STUDI DAN PERBANDINGAN ALGORITMA *ADFGVX CIPHER* DENGAN ALGORITMA *PLAYFAIR CIPHER* PADA PERANG DUNIA I

Rezza Mahyudin – NIM : 13505055

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-mail : if15055@students.if.itb.ac.id

Abstrak

Makalah ini mengulas secara mendalam algoritma ADFGVX *cipher* yang digunakan oleh pihak tentara Jerman ketika Perang Dunia I berlangsung. ADFGVX *cipher* merupakan salah satu dari sejumlah algoritma yang paling terkenal dalam seluruh sejarah kriptografi. Algoritma ADFGVX diciptakan dengan mengambil sebuah ide yang muncul pada masa lampau yaitu menghubungkan huruf-huruf alphabet dengan posisi-posisi yang terdapat di dalam sebuah jaringan atau tabel.

Algoritma kriptografi lain yang juga akan dibahas secara mendalam pada makalah ini adalah algoritma Playfair *cipher* yang digunakan oleh pihak tentara Inggris ketika Perang Dunia I berlangsung. Algoritma Playfair *cipher* ini termasuk ke dalam keluarga algoritma *cipher* subtitusi yang pertama kali diperkenalkan oleh Julius Caesar.

Selain pembahasan secara mendalam kedua algoritma tersebut, makalah ini juga akan membahas langkah-langkah di dalam melakukan pemecahan kedua algoritma kriptografi tersebut. Langkah-langkah pemecahan algoritma kriptografi ini akan menjadi dasar di dalam melakukan pembandingan penggunaan kedua algoritma pada Perang Dunia I pada bagian akhir makalah.

***Kata kunci**: ADFGVX cipher*, *Playfair chiper*, *world war I*, *Fritz Nebel*, *Georges Painvin*, *Sir Charles Wheatstone, Classic Crypthografi*, enkripsi, dekripsi.

1. Pendahuluan

Kriptografi adalah suatu ilmu dan seni untuk menjaga kerahasiaan pesan dengan cara menyandikannya ke dalam bentuk yang tidak dapat dimengerti lagi maknanya. Ilmu kriptografi sudah mulai dipelajari manusia sejak tahun 400 SM, yaitu pada zaman Yunani kuno. Dari catatan yang ada, dapat diperkirakan bahwa "Penyandian Transposisi" merupakan sistem kriptografi pertama yang digunakan atau dimanfaatkan. Bidang ilmu ini terus berkembang seiring dengan kemajuan peradaban manusia, dan memegang peranan penting dalam strategi peperangan yang terjadi dalam sejarah manusia.

Ada empat tujuan mendasar dari ilmu kriptografi ini yang juga merupakan aspek keamanan informasi yaitu :

- Kerahasiaan, adalah layanan yang digunakan untuk menjaga isi dari informasi dari siapapun kecuali yang memiliki otoritas atau kunci rahasia untuk membuka/mengupas informasi yang telah disandi.
- Integritas data, adalah berhubungan dengan penjagaan dari perubahan data secara tidak sah. Untuk menjaga integritas data, sistem harus memiliki kemampuan untuk mendeteksi manipulasi data oleh pihak-pihak yang tidak berhak, antara lain penyisipan, penghapusan, dan pensubsitusian data lain kedalam data yang sebenarnya.
- Autentikasi, adalah berhubungan dengan identifikasi ataupun pengenalan, baik secara kesatuan sistem maupun informasi itu sendiri. Dua pihak yang saling berkomunikasi harus saling memperkenalkan diri. Informasi yang dikirimkan melalui kanal harus diautentikasi keaslian, isi datanya, waktu pengiriman, dan lain-lain.
- Non-repudiasi, atau nirpenyangkalan adalah usaha untuk mencegah terjadinya penyangkalan terhadap pengiriman atau

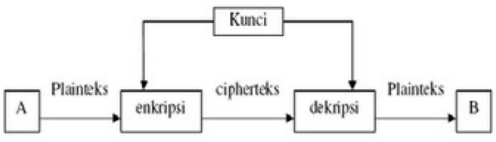
terciptanya suatu informasi oleh yang mengirimkan atau yang membuat pesan.

Ilmu kriptografi sendiri memiliki sejarah yang cukup panjang. Tercatat, pada 4000 tahun yang lalu, bangsa Mesir telah mengenal kriptografi melalui *Hyroglyph* yang tidak standard. Selanjutnya bangsa Sparta di Yunani juga menggunakan kriptografi pada permulaan tahun 400 SM. Mereka menggunakan alat kriptografi yang disebut dengan *Scytale*. Kemudian ilmu kriptografi mengambil peran yang cukup penting ketika Perang Dunia I dan Perang Dunia II berlangsung. Bahkan, beberapa ahli sejarah meyakini, keberhasilan pihak sekutu memenangi perang tidak lepas dari keberhasilan mereka dalam memecahkan algoritma kriptografi yang digunakan oleh pihak Jerman.

Ciri khas yang dimiliki oleh kriptografi klasik adalah sifatnya yang menitikberatkan kekuatan kriptografi pada kerahasiaan algoritma yang digunakan. Hal ini menyebabkan apabila algoritma yang digunakan telah diketahui maka pesan sudah jelas "bocor" dan dapat diketahui isinya oleh siapa saja yang mengetahui algoritma tersebut.

Kriptografi sendiri terdiri dari 2 buah proses utama yaitu proses enkripsi dan proses dekripsi. Proses enkripsi merubah plainteks menjadi cipherteks dengan menggunakan kunci tertentu. Sedangkan proses dekripsi adalah kebalikan dari proses enkripsi yaitu proses merubah cipherteks menjadi pesan awal (plainteks).

Skema proses enkripsi dan dekripsi secara sederhana dapat dilihat pada Gambar 1.



Gambar 1. Skema Proses Enkripsi dan Dekripsi

Ilmu kriptografi ikut memiliki andil pada saat Perang Dunia I berlangsung. Pada perang tersebut, masing-masing pihak yang bertikai saling mengembangkan algoritma kriptografi mereka sendiri demi ketersampaian pesan penting tanpa diketahui oleh pihak lawan. Algoritma kriptografi yang cukup dikenal dan dikembangkan ketika Perang Dunia I berlangsung adalah Algoritma ADFGVX cipher yang dikembangkan oleh pihak Jerman dan algoritma Playfair cipher yang dikembangkan oleh pihak Inggris.

2. Sejarah Singkat Perang Dunia I

Perang Dunia I adalah konflik militer dunia yang melibatkan mayoritas kekuatan-kekuatan dunia yang ada pada saat itu, yang selanjutnya terorganisasi menjadi 2 persekutuan militer yang saling berperang. Lebih dari 70 juta personel militer terlibat dalam perang yang merupakan terbesar sepanjang sejarah pada saat itu.

Penyebab utama terjadinya Perang Dunia I adalah pembuhunan terhadap pangeran Austria Franz Ferdinand oleh sekelompok pemberontak Serbia. Austria kemudian menyatakan perang kepada Serbia, dan dalam hitungan minggu, hampir seluruh kekuatan besar Eropa menyatakan terlibat dalam perang ini. bahkan, karena kekuasaan beberapa negara Eropa yang memiliki jajahan yang tersebar di seluruh dunia, perang inipun ikut menyebar dan melibatkan negara-negara lain di luar Eropa.

Pada Perang Dunia pertama ini, mereka yang terlibat perang tidak hanya berasal dari kalangan militer. Kalangan industri dan kalangan peneliti / *scientist* pun ikut terlibat dalam peperangan ini. tidak terkecuali adalah peneliti / ilmuwan yang berkecimpung di dalam dunia kriptografi. Bahkan kriptografi menjadi salah satu unsur yang dianggap penting dalam ketersampaian dan kerahasiaan pesan militer di kedua pihak yang bertikai.

Kriptografi klasik memasuki masa yang penting di dalam sejarah perkembangannya pada Perang Dunia I. hal ini ditandai dengan pengembangan dan penggunaan algoritma kriptografi di kedua pihak yang terlibat perang. Dua algoritma yang dikembangkan dan populer pada saat itu adalah algoritma ADFGVX *cipher* dan algoritma Playfair *cipher*.

3. Algoritma ADFGVX *Cipher*

Algoritma kriptografi ADFGVX yang digunakan oleh tentara Jerman pada Perang Dunia I adalah merupakan salah satu algoritma yang paling dikenal dalam sejarah kriptografi klasik. Algoritma ini ditemukan oleh seorang petugas radio tentara Jerman yang bernama Fritz Nebel (1891 - 1967). Algoritma ini pertama kali muncul pada tanggal 5 Maret 1918 ketika pihak

Jerman menggunakannya dalam sebuah transmisi pesan nirkabel di medan perang di bagian barat Eropa.

Dinamakan algoritma ADFGVX karena chiperteks hasil enkripsi pesan tentara Jerman hanya mengandung enam karakter alphabet tadi. Pada awalnya algoritma ini hanya menggunakan 5 karakter saja. Namun pada perkembangannya ditambahkan karakter huruf X agar algoritma ini dapat menangani 26 huruf alphabet dan 10 angka. Huruf-huruf A, D, F, G, V, dan X sendiri dipilih karena representasi huruf-huruf tersebut dalam sandi morse sangatlah berbeda dan oleh karenanya memperkecil kemungkinan terjadinya kesalahan dalam penerimaan pesan.

Algoritma ADFGVX *cipher* menggunakan tabel 6 x 6 yang berisi 26 huruf dan 10 angka (0-9). Enkripsinya terdiri dari dua proses, yaitu proses substitusi dan proses transposisi. Selain itu Setiap proses tersebut membutuhkan sebuah kunci.

3.1 Langkah-Langkah ADFGVX *Cipher*

Berikut ini adalah langkah-langkah dalam mengenkripsi sebuah pesan plainteks dengan menggunakan algoritma ADFGVX *Cipher*,

1. Tentukan kunci pertama yang terdiri dari huruf dan angka, misalkan "math08". Jika ada huruf yang berulang, maka cukup satu huruf yang muncul pertama yang dituliskan.
2. Buatlah sebuah tabel 6 x 6 dan isi dengan kunci pertama, kemudian huruf-huruf berurutan yang belum muncul, dan selanjutnya angka-angka berurutan yang belum muncul. Tabel berikut merepresentasikan tabel yang terbentuk dengan kunci "math08",

	A	D	F	G	V	X
A	m	a	t	h	0	8
D	b	c	d	e	f	g
F	i	j	k	l	n	o
G	p	q	r	s	u	v
V	w	x	y	z	1	2
X	3	4	5	6	7	9

3. Selanjutnya, setiap huruf dalam plainteks disubstitusi menjadi dua huruf yang ditentukan oleh posisi baris dan kolom. Sebagai contoh, huruf k menjadi FF, serta huruf g menjadi DX. Misalkan plainteksnya adalah "belajar sandi", maka hasil substitusinya adalah "DA DG FG AD FD AD GF GG AD FV DF FA".
4. Tentukan kata kunci kedua, terdiri dari huruf saja, dan boleh muncul berulang. Kunci ini digunakan dalam proses transposisi. Pertama buatlah sebuah tabel baru. Kemudian tulis kata yang menjadi kunci di bagian atas setiap kolomnya. Selanjutnya tulis hasil substitusi pada langkah 3 di bawahnya secara berurutan ke kanan lalu ke bawah. Jika ada sisa, diisi dengan huruf X atau sesuai dengan kesepakatan.
   Sebagai contoh, kata kunci kedua yang kita gunakan adalah "kunci". Maka tabel yang terbentuk adalah sebagai berikut,

1	2	3	4	5
k	u	n	c	i
D	A	D	G	F
G	A	D	F	D
A	D	G	F	G
G	A	D	F	V
D	F	F	A	X
X	X	X	X	X

5. Selanjutnya, urutkan huruf pada kunci kedua terurut sesuai dengan alfabet. Sebagai contoh, jika kunci kedua yang dipilih adalah kata "matahari" (1-2-3-4-5-6-7-8), menjadi "aaahimrt" (2-4-6-5-8-1-7-3).
   Hal yang sama diterapkan pada kata kunci kedua yang telah kita pilih yaitu kata "kunci" menjadi "ciknu" (4-5-1-3-2). Sehingga tabel menjadi,

4	5	1	3	2
c	i	k	n	u
G	F	D	D	A
F	D	G	D	A
F	G	A	G	D
F	V	G	D	A
A	X	D	F	F
X	X	X	X	X

6. Cipherteksnya adalah huruf-huruf yang berada di kolom pertama, dan seterusnya. Jadi, untuk contoh yang kita pilih, cipherteks yang dihasilkan adalah

"GFFFAX FDGVXX DGAGDX DDGDFX AADAFX".

3.2 Implementasi ADFGVX *Cipher*

Implementasi algoritma ADFGVX dilakukan melalui kode-kode PHP. Bentuk implementasi tersebut adalah sebagai berikut,

```
function encode($keyword,$msg){
  $keyword =
remove_duplicate_letters($keyword);
  $partially_done = encode_step1($msg);
  return
encode_step2($keyword,$partially_done);
}

function encode_step1($msg){

  global $ADFGVX; //Golbally defined
symbols
  global $grid;   //Globally defined grid

  // Form encoding map from the grid
  $encode=array();
  for ($i=0;$i<count($grid);$i++){
    for ($j=0;$j<count($grid);$j++){
      $encoding_map[$grid[$i][$j]] =
$ADFGVX[$i].$ADFGVX[$j];
    }
  }

  //Change message to upper case if using
original ADFGVX encoding
  if(!isset($_POST[tenbyten])) $msg =
strtoupper($msg);

  $plain = preg_split("//",$msg, -1,
PREG_SPLIT_NO_EMPTY);

  $encoded=array();
  foreach ($plain as $v){
    $encoded[] = $encoding_map[$v];
  }
  return join('',$encoded);
}

function encode_step2($keyword,
$partially_done){

  global $ADFGVX;
  $partially_done =
preg_split("//",$partially_done, -1,
PREG_SPLIT_NO_EMPTY);

  $KEY = preg_split("//",$keyword, -1,
PREG_SPLIT_NO_EMPTY);
  $KEY_sorted = preg_split("//",$keyword,
-1, PREG_SPLIT_NO_EMPTY);
  sort($KEY_sorted);

  //Charater to Column map
  $N = count($KEY);
  for ($i=0;$i<$N;$i++){
    $A[$KEY_sorted[$i]] = $i;
  }

  // Row-major filling of table using the
column scramble map
  $row=-1;
  for ($j=0; $j<count($partially_done);
$j++){
    if ($j%$N==0) $row++;
    $table[$row][$A[$KEY[$j%$N]]]=
$partially_done[$j];
  }

  // Column-major ordering
  $M = count($table);
  for ($c=0; $c<$N; $c++){
    for($R=0; $R<$M;$R++){
      $encoded[$R+$c*$M]=$table[$R][$c];
    }
  }

  // Make array a string and adding
spaces
  $encoded = join('',$encoded);
  $encoded = preg_replace("/(....)/","$1
",$encoded);
  return $encoded;
}
```

3.3 Langkah-Langkah Pemecahan ADFGVX

1. Lakukan pemisahan cipherteks dengan cara membagi panjang pesan / cipherteks yang ada dengan panjang kunci yang kita miliki. Pada titik ini, pengetahuan tentang kunci pesan mutlak kita miliki. Umunya kriptanalis menggunakan statistik sosial dalam menebak kunci pesan tersebut.
   Sebagai contoh, kita memiliki pesan cipherteks,

   "AGAGFXGAGAXFXVAVGXGXGAFV DGDFGAAG"

   Dan misalkan kita telah mengetahui kunci yang digunakan adalah "DORAEMON", maka kita dapat menduga tiap kelompok terdiri dari 32 / 8 = 4 karakter,

   "AGAG FXGA GAXF XVAV GXGX GAFV DGDF GAAG"

2. Selanjutnya, urutkan huruf pada kunci terurut sesuai dengan alfabet (sama dengan langkah ke-5 proses enkripsi plainteks). Untuk contoh kali ini kita dapatkan "ADEMNOOR" (4-1-5-6-8-2-7-3).

4	1	5	6	8	2	7	3
A	**D**	**E**	**M**	**N**	**O**	**O**	**R**
A	F	G	X	G	G	D	G
G	X	A	V	X	A	G	A
A	G	X	A	G	F	D	A
G	A	F	V	X	V	F	G

3. Urutkan kolom-kolom yang kita bentuk menjadi kata kunci yang kita miliki,

1	2	3	4	5	6	7	8
D	**O**	**R**	**A**	**E**	**M**	**O**	**N**
F	G	G	A	G	X	D	G
X	A	A	G	A	V	G	X
G	F	A	A	X	A	D	G
A	V	G	G	F	V	F	X

4. Lakukan pembacaan secara berurutan ke kanan lalu ke bawah,

   "FGGAGXDGXAAGAVGXGFAAXADG AVGGFVFX"

5. Langkah terakhir adalah mencari padanan karakter untuk setiap 2 huruf teks yang kita dapatkan di atas dengan tabel ADFGVX yang kita miliki,

	A	**D**	**F**	**G**	**V**	**X**
A	K	Z	W	R	I	F
D	9	B	6	C	L	S
F	Q	7	J	P	G	X
G	E	V	Y	3	A	N
V	8	O	D	H	0	2
X	U	4	1	S	T	M

   Setelah mencari padanan karakter pada tabel, kita dapatkan plainteks "PENCURINYA KUCING".

4. Algoritma Playfair *Cipher*

Algoritma Playfair *cipher* adalah algoritma kriptografi yang dikembangkan oleh ahli fisika berkebangsaan Inggris yang bernama Sir Charles Wheatstone (1802 - 1875). Algoritma ini dinamakan Playfair untuk menghargai jasa teman dari Wheatstone yang bernama Lyon Playfair yang telah membantunya mempopulerkan algoritma tersebut melalui usahanya dalam melobi pemerintah Inggris untuk menggunakannya secara resmi.

Algoritma ini digunakan untuk tujuan taktik oleh tentara Inggris pada Perang Dunia I. Algoritma ini dipilih karena algoritma ini terbilang cukup cepat untuk digunakan dan tidak membutuhkan peralatan khusus apapun. Skenario umum penggunaan algoritma Playfair adalah untuk melindungi pesan yang penting namun tidak kritikal selama perang berlangsung. Oleh karenanya, seandainya kriptanalis musuh dapat memecahkan algoritma itu, informasi yang mereka dapatkan tidaklah informasi yang penting bagi mereka.

Algoritma Playfair adalah merupakan algoritma digraphs *cipher*, yang artinya setiap proses enkripsi dilakukan pada setiap dua huruf. Misalkan plainteksnya "kriptologi", maka proses enkripsi dilakukan terhadap "kr ip to lo gi".

Kunci kriptografinya adalah 25 buah huruf alfabet yang disusun di dalam tabel 5 x 5 dengan menghilangkan huruf J dari abjad. Huruf J dianggap sama dengan huruf I, sebab dalam Bahasa Inggris, huruf J mempunyai frekuensi kemunculan yang paling kecil. Setiap elemen tabel berisi huruf yang berbeda satu sama lain.

4.1 Langkah-Langkah Playfair *Cipher*

Pesan yang akan dienkripsi diatur terlebih dahulu sebagai berikut,

1. Ganti huruf J (bila ada) dengan huruf I
2. Tulis pesan dalam pasangan huruf (*bigram*)
3. Pastikan tidak ada pasangan huruf yang sama. Jika ada, sisipkan hurf Z di tengahnya.
4. Jika jumlah huruf ganjil, tambahkan huruf Z di akhir pesan.

Contoh plainteks : "good brooms sweep clean"

➔ Tidak ada huruf J, maka langsung tulis pesan dalam pasangan huruf,
GO OD BR OZ OM SZ SW EZ EP CL EA NZ

Sementara itu, hal yang harus diperhatikan dalam menentukan kunci yang akan digunakan untuk algoritma Playfair ini adalah,

1. Kunci yang digunakan berupa kata dan tidak ada huruf sama yang berulang
2. Kunci dapat dipilih dari sebuah kalimat yang mudah diingat, misalnya

JALAN GANESHA SEPULUH

3. Buang huruf yang berulang dan huruf J jika ada

   ALNGESHPU

4. Lalu tambahkan huruf-huruf yang belum ada (kecuali J),

   ALNGESHPUBCDFIKMOQRTVWXYZ

5. Hasilnya masukkan ke dalam tabel, kemudian perluas tabel dengan menambahkan kolom dan baris ke-6,

A	L	N	G	E	A
S	H	P	U	B	S
C	D	F	I	K	C
M	O	Q	R	T	M
V	W	X	Y	Z	V
A	L	N	G	E	

Algortima enkripsi Playfair *cipher* sendiri adalah sebagai berikut,

1. Jika ada dua huruf yang terdapat pada baris kunci yang sama, maka tiap huruf diganti dengan huruf di kanannya (pada kunci yang sudah diperluas)
2. Jika dua huruf terdapat pada kolom kunci yang sama, maka tiap huruf diganti dengan huruf dibawahnya (pada kunci yang sudah diperluas)
3. Jika dua huruf tidak pada kolom yang sama dan baris yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua.
4. Huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang terbentuk dari ketiga huruf yang sudah digunakan sampai sejauh ini.
   Sebagai contoh, dengan menggunakan kunci

S	T	A	N	D	S
E	R	C	H	B	E
K	F	G	I	L	K
M	O	P	Q	U	M
V	W	X	Y	Z	V
S	T	A	N	D	

Kita akan mengenkripsi plainteks

GO OD BR OZ OM SZ SW EZ EP CL EA NZ

Menjadi chiperteks

FP UT EC UW PO DV TV BV CM BG CS DY

4.2 Implementasi Playfair *Chiper*

Implementasi algoritma Playfair dilakukan melalui kode-kode C#. Bentuk implementasi tersebut adalah sebagai berikut,

```
using System;
using System.Text;

public class Playfair
{

    public static string Prepare(string
originalText)
    {
       int length = originalText.Length;
       originalText =
originalText.ToLower();
       StringBuilder sb = new
StringBuilder();

       for(int i = 0; i < length; i++)
       {
          char c = originalText[i];
          if (c >= 97 && c <= 122)
          {
             if (sb.Length % 2 == 1 &&
sb[sb.Length - 1] == c)
             {
                sb.Append('x');
             }
             sb.Append(c);
          }
       }

       if (sb.Length % 2 == 1)
       {
          sb.Append('x');
       }

        return sb.ToString();
    }

public static string Encipher(string key,
string plainText)
    {
       int length = plainText.Length;
       char a,b;
       int a_ind, b_ind, a_row, b_row,
a_col, b_col;
       StringBuilder sb = new
StringBuilder();

       for(int i = 0; i < length; i+=2)
       {
          a = plainText[i];
          b = plainText[i+1];

          a_ind = key.IndexOf(a);
```

```
            b_ind = key.IndexOf(b);
            a_row = a_ind / 5;
            b_row = b_ind / 5;
            a_col = a_ind % 5;
            b_col = b_ind % 5;
            if(a_row == b_row)
            {
               if(a_col == 4)
               {
                 sb.Append(key[a_ind -4]);
                 sb.Append(key[b_ind + 1]);
               }
               else if(b_col == 4)
               {
                 sb.Append(key[a_ind + 1]);
                 sb.Append(key[b_ind - 4]);
               }
               else
               {
                 sb.Append(key[a_ind + 1]);
                 sb.Append(key[b_ind + 1]);
               }
            }
            else if(a_col == b_col)
            {
               if(a_row == 4)
               {
                 sb.Append(key[a_ind- 20]);
                 sb.Append(key[b_ind + 5]);
               }
               else if(b_row == 4)
               {
                 sb.Append(key[a_ind + 5]);
                 sb.Append(key[b_ind- 20]);
               }
               else
               {
                 sb.Append(key[a_ind + 5]);
                 sb.Append(key[b_ind + 5]);
               }
            }
            else
            {
               sb.Append(key[5*a_row +
b_col]);
               sb.Append(key[5*b_row +
a_col]);
            }
         }
         return sb.ToString();
      }
```

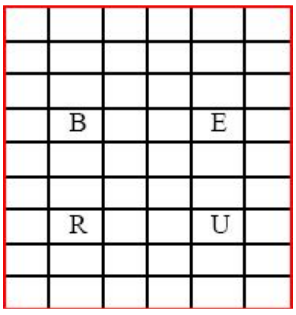
4.3 Langkah-Langkah Pemecahan Playfair

Salah satu metode yang cukup dikenal mampu memecah ciperteks hasil algoritma Playfair adalah metode analisis pasangan huruf. Namun di dalam menggunakan metode analisis ini diperlukan beberapa asumsi yang berlaku seperti,

- Pengetahuan mengenai algoritma yang digunakan (Playfair *cipher*)
- Pengetahuan mengenai bidang yang digemari oleh pembuat cipherteks
- Pengetahuan mengenai cara pembuatan kunci

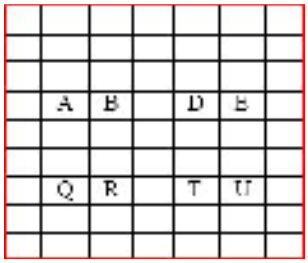
Langkah-langkah di dalam melakukan metode analisis pasangan huruf adalah sebagai berikut,

1. Cari kata yang mungkin berulang secara utuh dan tidak dipengaruhi oleh bentuk digraph Playfair *cipher*
   Contoh cipherteks :
   UQ SM BV DV UB UK DQ BU SI
   BU CU PN SE NS TO UC SI DQ DB
   QT AE DW SI DQ UB UK DQ BU
   Di sini terlihat bahwa ada pengulangan pasangan kata UB, UK, DQ, BU, SI lalu adanya pembalikan pasangan seperti UB→BU, CU→UC
2. Dengan mengetahui bidang yang digeluti oleh pembuat cipher yaitu jaringan komputer, tentukan beberapa kosakata yang mungkin dipakai seperti RE CE IV ER, RE PE AT ER , RE ND ER ER dan lain-lain.
3. Lakukan asumsi bahwa
   UB UK DQ BU→RE PE AT ER (1)
   Dari poin ini, dapat disimpulkan bahwa dalam tabel kunci huruf R dan E tidak terletak dalam baris dan kolom yang sama
4. Cari digraph berikutnya yang berdekatan dengan kata tersebut sehingga didapatkan
   SI BU CU → … ER …
5. Lalu dari poin ini kita gunakan digraph yang lain
   SI DQ … SI DQ
   Di baris paling bawah di cipherteks, SI DQ terletak sebelum UB UK DQ BU sedangkan UB UK DQ BU telah diterjemahkan menjadi kata benda yaitu RE PE AT ER sehingga besar kemungkinan SI DQ merupakan cipher dari petunjuk kata benda
   SI DQ → THAT, THIS
   Ini juga diperkuat dengan
   SI BU CU → … ER … → TH ER E
   Lalu kita asumsikan
   SI → TH
   Maka tabel kunci sementara yang mungkin dibentuk (untuk sementara pembentukan tabel kunci tidak dalam tabel 5 x 5)

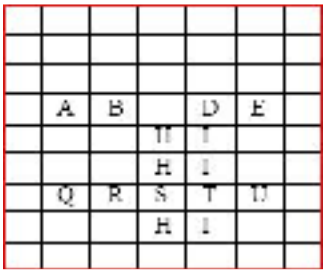


Ini didapatkan dari persamaan (1) yang menjelaskan bahwa B dan U tidak terletak pada baris dan kolom yang sama.

6. Lalu dari persamaan (1) juga diasumsikan
   DQ → AT
   Kita berspekulasi dengan meletakkan huruf A di sebelah huruf B dan meletakkan pasangannya huruf T di antara R dan U. Ini semakin diperkuat dengan posisi D dan Q yang memenuhi aturan pembuatan kunci sehingga



7. Selajutnya dari persamaan (2), kriptanalis mengasumsikan bahwa S terletak di antara R dan T sehingga tabel kunci yang mungkin



8. Sampai sini sudah banyak yang bisa dilakukan oleh kriptanalis seperti dengan *analytical attack* yaitu mencari kosakata lain dalam bahasa inggris lalu dengan digraph yang ada mencoba mencari kombinasi kunci yang mungkin dan mencocokkannya dengan tabel kunci.

5. Perbandingan ADFGVX Dengan Playfair

Kelebihan algoritma ADFGVX *cipher* terletak pada fakta bahwa algoritma ini berbeda dengan algoritma klasik lainnya dimana frekuensi tiap huruf seperti frekuensi huruf E tidaklah mudah untuk dikenali. Selanjutnya, kekuatan algoritma ini menjadi lebih ketika sistem transposisi diterapkan. Namun, kekuatan terbesar algoritma ini terletak pada kunci yang digunakan. Tentara Jerman menggunakan seluruh 26 huruf dalam alfabet dan 10 angka yang tersusun secara random sebagai kuncinya. Hal ini diperkuat lagi dengan kenyataan bahwa pihak Jerman mengganti kunci yang digunakan oleh mereka setiap harinya.

Algoritma ini berhasil dipecahkan oleh seorang kriptanalis berkebangsaan Perancis yang bernama Painvin. Dia berhasil memecahkan algoritma ADFGVX pada bulan April 1918, dan oleh karenanya rencana serangan besar Jerman menjadi tidak efektif. Namun Paivin hanya memecahkan algoritma ini dalam 2 kasus spesial saja, karena hingga tahun 1933 tidak ditemukan solusi umum pemecahan algoritma ini.

Playfair Cipher memiliki keunggulan tersendiri dibanding algoritma enkripsi klasik lainnya, karena dalam pemrosesannya algoritma ini menggunakan kombinasi dua huruf (Digraph) sehinggga Teknik Analisis Frekuensi sangat sukar digunakan untuk seorang kriptanalis menyerang algoritma ini. Ini disebabkan oleh banyaknya kombinasi yang diberikan oleh digraph (monograph = 26 sedangkan digraph = 26 x 25). Namun apabila cipherteks yang dimiliki cukup panjang, maka sama seperti algoritma klasik lainnya, Playfair cukup mudah untuk dipecahkan oleh kriptanalis. Hal ini karena Playfair *cipher* masih menyimpan adanya perulangan bentuk kata tidak seperti ADFGVX.

6. Kesimpulan

Kesimpulan yang dapat diambil dari studi dan perbandingan algoritma ADFGVX *cipher* dengan algoritma Playfair *cipher* adalah

1. Algoritma ADFGVX lebih baik dari algoritma Playfair untuk menyembunyikan pesan ketika Perang Dunia I berlangsung.
2. Algoritma Playfair sebaiknya digunakan untuk menyamarkan pesan penting yang dibutuhkan secara cepat, namun bukan merupakan pesan yang kritikal.

7. Daftar Pustaka

[1] Munir, Rinaldi. (2004). Bahan Kuliah IF5054 Kriptografi. Departemen Teknik Informatika, Institut Teknologi Bandung.

[2] Crypthology Club GMU. (2008). Sandi ADFGVX. *http://sandi.math.web.id/*. Tanggal akses: 25 Maret 2009 pukul 20:00.

[3] Schneier, Bruce. (1996). Applied Cryptography 2nd. John Wiley & Sons.