher* atau sandi Affine adalah *substitution cipher* yang dikhususkan untuk melakukan enkripsi atau dekripsi per karakter. Oleh karena itu, sandi Affine termasuk algoritma kriptografi *stream cipher*. Sistem sandi Affine merupakan sandi alfabetik yang menggunakan teknik substitusi fungsi linier *ap*

+ *b* untuk enkripsi teks asli *p* dan a^{-1} (*c* – *b*)
untuk dekripsi teks sandi *c*. (Sadikin, 2012:46) Di dalam sandi Affine, banyaknya alfabet /
karakter yang digunakan disimbolkan dengan
nilai dari 0 ... *m*-1. Pada tugas akhir ini, yang dienkripsi adalah *byte*, dengan nilai 0 … 255, sehingga nilai *m* yang digunakan adalah 256. Kemudian, metode ini menggunakan fungsi modulo aritmatika untuk mengubah nilai dari setiap karakter menjadi kode rahasia.
Fungsi enkripsi untuk setiap karakter adalah sebagai berikut:

$$E(p) = (a \times p + b) \bmod m$$

dimana:
p = *plaintext*
a dan *b* = kunci numerik yang digunakan.
m = 256 (maksimum nilai *byte* +1)
Syarat pemilihan nilai kunci adalah nilai *a* dan *m* harus relatif prima atau GCD(*a*, *m*) = 1, sedangkan nilai *b* dapat dipilih secara acak. Sedangkan, fungsi dekripsi untuk setiap karakter adalah:

$$D(c) = a^{-1}(c - b) \bmod m$$

dimana:
c = *ciphertext*
a dan *b* = kunci numerik yang digunakan.
m = 256 (maksimum nilai *byte* +1)
a^{-1} = invers dari perkalian modulo dengan *m*.

Nilai a^{-1} harus memenuhi ketentuan berikut:

$$1 = a \times a^{-1} \bmod m$$

Contoh dari pembuktian persamaan di atas dapat dilihat pada perhitungan berikut:
a = 9663 *m* = 256
$a^{-1} \bmod 256 = 9663^{-1} \bmod 256 = 63$
(gunakan algoritma *Extended Euclidean* untuk menyelesaikan $a^{-1} \bmod 256$)
Bukti:

$1 = a \cdot a^{-1} \bmod m$

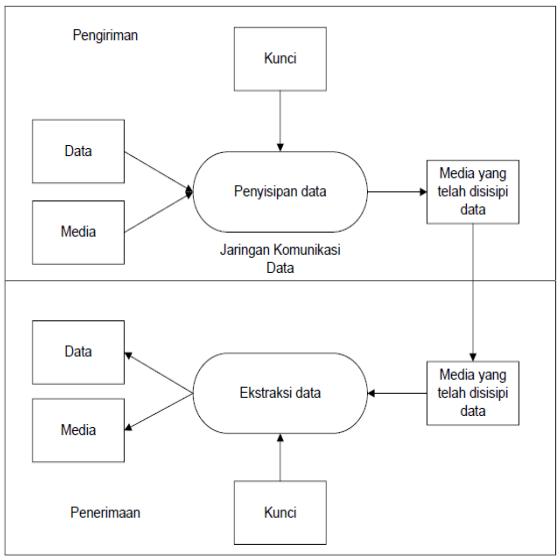
$1 = ((a \bmod m) \cdot (a^{-1} \bmod m)) \bmod m$
$1 = ((9663 \bmod 256) \cdot$
$(9663^{-1} \bmod 256)) \bmod 256$
$1 = (191 \cdot 63) \bmod 256$
$1 = 12033 \bmod 256$
1 = 1 **(**Benar**)**

2.3 Defenisi Steganografi

Steganografi merupakan cara menyembunyikan isi suatu data di dalam suatu sampul media atau data digital lain yang tidak dapat diduga oleh orang biasa sehingga tidak menimbulkan kecurigaan kepada orang yang melihatnya [8]. Gambar

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089

2. menggambarkan ilustrasi dasar dari konsep Steganografi.



Gambar 2. Illustrasi Steganografi

Penyembunyian data rahasia ke dalam citra digital akan mengubah kualitas citra tersebut.

Hal ini tergantung pada ukuran *file* media penyimpanan dan ukuran *file* pesan yang disisipkan. Untuk itu ada beberapa hal atau kriteria yang harus diperhatikan dalam penyembunyian data, yaitu:

1. *Fidelity*
   Mutu citra penampung data tidak jauh berubah. Setelah terjadi penambahan pesan rahasia, *stego-data* masih terlihat dengan baik. Pengamat tidak mengetahui kalau di dalam *stego-data* tersebut terdapat pesan rahasia.
2. *Recovery*
   Data yang disembunyikan harus dapat diungkapkan kembali (*recovery*). Karena tujuan steganografi adalah penyembunyian informasi maka sewaktu-waktu pesan rahasia di dalam *stego-data* harus dapat diambil kembali untuk digunakan untuk selanjutnya.

2.3 Metode End of File

Metode *End Of File* (EOF) merupakan salah satu teknik untuk menyisipkan data pada akhir *file* dan merupakan pengembangan daripada metode *Least Significant Bit* (LSB). Teknik ini dapat digunakan untuk menyisipkan data yang ukurannya sama dengan ukuran *file* asli (sebelum disisipkan data) ditambah dengan ukuran data yang akan disisipkan ke *file* tersebut. Dalam teknik EOF, data yang disisipkan pada akhir diberi tanda khusus sebagai pengenal *start* dari data tersebut dan pengenal akhir dari data tersebut.

Teknik EOF tidak akan mengubah isi awal dari *file* yang disisipi. Sebagai contoh, jika akan menyisipkan sebuah pesan ke dalam sebuah *file* audio, maka *file* audio tidak berubah, tidak rusak dan dapat diputar seperti *file* audio asli. Ini yang menjadi salah satu keunggulan metode EOF dibandingkan dengan metode steganografi yang lain. Karena disisipkan pada akhir *file*, pesan yang disisipkan tidak bersinggungan dengan isi *file*, hal ini menyebabkan integritas data dari *file* yang disisipi tetap dapat terjaga. Namun, metode EOF akan mengubah besar ukuran *file* sesuai dengan ukuran pesan yang disisipkan ke dalam *file* awal, namun tidak mengubah format *file* dari media yang dipakai sebagai tempat penyisipan pesan tersebut. Sebagai contoh, akan disisipkan sebuah pesan berjumlah 150 karakter pada sebuah citra digital dengan dimensi 300 x 300 pixel. Maka pesan akan ditempatkan pada akhir *file*. Berikut gambar citra yang disisipi pesan dengan citra yang tidak disisipi pesan.

2.4 Representasi Citra Digital

Sebuah citra digital dapat diwakili oleh sebuah matriks yang terdiri dari *M* kolom dan *N* baris, dimana perpotongan antara kolom dan baris disebut piksel (*pixel* = *picture element*), yaitu elemen terkecil dari sebuah citra. Piksel mempunyai dua parameter, yaitu koordinat dan intensitas atau warna. Nilai yang terdapat pada koordinat (*x*, *y*) adalah *f*(*x*, *y*), yaitu besar intensitas atau warna dari piksel di titik itu. Oleh sebab itu, sebuah citra digital dapat ditulis dalam bentuk matriks sebagai berikut:

$$f(x,y) = \begin{pmatrix} f(0,0) & f(0,1) & \dots & f(0,M) \\ f(1,0) & f(1,1) & \dots & f(1,M) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ f(N\text{-}1,0) & f(N-1,1) & \dots & f(N\text{-}1, M\text{-}1) \end{pmatrix}$$

Berdasarkan gambaran tersebut, secara matematis citra digital dapat dituliskan sebagai fungsi intensitas *f*(*x*, *y*) dimana harga *x* (baris) dan *y* (kolom) merupakan koordinat posisi dan *f*(*x*, *y*) adalah nilai fungsi pada setiap titik (*x*, *y*) yang menyatakan besar intensitas citra atau tingkat keabuan atau warna dari piksel di titik tersebut. (Sutoyo dkk, 2009:20)

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089

2.5 Jenis Citra

Nilai suatu piksel memiliki nilai dalam rentang tertentu, dari nilai minimum sampai nilai maksimum. Jangkauan yang digunakan berbeda-beda tergantung dari jenis warnanya[4]. Namun, secara umum jangkauannya adalah 0 sampai 255. Berikut adalah pembagian jenis citra berdasarkan nilai pikselnya:

1. Citra Biner
   Citra biner adalah citra digital yang hanya memiliki dua kemungkinan nilai piksel, yaitu hitam dan putih. Citra biner disebut juga sebagai citra B&W (*black and white*) atau citra monokrom. Hanya dibutuhkan 1 bit untuk mewakili nilai setiap piksel dari citra biner. Citra biner sering kali muncul sebagai hasil dari proses pengolahan citra, seperti proses segmentasi dan pengambangan.
2. Citra *Grayscale*
   Citra *grayscale* merupakan citra digital yang hanya memiliki satu nilai kanal pada setiap pikselnya, dengan kata lain nilai bagian *Red* = *Green* = *Blue*. Nilai tersebut digunakan untuk menunjukkan tingkat intensitas. Warna yang dimiliki adalah warna dari hitam, keabuan dan putih. Tingkat keabuan di sini merupakan warna abu dengan berbagai tingkatan dari hitam hingga mendekati putih (256 kombinasi warna keabuan).
3. Citra Warna (24 bit)
   Setiap piksel dari citra warna 24 bit diwakili dengan 24 bit sehingga total 16.777.216 variasi warna. Variasi ini sudah lebih dari cukup untuk memvisualisasikan seluruh warna yang dapat dilihat dengan penglihatan manusia. Penglihatan manusia dipercaya dapat membedakan hingga 10 juta warna saja. Setiap poin informasi piksel (RGB) disimpan ke dalam 1 *byte* data. 8 bit pertama menyimpan nilai biru, kemudian diikuti dengan nilai hijau pada 8 bit kedua dan pada 8 bit terakhir merupakan warna merah[9].

3. ANALISIS DAN PERANCANGAN

Analisis sistem didefenisikan sebagai teknik yang digunakan untuk memahami dan membuat sfesifikasi dengan detile apa yang harus dilakukan oleh sistem. Dengan adanya analisis sistem tersebut, maka sistem yang akan dirancang diharapkan akan lebih baik dan mudah dalam pengembangan sistem ini sendiri untuk membantu pemodelan rancangan dari sistem yang akan diimplementasikan dalam bentuk nyata.

3.1 Analisis Masalah

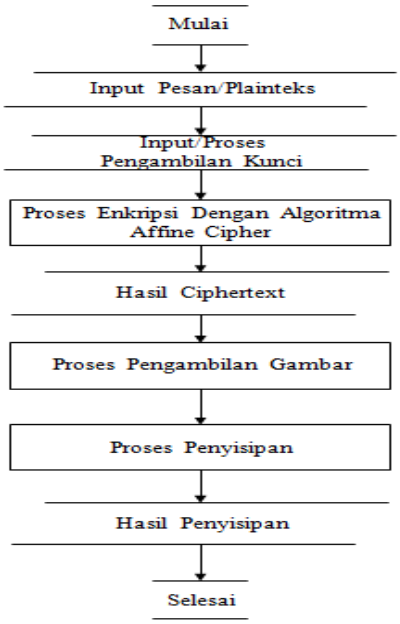
Masalah yang akan diselesaikan menggunakan sistem ini adalah masalah dalam pengamanan file citra yang berekstensi .bmp, .jpg, .jpeg, dan .png. Dalam sistem file ini, file akan diamankan menggunakan algoritma *Affine Cipher* dan metode *End Of File* (EOF), dimana file tersebut akan diamankan oleh kunci yang relatif prima. File ini mulanya akan dienkripsi menggunakan algoritma *Affine Cipher* dan file citra akan disisipkan pesan teks menggunakan metode *End Of File* (EOF).

3.2 Perancangan Sistem

Pada perancangan sistem yang dibuat menggunakan pemodelan fisik dengan membuat *flowchart program.* Bagan alir (flowchart) adalah bagan (*chart*) yang menunjukkan alir (*flow*) didalam program atau prosedur sistem secara logika. Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi.

3.2.1 Flowchart Aplikasi Sisip Teks (Enkripsi)

Adapun bentuk rancangan flowchart aplikasi proses kerja sisip teks yang penulis rancang ditunjukkan pada gambar 3 berikut.



Gambar 3. Flowchart Aplikasi Enkripsi/Penyisipan Teks

3.2.2 Flowchart Aplikasi Baca Teks (Dekripsi)

Untuk kemudian diperlukan juga aplikasi pembacaan teks yang akan disisipkan. Adapun bentuk perancangan *flowchart* aplikasi proses kerja baca teks dan

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089

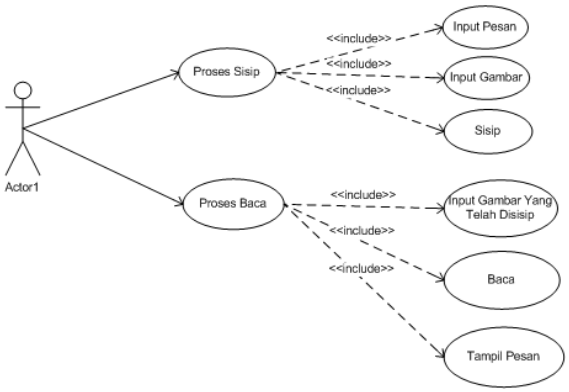
melakukan proses dekripsi pesan sehinggal pesan tersebut akan kembali ke pesan awal.

3.3. Perancangan *Requirement*

Struktur data yang digunakan penulis dalam perancangan perangkat lunak adalah *Unified Modeling Language* (UML). *Unified Modeling Language* (UML) adalah bahasa sfesifikasi standar untuk mendokumentasikan, mensfesifikasikan dan membangun sistem perangkat lunak. UML yang digunakan meliputi perancangan *Diagram Use Case.*

Use Case

Diagram Use Case digunakan untuk memberikan gambaran kebutuhan perangkat lunak secara virtual. *Use case* diagram yang memiliki *use case* proses sisip teks dan *use case* baca teks. *Use case* memilih berkas gambar oleh penerima.



Gambar 4. *Diagram Use Case*

3.4. Analisis Sistem

Pada analisa keamanan terdapat proses Enkripsi dan proses dekripsi pesan misalkan pada pesan dibawah ini, ada pun tahap-tahap proses analisanya adalah sebagai berikut :

Proses enkripsi terhadap masing-masing karakter:

Contoh plainteks : “ELVINDA”

i. Karakter ke-1 (p) = 69
E(p) = (a*p + b) mod m
E(p) = (9663* 69 + 1256) mod 256
E(p) = (666747 + 1256) mod 256
E(p) = 668003 mod 256
E(p) = 99, atau diubah ke karakter ”c”

ii. Karakter ke-2 (p) = 76
E(p) = (a*p + b) mod m
E(p) = (9663 * 76 + 1256) mod 256
E(p) = (734388 + 1256) mod 256
E(p) = 735644 mod 256
E(p) = 156, atau diubah ke karakter ”œ”

iii. Karakter ke-3 (p) = 86
E(p) = (a*p + b) mod m
E(p) = (9663*86 + 1256) mod 256
E(p) = (831018 + 1256) mod 256
E(p) = 832274 mod 256
E(p) = 18, atau diubah ke karakter ”↕ ”

iv. Karakter ke-4 (p) = 73
E(p) = (a*p + b) mod m
E(p) = (9663 * 73 + 1256) mod 256
E(p) = (705399 + 1256) mod 256
E(p) = 706655 mod 256
E(p) = 95, atau diubah ke karakter ”_”

v. Karakter ke-5 (p) = 78
E(p) = (a*p + b) mod m
E(p) = (9663 * 78 + 1256) mod 256
E(p) = (753714 + 1256) mod 256
E(p) = 754970 mod 256
E(p) = 26, atau diubah ke karakter ” → ”

vi. Karakter ke-6 (p) = 68
E(p) = (a*p + b) mod m
E(p) = (9663 * 68 + 1256) mod 256
E(p) = (657084 + 1256) mod 256
E(p) = 658340 mod 256
E(p) = 164, atau diubah ke karakter ”¤”

vii. Karakter ke-7 (p) = 65
E(p) = (a*p + b) mod m
E(p) = (9663 * 65 + 1256) mod 256
E(p) = (628095 + 1256) mod 256
E(p) = 629351 mod 256
E(p) = 103, atau diubah ke karakter ”g”

Pesan ”ELVINDA” dienkripsi menjadi ” cœ↕ _ →¤g”.

Dan pada proses tersebut dapatlah cipherteks yaitu “” cœ↕ _ →¤g”), dimana hasil cipherteks tersebut akan disisipkan kedalam citra menggunakan algoritma End Of File setelah selesai proses berikutnya adalah proses dekripsi.

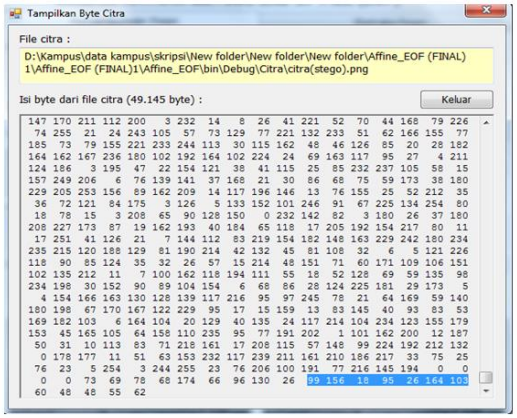
IMPLEMENTASI PROGRAM

Adapun beberapa langkah-langkah dalam mengimplementasi aplikasi yang telah dirancang diantaranya meliputi :

1. **FormViewByte**

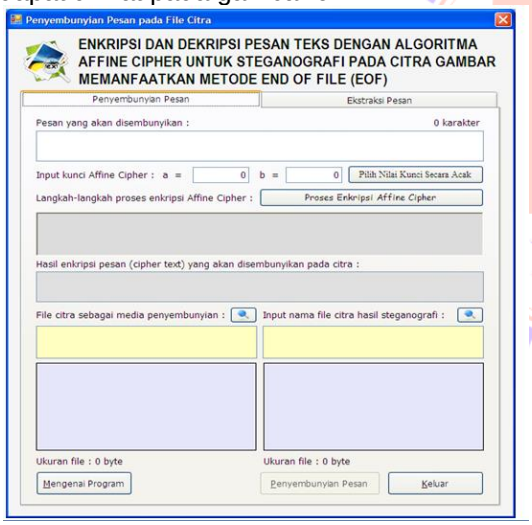
Form ini berfungsi untuk menampilkan byte yang terdapat dalam citra. Rancangan *form* dapat dilihat pada gambar 5.

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089



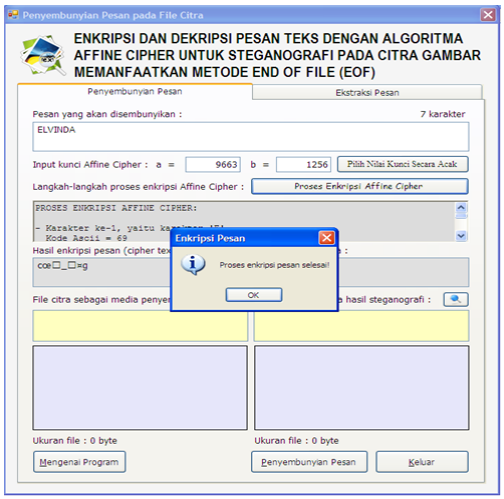
Gambar 5. Tampilan Byte Pada Gambar

Berikut akan dibahas hasil implementasi dari aplikasi enkripsi dan dekripsi pesan teks dengan algoritma *Affine Cipher* untuk steganografi pada citra gambar memanfaatkan metode *End of File* (*EOF*). Saat aplikasi dijalankan, *form* Utama akan muncul. *Form* utama terdiri dari dua buah tab, yaitu *tab* "Penyembunyian Pesan" dan *tab* "Ekstraksi Pesan". Tampilan tab "Penyembunyian Pesan" pada *form* Utama, dapat dilihat pada gambar 6.



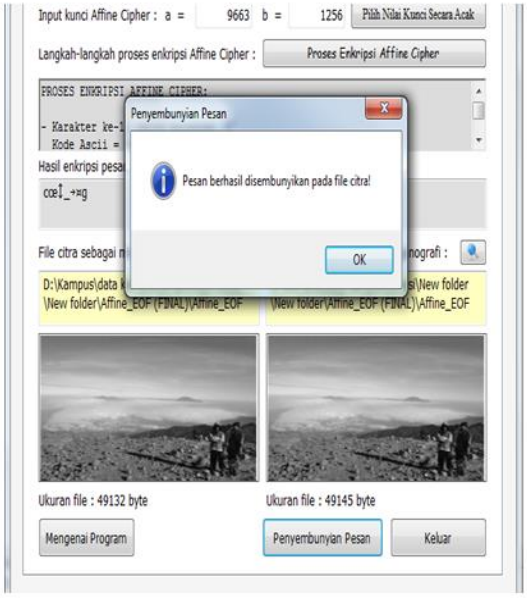
Gambar 6. Tampilan *Form* Utama (*Tab* Penyembunyian Pesan)

Tab "Penyembunyian Pesan" berfungsi untuk menyembunyikan pesan rahasia pada *file* citra. Sebelum pesan disembunyikan dengan metode EOF, pesan dienkripsi terlebih dahulu dengan kunci tertentu menggunakan algoritma *Affine Cipher*. Misalkan, pesan yang dimasukkan adalah "ELVINDA", dan kunci yang digunakan adalah *a* = 9663 dan *b* = 1256. Tekan tombol "Proses Enkripsi *Affine Cipher*", dan hasil enkripsi akan ditampilkan seperti terlihat pada gambar 7.



Gambar 7. Hasil Enkripsi Pesan

Pilih *file* citra sebagai media penyembunyian pesan dan tekan tombol "Penyembunyian Pesan", maka pesan terenkripsi akan disembunyikan pada citra seperti terlihat pada gambar 8.



Gambar 8. Hasil Enkripsi Pesan

KESIMPULAN DAN SARAN

Adapun kesimpulan dan saran pada penelitian diatas, diantaranya adalah berupa kelebihan-kelebihan dari program dan kelemahan dari program tersebut.

5.1. Kesimpulan

Setelah menyelesaikan perancangan aplikasi aplikasi enkripsi dan dekripsi pesan teks dengan algoritma *Affine Cipher* untuk steganografi pada citra gambar memanfaatkan metode *End of File* (*EOF*), beberapa hal yang dapat disimpulkan adalah sebagai berikut:

MEANS (Media Informasi Analisa dan Sistem)
p-ISSN : 2548-6985, e-ISSN : 2599-3089

1. Aplikasi dapat digunakan untuk mengenkripsi dan mendekripsi pesan menggunakan metode *Affine Cipher*, yaitu dengan melakukan perhitungan proses enkripsi dan dekripsi dengan rumus *Affine Cipher* terhadap setiap karakter pesan.
2. Aplikasi dapat digunakan untuk menyembunyikan pesan teks pada *file* citra dengan menggunakan metode EOF, yaitu dengan menyisipkan 1 *byte* karakter EOF dan pesan di akhir *file* citra, disertai dengan 5 *byte* penanda berupa panjang pesan yang disembunyikan.
3. Aplikasi dibangun dengan menggunakan bahasa pemrograman Microsoft Visual Basic .NET 2008, dan dirancang dapat dapat mengenkripsi pesan dengan menggunakan algoritma kriptografi *Affine Cipher* dan menyembunyikannya pada citra dengan menggunakan metode steganografi EOF.
4. *File* citra hasil steganografi memiliki jumlah piksel yang sama dengan *file* citra asli dan tidak terdapat perbedaan warna antara keduanya.
5. Ukuran *file* citra hasil steganografi akan bertambah, sesuai dengan panjang pesan yang disembunyikan ditambah dengan 6 *byte*, berupa 1 *byte* EOF dan 5 *byte* panjang pesan.
6. Apabila ada pihak yang mengetahui cara kerja teknik EOF, maka pesan yang berhasil diekstraksi adalah pesan terenkripsi. Tanpa mengetahui kunci yang digunakan pada saat penyembunyian pesan, maka pesan rahasia tidak dapat didekripsi dengan benar dan tidak dapat dibaca.

5.2. Saran

Untuk memperbaiki dan mengatasi kelemahan yang masih ada, maka beberapa hal yang dapat disarankan sebagai berikut:

1. Disarankan untuk menambahkan proses kompresi terhadap pesan sehingga dapat memperkecil ukuran pesan yang dapat menambah ukuran *file* asli.
2. Disarankan untuk menambahkan metode kriptografi yang lebih kuat daripada metode *Affine Cipher*, seperti metode AES atau *Serpent Cipher*.
3. Untuk perkembangan aplikasi ini dapat disarankan menggunakan bahasa pemograman java, android maupun aplikasi yang terupdate.

DAFTAR PUSTAKA

1. Anggraini, dkk, 2014, Penerapan Steganografi Metode *End of File* (EOF) dan Enkripsi Metode *Data Encryption Standard* (DES) pada Aplikasi Pengamanan Data Gambar, Universitas Budi Luhur, Jakarta.
2. Arini, G.M dan Widyawan, T.I, 2011, Pengamanan Pesan Steganografi dengan Metode LSB Berlapis Enkripsi, Budi Luhur, Jakarta Selatan.
3. Ariyus, D., 2008, Pengantar Ilmu Kriptografi, Penerbit Andi, Yogyakarta.
4. Basuki, dkk, 2005, Pengolahan Citra Digital Menggunakan Visual Basic, Penerbit Graha Ilmu, Yogyakarta.
5. Gunawan, 2013, Implementasi *Hidden Message* pada Citra Menggunakan Metode *End of File*, Universitas Widyatama, Bandung.
6. Jogiyanto H.M., *Analisa Dan Desain Sistem Informasi,* Penerbit Andi Yogyakarta, Yogyakarta 2005
7. Munir, R., 2006, Kriptografi, Penerbit Informatika, Bandung.
8. Munir, R., 2012, Matematika Diskrit, Penerbit Informatika, Bandung.
9. Putra, D., 2010, Pengolahan Citra Digital, Penerbit Andi, Yogyakarta.
10. Sadikin, R., 2012, Kriptografi untuk Keamanan Jaringan, Penerbit Andi, Yogyakarta.