Peningkatan Keamanan Pesan Dengan Kriptografi RC4 dan Steganografi LSB Pada File JPEG

Rahmat Sulaiman [1)], Burham Isnanto [2)]
Dosen STMIK Atma Luhur[1,2]
e-mail: rahmatsulaiman@atmaluhur.ac.id [1)], burham@atmaluhur.ac.id [2)]

Abstrak

Dalam suatu organisasi, keamanan pesan maupun keamanan data merupakan suatu hal yang sangat penting. Masalah integritas dan keamanan pesan adalah hal yang harus diperhatikan. Upaya menjaga informasi agar tidak jatuh ke pihak yang tidak seharusnya atau pihak yang tidak berkepentingan dituntut adanya penerapan suaku mekanisme yang baik dalam mengamankan pesan. Ilmu yang berkaitan untuk menjaga keamanan pesan adalah kriptografi, umumnya kriptorafi terbagi menjadi dua jenis, yaitu kriptografi simetris dan kriptografi asimetris. Selain itu untuk menjaga kerahasiaan pesan ilmu yang digunakan adalah steganogragi. Dalam penelitian ini penulis mengkombinasikan antara algoritma RC4 dengan metode LSB dengan menggunakan Visual Studio 2008. Hasilnya berupa waktu proses enkripsi dan dekripsi serta perubahan ukuran gambar setelah dienkripsi. Lamanya waktu proses enkripsi dan dekripsi sangat bergantung pada banyaknya jumlah karakter, lamanya waktu proses enkripsi dan dekripsi berbanding lurus dengan banyaknya jumlah karakter yang digunakan. Perubahan ukuran file setelah disisipkan pesan, mengalami perubahan walaupun tidak significant. Terlihat perubahan file hanya 1 KB tapi sesungguhnya jika dilihat dalam ukuran bytes gambar, gambar yang sudah dienkripsi mengalami perubahan sekitar 200 bytes.

Kata kunci: *Kriptografi, Steganografi, RC4, LSB*

1. Pendahuluan

Era globalisasi dan teknologi informasi berkembang sangat pesat, hal ini memungkinkan keamanan pesan atau data menjadi hal yang sanagt rawan. Banyak pihak yang dapat melihat bahkan merubah serta merusak data merupakan hal yang harus diperhatikan. Sehinggal hal ini membuat dibutuhkannya suatu system keamanan yang dapat menjaga keamanan pesan atau data serta menjaga kerahasiannya [1,2]. Tanpa adanya jaminan keamanan, orang lain dapat dengan mudah mendapatkan pesan atau informasi yang dikirimkan melalui jaringan/internet.

Kriptografi dan steganografi merupakan ilmu atau seni yang mempelajari pengamanan pesan atau data dan kerahasiaan pesan atau data. Dalam kriptografi, terdapat 2 proses utama, enkripsi dan dekripsi. Enkripsi adalah proses penyandian pesan asli atau *plainteks* menjadi *cipherteks* (teks tersandi). Sedangkan dekripsi adalah proses penyandian kembali *cipherteks* menjadi *plainteks.*

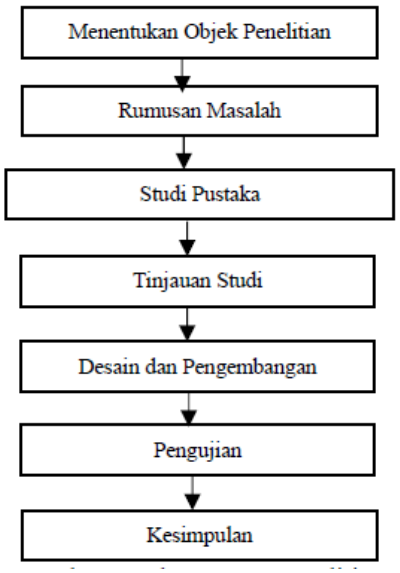
Salah satu metode *enkripsi* yang terkenal adalah metode RC4.RC4 pertama kali dibuat oleh Ron Rivest di Laboraturium RSA pada tahun 1987. Awalnya RC4 adalah sebuah rahasia dagang, akan tapi pada September1994, kode tersebut dikirim oleh seseorang yang tidak diketahui ke milist Chypermunks dan menyebar ke banyak situs internet. Kode yang bocor tersebut akhirnya dikonfirmasi sebagai RC4 karena memiliki output yang sama dengan *software* dengan *license* RC4 di dalamnya. Karena algoritma sudah diketahui, RC4 tidak lagi menjadi rahasia dagang. Nama "RC4" sekarang adalah sebuah merek dagang, namun sering disebut sebagai "*ARCFOUR*" atau "ARC4" (artinya diduga RC4, karena algoritma ini tidak pernah dirilis secara resmi oleh RSA), untuk menghindari kemungkinan masalah tentang merek dagang [3, 5]. Teknik lain yang dapat digunakan yaitu steganografi. Steganografi adalah seni dan ilmu untuk menyembunyikan pesan rahasia di dalam pesan lain sehingga keberadaan pesan rahasia tersebut tidak dapat diketahui. Berbeda dengan kriptografi yang merahasiakan makna pesan namun keberadaan pesan tetap ada, steganografi merahasiakan dengan menutupi atau menyembunyikan pesan. Implementasi steganografi saat ini telah menggunakan media digital sebagai media penampung atau penyembunyi pesan, salah satunya media gambar (citra digital). Salah satu metode steganografi citra digital adalah Least Significant Bit (LSB), dengan teknik penyembunyian pesan pada lokasi bit terendah dalam citra digital. Pesan dikonversi ke dalam bentuk bit biner dan disembunyikan pada citra digital dengan metode LSB. Implementasi metode LSB tanpa dilengkapi dengan sistem keamanan berpeluang untuk dapat dibongkar dengan mudah melalui teknik pemecahan frekuensi dengan membaca bit terendah [4].

Berdasarkan uraian di atas, penulis berinisiatif untuk mengangkat judul penelitian yaitu "Peningkatan Keamanan Pesan Dengan Kriptografi RSA dan Steganografi LSB Pada File JPEG". Dari latar belakang diatas, batasan masalah dalam penelitian ini adalah perancangan program Enkripsi pesan atau data yang kemudian di masukan kedalam gambar JPEG menggunakan metode LSB Steganografi dengan menggunakan visual studio 2008.

Penelitian ini bertujuan untuk mengetahui perubahan ukuran file gambar JPEG dan juga kecepatan waktu proses enkripsi dan dekripsi yang menggunakan algoritma RC4, kemudian akan dilihat perubahan ukuran file gambar JPEG sebelum dan sesudah enkripsi dan juga kecepatan waktu proses enkripsi dan kecepatan waktu proses dekripsi.

2. Metode Penelitian

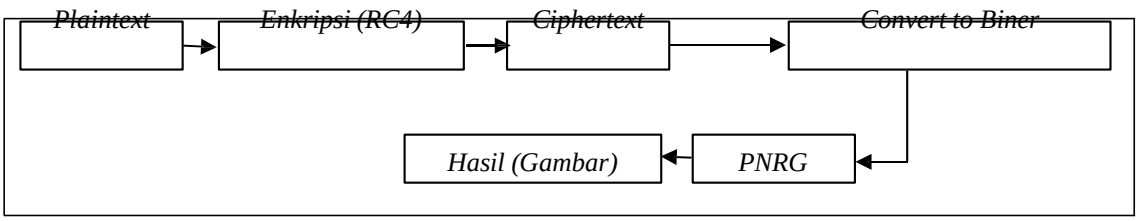
Untuk mencapai tujuan yang diharapkan, berikut ini merupakan bagian alur langkah-langkah dalam penelitian ini.



Dalam penelitian ini ada beberapa langkah-langkah atau tahapan-tahapan yang dilakukan penulis. Secara umum, tahapan-tahapan yang ada pada penelitian ini antara lain :

1. Identifikasi Masalah
2. Studi Pustaka dan Pengumpulan Data
3. Hipotesa
4. Perancangan enkripsi dengan Algoritma RC4
5. Perancangan penyisipan pesan pada gambar dengan metode LSB
6. Penggabungan enkripsi dengan algoritma RC4 pada penyisipan pesan pada gambar dengan metode LSB

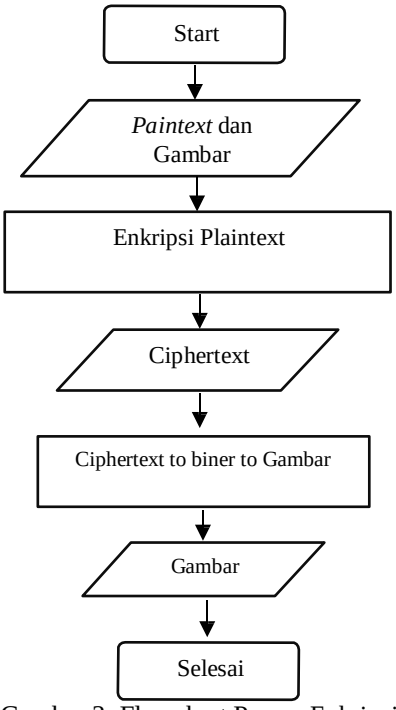
Penelitian ini menggunakan sistem kombinasi antara enkripsi dengan algoritma RC4 dengan metode LSB. Adapun rancangan sistem kombinasi yang dibuat adalah sebagai berikut:



Gambar 2. Alur Proses Sistem Kombinasi

1. Pesan (*Plaintext*) dimasukan dan kemudian dienkripsi menggunakan algoritma RC4
2. Hasil enkripsi (*Ciphertext*) kemudian dikonversikan ke dalam bentuk biner

3. *Ciphertext* yang sudah dikonversikan kedalam bentuk biner kemudia dimasukan kedalam bit-bit terendah pada Gambar dengan *Pseudo Number Random Generator*
4. Hasilnya berupa gambar yang sudah disisipkan pesan yang terenkripsi

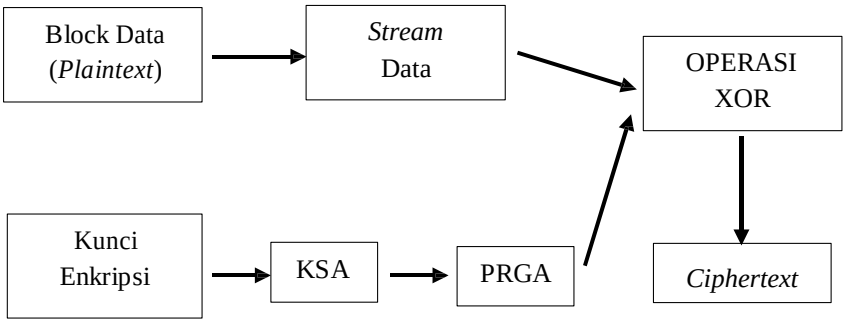


Gambar 3. Flowchart Proses Enkripsi

2.1. Rivest Code 4

RC4 merupakan salah satu algoritma kunci simetris yang berbentuk stream cipher, yaitu memproses unit atau input data pada satu saat. Unit atau data pada umumnya sebuah byte atau kadang-kadang bit. Algoritma ini tidak harus menunggu sejumlah input data tertentu sebelum diproses, atau menambahkan byte tambahan untuk mengenkripsi. Algoritma ini ditemukan pada tahun 1987 oleh Ronald Rivest dan menjadi simbol keamanan RSA. RC4 merupakan enkripsi stream simetrik proprietary yang dibuat oleh RSA *Data Security Inc* (RSADSI).

Penelitian ini menggunakan Kriptografi dengan algoritma RC4 untuk proses enkripsi pesan yang kemudian dikombinasikan dengan Steganografi dengan metode *Least Significant Bit* (LSB) untuk menyisipkan pesan dalam bentuk biner pada bit terkecil pada file dengan format JPEG.



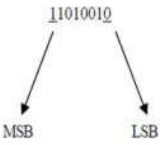
Gambar 4. Diagram Enkripsi RC4

KSA = *Key Scheduling Algorithm*
PRGA = *Pseudo Random Generation Algorithm*

2.2. Least Significant Bit (LSB)

Untuk menjelaskan teknik penyembunyian Least Significant Bit yang dipakai ini kita menggunakan citra digital sebagai *covertext*. Setiap *pixel* yang ada di dalam file citra berukuran 1 sampai

3 *byte*. Pada susunan bit dalam setiap *byte* (1 *byte* = 8 bit) ada bit yang paling berarti (*most significant bit*
atau MSB) dan bit yang paling kurang berarti (*least significant bit* atau LSB)



Gambar 5. MSB dan LSB

Dari contoh *byte* 11010010 pada gambar 2.2 diatas bit 1 pertama yang (di garis bawahi) adalah bit MSB dan bit 0 terakhir yang digaris bawahi adalah bit LSB. Bit yang cocok untuk diganti dengan bit pesan adalah bit LSB, karena modifikasi hanya mengubah nilai *byte* tersebut satu lebih tinggi atau satu lebih rendah dari nilai sebelumnya. Misalkan *byte* tersebut di dalam gambar memberikan persepsi warna merah, maka perubahan satu bit LSB hanya mengubah persepsi merah tidak terlalu berarti karena mata manusia tidak dapat membedakan perubahan sekecil ini.

Sebagai ilustrsi misalkan *cover-object* adalah citra sekempulan citra berwarna merah seperti yang terlihat pada contoh di bawah ini:

00110011 10100010 11100010 0110111

Jika pesan yang sudah dikonversikan kedalam bentuk biner, *embedded message*adalah 0110. Setiap bit yang ada pada *watermark* akan menggantikan posisi dari LSB (*Least Significant Bit*) dari segmen *pixel-pixel* citra yang ada menjadi:

00110010 10100011 11100011 01101110

Untuk dapat membuat *hiddentext* tidak dapat dilacak, bit-bit pesan tidak mengganti *byte-byte* yang berurutan, namun dipilih susunan *byte* secara acak. Misalnya jika terdapat 50 *byte* dan 6 bit data yang akan disembunyikan, maka *byte* yang diganti bit *LSB-nya* dipilih secara acak, misalkan *byte* nomor 36, 5, 21, 10, 18, 49. Pembangkitan bilangan acak dilakukan dengan *pseudo-random-number-generator* (*PRNG*) yang berlaku sebagai kunci.

Pada citra 8-bit yang berukuran 256 x 256 *pixel* terdapat 65536 *pixe*l, setiap *pixel* berukuran 1 *byte* sehingga kita hanya dapat menyisipkan 1 bit pada setiap *pixel*. Pada citra 24-bit yang berukuran 256 x 256 *pixel*, satu *pixel* berukuran 3 *byte* (atau 1 *byte* untuk setiap komponen R, G, B), sehingga kita bisa menyisipkan pesan sebanyak 65536 x 3 bit = 196608 bit atau 196608/8 = 24576 *byte*.

Pesan yang disembunyikan di dalam citra dapat diungkap kembali dengan mengekstraksinya. Posisi *byte* yang menyimpan bit pesan dapat diketahui dari bilangan acak yang dibangkitkan oleh *PRNG*. Jika kunci yang digunakan pada waktu ekstraksi sama dengan kunci pada waktu penyisipan, maka bilangan acak yang dibangkitkan juga sama. Dengan demikian, bit-bit data rahasia yang bertaburan di dalam citra dapat dikumpulkan kembali.

3. Hasil dan Pembahasan

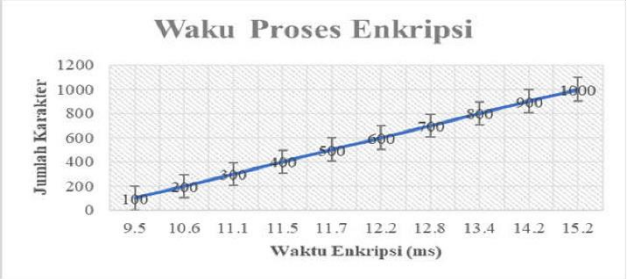
Pada bagian ini penulis menampilkan hasil dari pengujian dari proses enkripsi dan dekripsi antara algoritma RC4 yang dikombinasikan pada metode LSB steganografi. Hasil pengujian dilihat dari segi perubahan file JPEG dan juga waktu proses enkripsi dan waktu proses dekripsi. Untuk hasil pengujian dari segi waktu proses digunakan gambar yang sama dengan jumlah karakter yang berbeda ketika disisipkan dalam gambar tersebut. Pengujian dilakukan antara 5 – 10 kali untuk mendapatkan hasil yang stabil, dan juga pengujian dilakukan dengan menggunakan karakter kelipatan 100 (dari 100 karakter hingga 1000 karakter). Sedangkan untuk pengujian dari segi perubahan file JPEG dilihat dari ukuran gambar asli dibandingkan dengan ukuran gambar sesudah disisipkan pesan yang terenkripsi berdasarkan penyisipan kelipatan 100 karakter diatas.

Berikut adalah table dan grafik hasil dari proses waktu enkripsi menggunakan algoritma RC4 yang dikombinasikan pada LSB

Tabel 1. Tabel Waktu Enkripsi

Jumlah Karakter	Waktu Enkripsi (ms)
100	5.6
200	6.1
300	6.7
400	7.3
500	7.9
600	8.4
700	9
800	9.6
900	10.1
1000	10.6

Jika digambarkan dalam grafik, waktu proses enkripsi yang dibutuhkan oleh aplikasi yang dibangun adalah sebagai berikiut:



Gambar 6. Grafik Waktu Proses Enkripsi

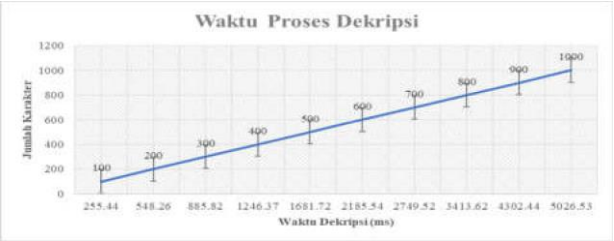
Hasil grafik diatas menggambarkan waktu yang dibutuhkan untuk proses enkripsi menggunakan algoritma RC4 yang dikombinasikan pada metode LSB. Pada sumbu y menyatakan banyaknya jumlah karakter yang dienkripsi dan sumbu x menyatakan lamanya proses enkripsi dalam satuan *millisecond*. Pengujian dilakukan sebanyak 10 kali untuk tiap jumlah karakter pada tiap gambar yang digunakan, nilai yang dimasukan kedalam grafik merupakan nilai rata-rata. Hal ini dilakukan untuk mendapatkan waktu yang konsisten, mengingat kinerja prosesor yang tidak stabil selama proses pengukuran waktu.

Selain pengujian dalam bentuk waktu proses enkripsi, penelitian ini juga menguji waktu proses dekripsinya. Berikut adalah table dan grafik hasil dari proses waktu dekripsi menggunakan algoritma RC4 yang dikombinasikan pada LSB:

Tabel2. Tabel Waktu Dekripsi

Jumlah Karakter	Waktu Dekripsi (ms)
100	255.44
200	548.26
300	885.82
400	1246.37
500	1681.72
600	2185.54
700	2749.52
800	3413.62
900	4302.44
1000	5026.53

Jika digambarkan dalam grafik, waktu proses enkripsi yang dibutuhkan oleh aplikasi yang dibangun adalah sebagai berikiut:



Gambar 7. Grafik Waktu Proses Dekripsi

Hasil grafik diatas menggambarkan waktu yang dibutuhkan untuk proses dekripsi menggunakan algoritma RC4 yang dikombinasikan pada metode LSB. Pada sumbu y menyatakan banyaknya jumlah karakter yang didekripsi dan sumbu x menyatakan lamanya proses dekripsi dalam satuan *millisecond*.

Tabel 3. Tabel Perubahan File Sebelum dan Sesudah Enkripsi

Ukuran File Sebelum Enkripsi	Jumlah Karakter	Ukuran File Sebelum Enkripsi
799 KB (819.115 bytes)	100	800 KB (819.328 bytes)
	200	800 KB (819.528 bytes)
	300	800 KB (819.728 bytes)
	400	800 KB (819.928 bytes)
	500	800 KB (820.128 bytes)
	600	801 KB (820.328 bytes)
	700	801 KB (820.528 bytes)
	800	801 KB (820.728 bytes)
	900	801 KB (820.928 bytes)
	1000	801 KB (821.128 bytes)

Untuk proses enkripsi semua gambar yang digunakan adalah sama, dengan tujuan agar nanti mudah membedakan antara file gambar yang sudah dienkripsi dan yang belum dienkripsi. Semua file gambar yang digunakan adalah gambar dengan format JPEG.

4. Simpulan

Berikut adalah beberapa kesimpulan yang dapat ditarik berdasarkan penelitian ini adalah:

1. Lamanya waktu proses enkripsi dan dekripsi sangat bergantung pada banyaknya jumlah karakter, lamanya waktu proses enkripsi dan dekripsi berbanding lurus dengan banyaknya jumlah karakter yang digunakan.
2. Perubahan ukuran file setelah disisipkan pesan, mengalami perubahan walaupun tidak significant. Terlihat perubahan file hanya 1 KB tapi sesungguhnya jika dilihat dalam ukuran bytes gambar, gambar yang sudah dienkripsi mengalami perubahan sekitar 200 bytes

Daftar Pustaka

[1] Basri. Kriptografi Simetris dan Asimetris dalam Perspektif Keamanan Data dan Kompleksitas Komputasi. *Jurnal Ilmiah Ilmu Komputer*. 2016; vol(2).

[2] Kurniawan A., Yuliana. M, Samsono .M.Zen., Hadi. Analisis dan Implementasi Sistem Keamanan Data dengan Menggunakan Metode Enkripsi Algoritma RC-5. Surabaya : Jurusan Teknologi Telekomunikasi Politeknik Elektonika Negeri Surabay

[3] Prasetyo. Enkripsi Menggunakan Algoritma RSA, Program Studi Teknik Informatika, Insitut Teknologi Bandung

[4] Handa, G., *Penggunaan Metode Steganografi dan Kriptografi Dengan Algoritma RC4 Stream Cipher Dalam Pengaanan Citra Digital Perusahaan*, Mahasiswa Pasca Sarjanan, Universitas Budi Luhur

[5] Cheddad, A., Joan, C., Curran, K. & Paul, M.K., *Digital image steganography: Survey and analysis of current methods Signal Processing 90,* 2010

[6] N.P, Indah, F.A. & Awang, H.K., *Jurnal Implementasi Kriptografi Pengamanan Data Pada File Teks, Isi File Dokumen dan File Dokumen Menggunakan Algoritma Advanced Encryption Standard.* Jurnal Informatika Mulawarman Vol. 10 No.1, 2015.