Bahan kuliah II4021 Kriptografi

03 - Kriptografi Klasik

(Bagian 2)

Oleh: Rinaldi Munir

Program Studi Sistem dan Teknologi Informasi

Sekolah Teknik Elektro dan Informatika

Institut Teknologi Bandung

2025

- Pada Bagian 2 dan 3 akan dibahas bebeapa *cipher* klasik popular lainnya:
    1. Vigenere Cipher
    2. Playfair Cipher
    3. Affine Cipher
    4. Hill Cipher
    5. Enigma Cipher

1. Vigènere Cipher

- Termasuk ke dalam *cipher* abjad-majemuk (*polyalpabetic substitution cipher*).

- Penemu cipher ini sebenarnya adalah Giovan Batista Belaso, karena ia menggambarkan pertama kali pada tahun 1553 seperti ditulis di dalam bukunya *La Cifra del Sig. Giovan Batista Belaso.*

- Namun, *cipher* ini disempurnakan dan dipublikasikan oleh diplomat (sekaligus seorang kriptologis) Perancis, Blaise de Vigènere pada abad 16 (tahun 1586).

- Pada abat ke-19, banyak orang yang mengira Vigenère adalah penemu cipher ini, sehingga dikenal luas sebagai *Vigenère Cipher*.

- *Cipher* ini berhasil dipecahkan oleh Babbage dan Kasiski pada pertengahan Abad 19 (akan dijelaskan pada materi selanjutnya).

- Kasiski menguraikan langkah-Langkah untuk menemukan panjang kunci (bukan huruf-huruf kuncikunci).

- *Vigènere Cipher* digunakan oleh Tentara Konfiderasi (*Confederate Army*) pada Perang Sipil Amerika (*American Civil war*).

- Perang Sipil terjadi setelah *Vigènere Cipher* berhasil dipecahkan.

- *Vigènere Cipher* menggunakan matriks *Vigènere* (*Vigenere square*) untuk melakukan enkripsi dan dekripsi.

Plaintext

Key	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

- Setiap baris *i* di dalam bujursangkar menyatakan huruf-huruf cipherteks yang diperoleh menggunakan *Caesar Cipher* dengan kunci *k* = *i*.
- Artinya, setiap baris *i* merupakan pergeseran huruf alfabet sejauh *i* ke kanan

Plaintext

Key

	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

→ baris ke-0

→ baris ke-25

A = 0, B = 1, C = 2, D = 3, E = 4, F = 5, G = 6, H = 7, I = 8, J = 9, K = 10
L = 11, M = 12, N = 13, O = 14, P = 15, Q = 16, R = 17, S = 18, T = 19, U = 20
V = 21, W = 22, X = 23, Y = 24, Z = 25

- Kunci adalah string: $K = k_1 k_2 \ldots k_m$

  k_i untuk $1 \leq i \leq m$ menyatakan huruf-huruf alfabet

- Jika panjang kunci lebih pendek daripada panjang plainteks, maka kunci diulang secara periodik.

- Misalkan panjang kunci m = 10, maka 10 huruf pertama plainteks dienkripsi dengan kunci K, setiap huruf ke-i menggunakan kunci k_i.

Contoh: kunci = `sony` (dalam angka: 18, 14, 13, 24)

Plainteks: `thisplaintext`

Kunci: `sonysonysonys`

Ini artinya setiap huruf plainteks dienkripsi menggunakan *Caesar Cipher* dengan kunci *k* yang berbeda-beda

Untuk 10 karakter berikutnya, kembali menggunakan pola enkripsi yang sama.

- Enkripsi dilakukan dengan mencari titik potong huruf plainteks dengan huruf kunci:

Plainteks : **t**hisplaintext
Kunci : **s**onysonysonys
Cipherteks: **L**

Plainteks

KUNCI	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 4.3 Enkripsi huruf T dengan kunci s

A = 0, B = 1, C = 2, D = 3, E = 4, F = 5, G = 6, H = 7, I = 8, J = 9, K = 10
L = 11, M = 12, N = 13, O = 14, P = 15, Q = 16, R = 17, S = 18, T = 19, U = 20
V = 21, W = 22, X = 23, Y = 24, Z = 25

- Hasil enkripsi seluruhnya adalah sebagai berikut:

  Plainteks : `thisplaintext`
  Kunci : `sonysonysonys`
  Cipherteks : `LVVQHZNGFHRVL`

- Tanpa menggunakan *Vigenere Square* pun enkripsi tetap dapat dihitung secara *Caesar Cipher* dengan menjumlahkan plainteks p_j dengan kunci k_i dalam modulus 26:

  Enkripsi: $c_j = E(p_j) = (p_j + k_i) \bmod 26$ (1)

  Dekripsi: $p_j = D(c_j) = (c_j - k_i) \bmod 26$ (2)

  Contoh:

  (t + s) mod 26 = (19 + 18) mod 26 = 37 mod 26 = 11 = L

  (h + o) mod 26 = (7 + 14) mod 26 = 21 mod 26 = 21 = V, dst

- Kelebihan Vigenere Cipher: huruf plainteks yang sama tidak selalu dienkripsi menjadi huruf cipheteks yang sama pula, bergantung huruf kunci yang digunakan.

  Contoh: pada contoh di atas, huruf plainteks `T` dapat dienkripsi menjadi **`L`** atau **`H`**, dan huruf cipherteks **`V`** dapat merepresentasikan huruf plainteks `H`, `I`, dan `X`

- Hal di atas merupakan karakteristik dari *cipher* abjad-majemuk: setiap huruf cipherteks dapat memiliki kemungkinan banyak huruf plainteks.

- Bandingkan dengan *cipher* abjad-tunggal, setiap huruf cipherteks selalu menggantikan huruf plainteks tertentu.

Plainteks:

Dinas Pendidikan Kota Ternate meminta kepada pihak sekolah dan orang tua siswa untuk jenjang pendidikan SD dan SMP se-Kota Ternate untuk melarang para siswa membawa permainan lato-lato yang sedang tren itu ke sekolah, karena akan mengganggu kegiatan belajar mengajar yang dinilai berbahaya sehingga mengantisipasi kecelakaan bagi anak di daerah itu.

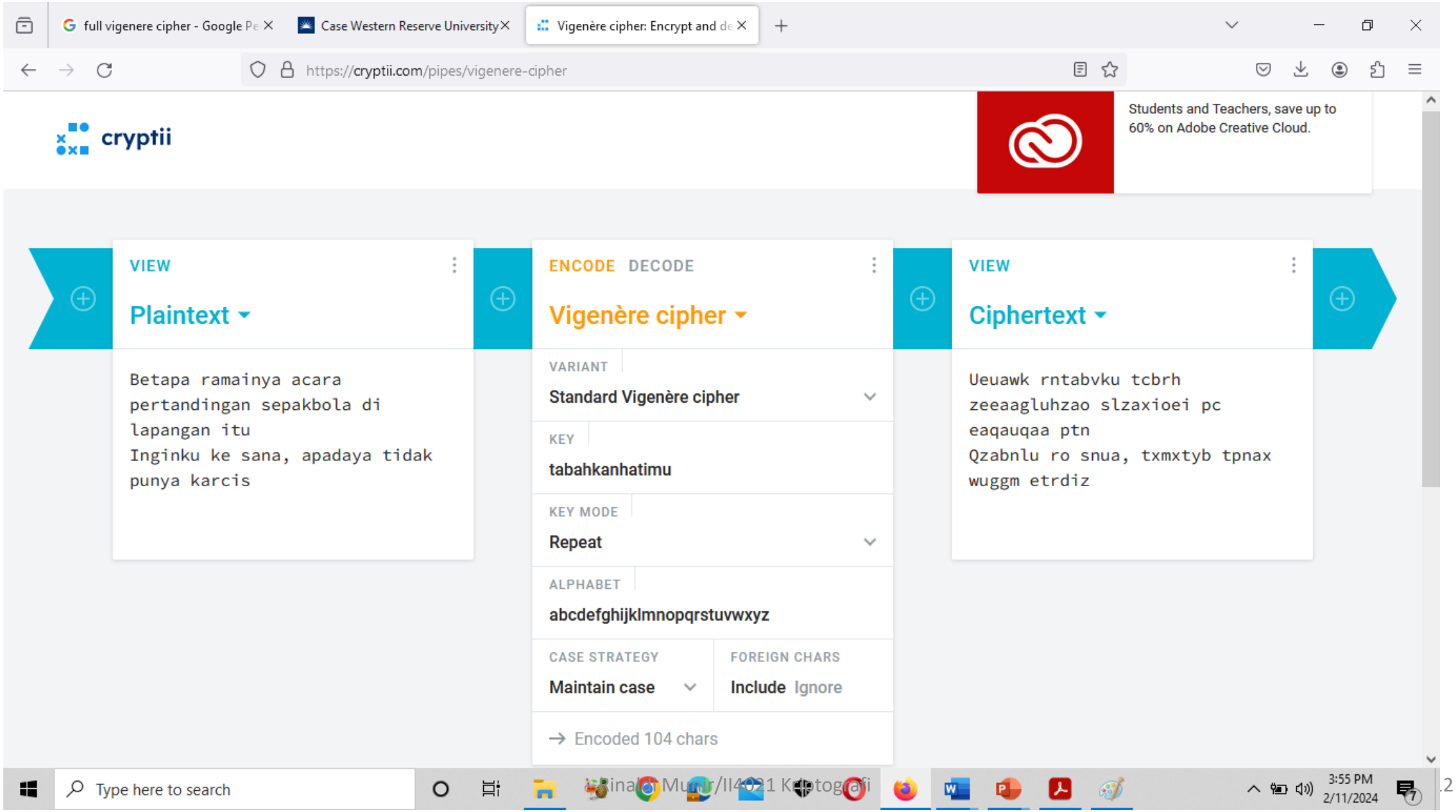
Kunci: selatsunda

Cipherteks:

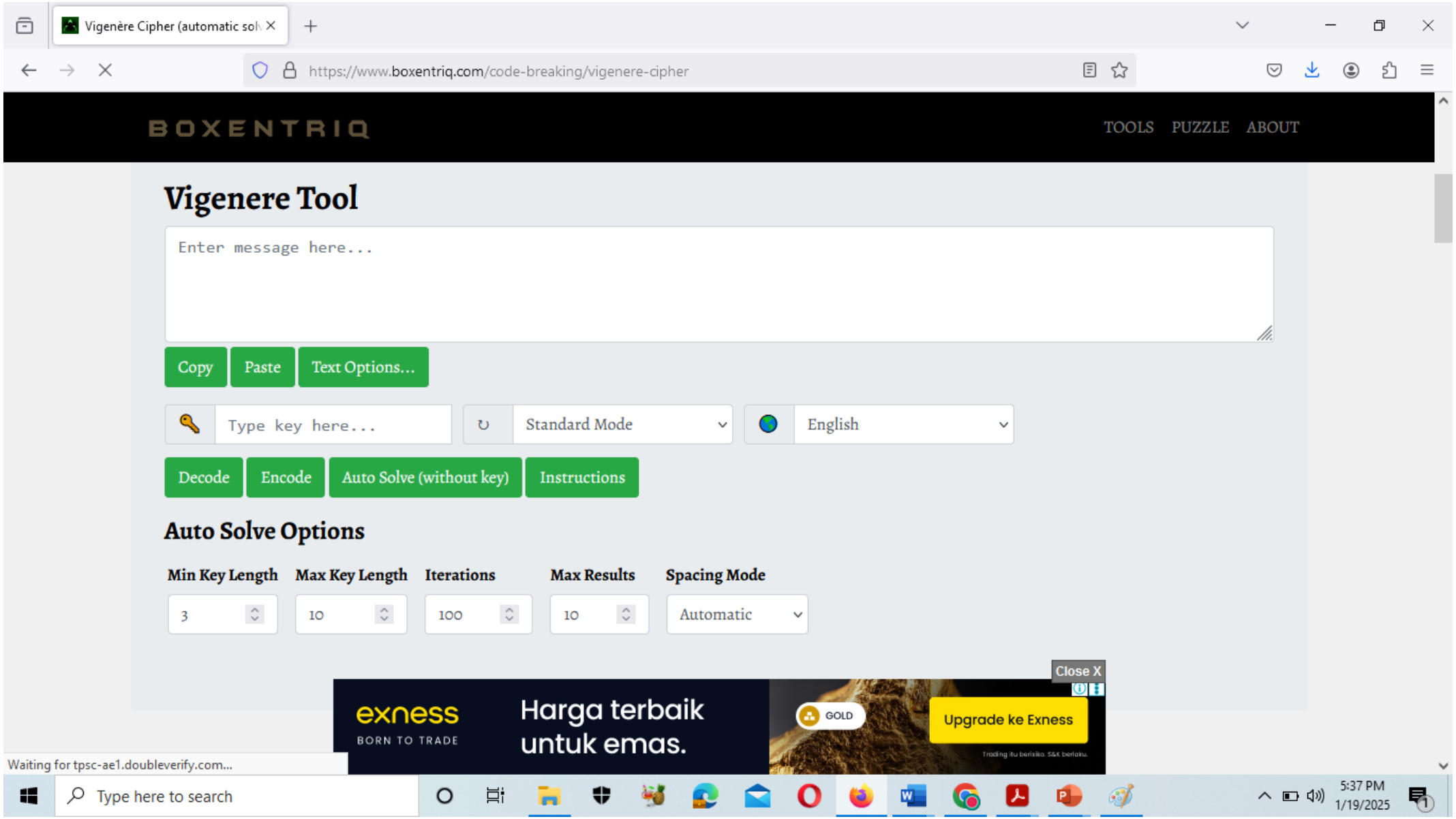
(dikelompokkan 4-huruf)

VMYAL HYAGI VMVAG CIGDT WVYAM WGRPI FXLKX HUQDP ALLKL WEBOA
ZHLNH JUAJT MEDIL OUHQT MOUEG BUAJP WROIW AENQS VHLNL EJFHK
GXLTX JHNWE MREUD EYYDR SRRPT JUFLS OEXEF TUJDP WVXAB FUAOA
LSWAM GSNQG KIOAG YNEHN AXFKX KYXRL SLVAK WHNDK SRXEG YANQG
YYVEZ AUGDN TIWAC SLZHN YEUAK QUAJD ARTLT AVRUB SLLYT KYULN YKLMX
FANQT AWTPT KCXHC WPLKT SHODG AEYAD VCQDE JESIM M

- Demo Vigenere Cipher online: https://cryptii.com/pipes/vigenere-cipher



https://www.boxentriq.com/code-breaking/vigenere-cipher



[7]:
```python
def Vigenere_cipher_encrypt(plainteks, kunci):
    cipherteks = ""
    kunci = kunci.lower()
    indeks_kunci = 0
    for char in plainteks:
        if char.isalpha():           # Hanya memproses huruf alfabet saja
            start = ord('A') if char.isupper() else ord('a')
            k = ord(kunci[indeks_kunci]) -  ord('a')
            c = (ord(char) - start + k) % 26   # Kodekan huruf ke angka 0 s/d 25, lalu enkripsi dengan Caesar Cipher
            c = c + start            # Kembalikan ke posisi semula
            cipherteks = cipherteks + chr(c)      # sambung setiap huruf cipherteks
            indeks_kunci = (indeks_kunci + 1) % len(kunci)
        else:
            cipherteks = cipherteks + char     # Pesan yang bukan huruf tidak dienkripsi, dibiarkan saja
    return cipherteks
```

[23]:
```python
def Vigenere_cipher_decrypt(cipherteks, kunci):
    plainteks = ""
    kunci = kunci.lower()
    indeks_kunci = 0
    for char in cipherteks:
        if char.isalpha():           # Hanya memproses huruf alfabet saja
            start = ord('A') if char.isupper() else ord('a')
            k = ord(kunci[indeks_kunci]) -  ord('a')
            c = (ord(char) - start - k) % 26   # Kodekan huruf ke angka 0 s/d 25, lalu enkripsi dengan Caesar Cipher
            c = c + start            # Kembalikan ke posisi semula
            plainteks = plainteks + chr(c)      # sambung setiap huruf plainteks
            indeks_kunci = (indeks_kunci + 1) % len(kunci)
        else:
            plainteks = plainteks + char     # Pesan yang bukan huruf tidak didekripsi, dibiarkan saja
    return plainteks
```

Vigenere Cipher dalam Python

[13]:
```
pesan = input("ketikkan pesan anda: ")
```

```
ketikkan pesan anda:  Sekarang tahun 2025, semoga kita sehat selalu, aamin!!
```

[5]:
```
kunci = input("kunci: ")
```

```
kunci:  apakabar
```

[21]:
```
cipherteks = Vigenere_cipher_encrypt(pesan, kunci)
```

[22]:
```
print(f"Pesan terenkripsi: {cipherteks}")
```

```
Pesan terenkripsi: Stkkrbnx tphen 2025, tedova uiua jewad sflrlj, akmjn!!
```

[24]:
```
plainteks = Vigenere_cipher_decrypt(cipherteks, kunci)
```

[25]:
```
print(f"Pesan hasil dekripsi: {plainteks}")
```

```
Pesan hasil dekripsi: Sekarang tahun 2025, semoga kita sehat selalu, aamin!!
```

Latihan

- Ubahlah program Vigenere Cipher di atas sehingga dapat melakukan enkripsi pesan berupa file teks.

- Jika periode kunci diketahui dan tidak terlalu panjang, maka kunci dapat ditentukan dengan menulis program komputer untuk melakukan *exhaustive key search*.

- Contoh: Diberikan cipherteks sbb:

```
TGCSZ GEUAA EFWGQ AHQMC
```

dan diperoleh informasi bahwa panjang kunci adalah *p* huruf dan plainteks ditulis dalam Bahasa Inggris, maka *running* program dengan mencoba semua kemungkinan kunci yang panjangnya tiga huruf, lalu periksa apakah hasil dekripsi dengan kunci tersebut menyatakan kata yang berarti.

Cara ini membutuhkan usaha percobaan sebanyak 26^p kali.

- Ada cara yang lebih sangkil menemukan panjang kunci yaitu dengan menggunakan Metode Kasiski sbb.

Kriptanalisis Vigenere Cipher

- Friedrich Kasiski adalah orang yang pertama kali memecahkan *Vigènere cipher* pada Tahun 1863.

Friedrich Kasiski
Born: November 29, 1805 @ Schlochau, Kingdom of Prussia
Died: May 22, 1881 (aged 75) @ Neustettin, German Empire
Nationality: German

- Metodenya dinamakan metode Kasiski

- Metode Kasiski tidak secara langsung menemukan kunci Vigenere Cipher, tetapi membantu menemukan panjang kunci *Vigenere cipher*.

- Metode Kasiski memanfaatkan keuntungan bahwa bahasa Inggris tidak hanya mengandung perulangan huruf,

- tetapi juga perulangan pasangan huruf atau tripel huruf, seperti `TH`, `THE`, `EN`, dsb.

- Perulangan kelompok huruf ini ada kemungkinan menghasilkan kriptogram yang berulang.

Contoh 1:

Plainteks : **crypto**isshortfor**crypto**graphy

Kunci : abcdabcdabcdabcdabcdabcdabcdabcd

Cipherteks : **CSASTP**KVSIQUTGQU**CSASTP**IUAQJB

- Pada contoh ini, crypto dienkripsi menjadi kriptogram yang sama, yaitu **CSATP**.

- Tetapi kasus seperti ini tidak selalu demikian, misalnya pada contoh berikut ini….

Contoh 2:

Plainteks : **crypto**isshortfor**crypto**graphy

Kunci : abcdefabcdefabcdefabcdefabcd

Cipherteks : **CSASXT**ITUKWSTGQU**CWYQVR**KWAQJB

- Pada contoh di atas, crypto tidak dienkripsi menjadi kriptogram yang sama.

- Mengapa bisa demikian?

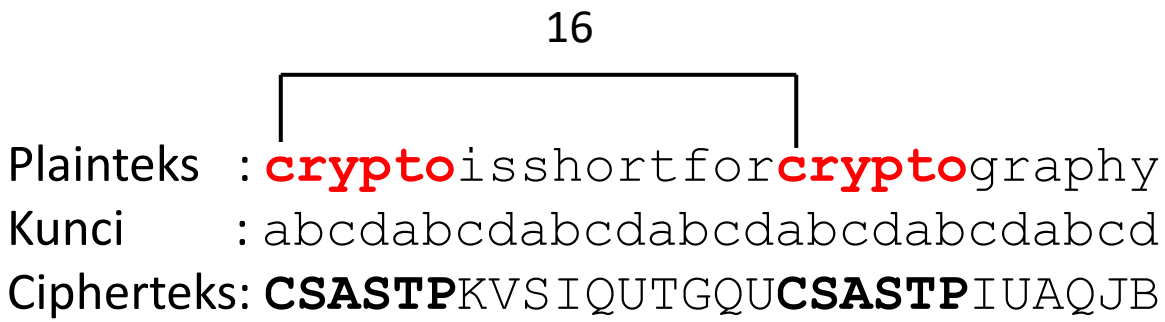
- Secara intuitif: jika jarak antara dua buah *string* yang berulang di dalam plainteks merupakan kelipatan dari panjang kunci,

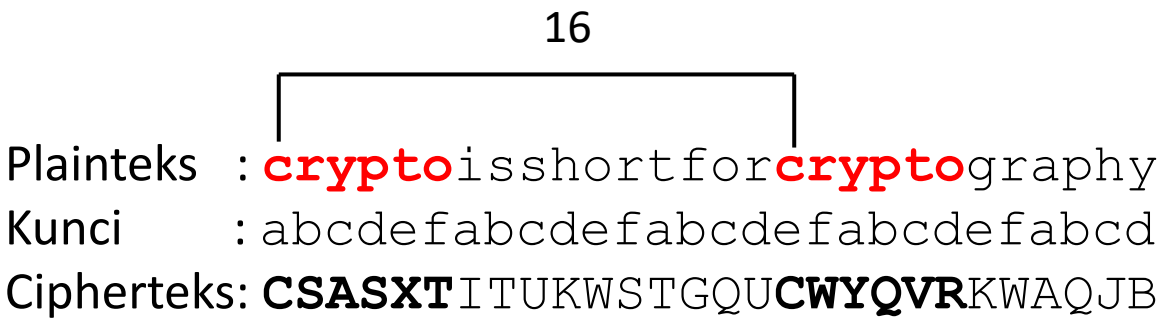
- maka *string* yang sama tersebut akan muncul menjadi kriptogram yang sama pula di dalam cipherteks.

- Pada Contoh 1,

  - kunci = `abcd`
  - panjang kunci = 4
  - jarak antara dua `crypto` yang berulang = 16
  - 16 = kelipatan 4

  ∴ `crypto` dienkripsi menjadi kriptogram yang sama





- Pada Contoh 2,
  - kunci = `abcdef`
  - panjang kunci = 6
  - jarak antara dua `crypto` yang berulang = 16
  - 16 bukan kelipatan 6

∴ `crypto` tidak dienkripsi menjadi kriptogram yang sama

- Goal metode Kasiski: mencari dua atau lebih kriptogram yang berulang untuk menentukan panjang kunci.

Langkah-langkah metode Kasiski:

1. Temukan semua kriptogram yang berulang di dalam cipherteks (pesan yang panjang biasanya mengandung kriptogram yang berulang).
2. Hitung jarak antara kriptogram yang berulang
3. Hitung semua faktor (pembagi) dari jarak tersebut (faktor pembagi menyatakan panjang kunci yang mungkin).
4. Tentukan irisan dari himpunan faktor pembagi tersebut. Nilai yang muncul di dalam irisan menyatakan angka yang muncul pada semua faktor pembagi dari jarak-jarak tersebut . Nilai tersebut mungkin adalah panjang kunci.

- Contoh:

DYDUXRMHTVDV**NQD**QNW**DYDUXRMH**ARTJGW**NQD**

Kriptogram yang berulang: **DYUDUXRMH** dan **NQD**.

Jarak antara dua buah perulangan **DYUDUXRMH** = 18.

Semua faktor pembagi 18 : {18, 9, 6, 3, 2}

Jarak antara dua buah perulangan **NQD** =20.

Semua faktor pembagi 20 : {20, 10, 5, 4, 2}.

Irisan dari kedua buah himpunan tersebut adalah 2

Panjang kunci kemungkinan besar adalah 2.

- Setelah panjang kunci diketahui, maka langkah berikutnya menentukan huruf-huruf kunci

- Huruf-huruf kunci dapat ditentukan dengan menggunakan *exhaustive key search*

- Jika panjang kunci = *p*, maka jumlah kunci yang harus dicoba sampai menemukan kunci yang benar adalah adalah maksimal 26^p kali.

- Namun lebih sangkil menemukan huruf-huruf kunci dengan menggunakan teknik analisis frekuensi.

Langkah-langkahnya sbb:

1. Misalkan panjang kunci yang sudah berhasil dideduksi adalah *n.* Setiap huruf kelipatan ke-*n* pasti dienkripsi dengan huruf kunci yang sama. Kelompokkan setiap huruf ke-*n* bersama-sama sehingga kriptanalis memiliki *n* buah "pesan", masing-masing dienkripsi dengan substitusi alfabet-tunggal (dalam hal ini *Caesar cipher*).

2. Tiap-tiap pesan dari hasil langkah 1 dapat dipecahkan dengan metode analisis frekuensi.

3. Dari hasil langkah 2 kriptanalis dapat menyusun huruf-huruf kunci. Atau, kriptanalis dapat menerka kata yang membantu untuk memecahkan cipherteks

- Contoh:

1 2 3 4

LJVBQ STNEZ LQMED **LJV**MA MPKAU FAVAT **LJV**DA YYVNF JQLNP **LJV**HK VTRNF
LJVCM LKETA **LJV**HU YJVSF KRFTT WEFUX VHZNP

5 6

Kriptogram yang berulang adalah **LJV**

Jarak **LJV** ke-1 dengan **LJV** ke-2 = 15

Jarak **LJV** ke-2 dengan **LJV** ke-3 = 15

Jarak **LJV** ke-3 dengan **LJV** ke-4 = 15

Jarak **LJV** ke-4 dengan **LJV** ke-5 = 10

Jarak **LJV** ke-5 dengan **LJV** ke-6 = 10

Faktor pembagi 15 = {3, 5, 15}

Faktor pembagi 10 = {2, 5, 10}

Irisan kedua himpunan ini = 5. Jadi, panjang kunci diperkirakan = 5

- Kelompokkan “pesan” setiap kelipatan ke-5, dimulai dari huruf cipherteks pertama, kedua, dan seterusnya. Setiap huruf kelipatan ke-5 pasti dienkripsi dengan huruf kunci yang sama.

```
LJVBQ STNEZ LQMED LJVMA MPKAU FAVAT LJVDA YYVNF JQLNP LJVHK
VTRNF LJVCM LKETA LJVHU YJVSF KRFTT WEFUX VHZNP
```

Kelompok	Pesan	Huruf paling sering muncul
1	LSLLM FLYJL VLLLY KWV	L
2	JTQJP AJYQJ TJKJJ REH	J
3	VNMVK VVVLV RVEVV FFZ	V
4	BEEMA ADNNH NCTHS TUN	N
5	QZDAU TAFPK FMAUF TXP	A

- Dalam Bahasa Inggris, 10 huruf yang yang paling sering muncul adalah E, T, A, O, I, N, S, H, R, dan D,

- Triplet yang paling sering muncul adalah THE. Karena **LJV** paling sering muncul di dalam cipherteks, maka dari 10 huruf tsb semua kemungkinan kata 3-huruf dibentuk dan kata yang yang cocok untuk **LJV** adalah THE.

- Jadi, kita dapat menerka bahwa **LJV** mungkin adalah THE.

- Huruf-huruf kunci lainnya dicoba dengan menerka dan menguji coba.

- Dari sini kita buat tabel yang memetakan huruf plainteks dengan cipherteks dan huruf-huruf kuncinya (ingatlah bahwa setiap nilai numerik dari huruf kunci menyatakan jumlah pergeseran huruf pada *Caesar cipher*):

Kelompok	Huruf plainteks	Huruf cipherteks	Huruf kunci
1	T	L	S (=18)
2	H	J	C (=2)
3	E	V	R (=17)
4	N	N	A (=0)
5	O	A	M (=12)

Jadi, kuncinya adalah SCRAM

- Dengan menggunakan kunci SCRAM cipherteks berhasil didekripsi menjadi:

```
THEBE ARWEN TOVER THEMO UNTAI NYEAH
THEDO GWENT ROUND THEHY DRANT THECA
TINTO THEHI GHEST SPOTH ECOUL DFIND
```

- atau dalam kalimat yang lebih jelas:

```
THE BEAR WENT OVER THE MOUNTAIN YEAH
THE DOG WENT ROUND THE HYDRANT
THE CAT INTO THE HIGHEST SPOT HE COULD FIND
```


Varian *Vigenere Cipher*

Untuk mengatasi serangan dengan metode Kasiski, maka dibuat varian Vigenere Cipher sebagai berikut:

1. *Full Vigènere cipher*
   - Setiap baris di dalam tabel tidak menyatakan pergeseran huruf, tetapi merupakan permutasi huruf-huruf alfabet (lihat contoh table pada halaman berikut).
   - Tabel tersebut harus dirahasiakan.

   - Proses enkripsi dan dekripsi tetap sama seperti Vigenere standard:

Contoh sebuah full Vigènere square

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

2. *Auto-Key Vigènere cipher*

- Jika panjang kunci lebih kecil dari panjang plainteks, maka kunci disambung dengan plainteks tersebut.

- Misalnya,

  Pesan: `negara penghasil minyak mentah di dunia`

  Kunci: `INDO`

  maka kunci tersebut disambung dengan plainteks semula sehingga panjang kunci menjadi sama dengan panjang plainteks:

  Plainteks: `negarapenghasilminyakmentahdidunia`

  Kunci: `INDONEGARAPENGHASILMINYAKMENTAHDID`

  Cipherteks: `VRJOEEVEEGWEFOSMAVJMSZCNDMLQBDBQQD`

3. Running-Key Vigènere cipher

- Kunci adalah string yang sangat panjang yang diambil dari teks bermakna (misalnya naskah proklamasi, naskah Pembukaan UUD 1945, terjemahan ayat di dalam kitab suci, dan lain-lain).

- Misalnya,

  Pesan: `negarapenghasilminyakmentahdidunia`

  Kunci: `KERAKYATANYANGDIPIMPINOLEHHIKMATPE`

- Selanjutnya enkripsi dan dekripsi dilakukan seperti Vigenere cipher biasa.

2. Playfair Cipher

- Termasuk ke dalam *polygram cipher*.

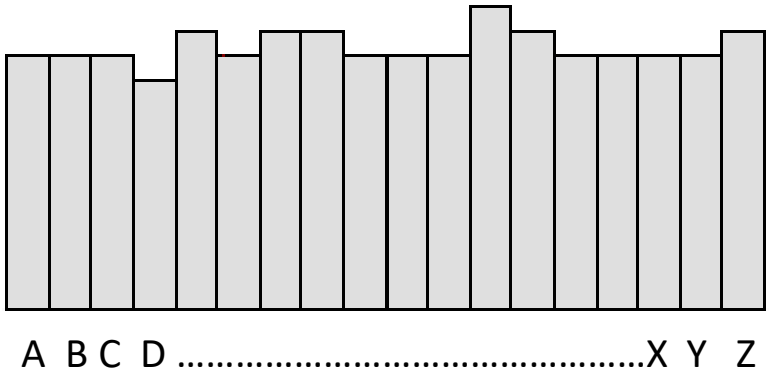
- Ditemukan oleh Sir Charles Wheatstone namun dipromosikan oleh Baron Lyon Playfair pada tahun 1854.

Sir Charles Wheatstone

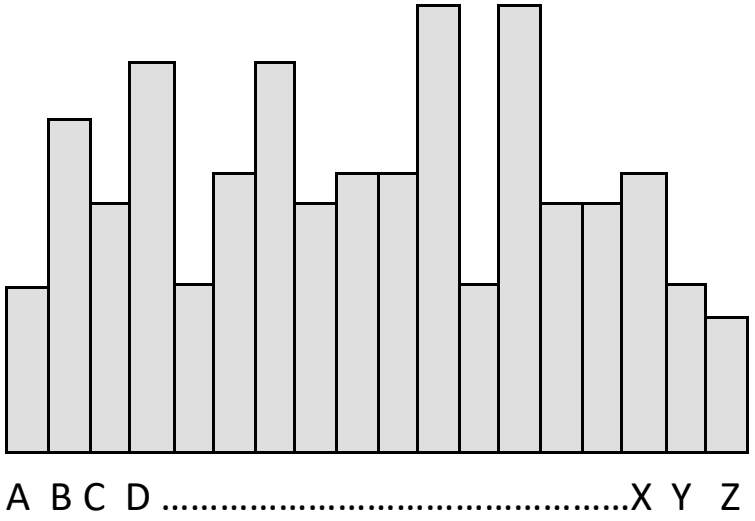
Baron Lyon Playfair

- *Cipher* ini mengenkripsi pasangan huruf (bigram), bukan huruf tunggal seperti pada *cipher* klasik lainnya.

- Tujuannya adalah untuk membuat analisis frekuensi menjadi sangat sulit sebab frekuensi kemunculan huruf-huruf di dalam cipherteks menjadi datar (*flat*).



Flat histogram



Bukan flat histogram

Kunci kriptografinya 25 buah huruf yang disusun di dalam bujursangkat 5x5 dengan menghilangkan huruf J dari abjad.

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

Jumlah kemungkinan kunci:

25!=15.511.210.043.330.985.984.000.000

Kunci dapat dipilih dari sebuah kalimat yang mudah diingat, misalnya:

JALAN GANESHA SEPULUH

Buang huruf yang berulang dan huruf J jika ada:

ALNGESHPU

Lalu tambahkan huruf-huruf yang belum ada (kecuali J):

ALNGESHPUBCDFIKMOQRTVWXYZ

Masukkan ke dalam bujursangkar:

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

Pesan yang akan dienkripsi diatur terlebih dahulu sebagai berikut:

1. Buang semua spasi

2. Ganti huruf `j` (bila ada) dengan `i`

3. Tulis pesan dalam pasangan huruf (*bigram*).

4. Jangan sampai ada pasangan huruf yang sama. Jika ada, sisipkan `x` di tengahnya

5. Jika jumlah huruf ganjil,tambahkan huruf `x` di akhir

Contoh plainteks: `temui ibu nanti malam`

→ Tidak ada huruf `j`, maka langsung tulis pesan dalam pasangan huruf (setelah semua spasi dibuang):

`te mu ii bu na nt im al am`

→ Ada bigram dengan huruf yang berulang (`ii`), sisipkan huruf `x` di tengahnya:

`te mu ix ib un an ti ma la m`

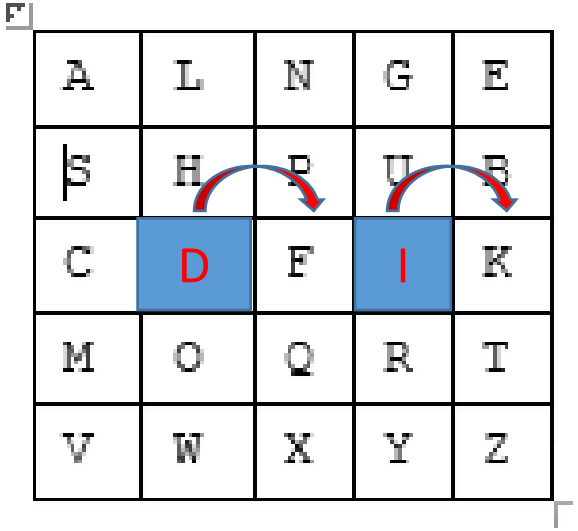
→ Tambahkan huruf `x` jika bigram terakhir hanya satu huruf:

`te mu ix ib un an ti ma la mx`

Algoritma enkripsi:

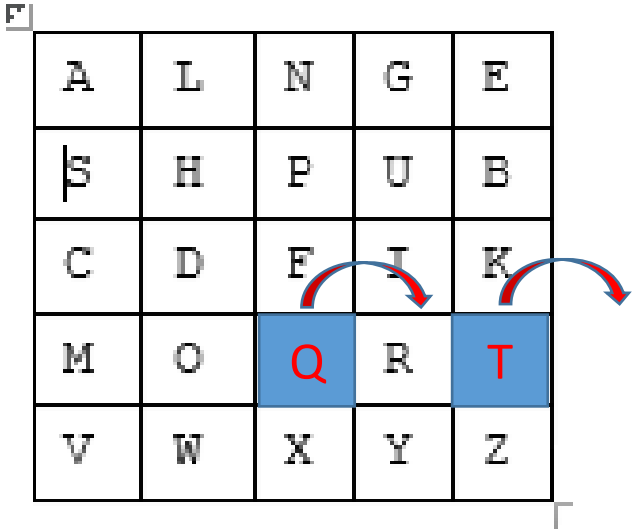
1. Jika dua huruf terdapat pada baris kunci yang sama maka tiap huruf diganti dengan huruf di kanannya (bersifat siklik).

Bigram: di



Cipherteks: FK

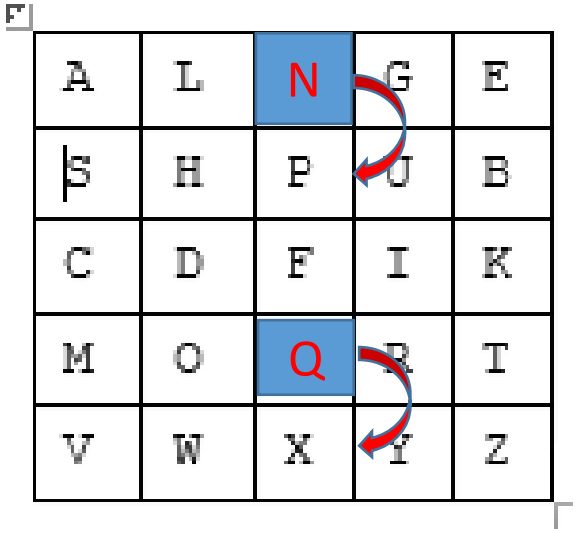
Bigram: qt



Cipherteks: RM

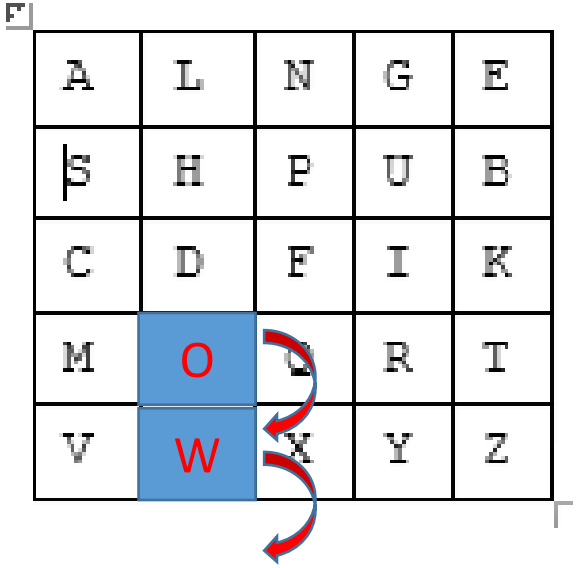
2. Jika dua huruf terdapat pada kolom kunci yang sama maka tiap huruf diganti dengan huruf di bawahnya (bersifat siklik).

Bigram: nq



Cipherteks: PX

Bigram: ow

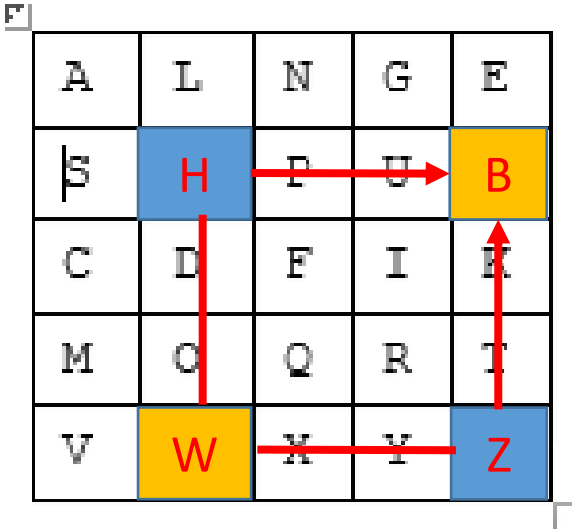


Cipherteks: WL

3. Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka:

- huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua.
- huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari tiga huruf yang digunakan sampai sejauh ini.

Bigram: `hz`



Cipherteks: `BW`

Plainteks: `temui ibu nanti malam`

Bigram: `te mu ix ib un an ti ma la mx`

Kunci:

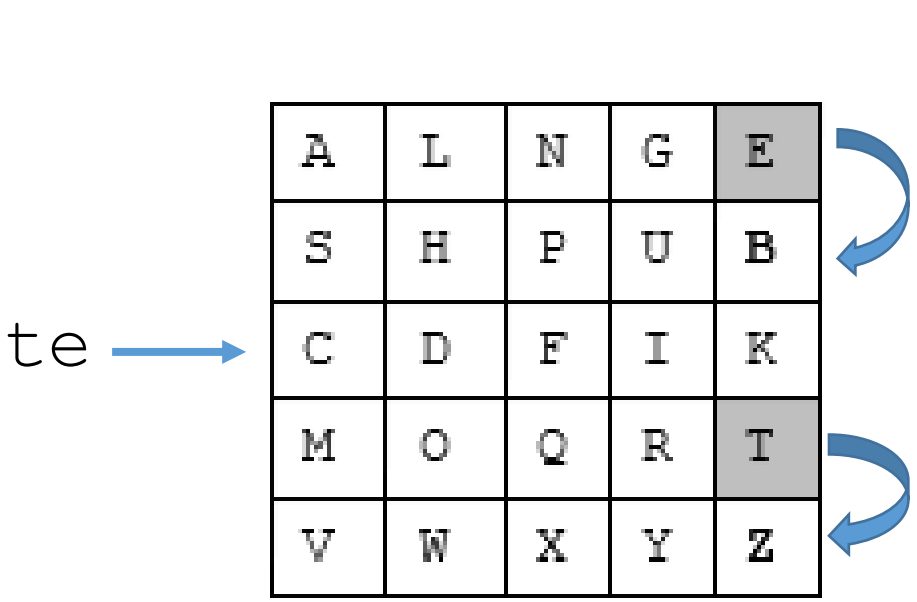
A	L	N	G	E
S	H	P	U	B
C	D	F	I	K
M	O	Q	R	T
V	W	X	Y	Z

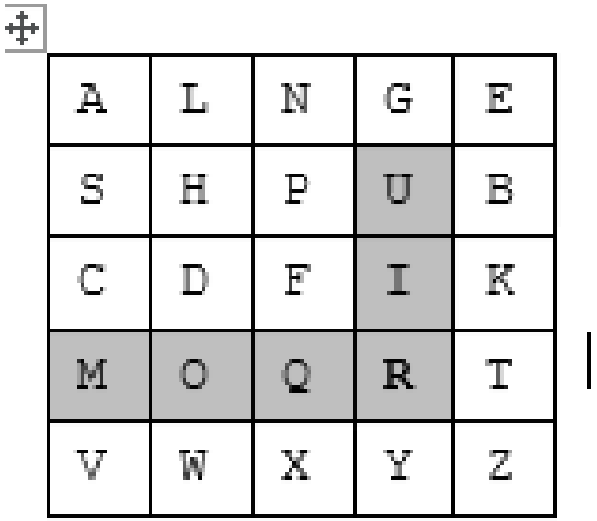
Cipherteks: `ZB RS FY KU PG LG RK VS NL QV`

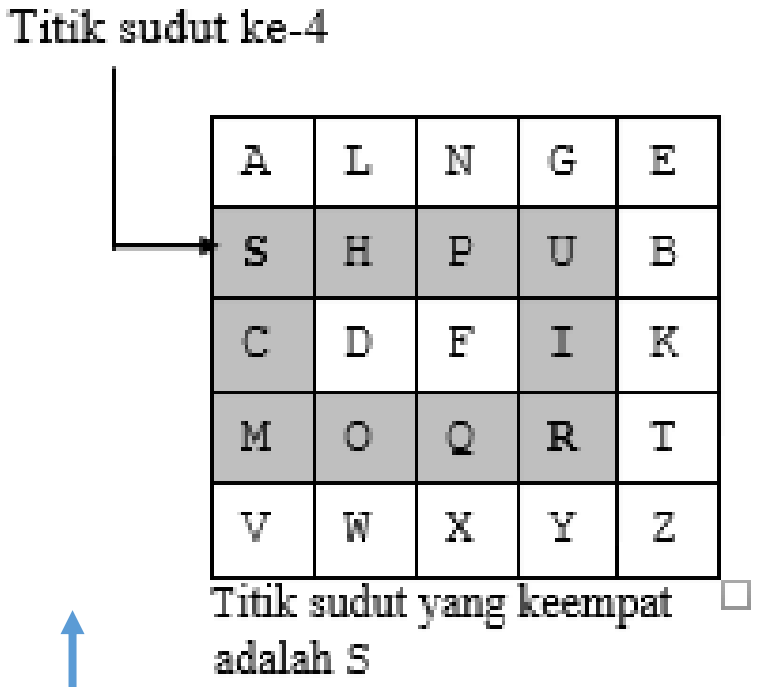
Cara enkripsinya sebagai berkut:

Bigram: te mu ix ib un an ti ma la mx

Cipherteks: ZB RS FY KU PG LG RK VS NL QV



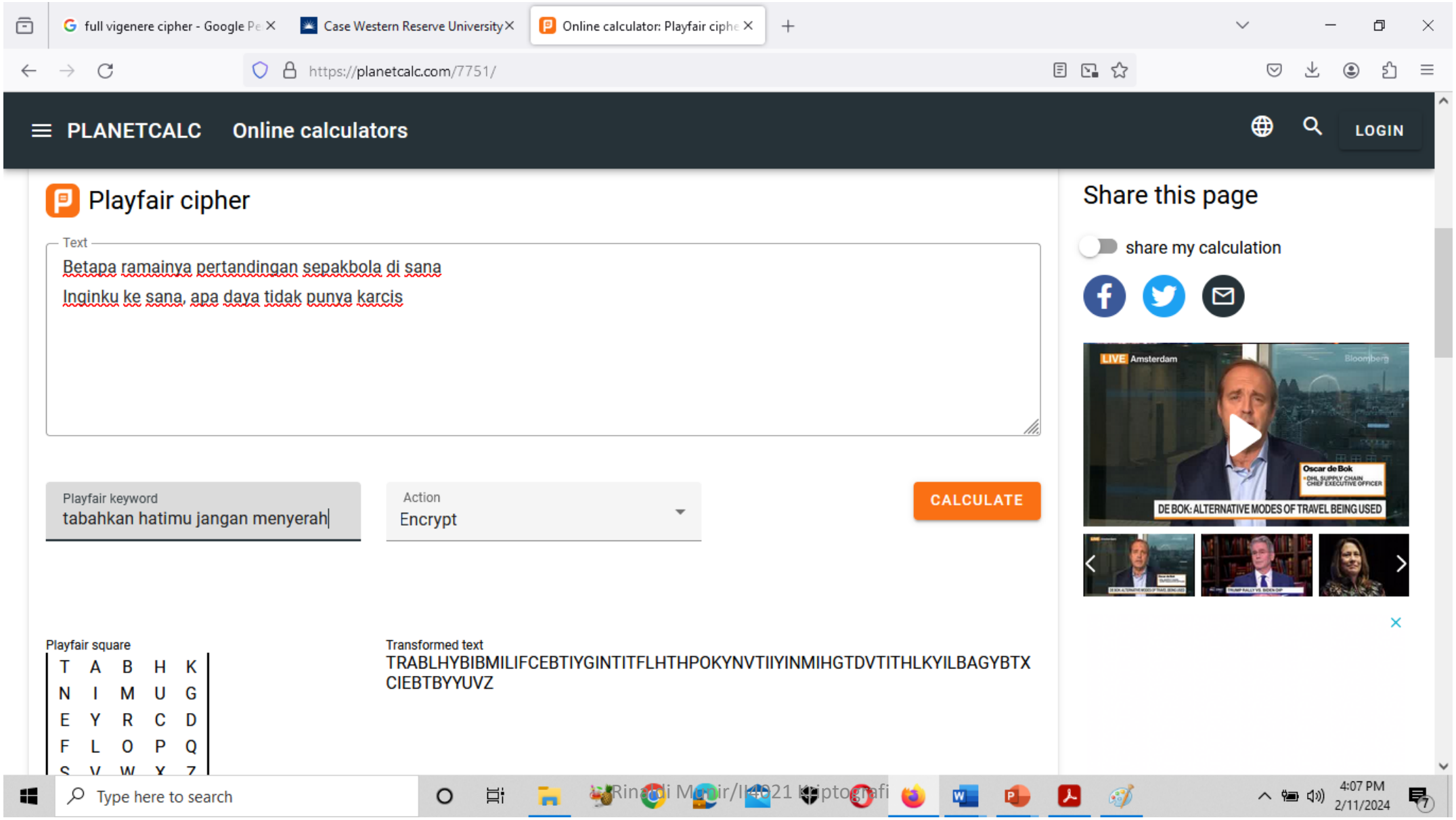




Algoritma dekripsi kebalikan dari algoritma enkripsi. Langkah-langkahnya adalah sebagai berikut:

1. Jika dua huruf terdapat pada baris bujursangkar yang sama maka tiap huruf diganti dengan huruf di kirinya.
2. Jika dua huruf terdapat pada kolom bujursangkar yang sama maka tiap huruf diganti dengan huruf di atasnya.
3. Jika dua huruf tidak pada baris yang sama atau kolom yang sama, maka huruf pertama diganti dengan huruf pada perpotongan baris huruf pertama dengan kolom huruf kedua. Huruf kedua diganti dengan huruf pada titik sudut keempat dari persegi panjang yang dibentuk dari tiga huruf yang digunakan sampai sejauh ini.
4. Buanglah huruf X yang tidak mengandung makna.

Demo Playfair cipher online: https://planetcalc.com/7751/



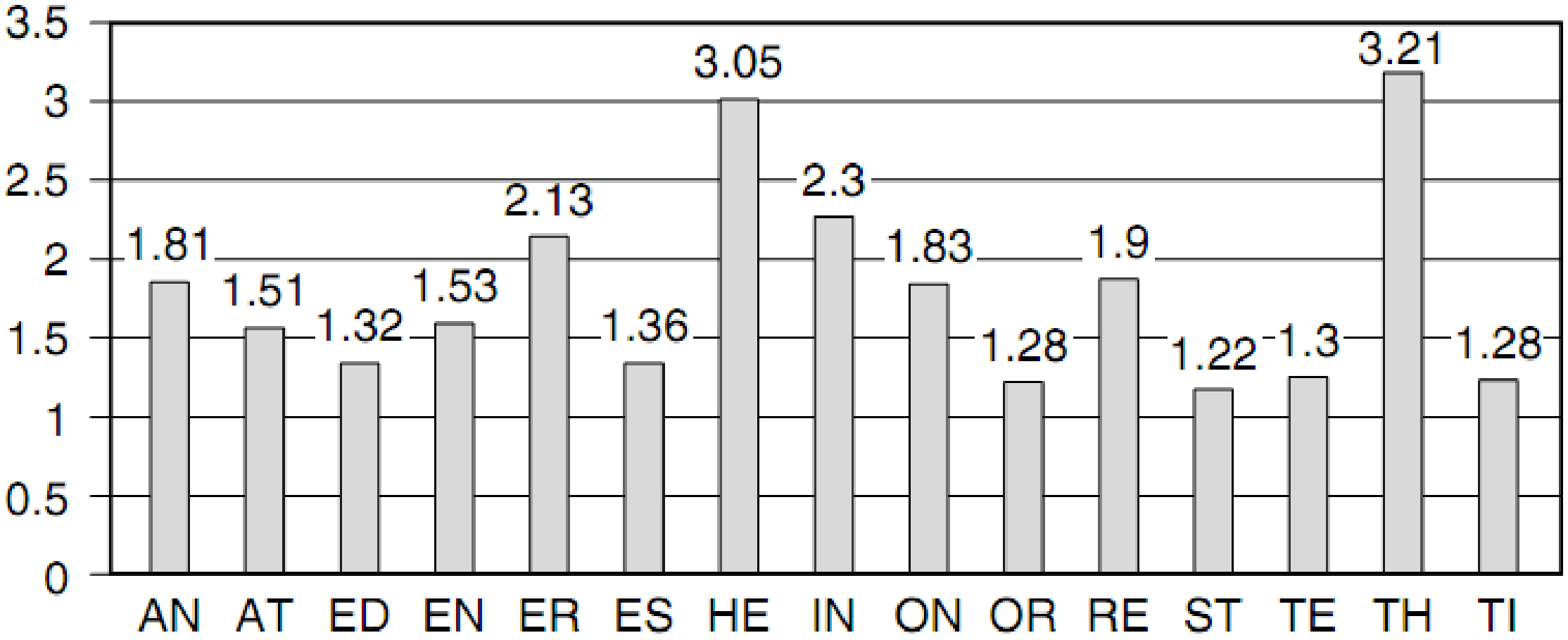
Kriptanalisis Playfair Cipher

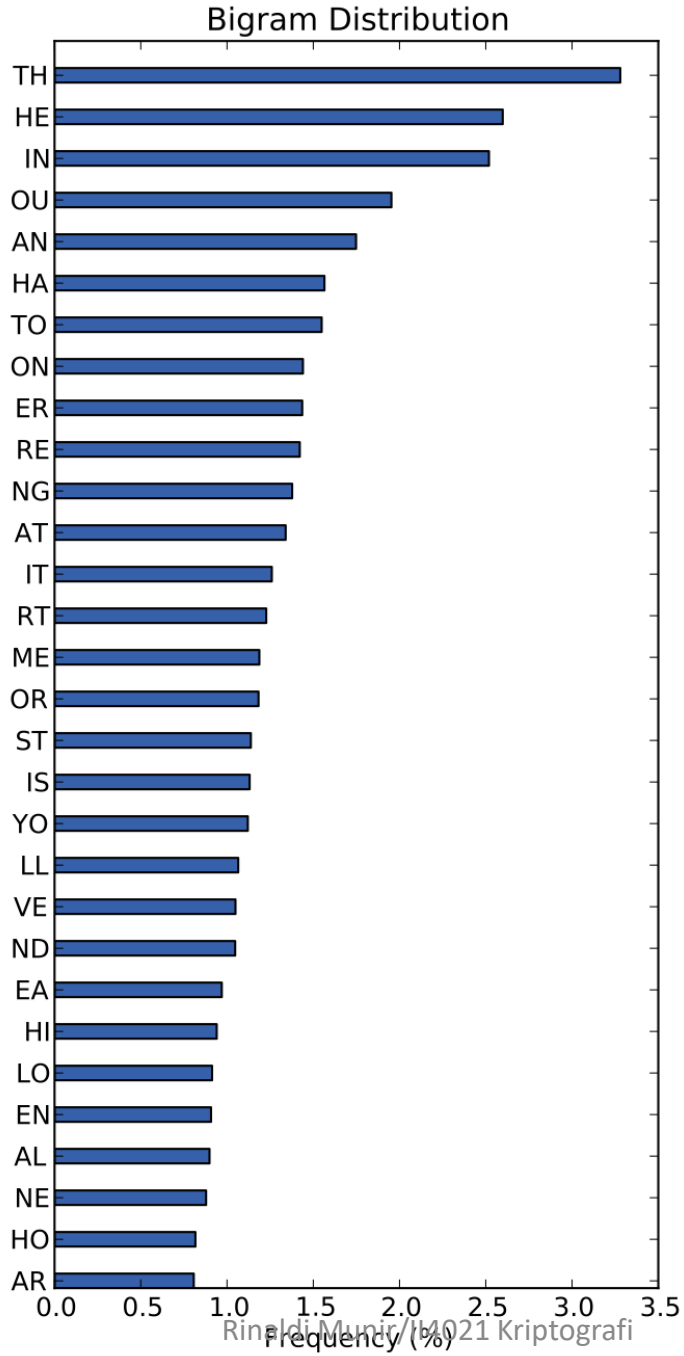
- Karena ada 26 huruf abjad, maka terdapat 26 x 26 = 677 bigram, sehingga identifikasi bigram individual lebih sukar.

- Sayangnya ukuran poligram di dalam *Playfair cipher* tidak cukup besar, hanya dua huruf sehingga *Playfair cipher* tetap tidak aman.

- Meskipun *Playfair cipher* sulit dipecahkan dengan analisis frekuensi relatif huruf-huruf, namun ia dapat dipecahkan dengan analisis frekuensi pasangan huruf.

- Dalam Bahasa Inggris kita bisa mempunyai frekuensi kemunculan pasangan huruf, misalnya pasangan huruf TH dan HE paling sering muncul.


3.5
3
2.5
2
1.5
1
0.5
0
1.81
1.51
1.32
1.53
2.13
1.36
3.05
2.3
1.83
1.28
1.9
1.22
1.3
3.21
1.28
AN
AT
ED
EN
ER
ES
HE
IN
ON
OR
RE
ST
TE
TH
TI


Bigram Distribution
TH
HE
IN
OU
AN
HA
TO
ON
ER
RE
NG
AT
IT
RT
ME
OR
ST
IS
YO
LL
VE
ND
EA
HI
LO
EN
AL
NE
HO
AR
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
Frequency (%)

- Dengan menggunakan tabel frekuensi kemunculan pasangan huruf di dalam Bahasa Inggris dan cipherteks yang cukup banyak, *Playfair cipher* dapat dipecahkan.

- Kelemahan lainnya, bigram dan kebalikannya (misal AB dan BA) akan didekripsi menjadi pola huruf plainteks yang sama (misal RE dan ER). Di dalam bahasa Inggris terdapat banyak kata yang mengandung bigram terbalik seperti REceivER dan DEpartED.

Bersambung