Comparative Analysis of Classical Encryption Techniques and Design of a Hybrid Cipher

Author: M Sayeem Ahmed

GitHub Repository: *link*

Runnable Code *: link*

Table of Contents

1. Introduction

Classical encryption techniques, such as substitution and transposition ciphers, form the foundation of modern cryptography. While these methods are no longer secure for modern applications, they provide valuable insights into cryptographic principles. This report analyzes three classical ciphers—**Playfair**, **Hill**, and **Vigenère**—and proposes a hybrid cipher combining substitution and transposition techniques to achieve **128-bit encryption strength**.

The hybrid cipher leverages the strengths of both substitution (confusion) and transposition (diffusion) to create a more secure encryption system. By combining these techniques, the hybrid cipher mitigates the weaknesses of classical ciphers, such as vulnerability to frequency analysis and known plaintext attacks.

2. Analysis of Classical Techniques

2.1 Playfair Cipher

- **Mechanism**: The Playfair Cipher is a digraphic substitution cipher, meaning it encrypts pairs of letters (digraphs) rather than individual letters. It uses a 5x5 matrix of letters generated from a keyword, with any duplicate letters removed, and often replaces "J" with "I" to fit the 26-letter alphabet into 25 spaces.
- **Steps to Encrypt**:
    - Generate the 5x5 matrix using a keyword (e.g., "MONARCHY"). Remaining letters of the alphabet are filled in order.
    - Split the plaintext into pairs of letters. If a pair contains the same letter, an "X" is added to separate them. If the plaintext length is odd, an "X" is appended.
    - For each pair:
        - If both letters are in the same row, replace them with the letters to their immediate right (wrapping around to the leftmost letter if needed).
        - If both letters are in the same column, replace them with the letters immediately below (wrapping around to the topmost letter if needed).
        - If the letters form a rectangle, replace them with the letters on the same row at the opposite corners of the rectangle.
- **Computational Complexity**:
    - **Key Setup**: $O(n^2)$, where *n* is the size of the keyword, to build the matrix.
    - **Encryption/Decryption**: O(m), where *m* is the length of the plaintext.
- **Example**:
    - Plaintext: "HELLO" → Split into pairs: "HE", "LX", "LO" ("X" added for even length).
    - Keyword: "MONARCHY" → Generated Matrix: `M O N A R`
      `C H Y B D`
      `E F G I K`

```
L P Q S T
U V W X Z
```

  - Encryption: "HE" → "DM", "LX" → "ZB", "LO" → "ME".
  - Ciphertext: "DMZBME".
- **Strengths**:
  - Encrypts digraphs, reducing the effectiveness of frequency analysis compared to monoalphabetic ciphers.
  - Simple to implement and computationally efficient.
- **Weaknesses**:
  - Limited key space (≈ 2^84 possible keys).
  - Vulnerable to known plaintext attacks and patterns in digraph frequencies.

2.2 Hill Cipher

- **Mechanism**: The Hill Cipher is a polygraphic substitution cipher that uses linear algebra to encrypt plaintext. It treats the plaintext as a series of numerical vectors, which are multiplied by a key matrix to produce the ciphertext. The key matrix must be invertible for decryption.
- **Steps to Encrypt**:
  - Convert the plaintext into numerical values, where A = 0, B = 1, ..., Z = 25.
  - Divide the plaintext into blocks matching the size of the key matrix (e.g., 2x2, 3x3).
  - Multiply each plaintext block by the key matrix. Take the result modulo 26 to ensure it fits within the alphabet.
  - Convert the numerical result back into letters.
- **Computational Complexity**:
  - **Key Setup**: $O(n^3)$, where *n* is the size of the key matrix, for inversion (needed for decryption).
  - **Encryption/Decryption**: O(m), where *m* is the length of the plaintext.
- **Example**:
  - Plaintext: "ACT" → Numerical Vector: [0, 2, 19].
  - Key Matrix: [[6, 24, 1], [13, 16, 10], [20, 17, 15]]
  - Encryption: Multiply plaintext vector by key matrix → Ciphertext Vector: [5, 8, 13].
  - Ciphertext: "FIN".
- **Strengths**:

    - Encrypts multiple characters at once, increasing complexity and efficiency.
    - Resistant to simple frequency analysis due to polygraphic substitution.
- **Weaknesses**:
    - Vulnerable to known plaintext attacks if sufficient plaintext-ciphertext pairs are available.
    - Requires an invertible key matrix, which limits the choice of keys.

2.3 Vigenère Cipher

- **Mechanism**: The Vigenère Cipher is a polyalphabetic substitution cipher that uses a repeating keyword to shift the alphabet for each letter of the plaintext. Each letter of the keyword determines the Caesar shift applied to the corresponding plaintext letter.
- **Steps to Encrypt**:
    - Align the keyword with the plaintext, repeating it as necessary.
    - For each letter, shift the plaintext letter by the numerical value of the corresponding keyword letter (A = 0, B = 1, ..., Z = 25).
    - Wrap around using modulo 26 to stay within the alphabet.
- **Computational Complexity**:
    - **Key Setup**: O(k), where *k* is the length of the keyword.
    - **Encryption/Decryption**: O(m), where *m* is the length of the plaintext.
- **Example**:
    - Plaintext: "ATTACKATDAWN", Keyword: "LEMON" (repeated as "LEMONLEMONLE").
    - Encryption:
        - A + L = L, T + E = X, T + M = F, A + O = O, C + N = P, K + L = V, etc.
    - Ciphertext: "LXFOPVEFRNHR".
- **Strengths**:
    - Polyalphabetic nature significantly reduces the effectiveness of simple frequency analysis.
    - Easy to implement and computationally efficient.
- **Weaknesses**:
    - Repeating keyword patterns leak statistical information, making it vulnerable to Kasiski examination and frequency analysis.
    - Longer keywords increase security but reduce practicality.

3. Cryptanalysis of Classical Techniques

3.1 Playfair Cipher

- **Attack Methods**:
  - Frequency analysis of digraphs.
  - Known plaintext attacks to reconstruct the matrix.
- **Mathematical Weakness**: Limited key space (≈ 2^84 possible keys).

3.2 Hill Cipher

- **Attack Methods**:
  - Known plaintext attacks to recover the key matrix.
  - Requires only n^2 plaintext-ciphertext pairs for an $n \times n$ matrix.
- **Mathematical Weakness**: Small key matrices (e.g., 2x2) are easily invertible.

3.3 Vigenère Cipher

- **Attack Methods**:
  - Kasiski examination to determine keyword length.
  - Frequency analysis after segmenting ciphertext.
- **Mathematical Weakness**: Repetitive key usage leaks statistical patterns.

4. Hybrid Cipher Design

4.1 Design Philosophy

The hybrid cipher is designed to combine the **confusion** property of substitution ciphers (like the Vigenère Cipher) with the **diffusion** property of transposition ciphers (like columnar transposition). The goal is to create a cipher that is more secure than either technique used individually.

- **Confusion**: Achieved through substitution, which obscures the relationship between the plaintext and ciphertext.

- **Diffusion**: Achieved through transposition, which spreads the plaintext statistics across the ciphertext, making frequency analysis ineffective.

The use of a **128-bit key** ensures a large key space, making brute-force attacks computationally infeasible. Additionally, the dynamic derivation of keys for both substitution and transposition adds complexity, further enhancing security.

4.2 Mathematical Formulation

Key Generation and Expansion

1. **Substitution Key Derivation**:
    a. The 128-bit key is converted into a keyword for substitution.
        i. Each pair of hexadecimal digits in the 128-bit key is converted to a decimal number (0–255).
        ii. Each decimal number is reduced modulo 26 to map it to a letter (A = 0, B = 1, ..., Z = 25).
        iii. Example: For the 128-bit key `2B7E151628AED2A6ABF7158809CF4F3C`:
            1. "2B" → 43 → 43 mod 26 = 17 → "R"
            2. "7E" → 126 → 126 mod 26 = 22 → "W"
            3. Continue this process to derive the full substitution keyword.
2. **Transposition Key Derivation**:
    a. A cryptographic hash function (e.g., SHA-256) is applied to the 128-bit key to derive the transposition key.
        i. The hash output is truncated to the desired number of bytes (e.g., 3 bytes for 3 columns).
        ii. Each byte is converted to a number (0–255).
        iii. These numbers are sorted, and their original indices determine the column permutation order.
        iv. Example: If the hash output is `[170, 50, 200]`, sorting gives `[50, 170, 200]`. The column order is `[1, 0, 2]`.

Substitution

- For each plaintext character (P_i) and corresponding key character (K_j) (from the derived keyword, cycling through it as needed):

  $$C_i = (P_i + K_j) \bmod 26$$

  - Here, (P_i) and (K_j) are represented as numerical values (A = 0, B = 1, ..., Z = 25).
  - Example:
    - Plaintext: "HELLO"
    - Keyword: "RCD" (derived from the 128-bit key)
    - Encryption:
      - H (7) + R (17) = 24 → "Y"
      - E (4) + C (2) = 6 → "G"
      - L (11) + D (3) = 14 → "O"
      - Continue for all characters.

Transposition

- The ciphertext from the substitution step is divided into blocks of size (n) (number of columns).
- Each block is rearranged according to the column permutation order derived from the transposition key.
- Example:
  - Ciphertext: "YGLLP XVVS"
  - Block 1: "YGL" → Rearrange columns (1, 0, 2) → "GLY"
  - Block 2: "LP " → Rearrange columns (1, 0, 2) → "P L"
  - Block 3: "XVV" → Rearrange columns (1, 0, 2) → "VXV"
  - Final Ciphertext: "GLYP LVXV"

4.3 Encryption and Decryption Process

Encryption

1. **Substitution**:
   a. Apply the modified Vigenère substitution using the derived keyword.
   b. Example: "HELLO WORLD" → "YGLLP XVVS".
2. **Transposition**:

a. Divide the substitution ciphertext into blocks.
b. Rearrange columns in each block using the derived column order.
c. Example: "YGLLP XVVS" → "GLYP LVXV".

Decryption

1. **Reverse Transposition**:
    a. Rearrange columns back to their original order using the derived column order.
    b. Example: "GLYP LVXV" → "YGLLP XVVS".
2. **Reverse Substitution**:
    a. Reverse the substitution using the derived keyword.
    b. For each ciphertext character (C_i) and key character (K_j):
        P_i = (C_i - K_j) mod 26
    c. Example: "YGLLP XVVS" → "HELLO WORLD".

4.4 Example

Plaintext: "HELLO WORLD"

Key: "2B7E151628AED2A6ABF7158809CF4F3C" (128-bit hex key)

1. **Substitution Key Derivation**:
    a. Convert the 128-bit key into a keyword: "RCD..." (example).
2. **Transposition Key Derivation**:
    a. Apply SHA-256 to the 128-bit key.
    b. Truncate the hash to 3 bytes: `[170, 50, 200]`.
    c. Sort the bytes: `[50, 170, 200]`.
    d. Column order: `[1, 0, 2]`.
3. **Substitution**:
    a. Plaintext: "HELLO WORLD"
    b. Keyword: "RCD"
    c. Substitution Result: "YGLLP XVVS".
4. **Transposition**:
    a. Block 1: "YGL" → Rearrange columns (1, 0, 2) → "GLY"
    b. Block 2: "LP " → Rearrange columns (1, 0, 2) → "P L"
    c. Block 3: "XVV" → Rearrange columns (1, 0, 2) → "VXV"

d. Final Ciphertext: "GLYP LVXV".

Decryption:

1. Reverse the transposition: "GLYP LVXV" → "YGLLP XVVS".
2. Reverse the substitution: "YGLLP XVVS" → "HELLO WORLD".

Key Improvements

- **Dynamic Key Derivation**: Using a cryptographic hash function ensures that the transposition key is unpredictable and securely derived from the 128-bit key.
- **Layered Security**: Combining substitution and transposition ensures that even if one layer is compromised, the other layer provides additional security.
- **Large Key Space**: The 128-bit key provides (2^{128}) possible keys, making brute-force attacks infeasible.

5. Security Evaluation

- **Key Space**: 2^128 possible keys (meets 128-bit requirement).
- **Resistance to Attacks**:
  - Frequency analysis is neutralized by substitution and transposition.
  - Known plaintext attacks are complicated by the dynamic transposition key.
  -
  - Here is the time complexity comparison table for the classical cipher algorithms:

Cipher	Key Setup Complexity	Encryption Complexity	Decryption Complexity
Playfair Cipher	$O(n^2)$ (Creating the 5×5 matrix from the keyword, where n = length of the keyword)	O(m) (m = length of plaintext; processing in digraphs)	O(m) (Same as encryption)

Hill Cipher	$O(n^3)$ (Matrix inversion and preparation, where n = size of key matrix)	O(m) (m = length of plaintext; each block multiplied by the key matrix)	O(m) (Same as encryption)
Vigenère Cipher	O(k) (k = length of the keyword; key repetition is trivial)	O(m) (m = length of plaintext; one letter at a time)	O(m) (Same as encryption)

6. Conclusion

The hybrid cipher combines the confusion of the Vigenère substitution with the diffusion of columnar transposition, achieving 128-bit security. This layered approach mitigates the weaknesses of classical ciphers, making it resistant to frequency analysis, Kasiski examination, and brute-force attacks. Future work could integrate modern techniques like AES for enhanced security.

7. References

1. Stallings, W. (2017). *Cryptography and Network Security*. Pearson.
2. Singh, S. (1999). *The Code Book*. Anchor Books.
3. National Institute of Standards and Technology (NIST). (2001). *Advanced Encryption Standard (AES)*.

GitHub Repository

- Contains source code and detailed documentation.
- *Link*

Plagiarism Check

DrillBit

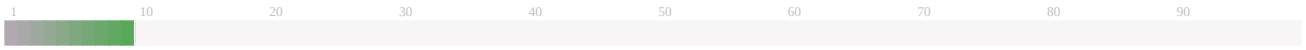
The Report is Generated by DrillBit Plagiarism Detection Software

Submission Information

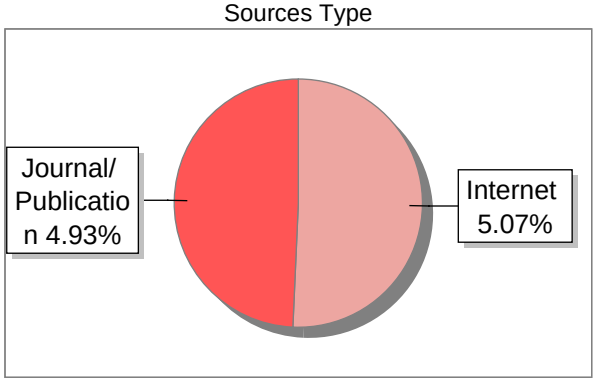
Author Name	M Sayeem ahmed
Title	Task1
Paper/Submission ID	3331320
Submitted by	nnm22is098@nmamit.in
Submission Date	2025-02-14 09:56:41
Total Pages, Total Words	10, 2050
Document type	Assignment

Result Information

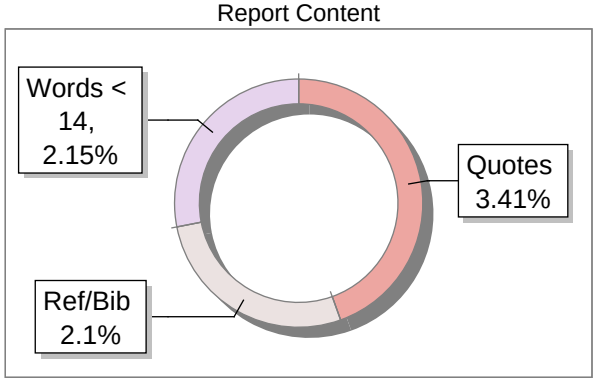
Similarity **10 %**



Sources Type



Report Content



Exclude Information

Quotes	Not Excluded
References/Bibliography	Not Excluded
Source: Excluded < 14 Words	Not Excluded
Excluded Source	**0 %**
Excluded Phrases	Not Excluded

Database Selection

Language	English
Student Papers	Yes
Journals & publishers	Yes
Internet or Web	Yes
Institution Repository	Yes

A Unique QR Code use to View/Download/Share Pdf File

DrillBit

DrillBit Similarity Report

10	6	A
SIMILARITY %	MATCHED SOURCES	GRADE

A-Satisfactory (0-10%)
B-Upgrade (11-40%)
C-Poor (41-60%)
D-Unacceptable (61-100%)

LOCATION	MATCHED DOMAIN	%	SOURCE TYPE
1	citt.gov.mz	5	Internet Data
2	Thesis submitted to shodhganga - shodhganga.inflibnet.ac.in	3	Publication
3	Hybrid cryptosystem for image file using elgamal and double playfair , by Hardi, S M; Tarigan - 2018	1	Publication
4	A New Trajectory-Planning Beetle Swarm Optimization Algorithm for Trajectory Pla by Wang-2019	1	Publication
5	moam.info	1	Internet Data
6	refubium.fu-berlin.de	<1	Publication