Bahan Kuliah ke-4

IF5054 Kriptografi

Algoritma Kriptografi Klasik

Disusun oleh:

Ir. Rinaldi Munir, M.T.

Departemen Teknik Informatika
Institut Teknologi Bandung
2004

4. Algoritma Kriptografi Klasik

4.1. Pendahuluan

- Sebelum komputer ada, kriptografi dilakukan dengan algoritma berbasis karakter.

- Algoritma yang digunakan termasuk ke dalam sistem kriptografi simetri dan digunakan jauh sebelum sistem kriptografi kunci publik ditemukan.

- Terdapat sejumlah algoritma yang tercatat dalam sejarah kriptografi (sehingga dinamakan algoritma kriptografi klasik), namun sekarang algoritma tersebut sudah usang karena ia sangat mudah dipecahkan.

- Tiga alasan mempelajari algoritma kriptografi klasik:
  1. Untuk memberikan pemahaman konsep dasar kriptografi.
  2. Dasar dari algoritma kriptografi modern.
  3. Dapat memahami potensi-potensi kelemahan sistem *cipher*.

- Algoritma kriptografi klasik:
  1. *Cipher* Substitusi (*Substitution Ciphers*)
  2. *Cipher* Transposisi (*Transposition Ciphers*)

  Keterangan: *cipher* = algoritma kriptografi

4.2. Cipher Substitusi

- Ini adalah algoritma kriptografi yang mula-mula digunakan oleh kaisar Romawi, Julius Caesar (sehingga dinamakan juga *caesar cipher*), untuk menyandikan pesan yang ia kirim kepada para gubernurnya.

- Caranya adalah dengan mengganti (menyulih atau mensubstitusi) setiap karakter dengan karakter lain dalam susunan abjad (alfabet).

- Misalnya, tiap huruf disubstitusi dengan huruf ketiga berikutnya dari susunan akjad. Dalam hal ini kuncinya adalah jumlah pergeseran huruf (yaitu $k = 3$).

Tabel substitusi:

p_i : 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
c_i : **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**

Contoh 1. Pesan

```
AWASI ASTERIX DAN TEMANNYA OBELIX
```

disamarkan (enskripsi) menjadi

```
DZDVL DVWHULA GDQ WHPDQQBA REHOLA
```

Penerima pesan men-dekripsi cipherteks dengan menggunakan tabel substitusi, sehingga cipherteks

```
DZDVL DVWHULA GDQ WHPDQQBA REHOLA
```

dapat dikembalikan menjadi plainteks semula:

```
AWASI ASTERIX DAN TEMANNYA OBELIX
```

- Dengan mengkodekan setiap huruf abjad dengan *integer* sebagai berikut: $A = 0$, $B = 1$, …, $Z = 25$, maka secara matematis *caesar cipher* menyandikan plainteks p_i menjadi c_i dengan aturan:

$$c_i = E(p_i) = (p_i + 3) \bmod 26 \qquad (1)$$

dan dekripsi cipherteks c_i menjadi p_i dengan aturan:

$$p_i = D(c_i) = (c_i - 3) \bmod 26 \qquad (2)$$

- Karena hanya ada 26 huruf abjad, maka pergeseran huruf yang mungkin dilakukan adalah dari 0 sampai 25. Secara umum, untuk pergeseran huruf sejauh k (dalam hal ini k adalah kunci enkripsi dan deksripsi), fungsi enkripsi adalah

$$c_i = E(p_i) = (p_i + k) \bmod 26 \qquad (3)$$

dan fungsi dekripsi adalah

$$p_i = D(c_i) = (c_i - k) \bmod 26 \qquad (4)$$

Catatan:

1. Pergeseran 0 sama dengan pergeseran 26 (susunan huruf tidak berubah)
2. Pergeseran lain untuk k > 25 dapat juga dilakukan namun hasilnya akan kongruen dengan bilangan bulat dalam modulo 26. Misalnya $k = 37$ kongruen dengan 11 dalam modulo 26, atau $37 \equiv 11 \pmod{26}$.
3. Karena ada operasi penjumlahan dalam persamaan (3) dan (4), maka *caesar cipher* kadang-kadang dinamakan juga *additive cipher*.

- Untuk mengenkripsi/dekripsi pesan yang disusun oleh karakter-karakter teks (ASCII, 256 karakter), maka persamaan 3 dan 4 ditulis

$$c_i = E(p_i) = (p_i + k) \bmod 256 \qquad (5)$$

$$p_i = D(c_i) = (c_i - k) \bmod 256 \qquad (6)$$

- Program Pascal sederhana untuk mengenkripsi dan dekripsi pesan teks (*text file*) dengan *caesar cipher*:

```
program enkripsi;
{ Mengenkripsi berkas 'plain.txt'
  menjadi 'cipher.txt' dengan
  metode caesar cipher }
uses
  crt;
var
   F1, F2 : text;
   p : char;
   c : integer;
   k : integer;

begin
   assign(F1, 'plain.txt');
   reset(F1);

   assign(F2, 'cipher.txt');
   rewrite(F2);

   write('k = ?'); readln(k);
   while not EOF(F1) do
    begin
      while not EOLN(F1) do
       begin
         read(F1, p);
         c := (ord(p) + k) mod 256;
         write(F2, chr(c));
       end;
      readln(F1);
      writeln(F2);
   end;
   close(F1);
   close(F2);
end.
```

```
program dekripsi;
{ Mendekripsi berkas 'cipher.txt'
  menjadi 'plain2.txt' dengan
  metode caesar cipher }
uses
  crt;
var
   F1, F2 : text;
   p : char;
   c : integer;
   k : integer;

begin
   assign(F1, 'cipher.txt');
   reset(F1);

   assign(F2, 'plain2.txt');
   rewrite(F2);

   write('k = ?'); readln(k);
   while not EOF(F1) do
    begin
      while not EOLN(F1) do
       begin
         read(F1, p);
         c := (ord(p) - k) mod 256;
         write(F2, chr(c));
       end;
      readln(F1);
      writeln(F2);
   end;
   close(F1);
   close(F2)
end.
```

Kriptanalisis Terhadap Caesar Cipher

- *Caesar cipher* mudah dipecahkan dengan metode *exhaustive key search* karena jumlah kuncinya sangat sedikit (hanya ada 26 kunci).

Contoh 2. Misalkan kriptanalis menemukan potongan cipherteks (disebut juga *cryptogram*) XMZVH. Diandaikan kriptanalis mengetahui bahwa plainteks disusun dalam Bahasa Inggris dan algoritma kriptografi yang digunakan adalah *caesar cipher*. Untuk memperoleh plainteks, lakukan dekripsi mulai dari kunci yang terbesar, 25, sampai kunci yang terkecil, 1. Periksa apakah dekripsi menghasilkan pesan yang mempunyai makna (lihat Tabel 1).

Tabel 1. Contoh *exhaustive key search* terhadap cipherteks XMZVH

Kunci (*k*) *ciphering*	'Pesan' hasil dekripsi	Kunci (*k*) *ciphering*	'Pesan' hasil dekripsi	Kunci (*k*) *ciphering*	'Pesan' hasil dekripsi
0	**XMZVH**	17	**GVIEQ**	8	**PERNZ**
25	**YNAWI**	16	**HWJFR**	7	**QFSOA**
24	**ZOBXJ**	15	**IXKGS**	6	**RGTPB**
23	**APCYK**	14	**JYLHT**	5	**SHUQC**
22	**BQDZL**	13	**KZMIU**	4	**TIVRD**
21	**CREAM**	12	**LANJV**	3	**UJWSE**
20	**DSFBN**	11	**MBOKW**	2	**VKXTF**
19	**ETGCO**	10	**NCPLX**	1	**WLYUG**
18	**FUHDP**	9	**ODQMY**		

Dari Tabel 1, kata dalam Bahasa Inggris yang potensial menjadi plainteks adalah CREAM dengan menggunakan k = 21. Kunci ini digunakan untuk mendekripsikan cipherteks lainnya.

- Kadang-kadang satu kunci yang potensial menghasilkan pesan yang bermakna tidak selalu satu buah. Untuk itu, kita membutuhkan informasi lainnya, misalnya konteks pesan tersebut atau mencoba mendekripsi potongan cipherteks lain untuk memperoleh kunci yang benar.

Contoh 3. Misalkan potongan cipherteks **HSPPW** menghasilkan dua kemungkinan kunci yang potensial, yaitu $k = 4$ menghasilkan pesan DOLLS dan $k = 11$ menghasilkan WHEEL. Lakukan deksripsi terhadap potongan cipherteks lain tetapi hanya menggunakan $k = 4$ dan $k = 11$ (tidak perlu *exhaustive key search*) agar dapat disimpulkan kunci yang benar.

- Cara lain yang digunakan untuk memecahkan cipherteks adalah dengan statistik, yaitu dengan menggunakan tabel kemunculan karakter, yang membantu mengidentifikasi karakter plainteks yang berkoresponden dengan karakter di dalam cipherteks (akan dijelaskan kemudian).

Jenis-jenis Cipher Subsbtitusi

a. **Cipher abjad-tunggal** (*monoalphabetic cipher* atau *cipher* substitusi sederhana - *simple substitution cipher*)
Satu karakter di plainteks diganti dengan satu karakter yang bersesuaian. Jadi, fungsi *ciphering*-nya adalah fungsi satu-ke-satu.

Jika plainteks terdiri dari huruf-huruf abjad, maka jumlah kemungkinan susunan huruf-huruf cipherteks yang dapat dibuat adalah sebanyak

$$26! = 403.291.461.126.605.635.584.000.000$$

Caesar cipher adalah kasus khusus dari *cipher* abjad tunggal di mana susunan huruf cipherteks diperoleh dengan menggeser huruf-huruf alfabet sejauh 3 karakter.

`ROT13` adalah program enkripsi sederhana yang terdapat di dalam sistem UNIX. `ROT13` menggunakan *cipher* abjad-tunggal dengan pergeseran k = 13 (jadi, huruf `A` diganti dengan **`N`**, `B` diganti dengan **`O`**, dan seterusnya).

Enkripsi arsip dua kali dengan `ROT13` menghasilkan arsip semula:

```
P = ROT13(ROT13(P))
```

b. **Cipher substitusi homofonik** (*Homophonic substitution cipher*)
Seperti *cipher* abjad-tunggal, kecuali bahwa setiap karakter di dalam plainteks dapat dipetakan ke dalam salah satu dari karakter cipherteks yang mungkin. Misalnya huruf `A` dapat berkoresponden dengan **`7`**, **`9`**, atau **`16`**, huruf `B` dapat berkoresponden dengan **`5`**, **`10`**, atau **`23`** dan seterusnya.

Fungsi *ciphering*-nya memetakan satu-ke-banyak (*one-to-many*).

Cipher substitusi homofonik digunakan pertama kali pada tahun 1401 oleh wanita bangsawan Mantua.

Cipher substitusi homofonik lebih sulit dipecahkan daripada *cipher* abjad-tunggal. Namun, dengan *known-plaintext attack*, cipher ini dapat dipecahkan, sedangkan dengan *ciphertext-only attack* lebih sulit.

c. ***Cipher* abjad-majemuk** (*Polyalpabetic substitution cipher*)
Merupakan *cipher* substitusi-ganda (*multiple-substitution cipher*) yang melibatkan penggunaan kunci berbeda.

Cipher abjad-majemuk dibuat dari sejumlah *cipher* abjad-tunggal, masing-masing dengan kunci yang berbeda.

Kebanyakan *cipher* abjad-majemuk adalah *cipher* substitusi periodik yang didasarkan pada periode *m*.

Misalkan plainteks *P* adalah

$$P = p_1 p_2 \ldots p_m p_{m+1} \ldots p_{2m} \ldots$$

maka cipherteks hasil enkripsi adalah

$$E_k(P) = f_1(p_1)\, f_2(p_2) \ldots f_m(p_m)\, f_{m+1}(p_{m+1}) \ldots f_{2m}(p_{2m}) \ldots$$

yang dalam hal ini p_i adalah huruf-huruf di dalam plainteks.

Untuk $m = 1$, *cipher*-nya ekivalen dengan *cipher* abjad-tunggal.

Contoh *cipher* substitusi periodik adalah cipher Vigenere yang ditemukan oleh kriptologi Perancis, Blaise de Vigenere pada abad 16. Misalkan *K* adalah deretan kunci

$$K = k_1\, k_2 \ldots k_m$$

yang dalam hal ini k_i untuk $1 \le i \le m$ menyatakan jumlah pergeseran pada huruf ke-*i*. Maka, karakter cipherteks $y_i(p)$ adalah

$$y_i(p) = (p + k_i) \bmod n \qquad (5)$$

Misalkan periode m = 20, maka 20 karakter pertama dienkripsi dengan persamaan (5), dimana setiap karakter ke-i menggunakan kunci k_i. Untuk 20 karakter berikutnya, kembali menggunakan pola enkripsi yang sama.

Cipher abjad-majemuk ditemukan pertama kali oleh Leon Battista pada tahun 1568. Metode ini digunakan oleh tentara AS selama Perang Sipil Amerika.

Meskipun *cipher* abjad-majemuk dapat dipecahkan dengan mudah (dengan bantuan komputer), namun anehnya banyak program keamanan komputer (*computer security*) yang menggunakan *cipher* jenis ini.

d. **Cipher substitusi poligram** (*Polygram substitution cipher*)
Blok karakter disubstitusi dengan blok cipherteks. Misalnya ABA diganti dengan **RTQ**, ABB diganti dengan **SLL**, dan lain-lain.

Playfair cipher, ditemukan pada tahun 1854, termasuk ke dalam *cipher* substitusi poligram dan digunakan oleh negara Inggris selama Perang Dunia I.

4.3. *Cipher* Transposisi

- Pada *cipher* transposisi, plainteks tetap sama, tetapi urutannya diubah. Dengan kata lain, algoritma ini melakukan *transpose* terhadap rangkaian karakter di dalam teks.

- Nama lain untuk metode ini adalah **permutasi**, karena *transpose* setiap karakter di dalam teks sama dengan mempermutasikan karakter-karakter tersebut.

Contoh 4. Misalkan plainteks adalah

```
DEPARTEMEN TEKNIK INFORMATIKA ITB
```

Untuk meng-enkripsi pesan, plainteks ditulis secara horizontal dengan lebar kolom tetap, misal selebar 6 karakter (kunci $k = 6$):

```
DEPART
EMENTE
KNIKIN
FORMAT
IKAITB
```

maka cipherteksnya dibaca secara vertikal menjadi

```
DEKFIEMNOKPEIRAANKMIRTIATTENTB
```

Untuk mendekripsi pesan, kita membagi panjang cipherteks dengan kunci. Pada contoh ini, kita membagi 30 dengan 6 untuk mendapatkan 5.

Algoritma dekripsi identik dengan algoritma enkripsi. Jadi, untuk contoh ini, kita menulis cipherteks dalam baris-baris selebar 5 karakter menjadi:

```
DEKFI
EMNOK
PEIRA
ANKMI
RTIAT
TENTB
```

Dengan membaca setiap kolom kita memperoleh pesan semula:

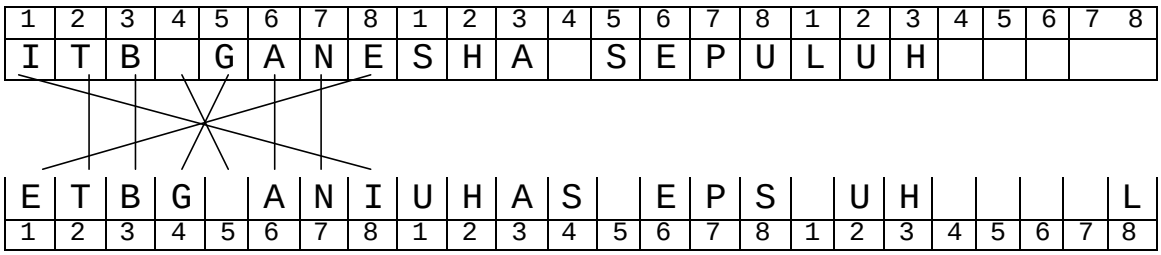
```
DEPARTEMEN TEKNIK INFORMATIKA ITB
```

Variasi dari metode transposisi lainnya ditunjukkan pada Contoh 5 dan Contoh 6.

Contoh 5. Misalkan plainteks adalah

```
ITB GANESHA SEPULUH
```

Plainteks diblok atas delapan karakter. Kemudian, pada tiap blok, karakter pertama dan karakter terakhir dipertukarkan, demikian juga karakter pertengahan:



maka cipherteksnya adalah

```
ETBG ANIUHAS EPS UH    L
```

Dekripsi dilakukan dengan cara yang sama, yaitu cipherteks diblok atas delapan karakter. Kemudian, pada tiap blok, karakter pertama dan karakter terakhir dipertukarkan, demikian juga karakter pertengahan.

Contoh 6. Misalkan plainteks adalah

```
CRYPTOGRAPHY AND DATA SECURITY
```

Plainteks disusun menjadi 3 baris (k = 3) seperti di bawah ini:

```
C     T     A    A      A      E      I
 R   P O   R P Y  N   D  T   S  C   R  T
   Y     G     H     D      A      U     Y
```

maka cipherteksnya adalah

```
CTAAAEIRPORPYNDTSCRTYGHDAUY
```

- Kriptografi dengan alat *scytale* yang digunakan oleh tentara Sparta pada zaman Yunani termasuk ke dalam *cipher* transposisi.

4.4 Lebih Jauh dengan *Cipher* Abjad-tunggal

- Seperti sudah disebutkan sebelum ini, metode *cipher* abjad-tunggal mengganti setiap huruf di dalam abjad dengan sebuah huruf lain dalam abjad yang sama.

- Jumlah kunci di dalam *cipher* abjad-tunggal sama dengan jumlah cara menyusun 26 huruf abjad tersebut, yaitu sebanyak

  $$26! = 403.291.461.126.605.635.584.000.000$$

  Ini berarti terdapat 26! buah kunci untuk menyusun huruf-huruf alfabet ke dalam tabel substitusi.

- Contohnya, susunan guruf-huruf untuk cipherteks diperoleh dengan menyusun huruf-huruf abjad secara acak seperti tabel substitusi berikut:

Tabel substitusi:

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

- Satu cara untuk membangkitkan kunci adalah dengan sebuah kalimat yang mudah diingat.
  Misal kuncinya adalah

  we hope you enjoy this book

  Dari kunci tersebut, buang perulangan huruf sehingga menjadi

  wehopyunjtisbk

  lalu sambung dengan huruf-huruf lain yang tidak terdapat di dalam kalimat tersebut sehingga menjadi

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

  Dengan demikian, tabel substitusi yang diperoleh adalah

Tabel substitusi:

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

4.5 Menerka Plainteks dari Cipherteks

- Kadang-kadang kriptanalis melakukan terkaan untuk mengurangi jumlah kunci yang mungkin ada.

- Terkaan juga dilakukan kriptanalis untuk memperoleh sebanyak mungkin plainteks dari potongan cipherteks yang disadap. Plainteks yang diperoleh dari hasil terkaan ini biasanya digunakan dalam *known-plaintext attack.*

- Asumsi yang digunakan: kriptanalis mengetahui bahwa pesan ditulis dalam Bahasa Inggris dan algoritma kriptografi yang digunakan adalah cipher abjad-tunggal.

  Contoh kasus 1: Kriptanalis mempunyai potongan cipherteks

  G WR W RWL

  Karena hanya ada dua kata yang panjangnya satu huruf dalam Bahasa Inggris (yaitu I dan A), maka **G** mungkin menyatakan huruf A dan **W** menyatakan huruf I, atau sebaliknya.

  Kemungkinan **G** adalah huruf A dapat dieliminasi, maka diapstikan **G** = I, sehingga dengan cepat kriptanalis menyimpulkan bahwa potongan cipherteks tersebut adalah

  I AM A MA*

  Dengan pengetahuan Bahasa Inggris, karakter terakhir (*) hampir dipastikan adalah huruf N, sehingga kalimatnya menjadi

  I AM A MAN

  Hasil ini mengurangi jumlah kunci dari 26! menjadi 22!

Contoh kasus 2: Kriptanalis mempunyai potongan cipherteks

HKC

Tidak banyak informasi yang dapat disimpulkan dari *cryptogram* di atas. Namun kriptanalis dapat mengurangi beberapa kemungkinan kunci, karena –sebagai contoh– tidak mungkin Z diganti dengan **H**, Q dengan **K**, dan K dengan **C**.

Namun, jumlah kemungkinan kunci yang tersisa tetap masih besar. Jika pesan yang dikirim memang hanya tiga huruf, maka *exhaustive key search* akan menghasilkan kata dengan tiga huruf berbeda yang potensial sebagai plainteks.

Contoh kasus 3: Kriptanalis mempunyai potongan cipherteks

HATTPT

Dalam hal ini, kriptanalis dapat membatasi jumlah kemungkinan huruf plainteks yang dipetakan menjadi T.

Kriptanalis mungkin mendeduksi bahwa salah satu dari **T** atau **P** merepresentasikan huruf vokal. Kemungkinan plainteksnya adalah CHEESE, MISSES, dan CANNON.

Contoh kasus 4: Kriptanalis mempunyai potongan cipherteks

HATTPT

dan diketahui informasi bahwa pesan tersebut adalah nama negara. Dengan cepat kriptanalis menyimpulkan bahwa *polygram* tersebut adalah GREECE.

Dalam hal ini, kriptanalis dapat membatasi jumlah kemungkinan huruf plainteks yang dipetakan menjadi T.

Kriptanalis mungkin mendeduksi bahwa salah satu dari **T** atau **P** merepresentasikan huruf vokal. Kemungkinan plainteksnya adalah CHEESE, MISSES, dan CANNON.

Metode Statistik dalam Kriptanalisis

- Metode yang paling umum digunakan dalam memecahkan cipherteks adalah menggunakan statistik.

- Dalam hal ini, kriptanalis menggunakan tabel frekuensi kemunculan huruf-huruf dalam teks bahasa Inggris. Tabel 2 memperlihatkan frekuensi kemunculan huruf-huruf abjad yang diambil dari sampel yang mencapai 300.000 karakter di dalam sejumlah novel dan suratkabar.

Tabel 2. Frekunsi kemunculan (relatif) huruf-huruf dalam teks Bahasa Inggris

Huruf	%	Huruf	%
A	8,2	N	6,7
B	1,5	O	7,5
C	2,8	P	1,9
D	4,2	Q	0,1
E	12,7	R	6,0
F	2,2	S	6,3
G	2,0	T	9,0
H	6,1	U	2,8
I	7,0	V	1,0
J	0,1	W	2,4
K	0,8	X	2,0
L	4,0	Y	0,1
M	2,4	Z	0,1

- Tabel 2 di atas pada mulanya dipublikasikan di dalam *Cipher-Systems: The Protection of Communications* dan dikompilasi oleh H. J. Beker dan F.C. Piper

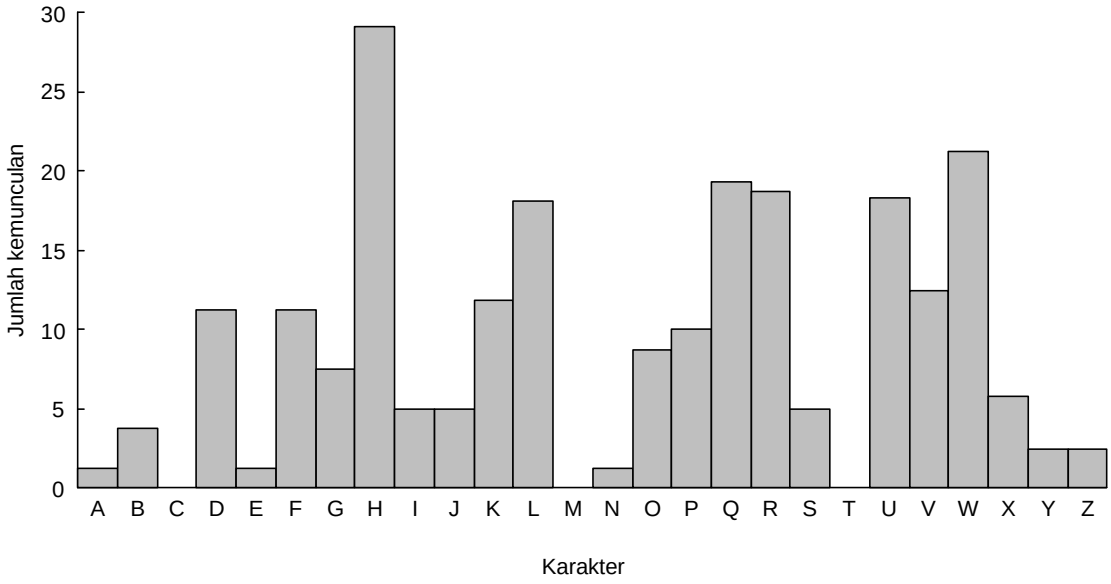
- Terdapat sejumlah tabel frekuensi sejenis yang dipublikasikan oleh pengarang lain, namun secara umum persentase kemunculan tersebut konsisten pada sejumlah tabel.

- Bila *cipher* abjad-tunggal digunakan untuk meng-engkripsi pesan, maka kemunculan huruf-huruf di dalam plainteks tercermin pada tabel 2 di atas. Misalnya bila di dalam *cipher* abjad-tunggal huruf **R** menggantikan huruf E, maka frekuensi **R** di dalam cipherteks relatif sama dengan frekuensi E di dalam plainteksnya.

Contoh 7. Misalnya terdapat cipherteks yang panjang sebagai berikut:

```
DIX DR TZX KXCQDIQ RDK XIHPSZXKPIB TZPQ TXGT
PQ TD QZDM TZX KXCJXK ZDM XCQPVN TZPX TNSX DR
HPSZXK HCI LX LKDUXI. TZX MDKJ QTKFHTFKX DR
TZX SVCPITXGT ZCQ LXXI SKXQXKWXJ TD OCUX TZX
XGXKHPQX XCQPXK. PR MX ZCJ MKPTTXI TZX.
HKNSTDBKCOPI BKDFSQ DR RPWX VXTTXKQ TZXI PT
MDFVJ ZCWX LXXI ZCKJXK. TD HDIWPIHX
NDFKQXVWXQ DR TZPQ SCPKQ SCPKQ DR KXCJXKQ
HCI SKDWPJX XCHZ DTZXK MPTZ HKNSTDBKCOQ
MPTZ TZPQ VXTTXK BKDFSIB
```

Histogram yang memperlihatkan frekuensi kemunculan relatif huruf-huruf di dalam cipherteks diperlihatkan di bawah ini:



Gambar 1. Histogram yang menyatakan frekuensi kemunculan (relatif) huruf-huruf di dalam cipherteks di dalam Contoh 7.

Dari histogram pada Gambar 1, karakter yang paling sering muncul di dalam cipherteks adalah **H**. Kita dapat menyimpulkan sementara bahwa **H** di dalam cipherteks menggantikan huruf E di dalam plainteks.

Cara yang sama dicoba untuk karakter-karakter lain di dalam cipherteks.

Tetapi kita belum dapat memastikannya. Masih diperlukan:

- cara *trial and error*
- pengetahuan tentang bahasa
- konteks plainteks
- intuisi