

**AHSANULLAH UNIVERSITY OF SCIENCE AND TECHNOLOGY** Department of Computer Science and Engineering

Program: Bachelor of Science in Computer Science and Engineering

Course Code: CSE 4174

Course Title: Cyber Security Lab

Academic Semester: Fall 2023

Assignment Topic: Substitution & Transposition Ciphers

Submitted on: **21/04/2024**

Submitted by

Name: **Nafisa Tasnim Neha**

Student ID: **20200204020**

Lab Section: **A2**

**Question :**

1. Write down how substitution and transposition ciphers differ from each other and detail the scenarios where one is more suitable than the other.

**Answer**: In cryptography, substitution and transposition ciphers are two methods for encrypting plaintext, but they function fundamentally in different ways.

|  |  |
| --- | --- |
| **Substitution Ciphers** | **Transposition Ciphers** |
| 1. A substitution technique is one in which the letters of plaintext are replaced by other letters or by numbers or symbols. | 1. In a transposition cipher, the characters in the plaintext are rearranged according to a specific system or key, but the characters themselves are not replaced. |
| 2. Substitution ciphers can involve complex mappings | 2. Transposition ciphers are generally simpler to implement. |
| 3. The time complexity of encryption and decryption is less. | 3. The time complexity of encryption and decryption is high. |
| 4. It’s a simple process. | 4. Complex than the substitution technique. |
| 5. Coding is simple to decipher. | 5. It is hard to decipher the code. |
| 6. Changes its identity while maintaining its status. | 6. Changes its place while maintaining its identity. |
| 7. Substitution provides encrypted one-to-one communication. | 7. Transposition allows for one-to-many encryption. |
| 8. **Example:** Caesar Cipher, Monoalphabetic Cipher, Playfair Cipher, Hill Cipher, Polyalphabetic Cipher, One-Time Pad etc. | 8. **Example**: Rail Fence Cipher, Row Transposition Cipher, Columnar Transposition Cipher etc. |

**Scenarios:**

**Message Length:** Substitution might be quicker for short messages, while transposition can handle medium lengths better.

**Security Level:** Substitution offers low-to-medium security, while transposition offers medium security.

**Content Sensitivity:** Substitution cipher might be useful for hiding specific keywords within a message for casual privacy but doesn't offer strong protection for the overall message content. On the other hand, by scrambling the message order, transposition adds an extra layer of protection, making it harder to understand.

**Hide individual Keywords:** For informal privacy, mask individual words or phrases within a message (Remember that this is not complete message encryption.) in substitution cipher.

**Obscuring Word Meaning:** Substitution ciphers are ideal for obscuring particular word meanings in messages. They replace letters, making it challenging to decipher based on typical patterns.

**Educational Purposes:** Demonstrating basic encryption and decryption concepts due to their simplicity in substitution cipher.

**Complex Key Handling**: Transposition ciphers provide enhanced security against cryptanalysis for encryption systems capable of handling complex keys or patterns. Using complex key layouts can enhance security by causing confusion and diffusion.

**Ease of Use:** Substitution ciphers are generally simpler to implement compared to transposition ciphers, especially for complex transposition methods.

**Modern Cryptography:** Both substitution and transposition ciphers are considered weak in modern cryptography. For high-security needs, more sophisticated algorithms and keys are use

2. Devise a code for implementation of Column Transposition cipher.

**Answer:**

**Column Transposition Cipher Code:**

#include <iostream>

#include <set>

#include <string>

#include <vector>

#include <algorithm>

using namespace std;

int main() {

string msg = "meet me after the party";

string key = "HACK";

set<pair<char, int>> keySet;

keySet.insert(make\_pair('H', 3));

keySet.insert(make\_pair('A', 1));

keySet.insert(make\_pair('C', 2));

keySet.insert(make\_pair('K', 4));

int col = key.length();

int row = (msg.length() + col - 1) / col;

vector<vector<char>> matrix(row, vector<char>(col, 'x'));

for (int i = 0, index = 0; i < msg.length(); ++i, ++index) {

matrix[index / col][index % col] = msg[i];

}

string cipher = "";

for (char keyChar : key) {

int permutationOrder = 0;

for (const auto& pair : keySet) {

if (pair.first == keyChar) {

permutationOrder = pair.second;

break;

}

}

for (int k = 0; k < row; ++k) {

char matrixChar = matrix[k][(permutationOrder - 1) % col];

if ((matrixChar >= 'A' && matrixChar <= 'Z') || (matrixChar >= 'a' && matrixChar <= 'z') || matrixChar == 'x')

cipher += matrixChar;

}

}

cout << "Cipher text: " << cipher << endl;

return 0;

}

**Output:**

**Plain Text**: meet me after the party

**Cipher Text**: eettpymareremfttehax

