KRIPTOGRAFI – ANALISIS VIGENÈRE CIPHER

Laporan ini disusun untuk memenuhi Tugas Besar 1 pada mata kuliah Kriptografi dengan dosen pengampu

Bapak Kodrat Mahatma S.T.,M.Kom.

LINTANG SARI IRVANIATI
(20123012)

C1.23 - S1 INFORMATIKA

UNIVERSITAS TEKNOLOGI DIGITAL

NOVEMBER 2025

ANALISIS VIGENÈRE CIPHER

A. Vigenère Cipher

Vigenère Cipher adalah algoritma kriptografi klasik yang termasuk ke dalam polyalphabetic substitution cipher. Setiap huruf pada plaintext digeser dengan nilai yang berbeda-beda, bergantung pada huruf kunci (key) yang digunakan. Jika key lebih pendek dari plaintext, maka key akan diulang sesuai panjang teks.

Setiap huruf dikonversi menjadi angka (A = 0, B = 1, …, Z = 25), kemudian dilakukan pergeseran:

- **Rumus Enkripsi:**
  - Ci = (Pi+Ki) mod 26

- **Keterangan:**
  - Pi = huruf plaintext
  - Ki = huruf key
  - Ci = huruf ciphertext

Karena menggunakan lebih dari satu pola pergeseran, hasil enkripsinya lebih sulit dipecahkan hanya dengan analisis frekuensi seperti pada Caesar Cipher.

B. Data Percobaan

Percobaan dilakukan dengan data:

Jenis Data	Keterangan
Plaintext	Hello World
Key	J

Key **J** memiliki nilai 9 (A=0 → J=9).
Pada proses enkripsi, huruf “J” diulang sepanjang plaintext (spasi diabaikan).

1. Implementasi di Google Colab

a. Kode Program

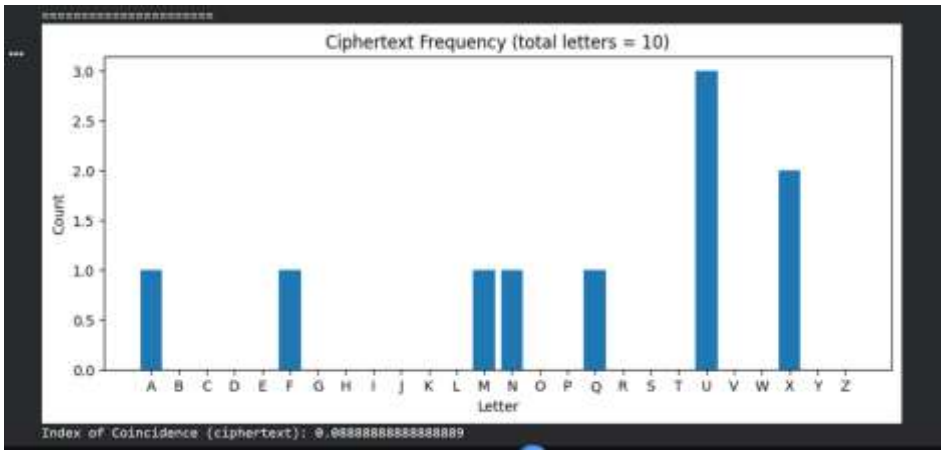
```
[1]
from datetime import datetime
import math
from collections import Counter
import matplotlib.pyplot as plt

%matplotlib inline
```

```
[2]
def vigenere_encrypt(plaintext, key):
    ciphertext = ""
    key = ''.join([c for c in key.upper() if c.isalpha()])
    if not key:
        raise ValueError("Key harus berisi huruf A-Z")
    plaintext_u = plaintext.upper()
    ki = 0
    for ch in plaintext_u:
        if ch.isalpha():
            shift = ord(key[ki % len(key)]) - ord('A')
            ciphertext += chr((ord(ch) - 65 + shift) % 26 + 65)
            ki += 1
        else:
            ciphertext += ch
    return ciphertext
```

```
[3]
def vigenere_decrypt(ciphertext, key):
    plaintext = ""
    key = ''.join([c for c in key.upper() if c.isalpha()])
    ki = 0
    for ch in ciphertext.upper():
        if ch.isalpha():
            shift = ord(key[ki % len(key)]) - ord('A')
            plaintext += chr((ord(ch) - 65 - shift) % 26 + 65)
            ki += 1
        else:
            plaintext += ch
    return plaintext

def print_result(mode, text, key, result, notes=""):
    print("==== Cipher Result ====")
    print("Date:", datetime.now().strftime("%Y-%m-%d %H:%M:%S"))
    print("Mode:", mode)
    if mode == "ENCRYPT":
        print("Plaintext:", text)
        print("Key:", key)
        print("Ciphertext:", result)
```

```
[8]
✓ 0d
plaintext = "Hello World"
key = "J"
ciphertext = vigenere_encrypt(plaintext, key)
print_result("ENCRYPT", plaintext, key, ciphertext, notes="Contoh run")

plot_frequency(ciphertext, title="Ciphertext Frequency")
print("Index of Coincidence (ciphertext):", index_of_coincidence(ciphertext))
show_kasiski(ciphertext)

==== Cipher Result ====
Date: 2025-11-07 08:42:31
Mode: ENCRYPT
Plaintext: Hello World
Key: J
Ciphertext: QNUUX FXAUM
Notes: Contoh run
======================
```

b. Grafik Ciphertext



Hasil yang diperoleh pada Google Colab:

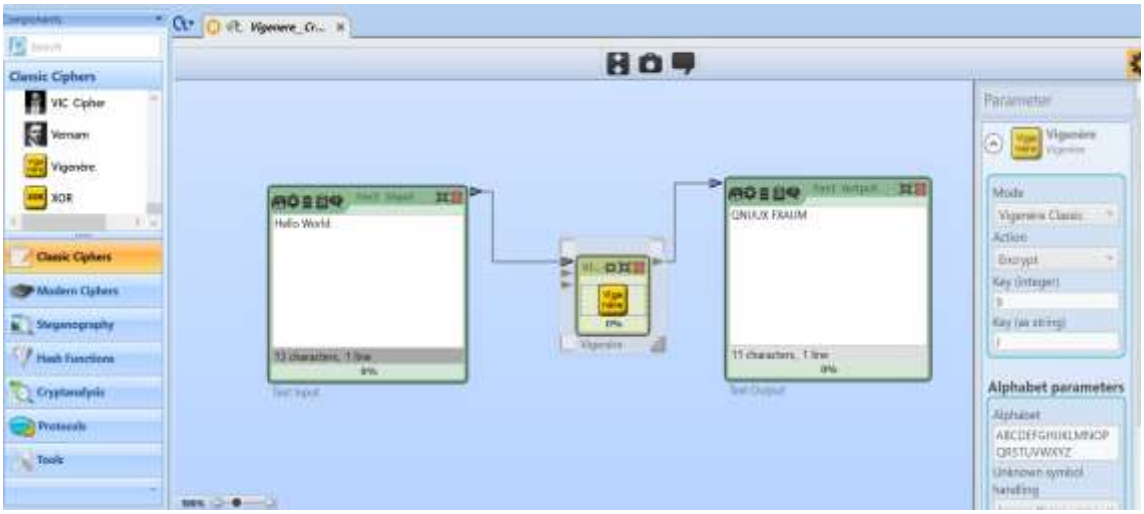
- **Plaintext** : HELLO WORLD
- **Key** : J
- **Ciphertext** : QNUUX FXAUM
- Pada percobaan ini, setiap huruf plaintext digeser mengikuti nilai alfabet dari huruf kunci. Karena key yang digunakan hanya satu huruf (**J** = pergeseran 9 langkah dari A), maka pergeseran akan konsisten di setiap karakter, sehingga menghasilkan ciphertext di atas.

Analisis frekuensi dilakukan untuk melihat sebaran huruf pada ciphertext. Hasil grafik menunjukkan:

1. Huruf U muncul 3 kali, sementara huruf lain hanya muncul 1 kali. Ini menandakan distribusi huruf lebih merata dibanding plaintext.
2. Pola frekuensi tidak menyerupai pola frekuensi bahasa Inggris. Misalnya pada teks asli, huruf seperti L dan O lebih sering muncul. Pola itu tidak lagi terlihat di ciphertext.
3. Index of Coincidence (IoC) lebih rendah dibanding plaintext**.**
   Pada teks normal, IoC mendekati pola bahasa alami. Penurunan IoC mengindikasikan kalau algoritma ini berhasil mengacak pola sehingga sulit ditebak dengan analisis statistik sederhana.

2. **Validasi pada CrypTool**

   Percobaan enkripsi yang sama dilakukan menggunakan aplikasi CrypTool untuk memastikan bahwa proses enkripsi di Google Colab tidak menghasilkan kesalahan pada perhitungan algoritma.



Hasil yang diperoleh pada CrypTool:

- **Plaintext** : HELLO WORLD
- **Key** : J
- **Ciphertext** : QNUUX FXAUM
- Proses enkripsi menggunakan CrypTool menghasilkan ciphertext QNUUX FXAUM, yang identik dengan hasil dari implementasi di Google Colab**.** Kesamaan ini menunjukkan bahwa perhitungan enkripsi pada kode yang dijalankan di Google Colab benar dan sesuai dengan teori Vigenère Cipher.

C. Kesimpulan

Dari percobaan yang telah dilakukan, dapat disimpulkan bahwa:

1. Vigenère Cipher melakukan enkripsi dengan pergeseran yang bervariasi berdasarkan huruf kunci, sehingga tidak mudah ditebak melalui pola sederhana.
2. Implementasi enkripsi menggunakan Google Colab menghasilkan ciphertext **QNUUX FXAUM**, dan hasil tersebut tervalidasi dengan CrypTool 2.
3. Analisis frekuensi ciphertext menunjukkan distribusi huruf yang lebih acak dibanding plaintext, sesuai karakter polyalphabetic cipher.
4. Secara keseluruhan, percobaan dan validasi membuktikan bahwa algoritma berjalan dengan benar.

❖ Kelebihan Vigenère Cipher

- Lebih aman dibanding Caesar Cipher karena menggunakan banyak pergeseran.
- Susah dipecahkan dengan analisis frekuensi biasa.
- Mudah diterapkan secara manual maupun dengan program komputer.
- Key dapat fleksibel (lebih panjang → keamanan meningkat).

❖ Kekurangan Vigenère Cipher

- Jika panjang key terlalu pendek, pola masih dapat dianalisis dan diserang (misalnya dengan Kasiski atau Friedman Test).
- Rentan jika key berhasil ditebak atau diketahui.
- Tetap dianggap kurang aman dalam konteks kriptografi modern karena masih berbasis substitusi huruf.