Experiment No. 2

Title: Transposition Ciph

Batch:A3 Roll No.:16010421073 Experiment No.: 2

Aim: To implement transposition cipher – Row transposition and column transposition cipher.

Resources needed: Windows/Linux

Theory

Pre Lab/ Prior Concepts:

Symmetric-key algorithms are a class of algorithms for cryptography that use the same cryptographic keys for both encryption of plaintext and decryption of cipher text. The keys may be identical or there may be a simple transformation to go between the two keys. The keys, in practice, represent a shared secret between two or more parties that can be used to maintain a private information link. This requirement that both parties have access to the secret key is one of the main drawbacks of symmetric key encryption, in comparison to public-key encryption. Symmetric-key encryption can use either stream ciphers or block ciphers. Transposition Cipher is block cipher. Ancient cryptographic systems are classified as: Substitution and Permutation/Transposition Ciphers.

Transposition Cipher/Permutation Cipher

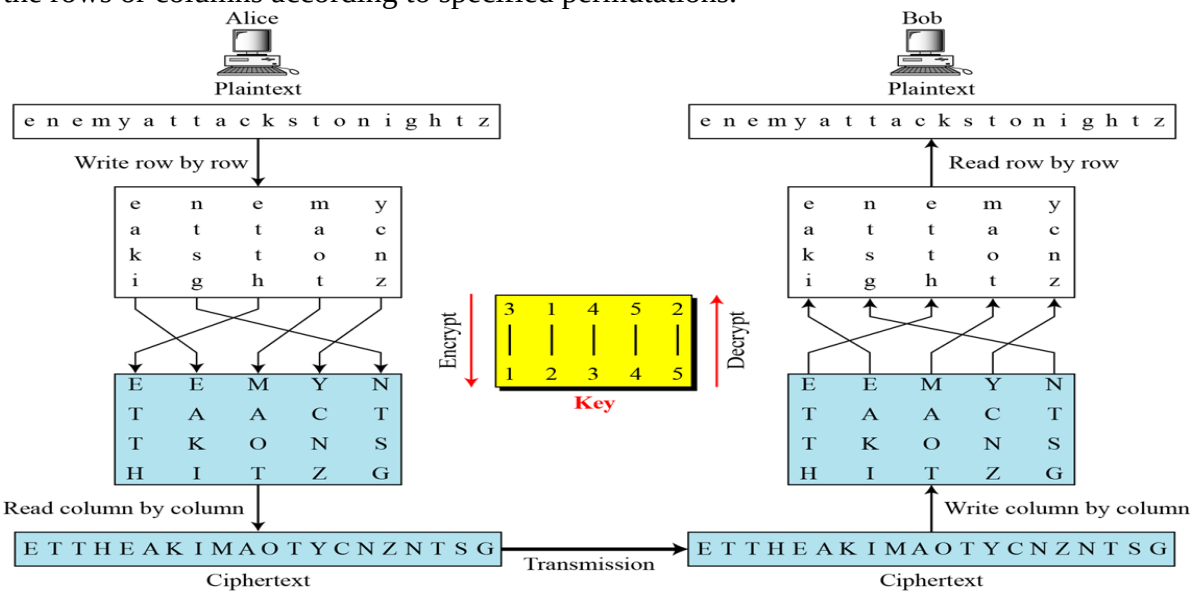
A transposition cipher rearranges (permutes) symbols in a block without altering actual values. It has the same frequency distribution as the original text .So it is easily recognizable.

EXAMPLE :

Plaintext: HELLO MY DEAR
Cipher text: ELHLMDOYAER

There are varieties of transposition ciphers like: keyless and keyed transposition ciphers.

Following figure shows the combination of both keyed and keyless. To encrypt with a transposition cipher, we first write the plaintext into a matrix of a given size and then permute the rows or columns according to specified permutations.



For the transposition, the key consists of the size of the matrix and the row or column permutations. The recipient who knows the key can simply put the cipher text into the appropriate sized matrix and undo the permutations to recover the plaintext.
Unlike a simple substitution, the transposition does nothing to disguise the letters that appear in the message But it does appear to thwart an attack that relies on the statistical information contained in the plaintext, since the plaintext statistics are disbursed throughout the cipher text. The double transposition is not a trivial cipher to break.

Activity :
Step 1 – Go through the theory explained and the instructions given by the instructor.

Step 2 – Derive/find encryption/decryption formula for each of the following transposition ciphers. Assume P as a plaintext square matrix, K as a row/column key and C as a ciphertext square matrix.

1) Row transposition cipher –

C[i][j] = M[(j-1) * n + K[i] - 1]
Where:
C[i][j] is the character at row i and column j in the encrypted grid.
M is the message.
L is the length of the message.
n is the number of rows in the grid.
K is the key permutation for rows.

P[i][j] =C[K[i] - 1][(j-1) * n + i]
Where:
P[i][j] is the character at row i and column j in the decrypted grid.
E is the encrypted message.
L is the length of the encrypted message.
n is the number of rows in the grid.
K is the key permutation for rows.

2) Column transposition cipher –

C[i][j] = M[K[j] * n - (n - i)]
Where:
C[i][j] is the character at row i and column j in the encrypted grid.
M is the message.
L is the length of the message.
n is the number of rows in the grid.
K is the key permutation for columns.

P[i][j] =C[L / m - (n - i)][K[j] * n - 1]
Where:
P[i][j] is the character at row i and column j in the decrypted grid.
E is the encrypted message.
L is the length of the encrypted message.
n is the number of rows in the grid.
K is the key permutation for columns.

3) Double transposition cipher (row followed by column)

C1[i][j] = M[(j-1) * n + K1[i] - 1]
c2[i][j] = C1[K2[j] - 1][(i-1) * m + j]
Where:
C1[i][j] is the character at row i and column j after the first permutation.
C2[i][j] is the character at row i and column j after the second permutation.
M is the message.
L is the length of the message.
n is the number of rows in the grid.
m is the number of columns in the grid.
K1 is the key for the first row permutation.
K2 is the key for the second column permutation.

D[i][j] =

D2[i][j] = E[(i-1) * m + K2'[j] - 1][j-1] D1[i][j] = D2[K1'[i] - 1][j]
Where:
D2[i][j] is the character at row i and column j after the second permutation inverse.
D1[i][j] is the character at row i and column j after the first permutation inverse.
E is the encrypted message.
L is the length of the encrypted message.
n is the number of rows in the grid.
m is the number of columns in the grid.
K1' is the inverse of the key for the first row permutation.
K2' is the inverse of the key for the second column permutation.

Step 3 – Implement

1) Row transposition cipher
2) Column transposition cipher
3) Double transposition cipher (row followed by column)

Implementation:
The program should have encryption function and decryption function for each cipher. Function should take message and a key as input from the user and display the expected output.

Results: (Program with output as per the format)
Menu Driven Program : (Row transposition cipher and column transposition cipher)

Code:

```
import math
```

```
def encryptColumn(msg, key):
    cipher = ""
    k_indx = 0
    msg_len = float(len(msg))
    msg_lst = list(msg)
    key_lst = sorted(list(key))
    col = len(key)
    row = int(math.ceil(msg_len / col))
    fill_null = int((row * col) - msg_len)
    msg_lst.extend('_' * fill_null)
    matrix = [msg_lst[i: i + col]
              for i in range(0, len(msg_lst), col)]
    for _ in range(col):
        curr_idx = key.index(key_lst[k_indx])
        cipher += ''.join([row[curr_idx]
                           for row in matrix])
        k_indx += 1
    return cipher

def decryptColumn(cipher, key):
    msg = ""
    k_indx = 0
    msg_indx = 0
    msg_len = float(len(cipher))
    msg_lst = list(cipher)
    col = len(key)
    row = int(math.ceil(msg_len / col))
    key_lst = sorted(list(key))
    dec_cipher = []
    for _ in range(row):
        dec_cipher += [[None] * col]
    for _ in range(col):
        curr_idx = key.index(key_lst[k_indx])
        for j in range(row):
            dec_cipher[j][curr_idx] = msg_lst[msg_indx]
            msg_indx += 1
        k_indx += 1
    try:
        msg = ''.join(sum(dec_cipher, []))
    except TypeError:
        raise TypeError("This program cannot handle repeating words.")
    null_count = msg.count('_')
    if null_count > 0:
        return msg[: -null_count]
    return msg

def encryptRow(msg, key):
    cipher = ''
    k = len(key)
    for i in range(k):
        j = int(key[i])
```

```
        cipher += msg[j::k]
    return cipher

def decryptRow(cipher, key):
    plain_text = ''
    k = len(key)
    columns = [''] * k
    for i in range(k):
        j = int(key[i])
        columns[j] = cipher[i::k]
    for i in range(len(columns[0])):
        for col in columns:
            plain_text += col[i] if i < len(col) else ''
    return plain_text

while True:
    print("1. Encrypt")
    print("2. Decrypt")
    print("3. Exit")
    choice = input("Choose an option: ")
    if choice == '1':
        key = input("Enter the key: ")
        msg = input("Enter the message to encrypt: ")
        cipherType = input("Choose the cipher type (row or column):
").strip().lower()
        if cipherType == "row":
            cipher = encryptRow(msg, key)
        elif cipherType == "column":
            cipher = encryptColumn(msg, key)
        else:
            print("Invalid cipher type. Please choose 'row' or 'column'.")
            continue
        print("Encrypted Message:", cipher)
    elif choice == '2':
        key = input("Enter the key: ")
        cipher = input("Enter the message to decrypt: ")
        cipherType = input("Choose the cipher type (row or column):
").strip().lower()
        if cipherType == "row":
            msg = decryptRow(cipher, key)
        elif cipherType == "column":
            msg = decryptColumn(cipher, key)
        else:
            print("Invalid cipher type. Please choose 'row' or 'column'.")
            continue
        print("Decrypted Message:", msg)
    elif choice == '3':
        break
    else:
        print("Invalid choice. Please try again.")
```

Output:

```
1. Encrypt
2. Decrypt
3. Exit
Choose an option: 1
Enter the key: 210
Enter the message to encrypt: keyur
Choose the cipher type (row or column): column
Encrypted Message: y_erku
1. Encrypt
2. Decrypt
3. Exit
Choose an option: 2
Enter the key: 210
Enter the message to decrypt: y_erku
Choose the cipher type (row or column): column
Decrypted Message: keyur
```

```
Decrypted Message: keyur
1. Encrypt
2. Decrypt
3. Exit
Choose an option: 1
Enter the key: 210
Enter the message to encrypt: attack
Choose the cipher type (row or column): row
Encrypted Message: tktcaa
1. Encrypt
2. Decrypt
3. Exit
Choose an option: 2
Enter the key: 210
Enter the message to decrypt: tktcaa
Choose the cipher type (row or column): row
Decrypted Message: tktaac
```

Questions:

1) **Compare substitution ciphers and transposition/permutation ciphers.**

Ans:

	Substitution Cipher	**Transposition cipher**
Basic principle	Substitution ciphers involve replacing each letter or character in the plaintext with another letter or character according to a fixed rule or key.	Transposition ciphers involve rearranging the characters of the plaintext without changing the actual characters themselves.
Encryption	The process involves creating a substitution table or key, which defines the mapping of each character in the plaintext to a corresponding character in the ciphertext.	The process involves reordering the characters of the plaintext based on a predetermined rule or key, such as reading the characters in a specific order (rows or columns) or applying a permutation.
Security	Substitution ciphers can be relatively easy to break using frequency analysis, especially if the language of the plaintext is known. This is because the frequency distribution of letters in most languages is predictable.	Transposition ciphers can be more secure than substitution ciphers, especially if the key used for rearrangement is complex and well-designed.
Examples	The Caesar cipher and the Affine cipher are examples of substitution ciphers.	The Rail Fence cipher and the Columnar Transposition cipher are examples of transposition ciphers.

Advantages	They are simple to implement and understand, making them suitable for educational purposes or situations where a basic level of security is sufficient.	They can provide a higher level of security compared to simple substitution ciphers, making them suitable for cases where stronger encryption is required.
Disadvantages	Due to their vulnerability to frequency analysis, substitution ciphers are not considered secure for serious encryption needs.	While they offer better security than substitution ciphers, they might still be vulnerable to attacks like pattern recognition if the structure of the transposition is too predictable.

2) Define confusion and diffusion properties. Comment on of both substitution and transposition ciphers w.r.t. confusion and diffusion properties.

Ans:

- **Confusion:** This property aims to make the relationship between the plaintext and the ciphertext as complex and unintuitive as possible. In other words, even if an attacker knows parts of the system (e.g., the algorithm used), they should not be able to easily deduce the relationship between the encryption key and the resulting ciphertext.
- **Diffusion:** This property focuses on spreading the influence of individual plaintext elements over multiple parts of the ciphertext, ensuring that small changes in the plaintext result in drastic changes in the ciphertext. Diffusion is about ensuring that the statistical properties of the plaintext do not show up in the ciphertext, making it difficult for an attacker to discern any patterns.

Substitution Ciphers and Confusion/Diffusion:

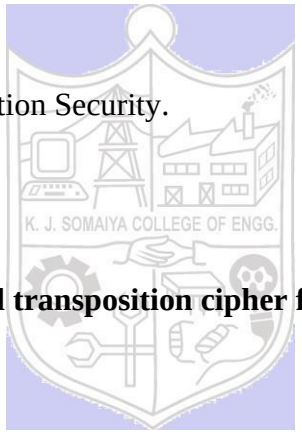
- **Confusion:**
  Substitution ciphers inherently provide some level of confusion by replacing one character with another. However, their susceptibility to frequency analysis and known language characteristics weakens their confusion property. In modern terms, they provide limited confusion.
- **Diffusion:**
  Substitution ciphers don't inherently offer diffusion. Changing one character in the plaintext only affects one character in the ciphertext, so local changes don't have a wide impact.

Transposition Ciphers and Confusion/Diffusion:

- **Confusion:** Transposition ciphers tend to provide a stronger level of confusion. The reordering of characters creates a complex relationship between the key and the ciphertext, making it harder for an attacker to infer any patterns.
- **Diffusion:** Transposition ciphers can offer a limited form of diffusion. Changes to the plaintext characters can influence multiple positions in the ciphertext, especially if the transposition rules are complex.

Outcomes:

CO1: Describe the basics of Information Security.



Conclusion:
Thus we successfully implemented transposition cipher for row and column.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date

References: Books/ Journals/ Websites:

1. Behrouz A. Forouzan, “Cryptography and Network Security”, Tata McGraw Hill
2. Mark Stamp, “Information Security Principles and Practice”, Wiley.
3. William Stalling, “Cryptography and Network Security”, Prentice Hall