Implementasi Vigenere Chiper Kunci Dinamis dengan Perkalian Matriks

Ahmad Zufri -13504044
1) Sekolah Teknik Elektro dan Informatika ITB, Bandung,40132, email: if14044@students.if.itb.ac.id

Abstract *–Semakin berkembangnya teknologi informasi membuat banyak masyarakat yang memanfaatkan teknologi informasi untuk berbagai keperluan dalam kehidupan sehari – hari. Teknologi informasi digunakan untuk berbagai hal yang sangat penting dan sering sekali digunakan untuk melakukan pengiriman data yang sifatnya rahasia dan tidak boleh diketahui oleh orang lain. Karena banyak sekali kejahatan cyber yang berusaha untuk mendapatkan pesan rahasia tersebut dibutuhkan suatu sistem pengamanan data yang disebut kriptografi. Salah satu metode yang digunakan dalam kriptografi ialah algoritma Vigenere Chiper yang termasuk kedalam algoritma kriptografi klasik yang sudah dikenal sejak lama. Algoritma ini melakukan enkripsi terhadap pesan yang aka dikirim sehingga pesan tersebut terjaga kerahasiaannya. Dalam makalah ini penulis akan membahas tentang varian baru yang menggabungkan antara algoritma vigenere Chiper dan konsep matriks dalam pembangkitan kunci dinamis yang digunakan untuk melakukan enkripsi. .Isi makalah ini akan mencakup pembahasan konsep dasar, implementasi dan juga analisis keamanan yang dilakukan berdasarkan pengujian.*

Kata Kunci*: analisis frekuensi,know-plaintext attack,kasiski,enkripsi,dekripsi,keystreaming.*

1. PENDAHULUAN

Perkembangan teknologi informasi yang begitu pesat pada saat sekarang ini membuat setiap orang menjadi lebih banyak melakukan pengiriman data dan pesan melalui media elektronik. Hal ini menyebabkan volume pertukaran data digital setiap hari mengalami penigkatan pesat. Data digital yang dipertukarkan disini sangat bervariasi dari mulai data yang tidak penting hingga data penting yang sifatnya sangat rahasia. Sering sekali pesan yang sifatnya rahasia tersebut dimanfaat oleh pihak lain yang tidak bertanggung jawab untuk kepentingannya sendiri.

Salah satu cara untuk mngatasi permasalahan diatas ialah dengan menggunakan kriptografi untuk melakukan enkripsi terhadap pesan yang dikirimkan. Sehingga hanya orang atau pihak yang berhak sajalah yang dapat mengetahui isi dari pesan tersebut.

Dalam ilmu kriptografi banyak dikenal berbagai macam algoritma baik yang klasik maupun yang modern. Salah satu algoritma kriptografi klasik yang terkenal ialah *Vigenere Chiper*. *Vigenere Chiper* merupakan algoritma yang sederhana dan mudah diimplementasikan. Namun algoritma ini sudah berhasil dipecahkan dengan suatu metode yang dinamakan dengan metode kasiski yang dapat mengetahui panjang kunci yang digunakan. Setelah panjang kunci ditemukan dilanjutkan dengan teknik analisis frekuensi untuk mengetahui isi pesan yang sebenarnya.

Karena kelemahan tersebut penulis berusaha mengembangkan algoritma vigenere chiper dengan pembangkitan kunci dinamis (*keystream*) dengan menggunakan konsep perkalian matriks yang sudah sangat umum dikenal dalam bidang matematika. Hal ini dimaksudkan untuk mengatasi kelemahan kunci berulang yang dimanfaatkan untuk memecahkan Vigenere Chiper.

2. KONSEP DASAR

2.1 Vigenere Chiper

Vigenere Chiper adalah suatu algoritma yang tergolong ke dalam algoritma subtitusi abjad majemuk. Ini artinya setiap huruf yang sama dalam plaintext tidak dipetakan atau disubtitusi oleh satu huruf. Melainkan di subtitusi oleh huruf yang berlainan bergantung dari kunci yang digunakan untuk melakukan enkripsi. Algoritma ini ditemukan oleh diplomat sekaligus kriptologis dari Prancis, Blaise de Vigènere pada abad 16. Vigènere cipher dipublikasikan pada tahun 1856, tetapi algoritma ini baru dikenal luas 200 tahun kemudian.

Proses enkripsi dalam *Vigenere Chiper* dilakukan dengan menggunakan bujur sangkar vigenere.Bujur sangkar vigenere berisi huruf huruf abjad yang disusun sedemikian rupa. Kolom paling kiri menyatakan huruf-huruf kunci sedangkan baris paling atas menyatakan huruf-huruf plainteks. Setiap baris dalam bujursangkar menyatakan huruf-huruf cipheteteks yang diperoleh dengan *Caesar Cipher,* yang mana jauh pergeseran huruf plainteks ditentukan oleh nilai decimal oleh huruf kunci tersebut (a = 0, b = 1,…, z = 25).

Berikut ini ialah bujur sangkar Vigenere:

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

Chipertext diperoleh dengan menggunakan bujur sangkar vigenere dan kunci yang telah ditentukan. Jika Panjang kunci lebih pendek daripada chipertext, maka penggunaan kunci akan diulang.

Berikut ini ialah contoh untuk menjelaskan cara eknripsi dengan metode vigenere ini. Plaintext "TUGAS MAKALAH" akan dienkripsi dengan kunci "WEW".

P	T	U	G	A	S	M	A	K	A	L	A	H
K	W	E	W	W	E	W	W	E	W	W	E	W
C	P	Y	C	W	W	I	W	O	W	H	E	D

Proses enkripsi dilakukan dengan cara menarik garis vertical ke bawah dari huruf plaintext hingga berada pada baris dari huruf kunci. Selanjutnya huruf yang merupakan perpotongan antara huruf plaintext dan huruf kunci merupakan huruf chipertext. Sedangkan proses enkripsi dilakukan sebaliknya,yaitu dengan cara menarik garis horizontal dari kunci ke kanan hingga menemukan huruf yang sesuai dengan huruf chipertext. Selanjutnya tariklah garis vertical ke atas hingga sampai ke huruf plaintext.

Dari contoh diatas terlihat bahwa setiap huruf tidak disubtitusi oleh huruf tertentu, akan tetapi selalu berubah – ubah sesuai dengan perulangan kunci. Sehingga tidak dapat dipecahkan dengan teknik analisis frekuensi. Akan tetapi, penggunaan kunci yang berulang ini merupakan kelemahan dari vigenere chiper. Hal ini dikarenakan adanya kemungkinan chipertext yang sama akan dienkripsi dengan kunci yang sama. Sehingga dapat ditentukan panjang kunci yang digunakan. Selanjutnya jika panjang kunci sudah diketahui, maka chipertext dapat dipecahkan dengan teknik analisis frekuensi. Yang pertama kali berhasil memecahkan vigenere chiper ini ialah . Babbage dan Kasiski pada pertengahan abad 19. Untuk mengatasi kelemahan ini hendaknya dibuat kunci yang panjangnya sama dengan panjang plaintext. Sehingga algoritma ini akan mirip dengan algoritma One Time Pad yang sampai sekarang belum dapate terpecahkan. Namun untuk membuat kunci yang panjangnya sama dengan panjang plaintext tidak praktis.

2.2 Konsep Dasar Matriks

Dalam matematika, matriks ialah sebuah table yang terdiri kumpulan element. Element dari sebuah matriks biasanya berupa angka, namun dapat juga type lain. Matriks terdiri dari baris dan kolom. Sebuah matriks dengan jumlah baris *m* dan jumlah kolom *n* disebut matriks *mxn.* Contoh matriks *5x3*

$$\begin{vmatrix} 6 & 5 & 9 \\ 5 & 9 & 4 \\ 7 & 7 & 2 \\ 9 & 0 & 6 \\ 8 & 7 & 8 \end{vmatrix}$$

Matriks dapat dikenakan operasi perkalian.Operasi perkalian pada matriks dapat dibedakan menjadi 2 jenis, yaitu perkalian dengan scalar dan perkalian dengan matriks.
Contoh perkalian dengan skalar ialah:

$$2 \cdot \begin{bmatrix} 1 & 8 & -3 \\ 4 & -2 & 5 \end{bmatrix} = \begin{bmatrix} 2 \cdot 1 & 2 \cdot 8 & 2 \cdot -3 \\ 2 \cdot 4 & 2 \cdot -2 & 2 \cdot 5 \end{bmatrix} = \begin{bmatrix} 2 & 16 & -6 \\ 8 & -4 & 10 \end{bmatrix}$$

Contoh perkalian matriks dan matriks ialah:

$$\begin{bmatrix} 1 & 0 & 2 \\ -1 & 3 & 1 \end{bmatrix} \times \begin{bmatrix} 3 & 1 \\ 2 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} (1 \times 3 + 0 \times 2 + 2 \times 1) & (1 \times 1 + 0 \times 1 + 2 \times 0) \\ (-1 \times 3 + 3 \times 2 + 1 \times 1) & (-1 \times 1 + 3 \times 1 + 1 \times 0) \end{bmatrix}$$

$$= \begin{bmatrix} 5 & 1 \\ 4 & 2 \end{bmatrix}.$$

3. Pembahasan Vigenere Chiper Kunci Dinamis dengan Perkalian Matriks

Algoritma yang akan dibahas disini merupakan pengembangan lebih lanjut dari vigenere chiper yang dilakukan oleh penulis. Algoritma ini menggunakan konsep dari aljabar linear yaitu matriks untuk men-*generate* kunci yang akan digunakan dalam enkripsi blok selanjutnya. Algoritma ini bertujuan menghilangkan penggunaan dari kunci yang berulang yang merupakan kelemahan dari vigenere chiper. Selain itu,algoritma ini juga di implementasikan dalam rangakaian huruf ASCII karakter(ASCII encoding pada C#) yang berjumlah 128 buah sehingga jumlah kemungkinan kunci menjadi lebih banyak.

Dalam algoritma ini digunakan konsep perkalian matriks yang biasa digunakan untuk merepesentasikan system persamaan linear. Secara umum, persamaan matematika untuk menghasilkan kunci proses enkripsi untuk blok ke m ialah:

$$\boldsymbol{Ak=l} \qquad (1)$$

Dimana **A** ialah sebuah matriks yang didapat dengan menggunakan fungsi permutasi dari kunci yang dipakai untuk melakukan enkripsi pada blok sebelumnya. Jika enkripsi blok sebelumnya dilakukan dengan kunci K yang terdiri atas $K_1,K_2 \ldots K_n$, maka matriks A ialah matriks *nxn* yang mana n merupakan panjang kunci yang dipakai untuk melakukan enkripsi pada blok sebelumnya (blok ke m-1). Fungsi permutasi untuk menghasilkan matriks A ialah:

$$\begin{matrix} K_1xK_1 & K_1xK_2 & \ldots\ldots\ldots & K_1xK_n \\ K_2xK_1 & K_2xK_2 & \ldots\ldots\ldots & K_2xK_n \\ \vdots & \vdots & \vdots & \vdots \\ K_nxK_1 & K_nxK_2 & \ldots\ldots\ldots & K_nxK_n \end{matrix}$$

Sedangkan **k** disini ialah matriks *nx1* yang merepresentasikan kunci yang digunakan untuk melakukan enkripsi pada blok sebelumnya.(blok ke m-1) yang mana **k** ialah

$$\begin{matrix} K_1 \\ K_2 \\ \vdots \\ K_n \end{matrix}$$

l disini merupakan hasil perkalian antara matriks A dan p yang digunakan sebagai kunci untuk melakukan enkripsi pada blok m. Selanjutnya ***l*** digunakan untuk melakukan enkripsi dengan menggunakan bujursangkar vigenere.

Sedangkan untuk melakukan proses dekripsi terhadap blok ke m, kunci untuk melakukan dekripsi didapat sesuai dengan persamaan (1). Selanjutnya dilakukan enkripsi biasa seperti pada vigenere chiper dengan menggunakan bujursangkar vigenere.

Contoh:

Diberikan sebuah plaintext (dalam Hexadecimal) sebagai berikut:
P= `TUGASM yang dalam byte(84 85 71 65 83 77 65)`
Akan dienkripsi dengan kunci
K=ABC dalam byte (65 66 67)

Proses Enkripsinya ialah:

1. Lakukan enkripsi blok pertama yaitu TUG dengan kunci awal ABC sehingga menghasilkan chipertext (dalam byte)
   (159 170 149 132 156)
2. Lakukan Enkripsi blok kedua dengan menggunakan kunci sebagai berikut:
   Bentuk matriks permutasi sehingga menjadi matriks tersebut menjadi:

4225	4290	4355
4290	4356	4422
4355	4422	4489

3. Selanjutnya kalikanlah matriks tersebut dengan kunci yang digunakan pada blok pertama sehingga kunci yang digunakan pada blok ini ialah:

862355 mod128	=19
875622 mod128	=102
888889 mod128	=57

4. Selanjutnya pakailah kunci tersebut untuk melakukan enkripsi pada blok kedua, sehingga didapatkan chipertext:
   (84,57,6).
5. Lakukan langkah 2 sampai 4 hingga semua plaintext habis terenkripsi.

Proses Dekripsinya sama dengan proses enkripsinya. Kecuali pada saat subtitusi huruf ke dalam bujursangkar vigenere.

Implementasi

Penulis telah mengimplementasikan algoritma ini kedalam sebuah program applikasi yang dibuat dengan menggunakan bahasa pemrograman C#. Berikut ini Fungsi – fungsi yang diimplementasikan ialah:

Fungsi Permutasi, fungsi ini dilakukan untuk membentuk matriks permutasi.

```
public static int[,]
fungsiPermutasi(char[] Kunci)
        {
            int[,] a=new
int[Kunci.Length,Kunci.Length];
            for (int i = 0; i <
Kunci.Length; i++)
            {
                for (int j = 0; j <
Kunci.Length; j++)
                {
                    a[i, j] =
(byte)Kunci[i] * (byte)Kunci[j];
                }
            }
            return a;
        }
```

Fungsi Perkalian Matriks, fungsi ini melakukan perkalian terhadap 2 buah matriks:

```
public static int[,] kaliMatriks(int[,]
a, int[,] b,int abaris,int akolom,int
bbaris,int bkolom)
        {
            if (akolom!=bbaris ||
a.Length == 0 || b.Length==0)
                return null;

            else
            {
                int[,] hasil = new
int[abaris,bkolom];
                for (int i = 0; i <
bkolom; i++)
                {
                    for (int j = 0; j <
abaris; j++)
                    {
                        int temp=0;
                        for (int k = 0;
k < akolom; k++)
                        {
                            temp +=
a[j,k] * b[k,i];
                        }
                        hasil[j,i] =
temp%128;
                    }
                }
                return hasil;
            }
        }
```

Fungsi vigenere, Fungsi ini digunakan untuk enkripsi dengan cara mensubtitusikan plaintext nya ke kotak vigenere.

```
public static char[] vigenere(Char[]
Plain,Char[] Key)
        {
            int keyPos = 0;
            int[] ktemp = new
int[Key.Length];
            int[] ptemp = new
int[Plain.Length];
            byte[] tbyte = new
byte[Plain.Length];
            for (int i = 0; i <
Plain.Length; i++)
            {
                ktemp[i%ktemp.Length] =
(int)Key[keyPos];
                ptemp[i] =
(int)Plain[i];
                int temp =
((int)Plain[i] + (int)Key[keyPos]) %
128;
                tbyte[i] = (byte)temp;
                if (keyPos ==
Key.Length - 1) keyPos = 0;
                else keyPos++;

            }
            ASCIIEncoding U = new
ASCIIEncoding();
            char[] t =
U.GetChars(tbyte);
            return t;
        }
```

Fungsi inverse vigenere, fungsi ini digunakan dekripsi dengan cara melakukan subtitusi chipertext menjadi plaintext pada kotak vigenere.

```
public static char[] rvigenere(Char[]
Plain,Char[] Key)
        {            int keyPos = 0;
            int[] ktemp = new
int[Key.Length];
            int[] ptemp = new
int[Plain.Length];
            byte[] tbyte = new
byte[Plain.Length];
            for (int i = 0; i <
Plain.Length; i++)
            {
                ktemp[i%ktemp.Length] =
(int)Key[keyPos];
                ptemp[i] =
(int)Plain[i];
                int temp =
((int)Plain[i] - (int)Key[keyPos]) %
128;
                tbyte[i] = (byte)temp;
                if (keyPos ==
Key.Length - 1) keyPos = 0;
                else keyPos++;

            }
            ASCIIEncoding U = new
ASCIIEncoding();
            char[] t =
U.GetChars(tbyte);
            return t;
        }
```

Fungsi Enkripsi yang dibuat ialah:

```
public static char[] Enkripsi(char[]
Plain, char[] Key)
        {
            char[] result = new
char[Plain.Length];
            int posisi = 0;
            int batas=Key.Length;
            char[] currKey = Key;
            while (posisi <
Plain.Length)
            {
                if(Plain.Length-
posisi<Key.Length) batas=Plain.Length-
posisi;
                char[] temp = new
char[batas];
                char[] temp_result =
getElement(Plain, posisi, batas);
                int[,] mA =
fungsiPermutasi(currKey);
                int[,] mHasil =
kaliMatriks(mA, charToint(currKey),
Key.Length, Key.Length, Key.Length, 1);
                currKey =
intTochar(mhasil);
                temp =
vigenere(temp_result, currKey);
                for (int j = 0; j <
batas; j++)
                {
                    result[posisi + i]
= temp[i];
                }
                posisi += batas;
            }
            return result;
        }
```

Fungsi Dekripsi yang dibuat ialah:

```
public static char[] Dekripi(char[]
Plain,char[] Key)
        {
            char[] result = new
char[Plain.Length];
            int posisi = 0;
            int batas=Key.Length;
            char[] currKey = Key;
            while (posisi <
Plain.Length)
            {
                if(Plain.Length-
posisi<Key.Length) batas=Plain.Length-
posisi;
                char[] temp = new
char[batas];
                char[] temp_result =
getElement(Plain, posisi, batas);
                int[,] mA =
fungsiPermutasi(currKey);
                int[,] mHasil =
kaliMatriks(mA, charToint(currKey),
Key.Length, Key.Length, Key.Length, 1);
                currKey =
intTochar(mhasil);
                temp =
rvigenere(temp_result, currKey);
                for (int j = 0; j <
batas; j++)
                {
                    result[posisi + i]
= temp[i];
                }
                posisi += batas;
            }
            return result
        }
```

Kekuatan Algoritma

Kekuatan Algoritma ini ialah:

1. Memakai kunci yang dinamis (keystream) sehingga algoritma ini akan mirip dengan algoritma one time pad yang unbreakable.
2. Metode kasiski tidak akan dapat digunakan disini karena algoritma ini tidak menggunakan perulangan kunci.
3. Fungsi permutasi dalam bentuk matriks yang digunakan dapat mempersulit kriptanalisis untuk memecahkan algoritma ini.
4. Fungsi pembangkitan kunci ***Ak=I*** akan membuat algoritma ini sangat sulit dipecahkan. Karena jika mengetahui kunci suatu blok, maka bagian blok sebelumya tidak akan dapat di dekripsi dan kriptanalis hanya dapat mendekripsi blok sesudahnya.
5. Penggunaan algoritma dengan kode ASCII yang berjumlah 128 membuat pencarian dengan exhaustive search terhadap kunci dengan panjang n akan memiliki 128^n kemungkinan.
6. Algoritma sederhana dan mudah untuk diimplemantasikan.
7. Jika terjadi perubahan terhadap chipertext, yang rusak hanya bagian yang berubah tersebut saja.sedangkan bagian lain tidak akan berpengaruh.

Kelemahan Algoritma

1. Jika kunci sangat panjang, maka operasi perkalian akan membutuhkan waktu yang sangat lama.
2. Fungsi permutasi yang digunakan sangat sederhana sekali.
3. Adanya kemungkinan kunci yang dipakai berulang.

4.Serangan Terhadap Algoritma Ini

4.1 Metode Kasiski

Algoritma ini sanagat kuat terhadap serangan dengan metode kasiski. Hal ini dikarenakan algoritma ini tidak menggunakan perulangan kunci seperti pada vigenere chiper biasa. Walaupun ada kemungkinan terjadi perulangan kunci, namun peluang hal ini terjadi sangat kecil. Dikarenakan algoritma ini menggunakan

fungsi permutasi yang dituangkan dalam bentuk matriks

4.2 Chipertext Only Attack

Algoritma ini sangat kuat terhadap serangan berjenis ini karena menggunakan subtitusi abjad majemuk yang mengakibatkan setiap huruf yang sama dapat di petakan ke huruf yang berbeda. Selain itu penggunaan fungsi permutasi yang berupa matriks membuat kriptanala

1. Menggunakan algoritma subtitusi abjad majemuk yang mana untuk setiap huruf yang sama tidak disubtitusi dengan huruf yang sama . melainkan bergantung dari kunci yang digunakan.

2. Fungsi permutasi yang berupa matriks membuat kriptanalis harus berusaha lebih keras untuk menyelesaikan persamaan kunci yang digunakan. Apalagi fungsi permutasi ini melibatkan kunci yang bersifat rahasia. Sehingga kecil sekali kemunginan serangan jenis ini dapat berhasil.

3. Penggunaan kunci yang dinamis membuat kriptanalis harus melakukan dekripsi secara manual terhadap selutuh chipertext dan keberhasilan memecahkan satu blok tidak akan membuat seluruh chipertext berhasil di dekripsi.

3. Penggunaan kunci yang dinamis yang bergantung satu sama lain menyebabkan kesalahan satu huruf kunci saja akan dapat menyebabkan kesalahan pada seluruh plaintext.

4.2 Known-plainteks Attack

Algoritma ini rawan terhadap serangan ini. Hal ini terjadi jika potongan plaintext dan chipertext yang bersesuaian merupakan pada blok pertama atau bagian awal dari plaintext. Hal ini dikarenakan pada fungsi pembangkit kunci ***Ak=l*** .Jika kita mengetahui kunci pada blok pertama maka matriks A akan diperoleh dan kunci – kunci blok berikutnya. Namun jika potongan plaintext dan chipertext tidak pada blok pertama, maka yang dapat didekripsi hanya blok sesudahnya. Sedangkan blok sebelumnya tidak anda dapat di dekripsi karena A dan k tidak diketahui kedua – duanya.

4.3 Exhaustif Search

Untuk jenis serangan ini, algoritma ni cukup kuat jika kunci yang digunakan panjag. Hal ini dikarenakan penggunaan kunci yang merupakan karakter ASCII yang berjumlah 128 buah.Sehingga kemungkinan kuncinya ialah 128^n buah kemungkian kunci yang harus dicari.

5.Kesimpulan

1. Algoritma vigenere chiper kunci dinamis ini lebih baik daripada algoritma vigenere chiper yang standar.
2. Algoritma ini hampir tidak mungkin tidak dapat dipecahkan melalui teknik kasiski karena algortima ini tidak menggunakan perulangan kunci.
3. Algoritma ini sangat rawan dengan serangan know-plaintext attack sepreti yang sudah dibahas sebelumnya
4. Algoritma ini sangat kuat terhadap serangan chipertext only attack.
5. Jmlah kemunkinan kunci dalam algoritma ini ialah 128^n yang mana n ialah panjang kunci.
6. Saran bagi pengguna algoritma ini ialah guenakanlah kunci yang panjang agar jumlah kemungkinan kunci lebih banyak.

DAFTAR PUSTAKA

[1] Anton, Howard., *Dasar – Dasar Aljabar Linear*, Interaksara, 2002

[2] Bishop,David*, Introduction to Cryptography with Java Applet*, Jones and Bartlet Computer Science, 2003

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