Enkripsi Image dengan Vigenere Cipher dan Chaos Transposition

Everaldo Sembiring(13510095)
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13510095@itb.ac.id

***Abstraksi*— Makalah ini menunjukkan pengacauan citra dengan mengganti warna setiap pixel dan mengubah posisi setiap pixel dari citra yang akan dienkripsi.Warna yang akan diubah adalah RGB dari setiap pixel.Warna akan diubah menggunakan Vigenere Cipher.Sedangkan posisi dari pixel-pixel akan diubah berdasarkan Chaos Function.Algoritma berfungsi untuk mengamankan data gambar yang penting.Hasil membuktikan bahwa sulit untuk mendekripsikan citra yang diamankan.**

***Kata Kunci*—citra,fungsi chaos,vigenere cipher,transposisi.**

I. PENDAHULUAN

Pada masa kini pertukaran informasi merupakan hal yang sudah biasa dilakukan.Media yang dipetukarkan banyak jenisnya,seperti:gambar,video,music dan lain-lain.Jika data yang dipetukarkan adalah media yang penting,maka dibutuhkan metode pengamanan yang kuat agar data tidak bisa dibuka oleh sembarang orang.Apalagi jika dokumen tersebut bersifat rahasia.Maka penulis mengembangkan salah satu metode kriptografi klasik sehingga dapat merahasiakan sebuah citra.Caranya adalah mengacaukan isi gambar tersebut menggunakan Extended Vigenere Cipher + Chaotic Generator.Sehingga gambar tersebut tidak bisa didekripsi dengan exhaustive search,karena chaotic generator bersifat sensitif terhadap perubahan.

Enkripsi citra ini bertujuan untuk merusak gambar,sehingga tidak dimengerti maknanya.Seperti disebutkan sebelumnya,enkripsi yang digunakan adalah Extended Vigenere Cipher + Chaotic Generator.Vigenere Cipher berfungsi untuk mengubah warna sebuah pixel,sedangkan Chaotic Generator berfungsi sebagai mengtranspose urutan pixel.Kunci Vigenere Cipher berupa karakter biasa ataupun khusus .Karakter-karakter tersebut berfungsi untuk mengubah warna merah,hijau,dan biru dari sebuah pixel.Key akan berulang sampai sejumlah pixel dari gambar.Key dari Chaotic Generator berupa sebuah bilangan real yang akan menjadi parameter fungsi suatu persamaan dan fungsi tersebut akan menghasilkan nilai yang akan digunakan untuk mengtranspose pixel.Maka perlu ditemukan fungsi yang menghasilkan nilai yang tidak berulang dan perubahan yang signifikan untuk mencegah brute force.

II. PENERAPAN VIGENERE CIPHER DAN FUNGSI CHAOS

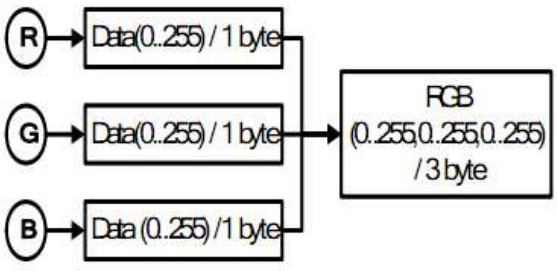
A. Image

Image adalah gabungan warna-warna pada bidang 2 dimensi. Jumlah warna terdiri atas beberapa jenis:

Nama	Jumlah Bit	Jumlah Warna
B/W	**1**	**2**
Windows Display	**4**	**16**
Grey Scale	**8**	**256**
256 Color	**8**	**256**
High Color	**16**	**65.535**
True Color	**24**	**16.777.216**
True Color	**32**	**4.294.967.296**

Tabel 1: Jenis Warna

Pada algoritma yang akan dilakukan nanti,warna RGB(*Red Green Blue*) pada setiap pixel image akan diubah.Masing-masing warna tersusun atas 8 bit,sehingga untuk satu warna memiliki rentang 0-255.Oleh karena itu dalam satu pixel terdiri atas 24 bit.Sehingga terdapat 16.777.216 kombinasi untuk satu pixel.



Gambar 1:RGB di dalam Pixel

B. Vigenere Cipher

Vigènere Cipher adalah metode enkripsi yang menggunakan Bujursangkar Vigènere untuk melakukan

enkripsi. Setiap baris di dalam bujursangkar menyatakan huruf-huruf cipherteks yang diperoleh dengan Caesar Cipher.

Kunci: $K = k_1 k_2 \ldots k_m$

k_i untuk $1 \le i \le m$ menyatakan jumlah pergeseran pada huruf ke-i. Karakter cipherteks: $c_i(p) = (p + k_i) \bmod 26$.

Jika panjang kunci lebih pendek daripada panjang plainteks, maka kunci diulang secara periodik.

Pada Algoritma yang akan dipakai,akan digunakan Extended Vigenere Cipher.Yaitu maksimum jumlah pergeseran maksimal 256.Sama dengan maksimal warna dari *Red/Green/Blue*.Setiap karakter dari kunci akan mengubah satu warna dari pixel.

$c(R) = (R + k_i) \bmod 256$

$c(G) = (G + k_{i+1}) \bmod 256$

$c(B) = (B + k_{i+2}) \bmod 256$

Panjang kunci akan diulang hingga sama dengan banyak RGB pada image.Sehingga pada akhir vigenere cipher,seluruh RGB pada image sudah berubah.

C.Fungsi Chaos

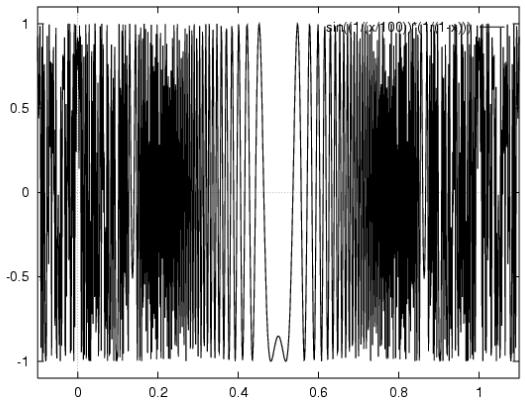
Fungsi chaos adalah fungsi yang sensitive terhadap nilai awal.Sifat dari fungsi ditentukan oleh nilai saat inisialisasi,sehingga jika berubah sedikit saja maka akan menghasilkan nilai yang jauh berbeda.

Untuk menghasilkan kumpulan nilai yang chaos,maka hasil dari fungsi chaos digunakan pada fungsi selanjutnya.

$$X_{n+1}=f(X_n)$$

Fungsi chaos yang digunakan pada algoritma adalah fungsi yang memiliki 2 daerah Chaos.

$$f(X_n)= \sin ((1/(x/100)) (1/(1-x)))$$



Gambar 2 : Fungsi $f(X_n)$
http://computing.dcu.ie/~humphrys/Notes/Neural/Bitmaps/sin.2.regions.2.gif

D.Algoritma Enkripsi

Pada algoritma yang akan digunakan,terdapat 2 bagian yaitu enkripsi warna dan transposisi.Berikut langkah-langkah yang dilakukan:

1. Mengubah RGB setiap pixel menggunakan kunci K.Kunci diulang terus-menerus hingga samadengan jumlah RGB.
2. Akan diperoleh image yang warnanya sudah tidak sama.
3. Iterasi fungsi chaos hingga N kali hingga sama dengan jumlah pixel.Sehingga terbentuk $X=\{X_1, X_2, X_3, \ldots, X_n\}$.
4. Pasangkan Setiap chaos function dengan pixel image dimulai dari kiri atas hingga kanan bawah.Sehingga X_1 akan berpasangan dengan pixel pada kiri atas dan X_n dengan pixel kanan bawah.Misalkan Y adalah pixel dan sebuah image tersusun atas $Y = \{ Y_1, Y_2, Y_3, \ldots, Y_n\}$.Sehingga akan terbentuk $(X,Y)=\{(X_1,Y_1), (X_2,Y_2), (X_3,Y_3), \ldots, (X_n,Y_n)\}$
5. Kemudian urutkan berdasarkan nilai X,sehingga setiap pixel akan berubah posisinya.Hal ini akan menyebabkan image menjadi kacau.

D.Algoritma Dekripsi

Untuk melakukan dekripsi cukup sederhana.Pada awalnya bangkitkan kembali fungsi chaos dengan nilai inisialisasi yang sama.Kemudian setiap X_i dipasangkan dengan i.Maka terbentuk $\{(X_1,1), (X_2,2), (X_3,3), \ldots, (X_N,N)\}$

Setelah itu urutkan berdasarkan X.Kemudian susunan yang sudah diurutkan dipasangkan dengan pixel image yang terenkripsi dimulai dari kiri atas hingga kanan bawah.Sehingga terbentuk $\{(X_a,Y_1,a), (X_b\ Y_2,b), (X_c, Y_3,c), \ldots, (X_Z, Y_N,Z)\}$

Urutkan kembali berdasarkan nilai i.Maka posisi pixel akan kembali seperti semula:

$\{(X_1,Y_a,1), (X_2,Y_b,2), (X_3,Y_c,3), \ldots, (X_N,Y_z,N)\}$

Setelah mengurutkan pixel dengan benar,maka dilakukan dekripsi vigenere cipher untuk setiap RGB.

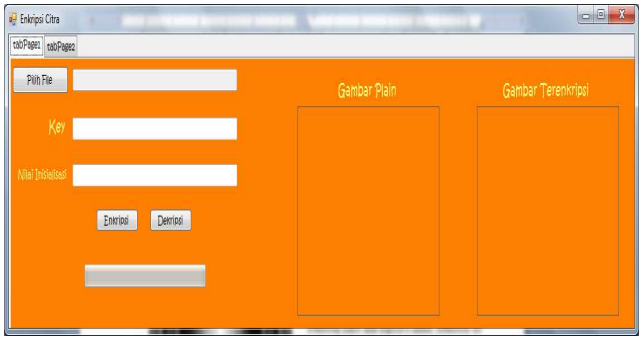
$$R = (c(R) - k_i) \bmod 256$$
$$G = (c(G) - k_{i+1}) \bmod 256$$
$$B = (c(B) - k_{i+2}) \bmod 256$$

Maka akan ditemukan plain Image yang sesuai.

III. APLIKASI ALGORITMA

A)Algoritma

Berikut adalah aplikasi yang dibuat oleh penulis untuk mengaplikasikan algoritma yang dirancang.



Gambar 3 : Aplikasi Enkripsi dengan vigenere dan

chaos function

Terdapat 3 input dari user.Yaitu file yang akan dienkripsi/dekripsi,Key untuk vigenere cipher,nilai inisialiasi fungsi chaos.

Berikut adalah penjelasan algoritma dengan menggunakan *pseudo code:*

```
function enkripsi(input I:image,input key:string,input
inisialisasi:double)->image
   V:VigenereTable ;
   V.CreateVigenereTable();

   int keyIndex <- 0;

   for (int y <- 0; y < I.Height; y++)
      {
         for (int x <- 0; x < I.Width; x++)
         {
         int tabelEntryR <- V[I.R(x,y)][key_keyindex];
         keyIndex++;
         if (keyIndex = key.Length) keyIndex<-0;

         int tabelEntryG <- V[I.G(x,y)][key_keyindex];
         keyIndex++;
         if (keyIndex = key.Length) keyIndex<-0;

         int tabelEntryB <- V[I.B(x,y)][key_keyindex];
         keyIndex++;
         if (keyIndex = key.Length) keyIndex<-0;

         I.SetPixel(x, y,
Color(tabelEntryR,tabelEntryG,tabelEntryB));
         }
      }

   generatefunction(inisialisasi);
   L:list
   L.add(everypixel,everyChaosXn);
   Lsort(Xn); //sort by chaos value

   Enkripsi:image;

   int iterate<-0 ;
   for (int y <- 0; y < I.Height; y++)
      {
         for (int x <- 0; x < I.Width; x++)
         {
            Enkripsi.setpixel(x,y, L[iterate].pixel);
         }
      }

   ->Enkripsi;
```

V adalah tabel vigenere table yang akan digunakan.Untuk mengurutkan pixel berdasarkan nilai chaos maka digunakan list.List yang sudah terurut kemudian setiap isi warna dalam list dimasukkan kembali ke dalam image.Image akhir itulah hasil enkripsi.

Sedangkan untuk dekripsi sebagai berikut:

```
function dekripsi(input I:image,input key:string,input
inisialisasi:double)->image
   generatefunction(inisialisasi);
   L:list
   L.addchaosvalue();
   L.addindex();
   L.sort(value);
   L.addpixel();
   L.sort(index);

   Plain:image;

   int iterate<-0 ;
      for (int y <- 0; y < I.Height; y++)
         {
            for (int x <- 0; x < I.Width; x++)
            {
               Plain.setpixel(x,y, L[iterate].pixel);
            }
         }

   V:VigenereTable ;
   V.CreateVigenereTable();

   int keyIndex <- 0;

   for (int y <- 0; y < I.Height; y++)
      {
         for (int x <- 0; x < I.Width; x++)
         {

         int tabelEntryR<-0;
         for (int pIndex <- 0; pIndex < vigenerecoloumnsize; pIndex++)
            {
               if (vigenereTabel[pIndex][key_keyindex] = Plain.R(x,y))
               {
                  tabelEntryR <- pIndex;
                  break;
               }
            }
         keyIndex++;
         if (keyIndex = key.Length) keyIndex <-0;

         int tabelEntryG<-0;
         for (int pIndex <- 0; pIndex < vigenerecoloumnsize; pIndex++)
            {
               if (vigenereTabel[pIndex][_keykeyindex] = Plain.G(x,y))
               {
                  tabelEntryG <- pIndex;
                  break;
               }
            }
         keyIndex++;
         if (keyIndex = key.Length) keyIndex <- 0;

         int tabelEntryB<-0;
         for (int pIndex <- 0; pIndex < vigenerecoloumnsize; pIndex++)
            {
               if (vigenereTabel[pIndex][key_keyindex] = Plain.B(x,y))
               {
                  tabelEntryR <- pIndex;
                  break;
               }
            }
         keyIndex++;
         if (keyIndex = key.Length) keyIndex <- 0;
         }
      }
   Plain.SetPixel(x, y, Color(tabelEntryR,tabelEntryG,tabelEntryB));
```

->Plain

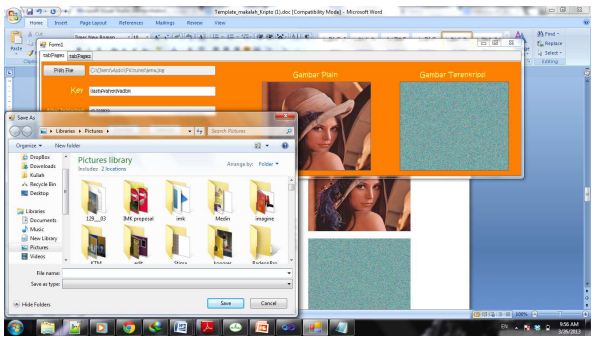
Dari kedua algoritma tersebut maka dapat dilakukan enkripsi citra seperti gambar berikut:

Gambar 4 : Contoh hasil enkripsi citra menggunakan algoritma yang diaplikasikan

B)Ilustrasi Aplikasi

Pada awalnya user melakukan input file path,key string,dan nilai inisialisasi.File path digunakan untuk mengambilfile image yang ingin dienkripsi,extensi file bisa apa aja.key string digunakan untuk algoritma vigenere sedangkan nilai inisialisasi digunakan untuk nilai awal fungsi chaos.

Jika user memilih enkripsi maka program akan memproses image seperti pada algoritma yang dijelakan pada bagian A.Kemudian image hasil enkripsi dapat disimpan.



Dapat dilihat pada gambar di atas,program langsung menampilkan hasil enkripsi.

Untuk melakukan dekripsi,diilakukan langkah yang sama.User memasukkan file path,string key,dan nilai inisialisasi.Kemudian memilih bagian dekripsi.

IV. EKSPERIMEN

1.Hasil Vigenere Cipher

Pada gambar 5 ditunjukkan hasil perubahan warna menggunakan algoritma vigenere ciperh.Kunci yang digunakan adalah “Lena”.Gambar 4 merupakan gambar asli dari image.Warnanya menjadi berubah tapi makna dari citra masih jelas.Pengacauan warna berguna saat gambar original maknanya bergantung terhadap warna.

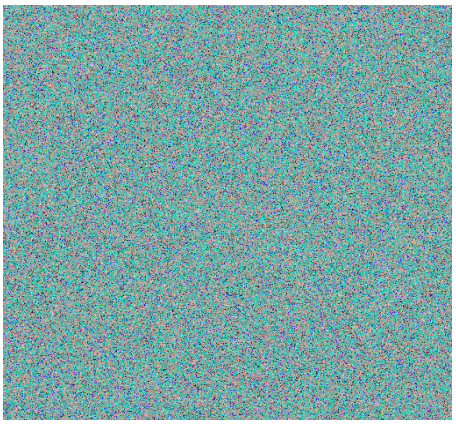
Gambar 4:Gambar Original http://www.freeappalliance.com/images/product_screenshots/146040/mzl.pcfsdift.480x480-75.jpg

Gambar 5:Gambar hasil enkripsi warna

Gambar diatas dapat menyebabkan kesalahan,karena warna biru,merah,dan hijau berubah warnanya setelah enkripsi.

2.Hasil akhir enkripsi

Pada enkripsi warna,image berubah menjadi memiliki warna yang berbeda.Namun makna dari image tersebut masih jelas.Oleh karena itu untuk membuat gambar tersebut tidak memiliki makna,maka digunakan transposisi.Berikut adalah hasil akhir enkripsi setelah ditransposisi setiap pixelnya.

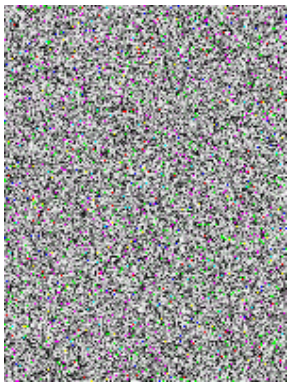


Gambar 6:Gambar original(atas) dan Gambar Akhir enkripsi(bawah)

Dapat dilihat bahwa hasil akhir tidak dapat ditentukan maknanya lagi.Hal ini dikarenakan posisi pixel letaknya sudah jauh berbeda dengan posisi semula.

Berikut contoh hasil enkripsi pada gambar grayscale:

Gambar 7:Gambar grayscale original http://upload.wikimedia.org/wikipedia/commons/f/fa/Grayscale_8bits_palette_sample_image.png



Gambar 8:Gambar grayscale terenkripsi http://upload.wikimedia.org/wikipedia/commons/f/fa/Grayscale_8bits_palette_sample_image.png

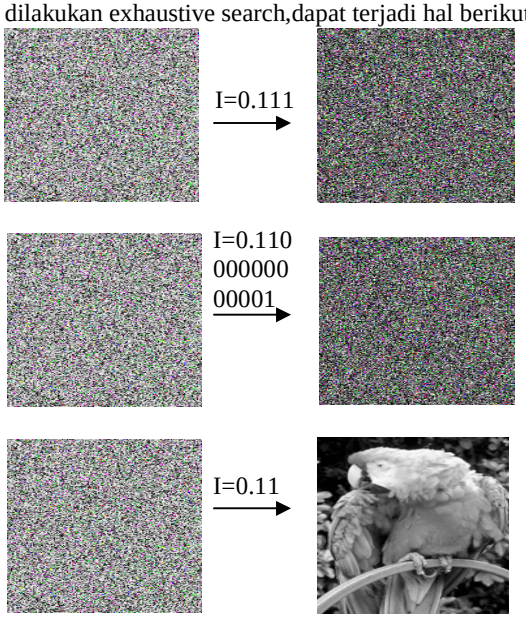
Dengan demikian maka algoritma ini cukup baik untuk merahasikan dokumen gambar yang penting.Karena orang asing yang mengambil gambar tidak akan bisa mengetahui isi dari gambar tersebut.

3.Dekripsi Menggunakan kunci yang salah

Dalam algoritmaini,nilaikey vigenere berfungsi untuk mengembalikan warna.Jika salah maka susunan warna akan salah.Sedangkan jika nilai inisialisasi salah,maka tetap tidak dapat diketahui makna dari gambar.

A.Nilai inisialisasi salah

Misalkan suatu gambar telah dienkripsi menggunakan nilai inisialisasi 0.11 dan kunci string "kriptografi".Asumsi jika kunci string diketahui.Namun nilai inisialisasi masih tidak diketahui.Maka jika dilakukan exhaustive search,dapat terjadi hal berikut:



Gambar 9:Dekripsi dengan berbagai macam nilai inisialisasi

Dapat dilihat bahwa nilai inisialisasi sangatlah penting.Karena perubahan yang sangat kecil pun dapat

menyebabkan gambar tetap tidak bermakna.Hal ini sudah pasti tidak memungkinkan penyerang melakukan exhaustive search.

B.Nilai kunci string salah

Nilai kunci string yang salah akan menyebabkan warna tidak sesuai dengan warna original.Namun untuk mencari susunan warna yang sesuai juga tidak mungkin menggunakan exhaustive search.Karena kombinasi warna dari 1 pixel sangatlah banyak.

IV. Analisis Keamanan

Untuk menyerang algoritma enkripsi ini sangatlah tidak mungkin jika tidak memiliki kunci.Jika penyerang berusaha untuk melakukan exhaustive search pada string key maka percobaan bergantung pada panjang key.Misalkan panjang string key adalah 8,maka banyak percobaan adalah 256^8.Hal ini tidak mungkin untuk dilakukan karena komputasinya tinggi.

Meskipun penyerang mengetahui string key tapi tidak mengetahui nilai inisialisasi,maka itu juga tidak mungkin.Karena nilai yang diinput bersifat kontinu,sehingga percobaan yang dilakukan dilakukan tak hingga.Perubahan sedikit saja dapat menghasilkan citra yang jauh berbeda dengan aslinya.Hal itu dibuktikan pada percobaan sebelumnya,dimana perbedaan nilai sebesar 0.000000000000001 masih tetap menghasilkan image yang tidak dimengerti.

Hal yang paling menyulitkan adalah jika penyerang tidak mengetahui algoritma yang digunakan,tidak akan mungkin penyerang dapat mengdekripsi citra yang dienkripsi.

Namun kelemahan dari algoritma ini adalah dapat menyebabkan kecurigaan.Jika image ini dipertukarkan,dapat memunculkan asumsi bahwa image ini mengandung pesan rahasia.Oleh karena itu baiknya dilakukan steganografi untuk merahasiakan transfer data.

V. Kesimpulan

Berdasarkan hasil eksperimen yang dilakukan,algoritma ini sangat aman digunakan jika ingin merahasikan suata data image.Untuk mengubah ke gambar aslinya,dibutuhkan kunci yang sebenarnya.Namun untuk merahasiakan pertukaran pesan rahasia,hasil image perlu disimpan pada media digital lainnya.

References

[1] Shanshan Li,Yinghai Zhao,Image Scrambling Based on Chaos Theory and Vigenere Cipher,7th International Conference on Computational Intelligence and Security,2011.W.-K. Chen, *Linear Networks and Systems* (Book style). Belmont, CA: Wadsworth, 1993, pp. 123–135.

[2] Lea Angelina,*Penerapan dari Pengembangan Algoritma Vigenere dalam Enkripsi Image*,Makakalah Kriptografi,Teknik Informatika,Institut Teknologi Bandung,2010.

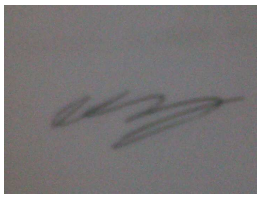
[3] http://computing.dcu.ie/~humphrys/Notes/Neural/chaos.html Waktu akses: 02.26 WIB , 26 Maret 2013

Pernyataan

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 26 Maret 2013

ttd



Everaldo Sembiring,13510095