RANCANGAN DAN ANALISIS *CIPHER* BERBASIS ALGORITMA TRANSPOSISI DENGAN PERIODISASI KUNCI

Boyke Ariesanda
NIM. 135 03 036

Program Studi Teknik Informatika STEI, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if13036@students.if.itb.ac.id

Abstrak

Makalah ini membahas tentang rancangan dan implementasi sebuah algoritma kriptografi yang tergolong algoritma kriptografi klasik. Secara umum, algoritma kriptografi klasik dibagi ke dalam dua jenis: algoritma berbasis substitusi serta algoritma berbasis transposisi. Makalah ini menitikberatkan pembahasan pada jenis yang kedua, yakni dengan menyusun rancangan sebuah algoritma transposisi yang menggunakan sebuah kunci untuk proses enkripsi dan dekripsi (kunci simetri). Pada algoritma yang akan dirancang ini, proses permutasi tersebut akan dikendalikan oleh suatu fungsi dengan sebuah kunci, berupa *string*, sebagai parameter.

Makalah ini membahas konsep dasar algoritma transposisi, metode periodisasi kunci yang diadaptasi dari *Vigenère cipher*, dan hasil implementasi sederhana algoritma tersebut, menggunakan bahasa pemrograman Java dan memanfaatkan kakas NetBeans 5.0. Selain itu, makalah juga membahas tingkat keamanan algoritma ini ditinjau dari aspek kekuatan algoritma, kekuatan kunci, serta daya tahan terhadap berbagai jenis serangan kriptanalisis. Akhirnya, makalah ini juga akan mengusulkan prospek algoritma ini untuk penggunaan praktis serta kemungkinan pengembangannya.

Kata Kunci : Kriptografi, Algoritma, Klasik, Transposisi, *Vigenère cipher*, Kriptanalisis.

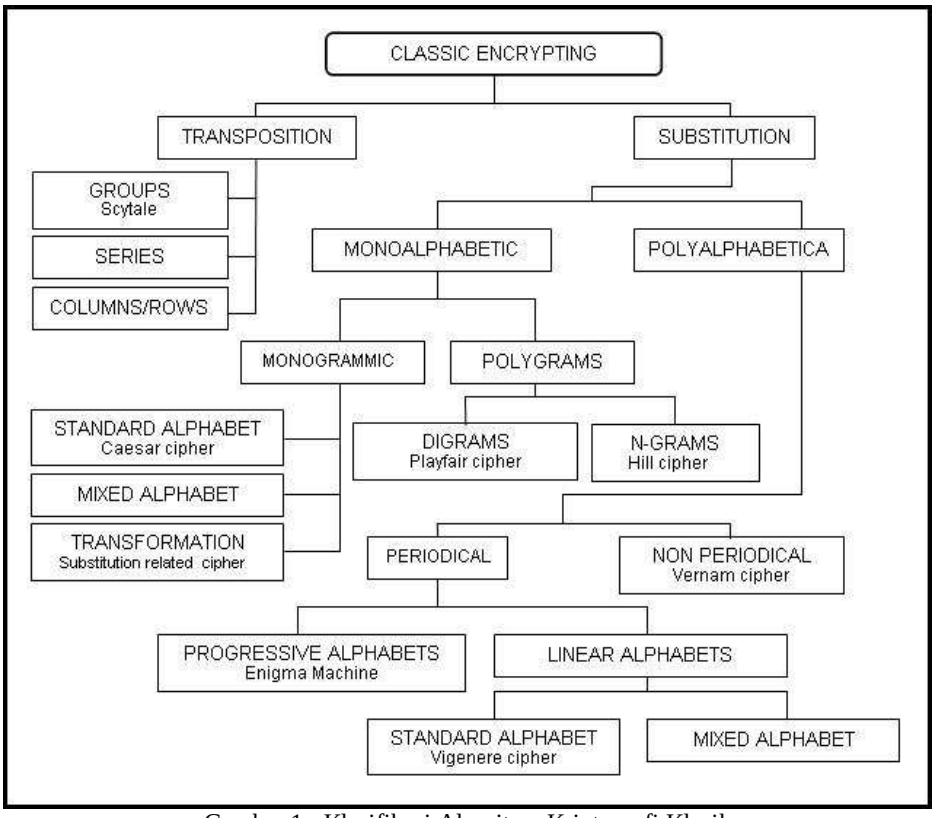
1. Pendahuluan

Saat ini, kriptografi memainkan peranan yang sangat penting di dunia, khususnya dalam bidang teknologi informasi. Ketika komputerisasi merambah hampir semua bidang kehidupan, kriptografi turut serta. Namun demikian, sejarah kriptografi ternyata jauh lebih panjang daripada sejarah komputer. Dengan kata lain, peradaban lebih dahulu mengenal kriptografi daripada komputer. Berdasarkan fakta ini, secara umum kriptografi, atau secara lebih khusus algoritma kriptografi, dibagi ke dalam dua golongan besar : klasik dan modern. Algoritma kriptografi klasik merupakan kumpulan berbagai teknik kriptografi yang ditemukan manusia sebelum ditemukannya komputer. Aplikasi teknik kriptografi ini secara umum dilakukan dengan pena dan kertas, serta menyandikan suatu dokumen huruf demi huruf (berbasis karakter). Setelah komputer diciptakan, kriptografi mulai diaplikasikan menggunakan komputer. Kumpulan teknik kriptografi baru ini yang dinamakan algoritma kriptografi modern. Karena berbasis komputer, maka ia bekerja mengikuti mekanisme kerja komputer, yakni (umumnya) melakukan penyandian bit per bit. Meski terdapat perbedaan yang cukup signifikan di antara kedua golongan itu, ternyata algoritma kriptografi modern tetap memiliki landasan konsep yang berasal dari algoritma kriptografi klasik.

Algoritma kriptografi klasik dibagi menjadi dua jenis yakni algoritma substitusi dan algoritma transposisi. Sesuai namanya, algoritma substitusi menyembunyikan isi pesan/dokumen dengan cara mengganti karakter-karakter yang menyusun pesan/dokumen tersebut, menurut aturan tertentu. Variasi aturan substitusi ini melahirkan berbagai jenis algoritma spesifik, yang biasanya dinamai menurut penemunya. Beberapa contoh algoritma substitusi diantaranya Algoritma Caesar, Algoritma Vigenère, dan Algoritma Affine. Namun, biasanya yang lebih sering dipakai sebagai sebutan bukanlah algoritmanya melainkan hasil dari algoritma kriptografi tersebut yakni *cipher* (pesan yang telah tersandikan). Jadi, contoh-contoh sebelumnya akan diacu sebagai *Caesar Cipher*, *Vigenère Cipher*, dan *Affine Cipher*.

Jenis algoritma kriptografi klasik yang kedua ialah algoritma berbasis transposisi. Cara kerja algoritma ini ialah dengan mengubah susunan / urutan karakter-karakter di dalam dokumen. Pada umumnya hal ini dilakukan dengan cara menuliskan plainteks dengan pola tertentu, kemudian dibaca menggunakan pola / aturan yang berbeda sehingga dihasilkan suatu cipherteks. Contoh teknik transposisi yang cukup populer dalam sejarah kriptografi adalah *scytale* yang berasal dari zaman Romawi Kuno. *Scytale* ialah sebuah silinder dengan diameter tertentu. Sebuah pita kertas dililitkan ke silinder tersebut, kemudian pesan ditulis ke pita dalam arah sejajar poros silinder. Pita kemudian diurai dari silinder sehingga huruf-huruf dalam pesan mengalami transposisi.

Skema yang cukup lengkap mengenai klasifikasi algoritma kriptografi klasik dapat dilihat pada Gambar 1.



Gambar 1 : Klasifikasi Algoritma Kriptografi Klasik

2. Dasar Teori

Bagian ini membahas beberapa algoritma kriptografi klasik yang akan digunakan sebagai landasan dalam merancang algoritma transposisi dengan periodisasi kunci.

Karena algoritma tersebut nantinya akan tergolong algoritma transposisi, bagian ini lebih ditekankan pada algoritma transposisi, sedangkan pembahasan algoritma substitusi dibatasi pada *Vigenère Cipher*, yang akan diadaptasikan ke dalam rancangan algoritma baru.

2. 1. Algoritma Transposisi

Seperti yang telah dijelaskan pada bagian sebelumnya, algoritma kriptografi klasik jenis transposisi bekerja dengan cara mengubah susunan karakter dalam pesan yang dienkripsi. Terdapat banyak cara / aturan dalam mengubah susunan karakter itu, namun sesuai Gambar 1, bagian ini akan membahas tiga jenis utama metode transposisi, yaitu:

1. Transposisi grup
2. Transposisi serial
3. Transposisi kolom / baris

2.1.1. Transposisi Grup

Pada metode ini, plainteks dibagi ke dalam blok-blok / grup yang ukurannya sama. Kemudian, kepada setiap blok pesan ini diaplikasikan suatu susunan karakter yang telah didefinisikan.

Misalkan sebuah susunan karakter didefinisikan pada sebuah grup karakter yang panjangnya delapan, sebagai "mengumpulkan dan mengurutkan karakter bernomor prima dan diikuti sisanya". Maka hasil permutasinya (∏) ialah

$$\prod = c_2, c_3, c_5, c_7, c_1, c_4, c_6, c_8$$

dengan c_n adalah karakter ke-n dalam blok karakter.

Misalkan plainteksnya adalah "TOLONG PERMUTASIKAN PESAN INI YA", maka langkah pertama ialah mengelompokkan plainteks itu ke dalam blok-blok yang panjangnya delapan karakter, sebagai berikut (spasi diperhitungkan)

```
TOLONG P
ERMUTASI
KAN PESA
N INI YA
```

Langkah berikutnya, aturan permutasi yang telah ada (∏), diterapkan ke masing-masing blok pesan, menjadi

```
OLN TOGP
RMTSEUAI
ANPSK EA
 IIYNN A
```

Terakhir, blok-blok pesan itu disatukan kembali menjadi cipherteks yang utuh:

```
OLN TOGPRMTSEUAIANPSK EA IIYNN
A
```

Bila jumlah karakter dalam plainteks bukan kelipatan dari panjang ∏, maka pada akhir pesan dapat ditambahkan (*padding*) karakter-karakter *dummy*. Selain itu, terdapat alternatif lain dalam metode ini, diantaranya dengan membuang spasi.

2.1.2. Transposisi Serial

Metode ini mengelompokkan seluruh karakter plainteks ke dalam beberapa grup dengan aturan tertentu, kemudian cipherteks disusun dengan menyatukan grup-grup tersebut secara berurutan (serial). Misalkan sebuah plainteks P terdiri atas 20 karakter, yakni

$p_1p_2p_3p_4p_5p_6p_7p_8p_9p_{10}p_{11}p_{12}p_{13}p_{14}p_{15}p_{16}p_{17}p_{18}p_{19}p_{20}$

Kemudian didefinisikan tiga grup secara berturut-turut: grup bilangan kelipatan 4, grup bilangan ganjil, dan grup sisa sebagai berikut

$G_1 = p_4p_8p_{12}p_{16}p_{20}$
$G_2 = p_1p_3p_5p_7p_9p_{11}p_{13}p_{15}p_{17}p_{19}$
$G_3 = p_2p_6p_{10}p_{14}p_{18}$

Maka, cipherteks C akan memiliki susunan

$$C = G_1G_2G_3 = p_4p_8p_{12}p_{16}p_{20}p_1p_3p_5p_7p_9p_{11}p_{13}p_{15}p_{17}p_{19}p_2p_6p_{10}p_{14}p_{18}$$

2.1.3. Transposisi Kolom / Baris

Dasar dari metode ini ialah menuliskan plainteks dalam beberapa baris, kemudian cipherteks diperoleh dengan cara membacanya kolom per kolom (karakter). Berikut ialah ilustrasi dari teknik ini.

Plainteks:

```
INI MIRIP OPERASI TRANSPOSE
PADA MATRIKS
```

ditulis dalam 4 baris dan spasi dihilangkan

```
INIMIRIPO
PERASITRA
NSPOSEPAD
AMATRIKS
```

kemudian dibaca kolom per kolom menjadi

```
IPNA NESM IRPA MAOT ISSR RIEI
ITPK PRAS OAD
```

Dari metode dasar tersebut, banyak variasi yang dapat dikembangkan. Salah satunya ialah melakukan pertukaran kolom dengan memanfaatkan kunci. Misalkan teknik ini akan dilakukan pada contoh sebelumnya. Karena ada sembilan kolom, maka kunci yang digunakan panjangnya sembilan karakter. Misalkan kuncinya ialah TRANSPOSE, maka

```
Kunci : T R A N S P O S E
Teks  : I N I M I R I P O
        P E R A S I T R A
        N S P O S E P A D
        A M A T R I K S
```

Pertukaran kolom dilakukan dengan cara mengurutkan karakter-karakter pada kunci menjadi

```
Kunci : A E N O P R S S T
Teks  : I O M I R N I P I
        R A A T I E S R P
        P D O P E S S A N
        A   T K I M R S A
```

kemudian dibaca kolom per kolom menjadi

```
IRPA OAD MAOT ITPK RIEI NESM
ISSR PRAS IPNA
```

Teknik-teknik di atas juga dapat dilakukan secara terbalik: mulai dengan plainteks yang menurun dalam kolom-kolom, kemudian membuat cipherteks dengan membacanya baris per baris. Selain itu, struktur transposisi juga tidak terbatas pada matriks (baris, kolom). Berikut ini ialah contoh transposisi yang memiliki struktur 'zig-zag'.

```
Plainteks: SULIT SEKALI MEMBACA
TEKS INI
```

Disusun menjadi

```
S   T   A   E   C   K   I
 U I S K L M M A A E S N
   L   E   I   B   T   I
```

Dibaca per baris menjadi

```
STAECKI UISKLMMAAESN
LEIBTI
```

2.2. *Vigenère Cipher*

Vigenère Cipher adalah salah satu algoritma kriptografi klasik yang tergolong ke dalam algoritma substitusi. Secara lebih spesifik, *Vigenère Cipher* termasuk *polyalphabetical substitution cipher* (cipher abjad-majemuk). Secara praktis hal ini berarti sebuah karakter di dalam plainteks dapat dienkripsi menjadi karakter yang berbeda pada setiap kemunculannya. Lawan dari jenis cipher ini ialah *monoalphabetical substitution cipher*, dimana suatu karakter plainteks pasti dienkripsi menjadi suatu karakter yang sama pada setiap kemunculannya.

Lebih lanjut, *Vigenère Cipher* adalah cipher yang bersifat periodik. Artinya, proses enkripsi terhadap keseluruhan plainteks mengikuti periode tertentu, yang besarnya sama dengan panjang kunci yang digunakan. Dengan kata lain, terjadi periodisasi kunci terhadap plainteks.

Sifat *polyalphabetic* yang dimiliki oleh *Vigenère Cipher* diimplementasikan dengan menggunakan Bujursangkar *Vigenère* (Gambar 2), sedangkan sifat periodik diwujudkan dengan cara menuliskan kunci yang digunakan secara berulang-ulang sampai sepanjang plainteks, kemudian memadankan setiap karakter ke-n dari plainteks dengan karakter ke-n di rangkaian perulangan kunci, dengan n bernilai dari 1 sampai panjang plainteks. Prosedur enkripsi pada *Vigenère Cipher* dapat dijelaskan melalui contoh sebagai berikut.

Plainteks: `CONTOH ENKRIPSI VIGENERE`
Kunci: `coba`

Langkah 1: Hilangkan spasi dari plainteks

```
CONTOHENKRIPSIVIGENERE
```

Langkah 2: Tuliskan kunci secara periodik

```
CONTOHENKRIPSIVIGENERE
cobacobacobacobacobaco
```

Langkah 3: Enkripsikan setiap karakter plainteks dengan karakter kunci yang berpadanan, menggunakan Bujursangkar *Vigenère*. Caranya, cari perpotongan antara baris karakter kunci dengan kolom karakter plainteks. Jadi, untuk karakter keempat misalnya, cari perpotongan antara baris a dan kolom T. Karakter yang merupakan perpotongan baris dan kolom menjadi substitusi dari karakter plainteks yang bersangkutan. Dengan mengulangi langkah ini untuk setiap karakter plainteks, maka didapatkan cipherteks

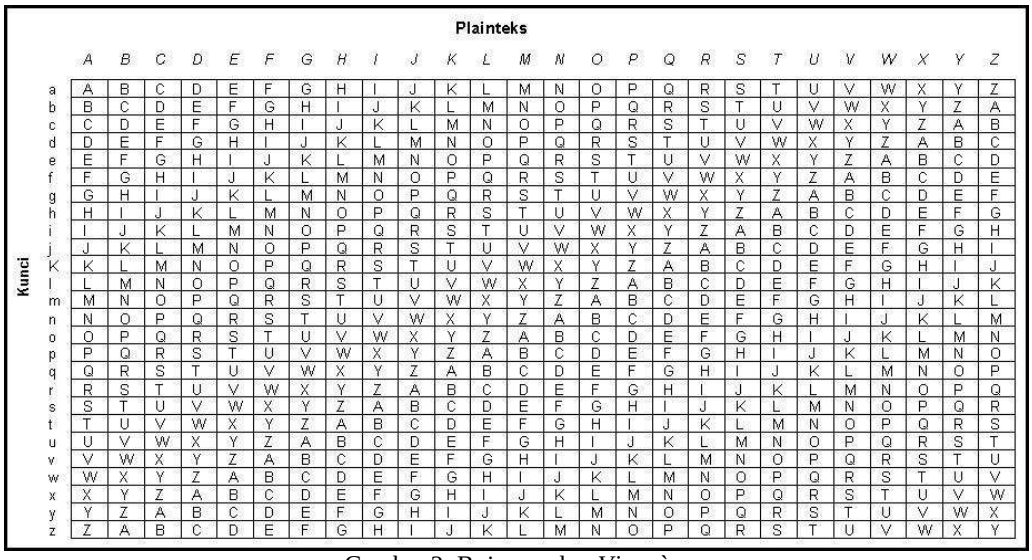
```
ECOTQVFNMFJPUWWIISOETS
```

Selain menggunakan Bujursangkar *Vigenère*, langkah ketiga dapat dilakukan dengan menggunakan rumus

$$|C_n| = (|P_n| + |K_n|) \bmod 26$$

dengan

$|C_n|$: representasi numerik dari karakter cipherteks ke-n
$|P_n|$: representasi numerik dari karakter plainteks ke-n
$|K_n|$: representasi numerik dari karakter kunci (yang berulang) ke-n
n : [1..jumlah karakter plainteks]



Gambar 2: Bujursangkar *Vigenère*

3. Rancangan Algoritma Baru

Bagian ini akan memaparkan rancangan algoritma baru yang akan diklasifikasikan sebagai algoritma transposisi. Algoritma ini akan bekerja dalam satuan karakter, oleh karena itu algoritma ini akan tergolong algoritma kriptografi klasik.

Langkah perancangan diawali dengan pemaparan umum mengenai proses enkripsi dan dekripsi yang dilakukan oleh algoritma baru ini. Setelah itu, prosedur enkripsi dan dekripsi tersebut akan diformalisasikan ke dalam suatu model matematika / formula enkripsi dan dekripsi. Berikutnya, algoritma disusun dalam notasi *pseudocode*. Akhirnya, algoritma baru ini diimplementasikan menjadi sebuah program kecil sehingga pengujian terhadap algoritma baru ini dapat dilakukan.

3.1. Proses Enkripsi dan Dekripsi

Bila diamati sekilas, algoritma baru ini memiliki prosedur enkripsi dan dekripsi yang sangat mirip dengan prosedur enkripsi-dekripsi pada *Vigenère Cipher*. Perbedaan dengan *Vigenère Cipher* akan tampak pada langkah utamanya, yakni substitusi digantikan oleh transposisi.

Prosedur enkripsi pada algoritma baru ini dapat diterangkan melalui contoh sebagai berikut.

Misalkan plainteks P terdiri dari 7 karakter, yaitu

$$P = p_1 p_2 p_3 p_4 p_5 p_6 p_7$$

Sedangkan kunci yang akan digunakan ialah K yang terdiri atas 4 karakter

$$K = k_1 k_2 k_3 k_4$$

Karakter-karakter yang menyusun P dan K di atas akan dipakai apa adanya. Dengan kata lain, tidak ada karakter yang akan diabaikan atau dibuang (misalnya spasi). Hal ini mengakibatkan langkah pertama pada prosedur enkripsi *Vigenère Cipher* tidak dilakukan. Jadi, langkah pertama algoritma ini ialah menuliskan kunci secara berulang / periodik sampai barisan karakter plainteks dan kunci sama panjangnya, seperti ilustrasi berikut

$$P = p_1\ p_2\ p_3\ p_4\ p_5\ p_6\ p_7$$
$$K = k_1\ k_2\ k_3\ k_4\ k_1\ k_2\ k_3$$

Langkah berikutnya, dan yang utama, ialah melakukan transposisi iteratif terhadap setiap posisi karakter pada plainteks. Jadi, pada contoh ini, transposisi karakter akan dilakukan tujuh kali. Adapun transposisi tersebut dilakukan dengan aturan sebagai berikut.

Mulai dari karakter pertama (p_1) dan karakter kuncinya (k_1), pertama-tama k_1 diubah ke dalam representasi numeriknya. Jika dilakukan dalam konteks kriptografi klasik (kakas pena dan kertas), prosedur ini dapat dilakukan dengan mengacu pada suatu tabel (disusun sebelumnya) yang menerjemahkan sebuah karakter menjadi sebuah bilangan bulat. Namun, mengingat algoritma baru ini nanti akan diimplementasikan secara modern, yakni menggunakan program komputer, maka pengubahan karakter menjadi bilangan bulat

ini akan dilakukan berdasarkan pengkodean karakter standar (*character encoding*) yang dipakai oleh bahasa pemrograman.

Demi kemudahan, representasi numerik karakter-karakter kunci pada contoh diberikan sebagai berikut.

$k_1 = 1$
$k_2 = 2$
$k_3 = 3$
$k_4 = 4$

Setelah representasi numerik dari karakter kunci telah diperoleh, langkah berikutnya ialah mencacah karakter plainteks sebanyak nilai karakter kunci yang sedang diproses, dimulai dari karakter setelah karakter plainteks yang sedang diproses. Kemudian, karakter plainteks tempat pencacahan berhenti akan dipertukarkan dengan karakter plainteks yang sedang diproses.

Pada contoh, jika proses enkripsi dilakukan pada p_1, maka pencacahan dilakukan satu kali (sebab $k_1 = 1$) dimulai dari p_2. Pencacahan akan berhenti pada p_2, sehingga pertukaran dilakukan antara p_1 dan p_2, menjadi

$$p_2 p_1 p_3 p_4 p_5 p_6 p_7$$

Jika suatu pencacahan belum selesai namun telah menemui karakter terakhir, maka pencacahan dilanjutkan ke karakter pertama (sirkuler).

Proses ini dilakukan sebanyak jumlah karakter pada plainteks, setiap iterasi ke-n dilakukan terhadap karakter ke-n pada teks hasil iterasi sebelumnya. Jadi, iterasi kedua ($k_2 = 2$) pada contoh di atas akan menghasilkan pertukaran p_1 dengan p_4, menjadi

$$p_2 p_4 p_3 p_1 p_5 p_6 p_7$$

Jika proses dilanjutkan sampai iterasi terakhir, maka akan diperoleh teks

$$p_5 p_4 p_7 p_2 p_3 p_1 p_6$$

Hasil terakhir inilah yang merupakan cipherteks pada contoh kali ini.

Proses dekripsi dilakukan secara terbalik, relatif terhadap proses enkripsi. Artinya, iterasi dimulai dari karakter terakhir pada cipherteks. Meskipun demikian, pencacahan tetap dilakukan maju. Adapun langkah periodisasi kunci identik dengan proses enkripsi. Jadi, bila cipherteks pada contoh sebelumnya akan didekripsi, maka pertama-tama dibuat susunan

$C = p_5\ p_4\ p_7\ p_2\ p_3\ p_1\ p_6$
$K = k_1\ k_2\ k_3\ k_4\ k_1\ k_2\ k_3$

Sebagai gambaran, iterasi pertama dalam proses dekripsi akan menghasilkan

$$p_5 p_4 p_6 p_2 p_3 p_1 p_7$$

Hasil akhir proses dekripsi akan sama dengan plainteks semula.

3.2. Formalisasi

Proses enkripsi dan dekripsi algoritma baru ini berintikan pada pertukaran antara dua karakter pada setiap iterasinya. Berdasarkan paparan proses di atas, aturan pertukaran karakter itu dapat dirumuskan sebagai berikut.

Enkripsi / Dekripsi

$$i' = (i + k_{(i \bmod K)}) \bmod N$$

Pertukaran dilakukan antara karakter ke-i dan karakter ke-i'

Keterangan:
- i : nomor iterasi [1..N]
- k : karakter kunci
- K : panjang kunci
- N : panjang plainteks

3.3. *Pseudocode*

Pseudocode fungsi enkripsi ditampilkan pada Kode 1, sedangkan *pseudocode* fungsi dekripsi pada Kode 2.

3.4. Implementasi

Rancangan algoritma baru ini diimplementasikan menggunakan bahasa pemrograman Java serta kakas NetBeans 5.0. Kode 3 menampilkan potongan kode implementasi tersebut, yakni fungsi enkripsi dan dekripsi yang dapat dipanggil oleh antarmuka melalui method ActionListener (tidak ditampilkan)

```
procedure NewEncrypt(P,K: array of karakter)
{  Melakukan permutasi terhadap elemen array of karakter P
menggunakan kunci K }
Deklarasi
t     : karakter
Ik    : integer
Np,Nk : integer
i,j,k : integer

Algoritma:
   {Inisialisasi}
   Np <- LengthOf(P)
   Nk <- LengthOf(K)
   i <- 0

   {iterasi}
   while (i<Np) do
      k <- i mod Nk              {periodisasi kunci}
      Ik <- IntegerValueOf(Kk)   {representasi numerik dari kunci}
      j <- (i + Ik) mod Np       {target pertukaran}

      {proses pertukaran}
      t <- Pj
      Pj <- Pi
      Pi <- t

      {increment}
      i <- i + 1
   endwhile

   { P telah dipermutasi}
```

Kode 1: Prosedur enkripsi

```
procedure NewDecrypt(C,K: array of karakter)
{  Melakukan de-permutasi terhadap elemen array of karakter C
menggunakan kunci K }
Deklarasi
t     : karakter
Ik    : integer
Nc,Nk : integer
i,j,k : integer

Algoritma:
   {Inisialisasi}
   Nc <- LengthOf(C)
   Nk <- LengthOf(K)
   i <- Nc

   {iterasi}
   while (i>=0) do
      k <- i mod Nk              {periodisasi kunci}
      Ik <- IntegerValueOf(Kk)   {representasi numerik dari kunci}
      j <- (i + Ik) mod Nc       {target pertukaran}
```

Kode 2: Prosedur dekripsi

```
      {proses pertukaran}
      t <- Pj
      Pj <- Pi
      Pi <- t

      {decrement}
      i <- i - 1
   endwhile

   {C telah di de-permutasi menjadi P semula}
```

Kode 2 (lanjutan): Prosedur dekripsi

```
private String Enkripsi(char[] in,char[] key) {
    int i = 0;
    int k,p;
    char tmp;

    int inlen = in.length;
    int keylen = key.length;

    while (i<inlen) {
        k = (int)(key[i%keylen]);
        p = (i+k)%inlen;

        tmp = in[p];
        in[p] = in[i];
        in[i] = tmp;
        i++;
    }

    return (new String(in));
}

private String Dekripsi(char[] in,char[] key) {
    int i,k,p;
    char tmp;

    int inlen = in.length;
    int keylen = key.length;

    i = inlen-1;
    while (i>=0) {
        k = (int)(key[i%keylen]);
        p = (i+k)%inlen;
        //if (p<0) {p = inlen+p;}
        tmp = in[p];
        in[p] = in[i];
        in[i] = tmp;
        i--;
    }

    return (new String(in));
}
```

Kode 3: Fungsi enkripsi dan dekripsi dalam Java

3.5. Pengujian

Untuk menguji program kecil yang telah dibuat, plainteks yang digunakan ialah

```
In George Lucas' Star Wars
trilogy, Jedi Knights were
expected to make their own
light sabers. The message was
clear: a warrior confronted by
a powerful empire bent on
totalitarian control must be
self-reliant. As we face a real
threat of a ban on the
distribution of strong
cryptography, in the United
States and possibly world-wide,
we should emulate the Jedi
masters by learning how to
build strong cryptography
programs all by ourselves. If
this can be done, strong
cryptography will become
impossible to suppress.
```

Kunci yang dipergunakan ialah

```
CipherSaber
```

Hasil enkripsi terhadap plainteks di atas adalah

```
K trsatee se oawyciLastcbofehw
rJibsIadbog stuc ehdeimche,t
tnmeoeGlyoar l r tnwloohw lrhb
psoUssesgi  la e .
sfetlwgetel  a  exro' naf  omar
hietoe oi etsiugnba p
otnrpstmcitrnag oo ye hdettr
rnnltdaSs, os sowsp-i nb,al drh
o i   esetlwoueal au dmr erdrie
lJ an biy otsee t bhg sna
nelgdipauotakrpycrt  omn spy
rhtguelrlaeg dyfsl ve
rsw.blieshe tearroondr  t gn
rcg nnip ohbypesa.ioiosmcetegis
bulislsttaarwepnmpi.ombag coe
iefeornaa:rteAcyr ,w  wearld hr
eroaonrtpSfht  pi
tIrbTnstpttruiaeurfnec Weoa-
yiatnyd lsr
```

Dekripsi menggunakan kunci yang sama menghasilkan

```
In George Lucas' Star Wars
trilogy, Jedi Knights were
expected to make their own
light sabers. The message was
clear: a warrior confronted by
a powerful empire bent on
totalitarian control must be
self-reliant. As we face a real
threat of a ban on the
distribution of strong
cryptography, in the United
States and possibly world-wide,
we should emulate the Jedi
masters by learning how to
build strong cryptography
programs all by ourselves. If
this can be done, strong
cryptography will become
impossible to suppress.
```

Jadi, program implementasi algoritma ini secara umum berjalan dengan baik. Untuk kepentingan analisis, beberapa uji coba berikut juga dilakukan:

a. Dekripsi cipherteks sebelumnya, menggunakan kunci

```
CipherSabeq
```

yang menghasilkan

```
In George rucas' StarnWars
trilooy, Jedi Knights were
xpected toamake theirmown light
fabers. TheAmessageawaa
cletr:oa aariLor eon rot bd ey
t pwgsrfel fmpric bontyonte,ta
it rin ycostrtl u wt re ael-s
el any. s  werfaee  sseal
tdretw of a hanofn the
tisraibutior o estrongocrp
tograpoy,ian the rnieod Statas
nsd possobl tworld-Iid,c we sh
ul hemulatt teb Jedi oasedrs by
leanning hop t  build  trnhg
cryp ogaUphy pr grmes all cy
uirselveb. fw this uanboe donee
sreong crlptgmraphy nil
lbecomebimowssibleitossuppres .
```

b. Dekripsi cipherteks sebelumnya, menggunakan kunci

```
CipherSabes
```

yang menghasilkan

```
In George oucas' StarbWars
trilosy, Jedi Kncghts were
,xpected toymake theirwown
light  abers. The message aas
cletr: o  arrLir cennron td be
i pogwrfue smpiir beotton ettal
tnria nconsrdl m ut br delfs-
eli nw. A swe ra e as eal
```

```
tereawtof a  an fo the
ristarbutioi ofe trongbcry
pograpgy, ai the pnitoe Statss
asn possrblyt orld- idec,we
sheuldh mulato thbeJedi rastdes
by bearnnng hos to  uild utrohn
crypwogrUahy pragraem all oy
oiuselvey. Iwfthis lan ob donen
sterng criptomgaphy tilll
ecomeeimpwosibleato ssppresf.
```

c. Dekripsi cipherteks sebelumnya, menggunakan kunci

```
ciphersaber
```

yang menghasilkan

```
nimeeaaeeLrcufr'r st sa Aa
aligs,aaiemrrtishea  o nheJp
ctrtoG sItty leit cuofnlek
hseWgyss.nag Kyy bge w s g.ear:
a wbxamor confem oed by
acopaerful eraiwe bent
omirotalitap tS contropr.rst be
snoiireliantinte we fac  wdreal
thlort of a b rton the d
nributioneeu strong na-
ptographsosin the Uo  ed
Statertand possi,dy world-wiwe,
we sho ed emulatelthe Jedi
mditers by ll.rning howtgo
build sstong cryptairaphy prot
ams all brrourselvesg If this
cre be done, htrong cry
Tography wnel become sapossible
tl suppressl
```

4. Analisis

Bagian ini menelaah tingkat keamanan data yang dienkripsi menggunakan algoritma yang telah dirancang. Analisis tersebut dilakukan dengan cara pertama-tama melakukan berbagai serangan kriptanalisis dan memeriksa hasilnya. Selain itu, beberapa kasus khusus dalam hal plainteks atau kunci diperiksa supaya dapat diperoleh pengetahuan mengenai perilaku algoritma baru ini.

4.1. Kekuatan Algoritma

Dari formula yang terdapat pada bagian 3.2., terlihat bahwa algoritma ini sangat sederhana. Selain itu, fungsi transposisi tersebut memetakan himpunan karakter penyusun plainteks ke dirinya sendiri. Sifat ini sangat bertentangan dengan prinsip *confusion* dalam teori dari Shannon yang menyatakan bahwa hubungan antara plainteks dan cipherteks (juga kunci) harus dibuat seminimal mungkin. Jadi, dapat dikatakan bahwa algoritma in lemah.

4.2. Kekuatan Kunci

Rancangan algoritma transposisi baru ini memiliki sifat bahwa jika kunci yang digunakan serta panjang plainteks dibuat tetap, maka pola permutasi yang dihasilkan juga pasti tetap. Pembuktian terhadap sifat ini dapat ditelaah dari bagian 3.1. Apapun karakter yang mengisi ketujuh posisi dalam contoh di bagian tersebut, pola cipherteksnya akan selalu sama, selama kunci yang digunakan juga sama. Di sisi lain, berapapun panjang kunci tidak akan berpengaruh terhadap kekuatan cipher sehingga dapat disimpulkan bahwa kekuatan kunci bukanlah kelebihan algoritma ini.

Analisa terhadap hasil pengujian di bagian 3.5 memperlihatkan bahwa perilaku algoritma baru ini mirip dengan perilaku Vigenere Cipher, yakni adanya independensi antar karakter kunci. Terbukti bahwa ketika kunci dekripsi sedikit menyimpang dari kunci enkripsi, sebagian besar isi pesan masih dapat dibaca dan dipahami.

4.3. Kriptanalisis

4.3.1. *Exhaustive Search / Brute Force Attack*

Pada umumnya, serangan kriptanalisis yang tergolong *Exhaustive Search / Brute Force Attack*, bertujuan untuk menemukan kunci yang digunakan untuk melakukan enkripsi sehingga analis dapat melakukan dekripsi. Namun, dalam kriptanalisis terhadap cipher transposisi, jenis serangan ini memiliki arah yang berbeda, yakni langsung mencari kemungkinan plainteks dalam susunan karakter yang benar. Dengan kata lain, analis mencoba melakukan ‘dekripsi’ tanpa kunci. Hal ini disebabkan oleh fakta bahwa karakter-karakter yang menyusun cipherteks transposisi sama dengan yang menyusun plainteksnya. Jenis serangan seperti ini dikenal dengan istilah *anagramming.*

Daya tahan cipher transposisi terhadap jenis serangan ini berbanding lurus dengan panjang plainteksnya. Hipotesis ini dapat memperoleh pembenaran melalui argumen berikut.

Misalkan panjang plainteks adalah N. Di dalam N buah karakter ini diasumsikan terdapat karakter yang berulang. Misalkan ke-26 abjad dapat ditemukan di dalam plainteks itu. Maka, jumlah permutasi dari multiset karakter penyusun plainteks tersebut ialah P, dengan

$$P = \frac{N!}{|'A'|!*|'B'|!*|'C'|!*...*|'Z'|!}$$

Setiap penambahan satu karakter pada plainteks, Nilai N bertambah satu, begitu pula salah satu karakter di bagian penyebut. Jadi, pertumbuhan nilai pembilang lebih besar daripada pertumbuhan nilai penyebut. Hal ini mengakibatkan peluang untuk ditemukannya susunan yang benar diantara permutasi itu semakin mengecil dengan cepat. Secara kasar, bahkan dapat disimpulkan bahwa kekuatan cipher berbanding lurus dengan faktorial dari panjang plainteks.

4.3.2. Analisis Frekuensi

Karena karakter-karakter dalam plainteks sama dengan karakter-karakter dalam cipherteks, maka analisis frekuensi akan menghasilkan statistik yang cocok dengan statistik bahasa yang digunakan untuk menulis plainteks tersebut. Di satu sisi, hal ini menandakan bahwa teknik analisis frekuensi tidak relevan untuk melakukan serangan terhadap cipher transposisi. Namun di sisi lain, kecocokan statistik tersebut memberi indikasi yang cukup kuat bagi analis bahwa cipher yang sedang diserang olehnya merupakan cipher transposisi. Dengan demikian pekerjaan kriptanalis menjadi lebih sederhana.

4.3.3. *Known-Ciphertext Attack*

Jenis serangan ini berusaha untuk mengumpulkan sebanyak-banyaknya cipherteks untuk kemudian dipecahkan dengan cara mencari hubungan antara cipherteks-cipherteks itu. Pemecahan dapat diarahkan untuk menemukan kunci atau langsung menyusun perkiraan plainteks, meski yang terakhirlah yang lebih umum dilakukan pada cipher transposisi. Algoritma transposisi yang baru relatif cukup kuat dalam menghadapi jenis serangan ini mengingat hubungan antar cipherteks baru terjadi ketika kunci yang digunakan identik dan panjang pesan-pesan tersebut sama besar. Adalah sangat mudah untuk menghindari dua hal di atas dalam pemakaian algoritma ini. Sebaliknya, peluang keberhasilan serangan ini menjadi sangat kecil.

4.3.4. *Known-Plaintext Attack*

Jenis serangan ini mengandalkan pengetahuan mengenai struktur plainteks, sebab beberapa dokumen (mis. e-mail, surat perjanjian, dsb) memiliki susunan yang fromatnya terstruktur. Definisi tersebut cukup untuk membuktikan bahwa jenis serangan ini tidak akan bisa diterapkan untuk melakukan serangan terhadap cipher transposisi karena yang terakhir ini justru mengacak struktur pesan tersebut.

4.3.5. *Chosen-Plaintext Attack*

Serangan ini dilakukan dengan memberi umpan beberapa plainteks pilihan kepada sistem kriptografi, kemudian mengambil cipherteks yang berpadanan. Pasangan cipher dan plainteks ini dapat digunakan untuk menemukan kunci dengan mudah.

Algoritma yang baru disusun ini terhitung lemah dalam menghadapi serangan jenis ini. Salah satu umpan yang paling ampuh untuk membongkar kekuatan algoritma ini ialah berupa himpunan (unik) karakter-karakter. Karena keunikan karakter tersebut, hubungan posisional antara plainteks dan cipherteks akan menjadi transparan (misalnya karakter ke-x pada plainteks pasti menjadi karakter ke-y dalam cipherteks.).

--

Secara umum, algoritma transposisi yang baru dirancang ini memiliki tingkat keamanan yang lebih rendah daripada algoritma substitusi. Akan tetapi, bila dibandingkan dengan cipher transposisi yang lain, cipher baru ini lebih baik karena proses permutasinya lebih sukar direkonstruksi tanpa menggunakan kunci.

5. Prospek dan Arah Pengembangan

Setelah mengetahui ide dasar dari algoritma ini, serta hasil analisa terhadap tingkat keamanannya, dapat dikatakan bahwa algoritma transposisi tidak memiliki prospek yang bagus sebagai suatu metode penyandian yang berdiri sendiri. Seperti yang telah dipaparkan pada bagian Analisis, kelemahan utama cipher transposisi ini memang terletak pada ketertutupan operasinya (pemetaan ke diri sendiri) sehingga usaha kriptanalisis terhadap suatu cipher transposisi menjadi lebih sederhana.

Bagaimanapun, metode transposisi tetap memiliki manfaat, jika dikombinasikan dengan teknik enkripsi yang lain. Bagian ini akan membahas kemungkinan-kemungkinan sinergi antara metode transposisi, khususnya metode baru yang dirancang dalam makalah ini, dengan metode yang lain.

5.1. Super Enkripsi

Pengertian dari super enkripsi ialah penggunaan secara bersama-sama, dalam satu algoritma, teknik substitusi dan teknik transposisi untuk menyandikan pesan. Dengan integrasi ini, kelemahan masing-masing teknik, bila berdiri sendiri, dapat dikurangi. Di sisi teknik transposisi, ketertutupan operasinya

dieliminasi oleh teknik substitusi sehingga range operasi enkripsi menjadi lebih terbuka, yang pada akhirnya mempersulit usaha kriptanalisis.

Karena kedekatannya, metode transposisi yang baru dirancang ini akan sangat cocok diimplementasikan bersama-sama dengan Vigenere Cipher. Sebagai contoh, dapat disusun langkah enkripsi yang melakukan substitusi (Langkah Vigenere) dilanjutkan dengan pertukaran (Langkah transposisi) dalam satu langkah iterasi yang sama. Secara teoretis, gabungan cipher ini cukup sulit untuk dipecahkan.

5.1. Kriptografi Modern

Algoritma kriptografi modern melakukan penyandian data pada level bit atau byte, yang lebih kecil daripada sebuah karakter. Namun demikian, prinsip-prinsip dasar yang digunakan tidak berbeda dengan kriptografi klasik, yakni teknik substitusi dan transposisi. Diantara dua teknik dasar itu, transposisi lebih mudah diadaptasikan ke dalam algoritma kriptografi modern karena metode transposisi tidak peduli terhadap semantik objek yang dipertukarkan posisinya: apakah karakter, angka, bit, dan sebagainya. Sebaliknya, algoritma substitusi sedikit banyak harus mempertimbangkan hal itu agar operasi yang dilakukan benar dan reversibel.

Berdasarkan pertimbangan di atas, penyisipan langkah transposisi yang mendasari algoritma baru ini, ke dalam algoritma kriptografi modern apapun, menjadi sangat mudah. Selain itu pengayaan algoritma modern menggunakan teknik transposisi sederhana ini mampu meningkatkan level kekuatan cipher tanpa meningkatkan biaya komputasi secara signifikan.

6. Kesimpulan

Algoritma kriptografi klasik memang terlalu sederhana jika diterapkan sendiri-sendiri. Namun, kekuatannya akan meningkat bila dikombinasikan, baik dengan sesama algoritma klasik maupun dengan algoritma modern.

Rancangan algoritma transposisi dengan kunci periodik memang lemah aspek keamanannya. Namun, kesederhanaannya membuat adaptasi dan integrasi dengan algoritma lain menjadi mudah, sehingga membuka peluang baru bagi tingkat keamanan data.

6. Pustaka

http://www.criptored.upm.es/guiateoria/gt_m001a_en.htm

http://en.wikipedia.org/

www.skypoint.com/~waltzmn/Cryptography.html

http://informatika.org/~rinaldi/Kriptografi/2006-2007/