

## IMPLEMENTASI SYSTEM SANDI *STREAM CIPHER* UNTUK PENGAMANAN DATA *IMAGE*

**Emy Setyaningsih**

*Program Studi Sistem Komputer Institut Sains & Teknologi AKPRIND Yogyakarta*

*Jln. Kalisahak No 28 Komplek Balapan, Yogyakarta, Telp. +62-274-563029,*

*E-mail: [emypurnomo@akprind.ac.id](mailto:emypurnomo@akprind.ac.id)*

### **Abstrak**

Makalah ini bertujuan untuk merancang sistem penyandian *image* yang dikembangkan dari algoritma kriptografi kunci simetri berbasis *stream cipher*, sehingga akan didapatkan *cipher* yang lebih kuat sehingga tidak mudah untuk dipecahkan. Algoritma yang dikembangkan adalah penggabungan antara teknik algoritma RC4 yang mewakili kriptografi modern yang digabungkan dengan teknik *Vigenere cipher* yang mewakili kriptografi klasik. Dari hasil pengujian terhadap *image* dengan ukuran, dan tingkat kedetilan yang berbeda berhasil didapatkan rata-rata nilai entropinya adalah 7.99721 serta rata-rata nilai korelasi antara *image* asli dengan *image* hasil enkripsi adalah 0.00042. Hal ini menunjukkan bahwa sistem enkripsi yang diusulkan aman dari serangan entropi serta sesuai dengan sistem keamanan yang baik karena nilai korelasinya mendekati nol. Dari histogram *plain image* dan *cipher image* terlihat perbedaan yang signifikan antara keduanya, pada histogram *cipher image* terlihat merata intensitas warnanya, hal ini menunjukkan bahwa algoritma ini cukup tangguh untuk bisa dibobol oleh kriptanalisis menggunakan metode *statistical attack*. Algoritma enkripsi *image* yang diusulkan juga cukup efektif karena tidak membutuhkan waktu yang lama untuk proses enkripsi maupun dekripsi.

**Kata Kunci:** *Stream Cipher*, RC4, *Vigenere Cipher*, Kriptografi.

### **Abstract**

*The aim of this study is to design an image encryption system which is developed from the cryptographic algorithms based symmetric key stream cipher, so it will be a strong not-easy-to-solve cipher. The algorithm that is developed is the combination of modern cryptography that is represented by the RC4 algorithm and classical cryptography that is Vigenere cipher. The results of the image test with different size, and a different level of detail shows a successful work. The average of entropy value is 7.99721, and the average of the correlation between the original image with the image encryption result is 0.00042. This indicates that the encryption system is safe from the entropy's attack and appropriate with the perfect security, because the correlation's value is close to zero. The plain image and cipher image histogram shows a significant differences between them, the cipher image histogram, the color intensity spreads evenly, it indicates that the algorithm is strong enough to be broken by cryptanalysis by using statistical attack methods. The suggested encryption algorithm image is also quite effective, because it takes a short time for the encryption and decryption process.*

**Key words:** *Stream Cipher*, RC4, *Vigenere Cipher*, cryptography.

## PENDAHULUAN

Aplikasi *secure image message* ini dikembangkan sejalan dengan meningkatnya pemanfaatan data dan informasi yang berupa *image* di berbagai bidang seperti perbankan, kedokteran, departemen pertahanan, perusahaan, pendidikan, dan lain sebagainya. Seiring dengan itu tuntutan akan keamanan terhadap kerahasiaan data dan informasi khususnya dalam bentuk *image* yang saling dipertukarkan tersebut juga semakin meningkat. Keamanan pada suatu informasi atau data pada saat ini dapat dibagi menjadi dua, yakni: kriptografi dan steganografi. Steganografi adalah suatu seni untuk menyembunyikan suatu data, di mana data tersebut disembunyikan ke dalam suatu media *image* yang tampak biasa saja sehingga tidak akan menimbulkan banyak perhatian dari pihak yang tidak dikehendaki [4]. Berbeda dengan steganografi, kriptografi adalah suatu seni untuk mengacak informasi atau data yang memiliki arti menjadi sesuatu yang tidak dapat dimengerti atau seakan-akan tidak berarti sehingga pesan yang dikirim pengirim dapat disampaikan kepada penerima dengan aman [8]. Kriptografi melingkupi proses transformasi informasi yang berlangsung dua arah. Proses transformasi tersebut terdiri dari proses enkripsi dan dekripsi [2].

Keamanan dari sebuah algoritma kriptografi diukur dari banyaknya kerja yang dibutuhkan untuk memecahkan *chiphertext* menjadi *plaintext*-nya tanpa mengetahui kunci yang digunakan. Kerja ini dapat diekivalenkan dengan waktu, memori, uang, dan lain-lain. Semakin banyak kerja yang diperlukan, yang berarti juga semakin lama waktu yang dibutuhkan, maka semakin kuat algoritma kriptografi tersebut, yang berarti semakin aman digunakan untuk menyandikan data.

Pada makalah ini akan difokuskan pada penyandian *image* yang menggunakan algoritma kriptografi kunci simetri berbasis *stream cipher*. Beberapa penelitian yang menggunakan algoritma kriptografi kunci simetri berbasis *stream cipher* telah dilakukan oleh beberapa peneliti untuk mendapatkan algoritma yang handal untuk mengamankan data [6] [7] [9]. Ketiga penelitian tersebut menggunakan algoritma kriptografi kunci simetri menggunakan basis *stream cipher*. Kekuatan dari RC4 adalah dari sisi kecepatan, meskipun adanya beberapa kelemahan yaitu terlalu tingginya kemungkinan terjadi tabel *S-box* yang sama, hal ini terjadi karena kunci *user* diulang-ulang untuk mengisi 256 bytes, sehingga 'aaaa' dan 'aaaaa' akan menghasilkan permutasi yang sama. Demikian juga pada algoritma *Vigenere cipher* apabila kuncinya terlalu pendek juga akan mudah di pecahkan menggunakan metode Kasisiki. Untuk itu pada makalah ini akan dilakukan analisis dan uji coba algoritma kriptografi kunci simetri berbasis *stream cipher* yang diterapkan pada algoritma RC4 yang digabungkan dengan algoritma *Vigenere cipher* dengan memodifikasi proses pembangkitan kunci untuk meningkatkan keamanannya.

## SISTEMATIKA

### A. Algoritma RC4

RC4 adalah *cipher* aliran yang digunakan secara luas pada sistem keamanan seperti protokol SSL (*Secure Socket Layer*). Algoritma kriptografi ini sederhana dan mudah diimplementasikan. RC4 dibuat oleh Ron Rivest dari Laboratorium RSA (RC adalah singkatan dari Ron's Code). RC4 membangkitkan aliran kunci (*keystream*) yang kemudian di-XOR-kan dengan *plaintext* pada waktu enkripsi (atau di-XOR-kan dengan *bit-bit ciphertext* pada waktu dekripsi). RC4 tidak seperti *cipher* aliran yang memproses data dalam *bit*, RC4 memproses data dalam ukuran *byte* (1 *byte* = 8 *bit*). RC4 menggunakan dua buah kotak substitusi (*S-box*) array 256 *byte* yang berisi permutasi dari bilangan 0 sampai 255 dan *S-box* kedua yang berisi permutasi fungsi dari kunci sepanjang variabel. Langkah algoritma enkripsi RC4 [3] yang diilustrasikan pada Gambar 3(a) adalah sebagai berikut:

- 1) Inisialisasi array *S-box* pertama,  $S[0], S[1], \dots, S[255]$ , diisi dengan bilangan 0 sampai 255, sehingga array *S-box* array *S* berbentuk  $S[0] = 0, S[1] = 1, \dots, S[255] = 255$ .  
For  $r = 0$  to 255  
 $S[r] = r$
- 2) Inisialisasi array kunci (*S-box* lain), misal array kunci *K* dengan panjang 256. Jika panjang kunci  $K < 256$ , maka dilakukan *padding* yaitu penambahan *byte* semua sehingga panjang kunci menjadi 256 *byte*. Misalnya  $K = \text{"abc"}$  yang hanya terdiri 3 *byte* (3 huruf), maka lakukan *padding* dengan

penambahan *byte* (huruf) semu, misalnya  $K = \text{"abcabcabc..."}$  sampai panjang  $K$  mencapai 256 *byte*, sehingga *S-box Array* kunci  $K$  berbentuk  $K[0], K[1], \dots, K[255]$ .

for  $i = 0$  to 255

$K[i] = \text{Kunci}[i \bmod \text{length}]$ ;

- 3) Permutasi terhadap nilai-nilai di dalam *array*  $S$  dengan cara menukarkan isi *array*  $S[i]$  dengan  $S[j]$ , prosesnya adalah sebagai berikut:

$j = 0$

For  $i = 0$  to 255

$j = (j + S[i] + K[i]) \bmod 256$

isi  $S[i]$  dan isi  $S[j]$  ditukar

- 4) Membangkitkan aliran kunci (*keystream*) selanjutnya digunakan untuk enkripsi.

$i = j = 0$

$i = (i + 1) \bmod 256$

$j = (j + S[i]) \bmod 256$

isi  $S[i]$  dan  $S[j]$  ditukar

$t = (S[i] + S[j]) \bmod 256$

$K = S[t]$ ;

Proses pembangkitan aliran kunci  $K$  dipilih dengan mengambil nilai  $S[i]$  dan  $S[j]$  dan menjumlahkannya dalam *modulo* 256. Hasil penjumlahan adalah nilai indeks  $t$  sedemikian sehingga  $S[t]$  menjadi kunci aliran  $K$ .

- 5) Kunci aliran  $K$  kemudian digunakan untuk mengenkripsi *plaintext* ke-idx sehingga didapatkan *ciphertext*, sedangkan untuk mendapatkan *plaintext* dengan cara *ciphertext* di-XOR-kan dengan kunci yang sama dengan proses enkripsi.

## B. Algoritma Vigenere Cipher

*Vigenere Cipher* adalah algoritma substitusi jamak (*polyalphabetical substitution cipher*) dimana suatu huruf *plaintext* tidak selalu disubstitusi menjadi huruf yang sama, namun disubstitusi berdasarkan kunci yang digunakan [10]. Secara matematis, misalkan kunci  $K$  dengan panjang  $m$  adalah rangkaian huruf-huruf  $K = k_1 \dots k_m$  dimana  $k_i$  didapat dari banyak penggeseran pada alfabet ke- $i$ , *plaintext* adalah rangkaian  $p_1, p_2, \dots, p_m$ , dan *ciphertext* adalah rangkaian  $c_1, c_2, \dots, c_m$  dapat dinyatakan dengan formula [10]. Misalkan  $m$  menentukan beberapa nilai integer positif dimana  $P = C = K = (Z_{26})^m$ , untuk sebuah kunci  $K = (k_1, k_2, \dots, k_m)$ , kita definisikan :

$$e_K(c_1, c_2, \dots, c_m) = (p_1 + k_1, p_2 + k_2, \dots, p_m + k_m) \bmod 26 \quad (1)$$

$$\text{dan } d_K(p_1, p_2, \dots, p_m) = (c_1 - k_1, c_2 - k_2, \dots, c_m - k_m) \quad (2)$$

dimana semua operasi adalah berbasis pada  $Z_{26}$

Pengembangan dari algoritma *Vigenere cipher* untuk penyandian *image* dilakukan dengan menggunakan formula *Vigenere cipher* dengan menggunakan nilai basis modulo 256 sesuai dengan intensitas warna pada *image*. Kunci-kunci tersebut disebut dengan *Vigenere table*. Dalam implementasinya tabel tersebut dikembangkan dengan nilai *plaintext* dari 0 sampai dengan 255 seperti terlihat pada Gambar 1.

|     | 0   | 1   | 2   | 3   | ... | 250 | 251 | 252 | 253 | 254 | 255 |
|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| 0   | 0   | 1   | 2   | 3   | ... | 250 | 251 | 252 | 253 | 254 | 255 |
| 1   | 1   | 2   | 3   | 4   | ... | 251 | 252 | 253 | 254 | 255 | 0   |
| 2   | 2   | 3   | 4   | 5   | ... | 252 | 253 | 254 | 255 | 0   | 1   |
| 3   | 3   | 4   | 5   | 6   | ... | 253 | 254 | 255 | 0   | 1   | 2   |
| 4   | 4   | 5   | 6   | 7   | ... | 254 | 255 | 0   | 1   | 2   | 3   |
| 5   | 5   | 6   | 7   | 8   | ... | 255 | 0   | 1   | 2   | 3   | 4   |
| 6   | 6   | 7   | 8   | 9   | ... | 0   | 1   | 2   | 3   | 4   | 5   |
| 7   | 7   | 8   | 9   | 10  | ... | 1   | 2   | 3   | 4   | 5   | 6   |
| 8   | 8   | 9   | 10  | 11  | ... | 2   | 3   | 4   | 5   | 6   | 7   |
| 9   | 9   | 10  | 11  | 12  | ... | 3   | 4   | 5   | 6   | 7   | 8   |
| 10  | 10  | 11  | 12  | 13  | ... | 4   | 5   | 6   | 7   | 8   | 9   |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| 253 | 253 | 254 | 255 | 0   | ... | 247 | 248 | 249 | 250 | 251 | 252 |
| 254 | 254 | 255 | 0   | 1   | ... | 248 | 249 | 250 | 251 | 252 | 253 |
| 255 | 255 | 0   | 1   | 2   | ... | 249 | 250 | 251 | 252 | 253 | 254 |

Gambar 1. Pengembangan bujursangkar *Vigenere cipher*

Angka pada baris pertama pada Gambar 1. dengan arsiran adalah *index* nilai *pixel image* yang dikodekan (*plaintext* yang berupa *image*), angka pada kolom pertama dengan arsiran adalah kode kunci (*key*). Sedangkan Angka tanpa arsiran adalah hasil (*cipher image*). Rumus enkripsi yang digunakan untuk menghitung nilai *cipher image* tiap *pixel* menggunakan persamaan 3, sedangkan rumus dekripsi menggunakan persamaan 4:

$$E_{k_i}(a) = (a + k_i) \bmod 256 \quad (3)$$

$$D_{k_i}(a) = (a - k_i) \bmod 256 \quad (4)$$

Keterangan:

a = intensitas ke-i,j *image* yang akan di enkripsi atau di dekripsi

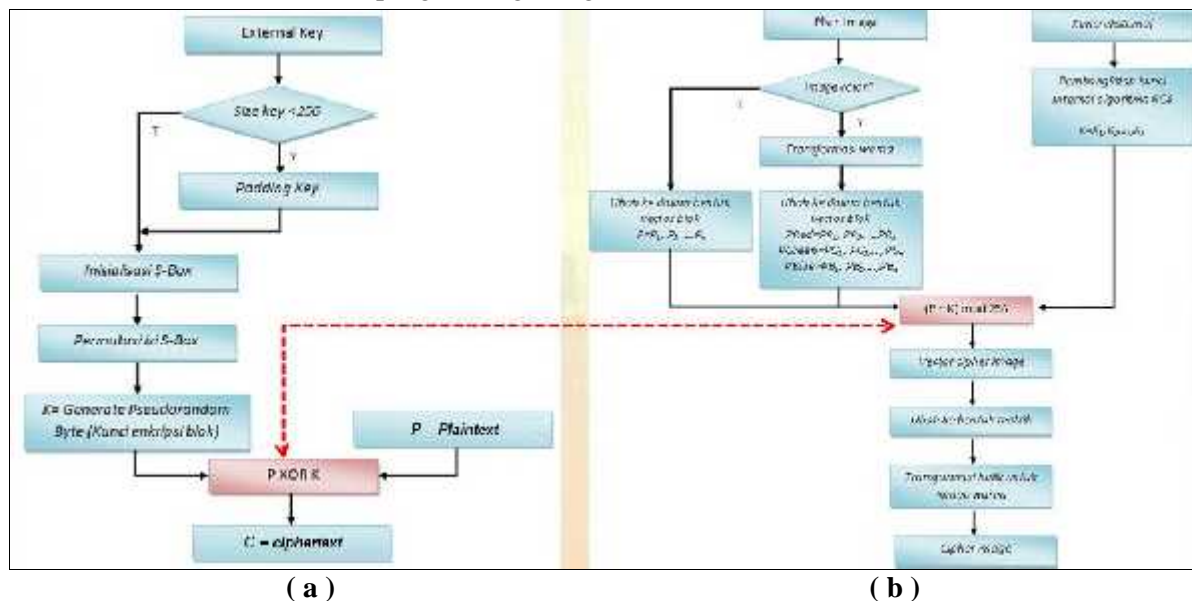
$k_i$  = kunci ke-i

Algoritma *Vigenere cipher* masih dapat dipecahkan dengan metode *exhaustive search* apabila panjang kunci diketahui karena kunci berikutnya merupakan pengulangan dari kunci apabila panjang kunci tidak sama dengan panjang *plaintext* [1]. Untuk mengatasi kelemahan ini, digunakan metode *keystream generator* untuk mengacak urutan kunci berikutnya agar kriptanalis kesulitan mendapatkan kuncinya. Persamaan 5 digunakan untuk membangkitkan kunci ke-i menggunakan *keystream*.

$$k_i = (k_{i-1} + k_{i-m}) \bmod 256 \quad (5)$$

### C. Usulan Pengembangan Algoritma RC4

Berdasarkan kelemahan pada algoritma RC4 dan juga algoritma *Vigenere cipher* yang telah dijelaskan di atas, maka untuk mengatasi serangan *known-plaintext attack* pada makalah ini akan dikembangkan algoritma RC4 yang diimplementasikan pada pengamanan data *image*. Gambar 2 memberikan ilustrasikan skema pengembangan algoritma RC4.



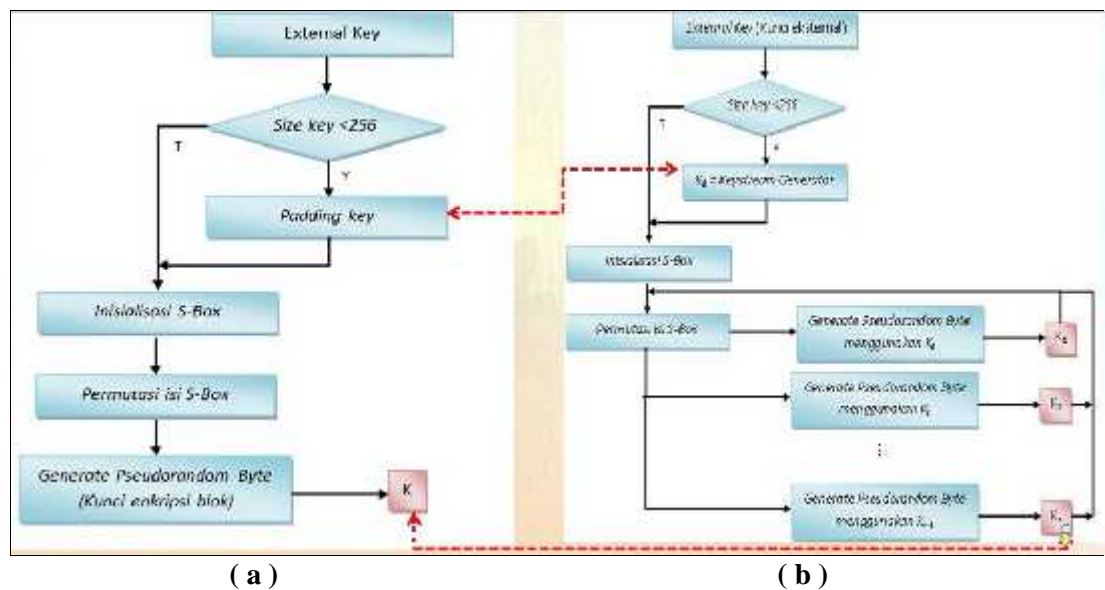
**Gambar 2** (a) Skema enkripsi algoritma RC4 (b) Skema pengembangan algoritma RC4

Langkah-langkah enkripsi *image* menggunakan pengembangan algoritma RC4 yang diilustrasikan pada Gambar 2(b) adalah sebagai berikut:

1) Mengambil *image* yang akan dienkripsi

- Lakukan transformasi warna yang berfungsi untuk memisahkan warna RGB untuk *image* berwarna sehingga menjadi 3 buah matrik. Untuk *image* grayscale tidak perlu dilakukan transformasi warna.
- Masing-masing *plain image* diubah menjadi vektor 1 x m dimana m adalah perkalian jumlah baris dan kolom dari *image* yang akan dienkripsi. Misalkan sebuah *image* berukuran 10 x 10 *pixel* maka diubah menjadi vektor 1 x 100 *pixel*
- Bagi setiap kanal warna menjadi blok-blok cipher dimana setiap blok cipher berisi 256 *pixel image*, sehingga didapatkan n blok *plain image* ( $P = p_1, p_2, p_3, \dots, p_n$ ).

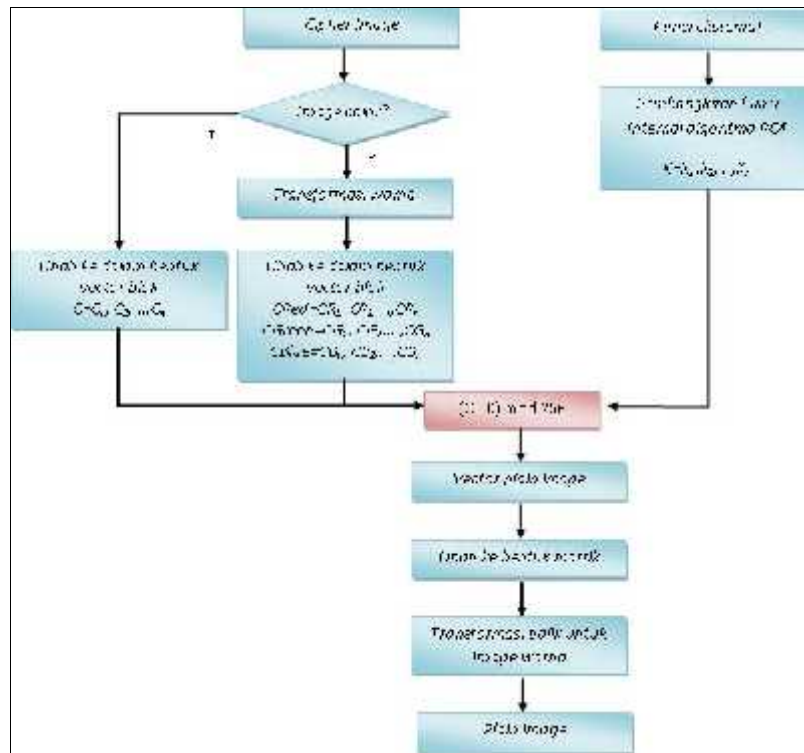
- 2) Membangkitkan kunci internal ( $K = K_1, K_2, \dots, K_n$ ) untuk proses enkripsi seperti diilustrasikan pada Gambar 3(b). Langkah untuk membangkitkan kunci internal sebagai berikut:
  - a. Mengisi nilai-nilai di dalam *array* S dengan menggunakan kunci *eksternal* yang lebih panjang yaitu 256 *byte*. Agar mendapatkan kunci *eksternal* yang lebih panjang. Metode pengisian kunci ke dalam kunci *array* dengan cara kunci *eksternal* yang pendek cukup diisikan sekali dalam *array* kemudian sisa variabel kunci *array* yang lainnya akan diisi dengan nilai yang dibangkitkan secara acak menggunakan persamaan 5.
  - b. Kunci internal didapatkan dengan melakukan proses *keystream generator* RC4 secara berulang untuk tiap blok kunci internal menggunakan kunci eksternal yang berbeda. Kunci eksternal  $K_2$  sampai  $K_n$  didapatkan dari proses *keystream generator* RC4 pada iterasi ke-1 sampai  $n-1$ . Artinya kunci  $K_2$  didapatkan dari proses *keystream generator* RC4 pada iterasi ke-1 menggunakan kunci  $K_1$ . Untuk kunci  $K_3$  didapatkan dari proses *keystream generator* RC4 pada iterasi ke-2 menggunakan kunci  $K_2$  dan seterusnya sampai dengan iterasi ke  $n-1$ .



**Gambar 3** Proses pembangkitan kunci internal (a) RC4 (b) pengembangan algoritma RC4

- 3) Proses enkripsi dilakukan menggunakan konsep blok dimana setiap blok menggunakan 256 *byte/pixel*. Proses enkripsi menggunakan persamaan 3 seperti yang digunakan pada algoritma *Vigenere cipher* menggunakan kunci hasil langkah 2 dengan blok *cipher plain image* untuk masing- komponen warna.
- 4) Vektor hasil enkripsi dikembalikan sebagai nilai matrik RGB menggunakan transformasi warna balik sehingga menghasilkan *image* baru yang sudah tersandikan.

Proses dekripsi dilakukan dengan cara yang sama dengan proses enkripsi. Sedangkan skema dekripsi pengembangan RC4 diperlihatkan pada Gambar 4.



**Gambar 4** Skema dekripsi pengembangan algoritma RC4













## PEMBAHASAN

Untuk mengetahui kekuatan dari algoritma enkripsi dan dekripsi *image* yang diusulkan, maka dilakukan pengujian dan analisis menggunakan beberapa metode yang biasa digunakan untuk mengukur kekuatan dari sebuah cipher.

### A. Uji Visual dan Analisis Histogram

Dari hasil pengujian 3 *image* yang berbeda ketiditan dan ukurannya menggunakan kunci yang sama, maka berdasarkan uji secara visual seperti yang terlihat pada Tabel 1. *image* asli tidak dapat terlihat setelah dilakukan proses enkripsi. Hasil penyandian *image* menunjukkan keteracakan warna dan perubahan intensitas warna yang cukup signifikan, hal ini menunjukkan bahwa proses enkripsi menggunakan algoritma pengembangan algoritma RC4 berhasil dengan baik. Apabila dilihat secara visual dari histogram *image* asli dengan histogram dari *image* yang tersandikan, maka terlihat perbedaan yang signifikan antara keduanya. Hal ini menunjukkan bahwa algoritma enkripsi yang digunakan tidak dapat memberikan petunjuk apa-apa untuk dilakukan *statistical attack* oleh kriptanalisis.

**Tabel 1.** Uji Visual dan Analisis Histogram Warna

| Plain Image  | Cipher Image  | Histogram   |   |
|--|---|---|---|
|  |   | Asli  | Enkripsi  |
| <br>bunga.bmp   |  |  |  |
| <br>pesawat.bmp |  |  |  |
| <br>nohan.bmp   |  |  |  |





## B. Uji Statistik

Uji statistik ini meliputi uji korelasi, entropi, kualitas hasil enkripsi dan waktu proses enkripsi dan dekripsi seperti terlihat dari pada Tabel 2.

**Tabel 2.** Hasil Uji Statistik

| Nama <i>Image</i>            | bunga.bmp | nohan.bmp | pesawat.bmp | godhill.bmp |
|------------------------------|-----------|-----------|-------------|-------------|
| Ukuran <i>Image</i> (piksel) | 192 x 256 | 256 x 256 | 200 x 200   | 576 x 720   |
| Type <i>image</i>            | Color     | Color     | gray        | Color       |
| Nilai Entropi                | 7.99610   | 7.99743   | 7.99576     | 7.99953     |
| Nilai Korelasi               | 0.00069   | -0.00052  | -0.00024    | -0.00024    |
| Kualitas Enkripsi            | 166.917   | 201.826   | 247.172     | 880.081     |
| Waktu Enkripsi (detik)       | 1.20121   | 1.70041   | 1.07641     | 6.27124     |
| Waktu Dekripsi (detik)       | 1.12321   | 1.45081   | 0.904806    | 6.06844     |

Jika sebuah informasi dienkripsi dan dalam kondisi teracak maka nilai entropi yang idealnya adalah mendekati nilai 8 ( 8) [5]. Dari Tabel 2 terlihat untuk 4 *image* yang diujikan rata-rata nilai entropinya adalah 7.99721, maka sistem enkripsi yang dirancang ini aman dari serangan entropi karena nilainya sangat dekat dengan 8. Nilai korelasi antara *image* asli dengan *image* hasil enkripsi dari 4 *image* yang diuji rata-rata bernilai 0.00042. Hal ini menunjukkan bahwa sistem enkripsi yang diusulkan sesuai dengan *perfect security* yang dikemukakan oleh Shannon karena nilai korelasinya mendekati nol [10]. Sedangkan untuk mengukur kualitas enkripsi *image* dilakukan dengan membandingkan nilai *pixel image* sebelum dan sesudah dienkripsi yang dinyatakan sebagai deviasi antara *image* asli dan *image* hasil enkripsi [11]. Nilai kualitas enkripsi dari hasil pengujian cukup tinggi yang artinya tingkat perubahan *pixel*-nya pun juga tinggi sehingga sistem ini dapat dikatakan efektif dan aman. Analisis waktu proses enkripsi dan dekripsi terhadap berbagai ukuran *image* yang diujikan cukup cepat dalam hitungan detik. Dari hasil tersebut dapat dinyatakan bahwa algoritma ini cukup efektif untuk penyandian data *image* karena tidak membutuhkan waktu proses yang lama.

## KESIMPULAN

Berdasarkan hasil penelitian dapat diambil beberapa kesimpulan antara lain: 1) Algoritma enkripsi *image* yang diusulkan cukup efektif karena tidak membutuhkan waktu yang lama untuk proses enkripsi maupun dekripsi. 2) Hasil pengujian algoritma pengembangan RC4 menunjukkan secara visual *image* hasil enkripsi tidak terlihat lagi disebabkan oleh keteracakan warna dan perubahan intensitas warna yang cukup signifikan. 3) Dari histogram *plain image* dan *cipher image* menunjukkan bahwa algoritma ini cukup tangguh untuk bisa dibobol oleh kriptanalisis menggunakan metode *statistical attack* karena pada histogram *cipher image* terlihat merata intensitas warnanya 4) Dari uji statistik juga menunjukkan hasil yang cukup bagus dengan rata-rata nilai entropinya adalah 7.99721 yang berarti algoritma enkripsi yang dirancang ini aman dari serangan entropi karena nilainya sangat dekat dengan 8. Nilai korelasi antara *image* asli dengan *image* hasil enkripsi rata-rata bernilai 0.00042, hal ini menunjukkan bahwa sistem enkripsi yang diusulkan sesuai dengan *perfect security* yang dikemukakan oleh Shannon karena nilai korelasinya mendekati nol.

## DAFTAR PUSTAKA

- [1] Abrihama D. *Keystream Vigenere Cipher: Modifikasi Vigenere Cipher dengan Pendekatan Keystream Generator*. Program Studi Informatika ITB. Bandung. 2008.
- [2] Bishop D. *Introduction to Cryptography with Java Aplets*. John and Batrlet Publisher. 2002.

- [3] Hari W.H. dan Mulyana S. Implementasi RC4 Stream Cipher untuk Keamanan Basis Data. *Seminar Nasional Aplikasi Teknologi Informasi 2012 (SNATI 2012)*. 2012.
- [4] Ibrahim A. and Zabian A. Algorithm for Text Hiding in Digital Image for Information Security. *International Journal of Computer Science and Network Security*. 9(6):262 – 268. 2009.
- [5] Jolfaei A. And Mirghadri A. Image Encryption Using Chaos and Block Cipher. *Computer and Information Science*. 4(1). 2011.
- [6] Pardeep Pateriya P.K. PC1-RC4 and PC2-RC4 Algorithms: Pragmatic Enrichment Algorithms to Enhance RC4 Stream Cipher Algorithm. *International Journal of Computer Science and Network (IJCSN)*. 1(3). 2012. [www.ijcsn.org](http://www.ijcsn.org).
- [7] Risal A. dan Suharto. Implementasi Algoritma RC4 Untuk Keamanan Login Pada Sistem Pembayaran Uang Sekolah (Studi Kasus : Di Yayasan YABIS Bontang). *Jurnal Dielektrika*. 2(2): 142-149. 2011.
- [8] Schneier B. *Applied Cryptography 2nd* . New York: John Wiley & Sons. 1996.
- [9] Setyaningsih E. Pengembangan Metode Vigenere Cipher Untuk Pengamanan Data Citra. Laporan Penelitian. Lembaga Penelitian IST AKPRIND Yogyakarta. Februari 2010.
- [10] Stinson R. Douglas. *Cryptography Theory and Practice 2nd Edition*. London: CRC Press. Inc. 2002.
- [11]Younes M.A.B. and Jantan A. Image Encryption Using Block-Based Transformation Algorithm. *IAENG International Journal of Computer Science*. 35(1). 2008.