Vigènere Cipher Dengan Kunci Substitusi Inkremental Berdasarkan Caesar Cipher

Bhimantyo Pamungkas - 13504016
Program Studi Teknik Informatika ITB, Bandung 40132, email: btyo_pamungkas@yahoo.co.id

Abstract – *Dewasa ini dunia teknologi informasi semakin berkembang sejak munculnya komputer pertama kali. Banyak hal yang berubah termasuk bentuk penyimpanan dan penyampaian data, dimana sebelum adanya komputer data masih berbentuk analog, sedangakan sekarang sudah banyak data dalam bentuk digital. Perlu adanya perindungan terhadap isi dari data digital tersebut karena semakin canggih teknologi yang ada maka semakin canggih juga kejahatan yang mungkin terjadi, termasuk kejahatan yang berkaitan dengan pencurian data. Oleh karena itu pengamanan data menggunakan kriptografi semakin berkembang dan semakin banyak digunakan. Salah satu metode kriptografi klasik yang dapat digunakan dan yang akan menjadi pembahasan dalam makalah ini adalah vigènere cipher.*

Dalam makalah ini, penulis akan mencoba untuk mengembangkan algoritma vigènere cipher dengan cara menggabungkannya dengan algoritma caesar cipher yang akan digunakan untuk melakukan enkripsi terhadap kunci yang digunakan dalam vigènere cipher. Dasar ide pengusulan metode ini adalah karena kunci dari vigènere cipher yang berulang akan memberi peluang untuk kriptanalis untuk dapat memecahkan algoritma vigènere cipher yang biasa.

Isi dari makalah ini meliputi kosep dasar, implementasi, dan pengujian tingkat keamanan dari algoritma yang diusulkan. Konsep dasar meliputi landasan teori yang digunakan, implementasi meliputi penjelasan mengenai algoritma yang diusulkan, dan pengujian terhadap algoritma untuk mengetahui tingkat keamanannya.

Kata Kunci *analisis kasiski, enkripsi, dekripsi, kunci inkremental, kode*

1. PENDAHULUAN

Vigènere cipher merupakan sebuah metode enkripsi yang dapat menghilangkan kemungkinan untuk dipecahkan dengan analisis frekuensi, tetapi sayangnya memiliki pola yang dapat dipecahkan dengan metode kasiski. Untuk menambah kerumitan dari vigènere cipher dan mengurangi kemungkinan terpecahkannya ciphertext dengan menggunakan metode kasiski, kunci harus dibuat dengan pola yang lebih rumit. Salah satu cara yang dapat dilakukan adalah melakukan pengacakan terhadap kunci di setiap pengulangan kunci, tetapi pengacakan tanpa pola dapat menyebabkan ciphertext tidak dapat didekripsi oleh penerima ciphertext. Untuk dapat didekripsikan kembali, perubahan kunci pada setiap pengulangannya harus dapat memiliki paramter-parameter tertentu sehingga perubahan kunci dapat diketahui oleh penerima pesan.

Pengacakan kunci dari vigènere cipher dapat dilakukan dengan cara mengenkripsi kunci tersebut setiap kali pengulangan kunci terjadi. Metode enkripsi yang digunakan untuk mengenkripsi kunci tidak harus rumit, misalnya saja Caesar cipher. Dengan mengenkripsi kunci setiap terjadi pengulangan maka akan semakin kecil kemungkinan kunci dapat dideteksi dengan metode kasiski oleh kriptanalis. Ada beberapa hal yang harus diperhatikan dalam menggunakan caesar cipher dalam mengenkripsi kunci dari vigènere cipher yang tentunya akan dibahas pada bab implementasi.

2. KONSEP DASAR

2.1. Caesar Cipher

Caesar cipher merupakan salah satu algoritma kriptografi klasik yang sudah ada pada zaman kerajaan romawi. Caesar cipher digunakan untuk menyembunyikan pesan yang dikirim oleh kaisar romawi, proses enkripsi dilakukan dengan menggeser tiga karakter ke kanan terhadap setiap huruf dari plainteks, misal huruf ‘A’ diganti dengan huruf ketiga setelah huruf tersebut, yaitu huruf ‘D’. Tinjau plainteks

KRIPTOGRAFI

dienkripsi menggunakan caesar cipher dengan menggeser tiga karakter ke kanan pada setiap karakter pada plainteks sehingga cipherteks yang dihasilkan adalah

NULSWRJUDIL

Secara sistematis, proses enkripsi menggunakan caesar cipher dapat dituliskan sebagai

$$C_i = E(P_i) = (P_i + n) \bmod 26$$

sedangkan proses dekripsi dapat dituliskan sebagai

$$P_i = D(C_i) = (C_i - n) \bmod 26$$

dengan n adalah jumlah pergeseran, pada kasus caesar cipher n=3.

Jika dilihat lebih lanjut, caesar cipher dapat digolongkan kedalam cipher substitusi tunggal,

dimana setiap karakter selalu dienkripsikan menjadi karakter yang sama. Oleh karena itu, caesar cipher sangat rawan untuk dipecahkan oleh kriptanalis dengan metode analisis frekuensi.

2.2. Vigènere Cipher

Vigènere cipher merupakan salah satu algortima kriptografi klasik yang dirancang untuk mengatasi kekurangan pada algoritma substitusi tunggal dimana setiap karakter pada plainteks pasti disubstitusikan dengan karakter yang sama. Hal tersebut dapat dipecahkan dengan mudah dengan melihat pola-pola kata yang mirip dan dengan analisis frekuensi, yaitu dengan menghitung jumlah kemunculan setiap huruf pada cipherteks lalu membandingkannya dengan kecenderungan jumlah huruf yang muncul pada suatu bahasa tertentu (contohnya pada bahasa Inggris huruf 'e' adalah huruf yang paling banyak muncul).

Algoritma vigènere cipher dirancang untuk menghilangkan pola huruf sehingga tidak dapat dilakukan analisis frekuensi pada cipherteks. Plainteks pada vigènere cipher akan digeser (seperti pada caesar cipher) berdasarkan sebuah kunci. Panjang kunci dari vigènere cipher lebih kecil atau sama dengan panjang dari plainteks, jika panjang kunci lebih kecil daripada panjang plainteks maka kunci akan diulang hingga panjangnya sama dengan panjang plainteks.

Tinjau plainteks "KRIPTOGRAFI" dengan kunci "TES", maka proses enkripsi dengan vigènere cipher adalah sebagai berikut:

plainteks :	KRIPTOGRAFI
kunci :	TESTESTESTE
cipherteks :	DVAIXGZVSYM

Dapat terlihat bahwa setiap setiap huruf belum tentu dienkripsi menjadi cipherteks yang sama. Dalam proses enkripsi plainteks yang panjang, karakter separator (seperti spasi, semicolon, kome, titik, dsb) dihilangkan terlebih dahulu dari plainteks, kemudian hasil cipherteks dapat disatukan begitu saja tanpa ada jeda atau dikelompokan menjadi beberapa karakter (misal empat) lalu ditambahkan dengan spasi. Contoh pada hasil cipherteks diatas dikelompokan menjadi:

DVAI XGZV SYM

Kelemahan vigènere cipher adalah jika panjang kunci lebih pendek daripada panjang plainteks, kunci akan diulang hingga sepanjang plainteks sehingga ada kemungkinan bahwa sepotong plainteks akan dienkripsikan menjadi cipherteks yang sama.

2.3. Kunci Berulang Inkremental

Kelemahan vigeènere cipher tersebut dapat memberikan informasi kepada kriptanalis untuk mengetahui panjang kunci dengan metode Kasiski (metode yang diberi nama salah satu orang yang memecahkan vigènere cipher). Oleh karena itu, untuk kunci yang pendek perlu dilakukan penangan khusus agar kunci tidak mudah ditemukan dengan analisis Kasiski. Cara yang bisa dilakukan adalah mengenkripsi secara inkremental kunci setiap pengulangan dengan caesar cipher.

Algoritma ini membutuhkan dua buah kunci, kunci pertama adalah kunci untuk vigènere cipher, kunci kedua adalah kunci inkremental terhadap kunci dari vigènere cipher yang memiliki panjang yang sama tetapi berupa angka. Tinjau kunci vigènere cipher adalah "TES" sedangkan kunci kedua adalah 135. Setiap digit pada kunci kedua berkoresponden dengan setiap karakter pada kunci vigènere cipher sesuai dengan posisinya sebagai jumlah pergeseran yang akan dilakukan terhadap karakter tersebut. Dengan demikian karakter pertama akan digeser 1 karakter, karakter kedua digeser 3 karakter, karakter ketiga digeser 5 karakter, sehingga pada hasil kunci pada saat pengulangan akan menjadi

TESUHXVKC

Karena menggunakan prinsip yang sama dengan caesar cipher, pada saat pengulangan ke-26 akan kembali membentuk kunci semula. Untuk mencegah hal itu terjadi, setiap 25 kali pengulangan kunci kedua akan diinkremen setiap digitnya, misal pada kunci kedua pada contoh diatas, yaitu 135, akan diinkremen menjadi 246 setelah 25 kali pengulangan.

Perlu dicatat bahwa representasi setiap digit pada kunci kedua pada akhirnya akan berupa larik, sehingga jika angka sudah melebihi digit 9 tidak akan menjadi masalah, misalnya 11,31,54, karena adanya operasi modulo pada caesar cipher.

3. IMPLEMENTASI

3.1. Lingkungan pengembangan

Aplikasi algoritma kunci inkremental ini dikembangkan dengan lingkungan perangkat lunak:
Sistem operasi Windows XP SP2

- .Net Framework 2.0
- Microsoft Visual Studio 2005

- Bahasa pemograman Visual Basic

3.2. Fungsi Pergeseran Karakter

Fungsi untuk menggeserkan karakter memiliki prinsip yang sama dengan caesar cipher. Beberapa keterangan yang perlu diberikan adalah fungsi AscW adalah fungsi untuk mengubah karakter menjadi bilang Unicode, sedangkan ChrW fungsi untuk mengubah Unicode menjadi karakter. Representasi huruf 'z' pada Unicode adalah 122, sedangkan huruf 'a' adalah 97.

```
function GeserChar(
  c:char,
  n:integer
) -> Char
{Melakuan pergeseran karakter c
sebanyak n}

Deklarasi
  ni  : integer;
  temp: integer;

Algoritma
  begin
    ni:= n mod 26;
    temp:= AscW(c) + ni;
    if temp > 122 then
    begin
      temp:= (temp-122) + 96;
    end;
    else if temp < 97 then
    begin
      temp:= 122 - (96 - temp);
    end;
    return ChrW(temp);
  end.
```

3.3. Fungsi pembentuk kunci

Dalam membentuk kunci dibutuhkan dua fungsi, yaitu fungsu untuk menginkremen kunci kedua, kemudian fungsi untuk membentuk kunci gabungan kunci pertama dan kedua. Fungsi untuk menginkremen kunci kedua adalah sebagai berikut:

```
procedure InkremenKey2(
  k : arrayOfInteger
);
{Melakukan inkremen terhadap kunci
kedua}

Deklarasi
  i : integer;

Algoritma
  begin
    for i:=0 to k.Lenght-1 do
    begin
      k[i] := k[i] + 1;
    end;
    {endfor}
  end.
```

Dengan berbekal fungsi diatas, fungsi selanjutnya adalah untuk membentuk kunci yang memiliki panjang sama dengan teks yang akan dienkripsi maupun didekripsi menggunakan kunci yang pendek dan kunci kedua.

```
function GenerateKey(
  k1:arrayOfChar,
  k2:arrayOfInteger,
  l :integer
) -> arrayOfChar
{Membentuk kunci inkremental}

Deklarasi
  temp : arrayOfChar;
  temp1: arrayOfChar;
  temp2: arrayOfInteger;
  i,j,counter: integer;

Algoritma
  begin
    temp1:=k1;
    temp2:=k2;
    i:=0;
    counter:=0;

    while i<k1.Length do
    begin
      temp[i] := k1[i];
      i:= i+1;
    end;
    {endwhile}
    while i<l do
    begin
      if counter>25 then
      begin
        InkremenKey(temp2);
        counter:=0;
      end;
      {endif}
      j:=0;
      while j<temp1.Length and i<l
do
      begin

temp1[j]:=GeserChar(temp1[j],temp2
[j]);
        temp[i]:=temp1[j];
        j:=j+1;
        i:=i+1;
      end;
      {endwhile}
      counter:=counter+1;
    end;
    {endwhile}
    return temp;
  end.
```

3.4. Fungsi Enkripsi

Pada vigènere cipher, proses pergeseran karakter memiliki parameter sebuah karakter juga (berbeda

dengan caesar cipher yang paramaternya adalah sebuah bilangan). Oleh karena itu dibutuhkan fungsi untuk mengkonversi karakter menjadi indeks pergeseran berupa bilangan.

```
function CharToIndex(
  b:char
)-> integer
{Mengubah karakter menjadi index
pegeseran}

Deklarasi

Algoritma
  begin
    return AscW(b)-96;
  end.
```

Proses enkripsi dilakukan dengan meng-*generate* kunci terlebih dahulu kemudian melakukan pergeseran teks menggunakan kunci tersebut.

```
function Encrypt(
  teks:arrayOfChar;
  key1:arrayOfChar;
  key2:arrayOfInteger;
)-> arrayOfChar
{Melakukan enkripsi dengan
algoritman vigenere cipher dengan
kunci inkremental}

Deklarasi
  vkey:arrayOfChar;
  res :arrayOfChar;
  i   :integer;

Algoritma
  begin

vkey:=GenerateKey(key1,key2,teks.L
ength);
    for i:=0 to teks.Length-1 do
    begin

res[i]:=GeserChar(teks[i],CharToIn
dex(vkey[i]));
    end;
    return res;
  end.
```

3.5. Fungsi Dekripsi

Proses pada dekripsi kurang lebih sama dengan saat enkripsi, hanya saja parameter pergeseran karakter dijadikan minus sehingga karakter akan bergeser kearah yang berlawanan dan membentuk kembali plainteks seperti semula.

```
function Decrypt(
  teks:arrayOfChar;
  key1:arrayOfChar;
  key2:arrayOfInteger;
)-> arrayOfChar
{Melakukan enkripsi dengan
algoritman vigenere cipher dengan
kunci inkremental}

Deklarasi
  vkey:arrayOfChar;
  res :arrayOfChar;
  i   :integer;

Algoritma
  begin

vkey:=GenerateKey(key1,key2,teks.L
ength);
    for i:=0 to teks.Length-1 do
    begin
      res[i]:=GeserChar(teks[i],-
CharToIndex(vkey[i]));
    end;
    return res;
  end.
```

4. PENGUJIAN

Pengujian akan lebih ditekankan pada kunci yang dibentuk dengan metode inkremental, karena pada dasarnya algoritma enkripsi vigènere cipher tidak berubah.

Pengujian yang dilakukan adalah me-*generate* kunci sepanjang 10000 karakter dengan kunci awal "KRIPTO" dan kunci keduanya adalah 873592. Kunci sepanjang 10000 karakter yang terbentuk ternyata terdapat 3 kali pengulangan pola yang sama persis, dengan demikian dengan kunci awal sepanjang 6 karakter dapat membentuk kunci sepanjang sekitar 3300 karakter. Hal tersebut cukup baik untuk mempersulit metode analisis kasiski dalam memperkirakan panjang kunci yang sebenarnya.

Walau demikian, sepanjang 10000 karakter tersebut, kata "KRIPTO" muncul sebanyak 65 kali, berarti setiap pengulangan pola yang terjadi terdapat kemunculan kunci sebenarnya sebanyak sekitar 22 kali. Hal ini dapat memberikan dampak baik maupun buruk terhadap kriptanalis, yaitu dengan banyaknya kemunculan kunci yang sebenarnya dapat memperbesar peluang untuk menemukan kunci, tetapi hal tersebut juga dapat membingungkan kritanalis karena seakan-akan terdapat banyak kemungkinan panjang kunci.

Statistik kemungkinan pengulangan dan kemunculan kata kunci sebenarnya ternyata tida berpengaruh pada kunci kedua selama kunci kedua tidak seluruhnya terdiri dari angka 0. Statistik dipengaruhi oleh panjangnya kunci pertama, semakin panjang kunci

pertama maka semakin kecil statistik pengulangan dan kemunculan kunci.

5. KESIMPULAN

Berdasarkan percobaan dan analisis yang telah dilakukan, dapat ditarik beberapa kesimpulan terkati dengan algoritma vigènere cipher dengan kunci inkremental:

- Menggunakan kunci inkremental lebih baik dibandingkan dengan kunci biasa karena statistik kemunculan kunci jauh berkurang dibandingkan dengan kunci biasa.
- Panjang kunci yang terbentuk bila tidak ada pengulangan rata-rata hampir 500 kali lipat dari panjang kunci semula.
- Statistik pengulangan dan kemunculan tidak dipengaruhi oleh kunci kedua selama kunci kedua tidak seluruhnya terdiri dari angka 0.
- Statistik pengulangan dan kemunculan dipengaruhi oleh panjangnya kunci pertama.
- Algoritma ini tidak menjamin kuat terhadap analisis kasiski, tetapi dapat sangat memperumit pencarian dengan metode tersebut.
- Saran penggunaan algoritma ini adalah menggunakan kunci pertama yang panjang dan kunci kedua yang sesedikit mungkin terdapat angka 0.
- Saran pengembangan adalah algoritma untuk me-*generate* kunci sebaiknya dibuat lebih kompleks lagi, bisa saja menggunakan rumus matematika atau sejenisnya.

DAFTAR REFERENSI

[1] Munir, Rinaldi, *Diktat Kuliah IF5054 Kriptografi,* Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung, 2006.