# Tripel Vigenere Super Encrypsi

Edwin Romelta / 13508052
*Program Studi Teknik Informatika*
*Sekolah Teknik Elektro dan Informatika*
*Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia*
*edro_ooo@yahoo.co.id*

***Abstract*—Makalah ini membahas tentang memodifikasi sebuah algoritma kriptografi. Algoritma yang dimodifikasi pada makalah ini adalah algoritma kriptografi Vigenere Chiper. Modifikasi yang dilakukan di sini adalah dengan menambahkan sebuah algoritma transposisi setelah enkripsi oleh algoritma kriptografi Vigenere Chiper. Hal ini mengikuti sebuah metode kriptografu yang disebut Super Enkripsi yaitu pengabungan algoritma subtitusi dan transposisi. Dimana algoritma subtitusinya adalah algoritma Vigenere Chiper dan algoritma transposisinya adalah mengacak secara random yang memiliki nilai seed. Dengan algoritma super enkripsi ini penguna hanya cukup menghafal 1 buah key karena dengan 1 buah key tersebut dapat dibangun 2 buah key, 1 key untuk key Vigenere Chiper dan 1 key lagi untuk seed randomize algoritma transposisi.**

**Namum penulis masih belum merasa cukup aman dengan algoritma super enkripsi ini. Lalu penulis melakukan iterasi looping selama 3 kali yang dilakukan oleh tripel DES sehingga algoritma Super Enkripsi yang dilakukan sebanyak 3 kali. Dengan begitu penulis mengharapkan membentuk suatu algoritma kriptografi yang cukup aman dan mudah digunakan.**

**Penulis menamakan algoritma ini Tripel Vigenere Super Enkripsi karena algoritma yang digunakan adalah algortma vigenere chiper yang di super enkripsikan dengan algoritma transposisi randomize dan semua itu dilakukan 3 kali.**

***Index Terms*—Vigenere Chiper, algoritma transposisi, Tripel DES.**

## 1. PENDAHULUAN

### 1.1 LatarBelakang

Sejak zaman dahulu banyak dokumen – dokumen milik pribadi atau kelompok yang besifat rahasia. Dokumen tersebut bebahaya untuk dibaca oleh pihak yang tidak berwenang. Sehingga dibutuhkan keamanan pada saat mengirimkan dokumen – dokumen tersebut. Namun keamanan seperti menjaga kurir yang membawa dokumen tersebut tidak cukup. Untuk itu diperlukan suatu cara agar meskipun dokumen itu berada di pihak yang tidak berwenang dokument tersebut tidak bisa dibaca oleh pihak itu. Namun pihak yang berwenang masih tetap dapat membaca dokumen tersebut. Mulailah kriptografi atau seni menyembunyikan muncul.

Pada zaman tersebut muncul sebuah algoritma yang terkenal dalam menyembunyikan informasi dokemen dan sulit di pecahkan. Algoritma tersebut bernama Vigenere Chiper. Namun dengan berkembangnya zaman algoritma ini sudah memiliki banyak kelemahan dan mudah untuk di pecahkan. Karena itu perlu cara untuk mempersulit agar algoritma ini dapat bertahan.

Untuk membuat algoritma ini lebih sulit dipecahkan kita dapat menambahkan algoritma transposisi setelah mengenkirpsi dokemen. Dan agar lebih sulit lagi kita melakukan semuanya sebanyak 3 kali seperti algoritma tripel DES.

### 1.2 Tujuan

- Mengetahuai apakah dengan merandom posisi setelah enkripsi dengan vigenere dapat mengurangi permasalahan diatas.
- Menganalisis keamanan dari algoritma super – enkripsi
- Menganalisi kemananan dari algoritma tripel enkripsi
- Menciptakan algoritma yang sulit untuk dipecahkan tetapi mudah untuk digunakan.

### 1.3 Rumusan Masalah

Pada Makalah ini membahas:

1. Menciptakan algoritma yang tidak mungkin / mempersulit untuk dipecahkan dengan analisi frekuensi
2. Bagaimana menciptakan 1 buah key untuk 2 buah algoritma yang berbeda.

### 1.4 Batasan Masalah

Hal yang dibahas di makalah ini adalah menciptakan tripel super enkripsi dengan algoritma vigenere dan di acak setelah enkripsinya.
Pada makalah ini tidak membahas tentang menciptakan key yang memperkecil kemungkinan memecahkan algoritma ini dengan analisis frekuensi.
Random yang digunakan adalah random yang

diciptakan dari seed , bukan random yang absolut karena sangat sulit untuk mengimplementasinya.

## 2. DASAR TEORI

### 2.1 Vigenere Chiper

Vigenere Chiper mengunakan metode enkripsi abjad majemuk. Metode ini dipublikasikan oleh diplomat (sekaligus seorang kriptologis) Perancis, Blaise de Vigènere pada abad 16 (tahun 1586). Tetapi sebenarnya Giovan Batista Belaso telah menggambarkannya pertama kali pada tahun 1553 seperti ditulis di dalam bukunya *La Cifra del Sig. Giovan Batista Belaso*. Algoritma tersebut baru dikenal luas 200 tahun kemudian yang oleh penemunya *cipher* tersebut kemudian dinamakan *Vigènere Cipher*.

*Cipher* ini berhasil dipecahkan oleh Babbage dan Kasiski pada pertengahan Abad 19.

Metode ini mengunakan bujursangkar vigenere untuk melakukan enkripsi

| | 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 |

Gambar 1. Bujursangkar Vigenere

Sehingga untuk mendapatkan satu buah chipertext didapat dengan melihat plaintext pada colom dan kuncinya pada baris.

Contoh :

PlainText = C

Key = G

Sehingga dengan melihat bujursangkar vigenere didapatkan chipertext I

Untuk mendapatkan Plaintext dari chipertext cukum membalikan hal tersebur.

Dengan melihat key terlebih dahuli pada baris mana lalu menelusuri sampai menemukan chipertext yang dimaksud. Setelah itu melihat chipertext tersebut berada pada kolom mana.

Contoh:

Key = G

ChiperText = I

Sehingga dengan melihat bujursangkar vigenere didapatkan Plaintext C.

Selain cara diatas dapat juga dengan mengunakan rumus berikut:

$$Ci = (Pi+Ki) \bmod 26$$

sedangkan proses dekripsinya:

$$Pi = (Ci-Ki) \bmod 26$$

dalam hal ini,

Pi: karakter plainteks;

Ci: karakter cipherteks;

Ki: karakter kunci.

Namun hal diatas hanya dapat digunakan untuk karakter 26 bit. Dengan perkembangan zaman jumlah karakterpun juga sangat beragam. Semua bergantung pada jenis karakter yang digunakan. Karakter yang biasa digunakan adalah karakter ASCII yang memiliki 256 jumlah karakternya. Berarti vigenerepun diubah menjadi 256bit. Sehingga rumus vigenere pun berubah menjadi seperti berikut :

$$Ci = (Pi+Ki) \bmod 256$$

sedangkan proses dekripsinya:

$$Pi = (Ci-Ki) \bmod 256$$

dalam hal ini,

Pi: karakter plainteks;

Ci: karakter cipherteks;

Ki: karakter kunci.

Untuk karakter jenis lainnya cukup menganti pembagian mod dengan jumlah jenis karakter yang digunakan.

Dengan begitu kita dapat mengubahnya ke bahasa pemprograman dengan mudah. Karena kita tidak perlu mengunakan bujursangkar vigenere.

Berikut adalah source code vigenere chiper dengan mengunakan bahasa pemprograman java dengan karakter ASCII 256 bit:

```
 public static String EncriptVigenere
(String text,String key){
          String temp = "";
          int k=0;
          char c;
          for(int i =0;i<=text.length()-
1;++i){
               c =
(char)((text.charAt(i)+key.charAt(k))%25
6);
               temp += c;
               ++k;
               if(k >= key.length())
                    k=0;
          }
          return temp;
     }
```

```
      public static String
DecriptVigenere (String text,String
key){
          String temp = "";
          int k=0;
          char c;
          int value;
          for(int i =0;i<=text.length()-
1;++i){
              value = ((text.charAt(i)-
key.charAt(k))%256);
              if(value <0){
                      value += 256;
                  }
              c = (char)value;
              temp += c;
              ++k;
              if(k >= key.length())
                  k=0;
          }
          return temp;
      }
```

### 2.2 Algoritma Transposisi

Algoritma kriptografi Transposisi adalah memindahkan posisi setiap karakter pada plaintextnya sehingga terbentuk sebuah chipertext dari anagram plaintext.

Algoritma transposisi yang akan digunakan memiliki ide seperti berikut:

- Buat sederetan angka mulai dari 1 sampau dengan jumlah karakter pada plaintext yang akan di transposisi
- Mengacak seluruh deretan angka – angka tersebut dengan seed yang didapat dari key yang dimasukan
- Bentuk sebuah deretan chipertext sebanyak jumlah karakter plaintext
- Setelah itu pindahkan karakter plaintext pertama ke nomor yang didapat bada baris pertama pada chipertext
- Berikutnya dilakukan sama dengan diatas sampai pada akhirnya terbentuk chipertext yang merupakan anagram plaintext

Untuk mengembalikannya tidak sulit. Cukup membalikan algoritma diatas.

Berikut cara mengembalikannya kembali seperti plaintext:

- Buat sederetan angka mulai dari 1 sampau dengan jumlah karakter pada plaintext yang akan di transposisi
- Mengacak seluruh deretan angka – angka tersebut dengan seed yang didapat dari key yang dimasukan
- Bentuk sebuah deretan Plaintext sebanyak jumlah karakter Chipertext
- Letakkan ke plaintext pertama karakter chipertext sesuai nomor yang didapat pada baris random pertama.
- Lakukan sampai akhir dari baris random yang diciptakan dan akhirnya terbentuk plaintext yang sama seperti awalnya.

Berikut adalah souce code transposisi yang digunakan pada makalah ini:

```
 public static String EncriptRandomize
(String text,int seed){
          String temp="";
          char[] c = text.toCharArray();
          char[] val = new
char[c.length];
          ArrayList<Integer> array =
CreateRandomizeArray(text.length(),seed)
;
          for(int i
=0;i<text.length();++i){
              val[array.get(i)] = c[i];
          }
          for(int i
=0;i<text.length();++i){
              temp+=val[i];
          }
          return temp;
      }

     public static String
DecriptRandomize(String text,int seed){
          String temp="";
          ArrayList<Integer> array =
CreateRandomizeArray(text.length(),seed)
;
          char[] c = text.toCharArray();
          char[] val = new
char[c.length];
          for(int i
=0;i<text.length();++i){
              val[i] = c[array.get(i)];
          }
          for(int i
=0;i<text.length();++i){
              temp += val[i];
          }
          return temp;
      }

 public static ArrayList<Integer>
CreateRandomizeArray(int size, int seed)
{
          Random rand = new
Random(seed);
          ArrayList<Integer> array = new
ArrayList<Integer>();
          for (int i = 0; i < size; ++i)
{
              array.add(i);
          }

java.util.Collections.shuffle(array,
rand);
          return array;
```

```
        }
```

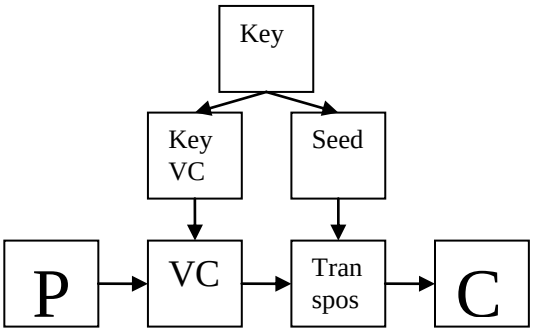
### 2.3 Algoritma Super Enkripsi

Super Enkripsi adalah algoritma kriptografi kombinasi dari algoritma kriptografi subtitusi dan algoritma kriptigrafi transposisi.

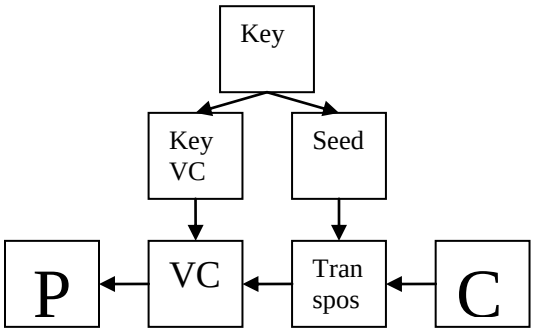
Algoritma super enkirpsi yang dilakukan adalah melakukan algoritma vigenere chiper yang diberikan pada bagian sebelumnya lalu di transposisi dengan algoritma tranposisi yang diberikan sebelumnya juga.

Sementara itu algoritma Super Dekripsinya adalah melakuan dekripsi transposisi baru dekripsi vigenere chiper

Berikut adalah bagan dari algoritma Super Enkripsi yang digunakan.




Gambar 2. Bagan Super Enkripsi




Gambar 3. Bagan Super Dekripsi

Keterangan

P = PlainText
VC = Vigenere Chiper
Transpos = Algoritma Transposisi
C = ChiperText
Key = Key yang dimasukan Pengguna
Key VC = Key Vigenere Chiper
Seed = Seed untuk random pada transposisi

Berikut adalah source code algoritma super enkirpsi

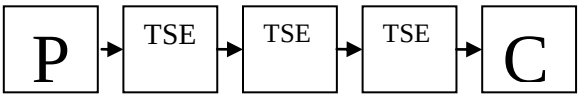
```
    public static String SuperEncript
(String text,String Key){
          String temp =text;
          int seed = SeedMaker(Key);
          temp =
EncriptVigenere(temp, Key);
          temp =
EncriptRandomize(temp, seed);
          return temp;
      }

      public static String
SuperDecript (String text,String
Key){
          String temp =text;
          int seed = SeedMaker(Key);
          temp =
DecriptRandomize(temp, seed);
          temp =
DecriptVigenere(temp, Key);
          return temp;
    }

  public static int SeedMaker(String
Key){
          int val=0;
          for(int
i=0;i<Key.length();++i){
              val+= Key.charAt(i);
          }
          return val;
    }
```

Untuk algoritma SeedMaker dapat diganti – ganti yang penting hasil seed yang didapat adalah hasil seed yang merupakan hasil dari key yang dibangkitkan menjadi seed. Demikian pula dengan key VC. Pada makalah ini tidak dibahas menciptakan key yang baik juga dari key yang dimasukan user. Key yang akan dimasukan ke vigenere chiper dapat juga di bangkitkan dari key yang dimasukan user sehingga key vigenere dan seed selalu unik dan tidak sama dengan key yang dimasukan user.

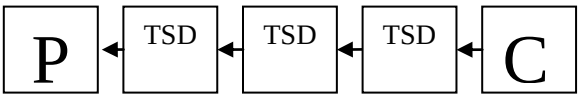
### 2.4 Algoritma Tripel Enkripsi

Pada algoritma ini, yang dilakukan adalah 3 kali proses Super Enkripsi. Hal ini didasari oleh algoritma Tripel DES yang melakukan enkripsi 3 kali. Cara mendekripsikannya juga mudah. Cukup melakukan super dekripsi 3 kali.

Berikut Adalah bagan dari tripel super enkipsi dan tripel super dekripsi




Gambar 4. Bagan Tripel Super Enkripsi




Gambar 5. Bagan Tripel Super Dekripsi

Keterangan

P = PlainText

C = ChiperText

TSE = Tripel Super Enkripsi

TSD = Tripel Super Dekripsi

Berikut adalah source code algoritma tripel enkripsi dan tripel dekripsi

```
  public static String
TripelSuperEncript(String text,String
key){
          String temp = "";
          temp = Encript(text, key);
          temp = Encript(temp, key);
          temp = Encript(temp, key);
          return temp;
      }

     public static String
TripelSuperDecript(String text, String
key){
          String temp = "";
          temp = Decript(text, key);
          temp = Decript(temp, key);
          temp = Decript(temp, key);
          return temp;
      }
```

## III. ANALISIS

### 3.1 Uji Coba

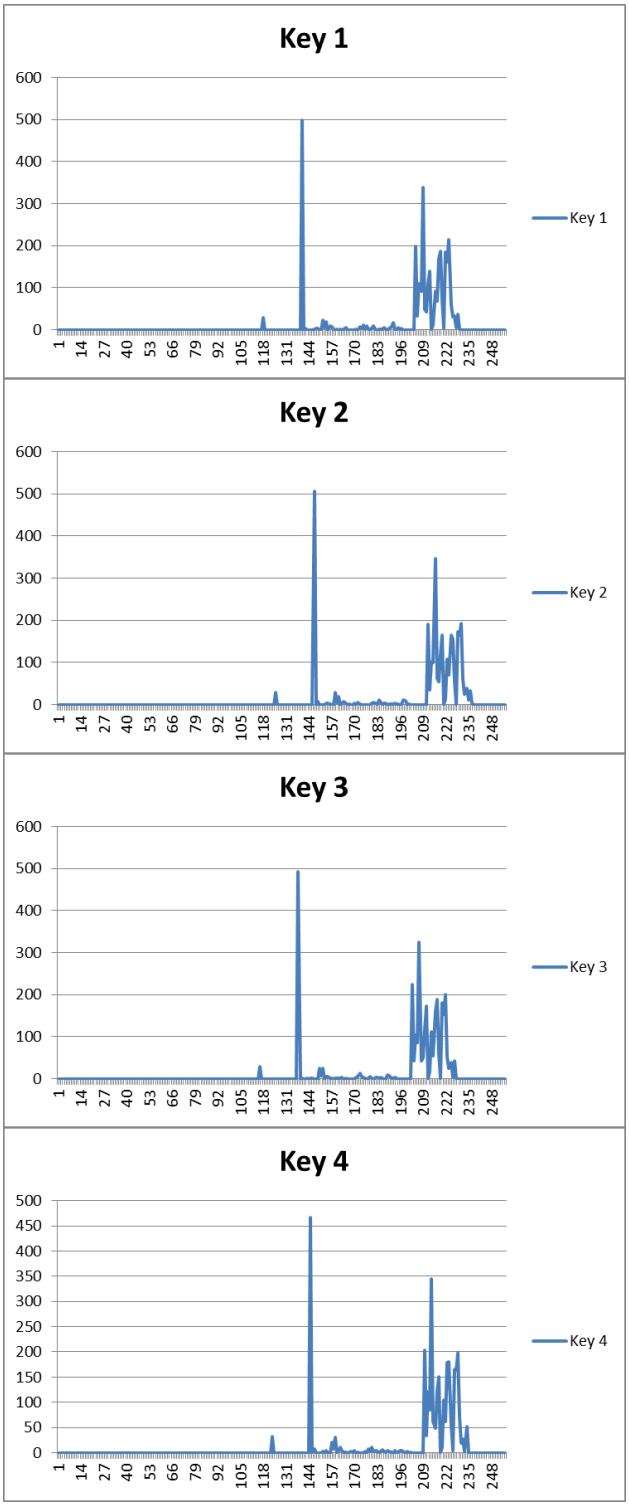
Plaintext yang dimasukan adalah seluruh page http://www.crystalinks.com/croptheories.html

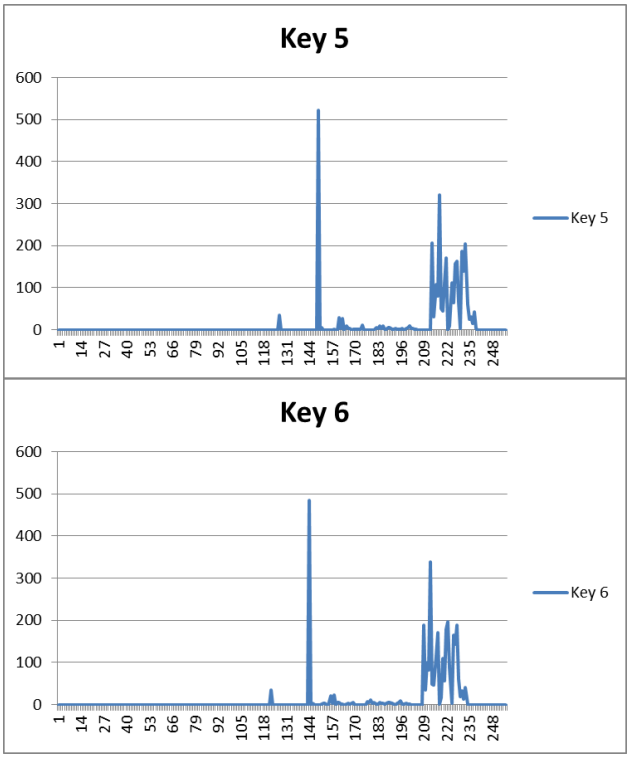
Dan key yang digunakan adalah kripto

Sumbu Y adalah jumlah karakter yang keluar dan Sumbu X adalah karakter pada ASCII ke-X

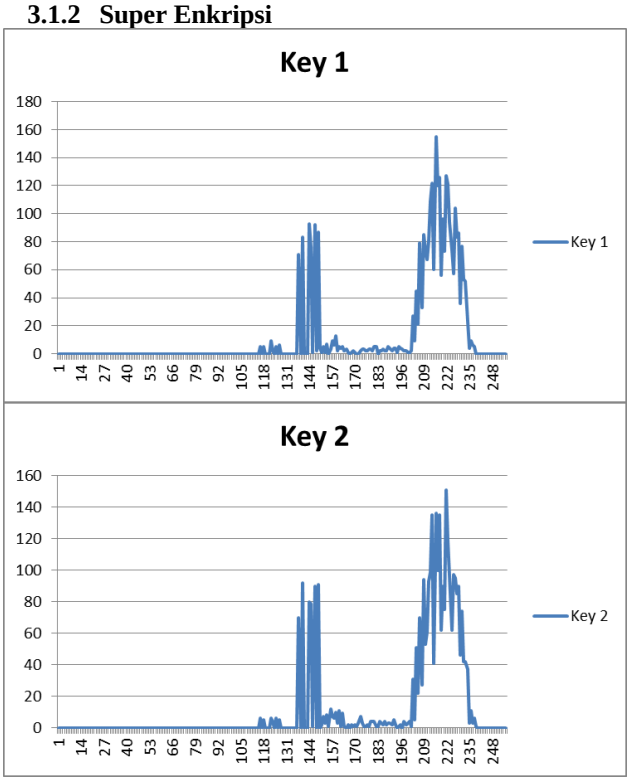
Key ke-n adalah posisi karakter yang di enkirpsi dengan key vigenere chiper ke-n

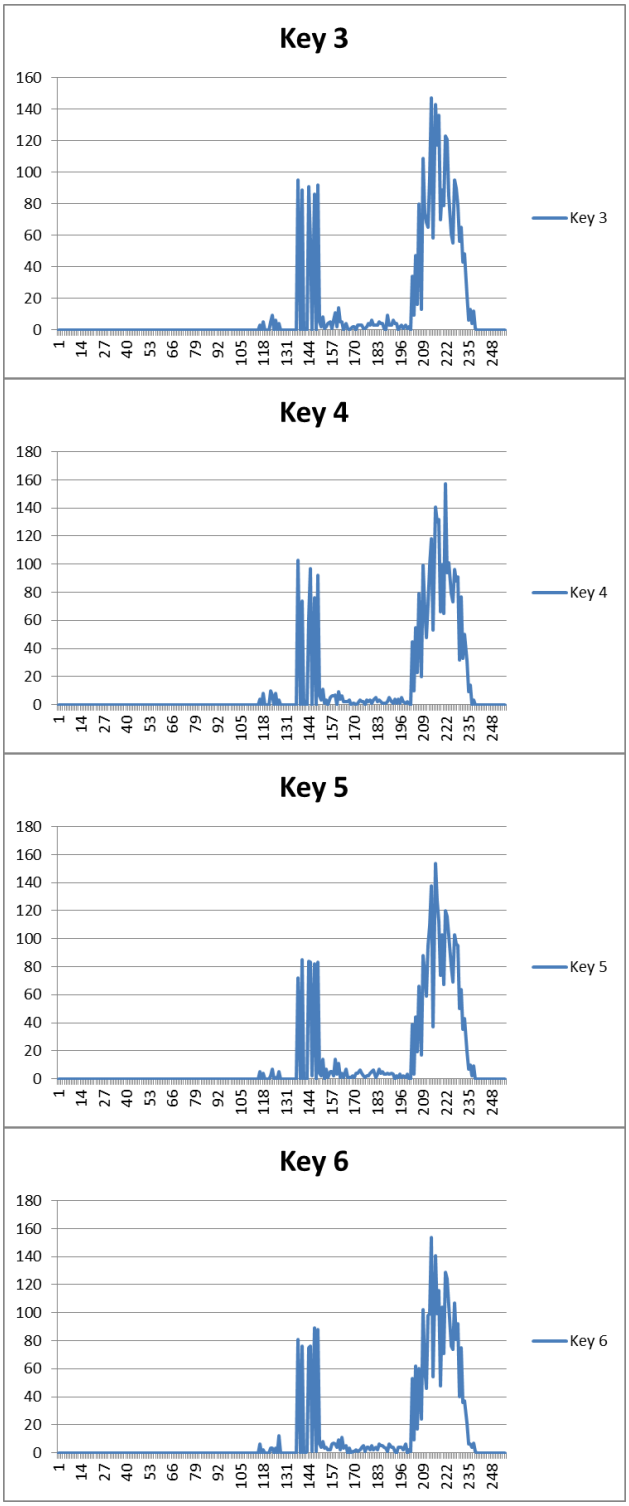
#### 3.1.1 Vigenere Chiper



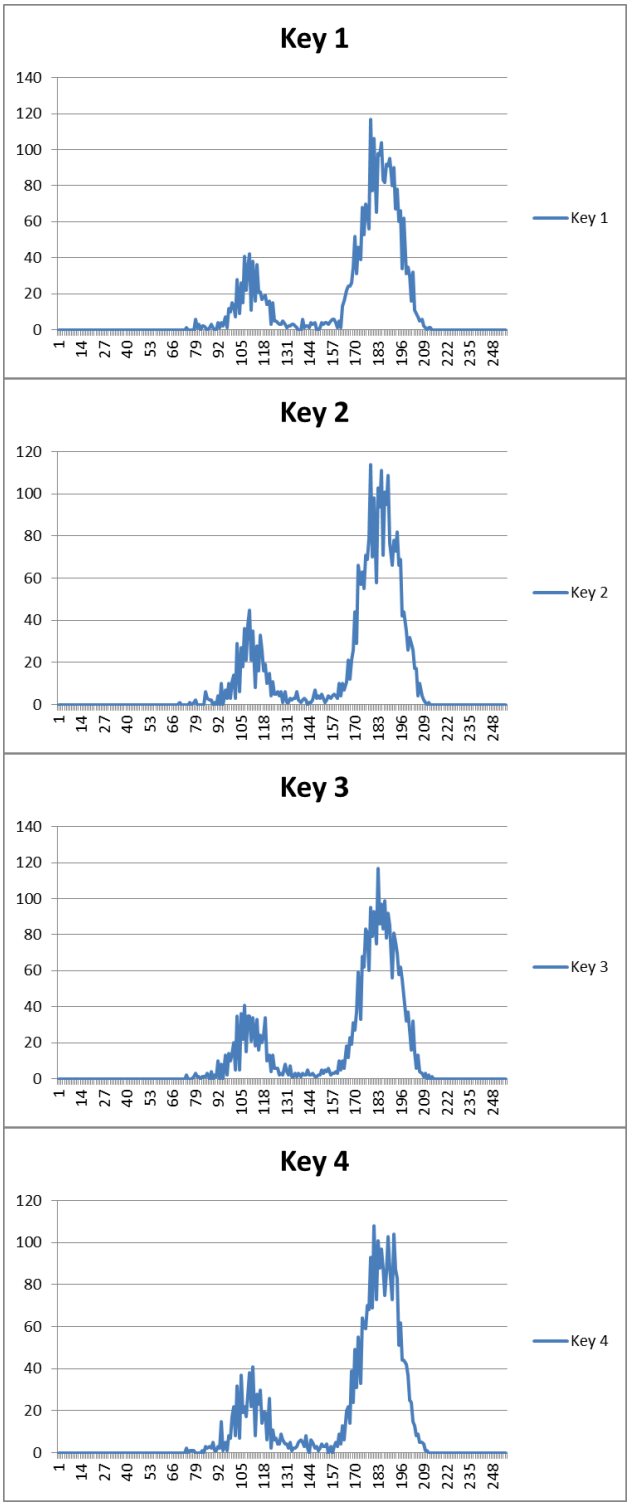


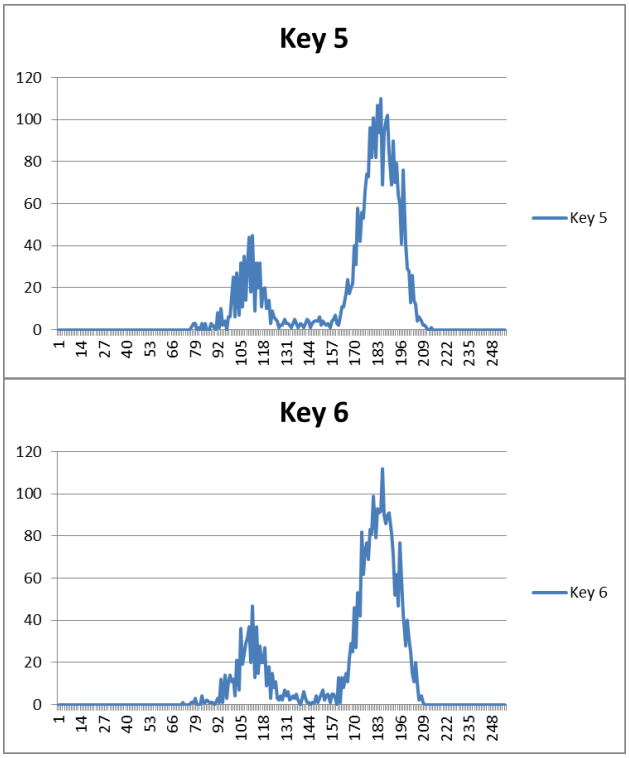

### 3.1.2 Super Enkripsi



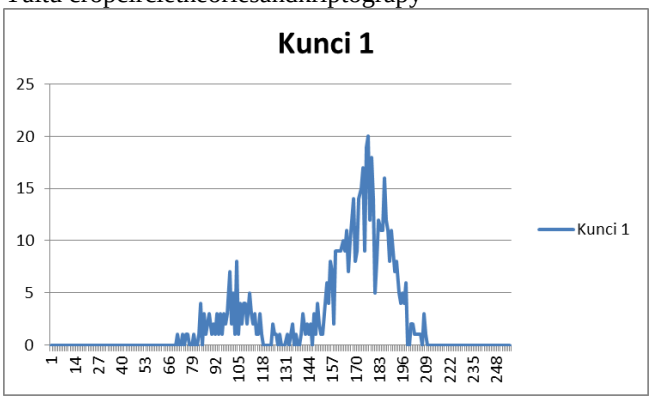




### 3.1.3 Tripel Super Enkripsi




Salah Satu Bagian Kunci Dengan Panjang Kunci 31
Yaitu cropcircletheoriesandkriptograpy




### 3.2 Analisis

Dilihat dari key – key pada vigenere masih jelas analisis dilakukan karena masih dapat di lihat bahwa ada frekuensi karakter yang menonjol. Yaitu sebanyak 500 kali keluar. Dan posisi karakter berada pada karakter ke 110-120 & 140 – 235.

Setelah dilakukan super enkripsi, frekuensi maksimum menurun sampai dengan 160. Dan karakter yang muncul adalah karakter ke 110 – 235

Setelah dilakukan tripel super enkripsi frekuensi maksumum berada pada 110 dan karakter ke 70 – 210

Tripel Super Enkripsi tidak dapat meratakan distribusi karakter, perbedaan karakter terbanyak dan terkecil masih cukup jauh.

Jarak panjang karakter maksumum dapat ditekan dengan memanjangkan kunci. Namun masih tidak dapat meratakan dengan sempurna

## IV. KESIMPULAN

Dengan Super Enkripsi dapat menekan frekuensi maksimum dan menekannya hingga 70% dan mendistribusikan pemerataan karakternya

Dengan Tripel Super Enkripsi menekan hingga 80% dan mendistribusikan pemerataan karakternya lebih baik lagi dibanding dengan Super Enkripsi

Tripel super enkripsi tidak dapat meratakan distribusi karakter yang muncul. Perbedaan karakter terbanyak dengan tersedikit masih cukup mencolok pada grafik diatas padahal penulis mengira pembagian karakter sudah tidak begitu banyak perubahan.

Memecahkan kriptografi ini dengan strategi analisis frekuensi sudah tidak mungkin lagi dipecahkan. Karena adanya transposisi algoritma yang dapat menyalahkan analisisna.

Masih dapat menekan kemunculan frekuensi dengan memanjangkan kunci yang digunakan. Namun distribusi karakter masih tidak sempurna

## REFERENCES

[1] Slide Algoritma Kriptografi Klasik
[2] http://www.crystalinks.com/croptheories.html

## PERNYATAAN

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

Bandung, 19 Maret 2011

Edwin Romelta / 13508052