FR. CONCEICAO RODRIGUES COLLEGE OF ENGINEERING

Department of Computer Engineering

1. Course , Subject & Experiment Details

Academic Year	**2021-22**	**Estimated Time**	**03 - Hours**
Course & Semester	**T.E. (CMPN)- Sem VI**	**Subject Name & Code**	**CSS - (CSL604))**
Chapter No.	**02 – Mapped to CO- 1**	**Chapter Title**	**Basics of Cryptography**

Practical No:	**1**
Title:	Design and Implementation of a product cipher using Substitution and Transposition ciphers
Date of Performance:	**08/02/2022**
Date of Submission:	**02/03/2022**
Roll No:	**8940**
Name of the Student:	**Warren Fernandes**

Evaluation:

Sr. No	Rubric	Grade
1	On time submission Or completion (2)	
2	Preparedness(2)	
3	Skill (4)	
4	Output (2)	

Signature of the Teacher:

Date:

MNS

Title: Design and Implementation of a product cipher using Substitution and Transposition Ciphers.

Lab Objective :

This lab provides insight into:

· How different types of Substitution Ciphers and Transposition Ciphers like Hill cipher, Verman cipher, Playfair cipher, Vigenere cipher works and their advantages and disadvantages.

Reference : "Cryptography and Network Security" B. A. Forouzan
"Cryptography and Network Security" Atul Kahate **Pre-requisite**

: Any Programming language and Knowledge of Ciphering .

Theory:

Cryptography is the practice and study of hiding information. It is the process of converting ordinary information (plain text) into cipher text and converting cipher text again to plain text, A cipher is a pair of algorithms which create the encryption and decryption.

Substitution Cipher: In cryptography, a **substitution cipher** is a method of encryption by which units of plaintext are replaced with cipher text according to a regular system; the "units" may be single letters (the most common), pairs of letters, triplets of letters, mixtures of the above, and so forth. The receiver deciphers the text by performing an inverse substitution.

Types of substitution cipher:

· **Monoalphabatic Cipher:** A *monoalphabetic substitution cipher*, also known as a simple substitution cipher, relies on a fixed replacement structure. That is, the substitution is fixed for each letter of the alphabet. Thus, if "a" is encrypted to "R", then every time we see the letter "a" in the plaintext, we replace it with the letter "R" in the ciphertext.

Ex. If **a** is substituted by **'x'** and **b** is substituted by **'y'** and so on than

"starbucks at three" encrypted as
PQXOYRHPXQQEOBB

1) Caesar Cipher/Additive/ Shift: It is a type of substitution cipher in which each letter in the plaintext is replaced by a letter some fixed number of positions down the

alphabet. For example, with a shift of 3, A would be replaced by D, B would become E, and so

on. The method is named after Julius Caesar, who used it to communicate with his generals.

MNS

Example

Plain: ABCDEFGHIJKLMNOPQRSTUVWXYZ
Cipher: DEFGHIJKLMNOPQRSTUVWXYZABC

Like Plaintext: the quick brown fox jumps over the lazy dog
Cipher text: WKH TXLFN EURZQ IRA MXPSV RYHU WKH ODCB GRJ

· **Polyalphabetic Cipher :** In this cipher, we are using no. of substitutions at different positions in the message.

1) Vigenere Cipher
2) Hill Cipher

1) **Hill Cipher** : It is a block cipher.

Key: An invertible m*m matrix (where m is the block length) i.e the sender & receiver must first agree upon a key matrix A of size m*m. A must be invertible mod 26.
Encryption: To encrypt a message using a message using the Hill Cipher we must first turn our keyword into a key matrix (a 2*2 matrix for working with digraphs). We also turn the PT into digraphs and each of these into a column vector. We then perform matrix multiplication modulo the length of the alphabet (i.e 26) on each vector. These vectors are then converted back into letters to produce the ciphertext.
Decryption:
To decrypt a ciphertext encoded using the Hill Cipher, we must find the inverse matrix. Once we have the inverse matrix, the process is the same as encrypting. That is we multiply the inverse key matrix by the column vectors that the ciphertext is split into, take the results modulo the length of the alphabet, and finally convert the numbers back to letters.

2) **Verman Cipher:** it is a stream, polyalphabetic cipher in which the plaintext is XORed with a random or pseudorandom stream of data to generate the ciphertext. If the stream of data is truly random and used only once, this is the one-time pad.

Ex. H E L L O message
7 (H) 4 (E) 11 (L) 11 (L) 14 (O) message
+ 23 (X) 12 (M) 2 (C) 10 (K) 11 (L) key
= 30 16 13 21 25 message + key
= 4 (E) 16 (Q) 13 (N) 21 (V) 25 (Z) message + key (mod 26)
E Q N V Z → ciphertext

A **transposition cipher** is a method of encryption by which the positions held by units of plaintext (which are commonly characters or groups of characters) are shifted according to a regular system, so that the ciphertext constitutes a permutation of the plaintext. That is, the order of the units is changed.

MNS

1. **Rail Fence cipher:** The Rail Fence cipher is a form of transposition cipher that gets its name from the way in which it is encoded. In the rail fence cipher, the plaintext is written downwards on successive "rails" of an imaginary fence, then moving up when we get to the bottom. The message is then read off in rows. For example, using three "rails" and a message of 'WE ARE DISCOVERED. FLEE AT ONCE', the cipher writes out:

Example:

W . . . E . . . C . . . R . . . L . . . T . . . E
. E . R . D . S . O . E . E . F . E . A . O . C .
. . A . . . I . . . V . . . D . . . E . . . N . .

Then reads off:

WECRL TEERD SOEEF EAOCA IVDEN

2. **Single Columnar transposition**: In a columnar transposition, the message is written out in rows of a fixed length, and then read out again column by column, and the columns are chosen in some scrambled order. Both the width of the rows and the permutation of the columns are usually defined by a keyword. For example, the word `ZEBRAS` is of length 6 (so the rows are of length 6), and the permutation is defined by the alphabetical order of the letters in the keyword. In this case, the order would be "6 3 2 4 1 5".

In a regular columnar transposition cipher, any spare spaces are filled with nulls; in an irregular columnar transposition cipher, the spaces are left blank. Finally, the message is read off in columns, in the order specified by the keyword. For example, suppose we use the keyword `ZEBRAS`and the message `WE ARE DISCOVERED. FLEE AT ONCE`. In a regular columnar transposition, we write this into the grid as:

Example:

ZEBRAS - 632415

6 3 2 4 1 5
W E A R E D
I S C O V E
R E D F L E
E A T O N C
E Q K J E U
The ciphertext is then read off as:

EVLNE ACDTK ESEAQ ROFOJ DEECU WIREE

3. **Double Columnar transposition:** A single columnar transposition could be attacked by guessing possible column lengths, writing the message out in its columns (but in the wrong order, as the key is not yet known), and then looking for possible anagrams. Thus to make it

stronger, a double transposition was often used. This is simply a columnar transposition applied twice. The same key can be used for both transpositions, or two different keys can be used.

As an example, we can take the result of the irregular columnar transposition in the previous section, and perform a second encryption with a different keyword, `STRIPE`, which gives the permutation "564231"

Example:

5 6 4 2 3 1
E V L N A C
D T E S E A
R O F O D E
E C W I R E
E
This is read off column wise to give the cipher text.

CAEEN SOIAE DRLEF WEDRE EVTOC
Algorithm of Proposed Product Cipher :

1. **Take the plaintext from the user . Apply substitution cipher to the text(Here we are using Ceaser cipher)**
2. **Take the encryption key from the user .**
3. **for every i in plaintext:**

   if i is uppercase:
   temp = 65 + ((i) - 65 + key) % 26)(ASCII value of upper case starts at 65) else:

   temp = 97 + ((i) - 97 + key) % 26)(ASCII value of upper case starts at 97)

4. **Take the encrypted ciphertext and again apply transposition cipher(Here we are using Railfence).**
5. **Create a matrix where Number of columns in matrix = len(cipher-text) .Number of rows = key**
6. **fill the rail matrix to distinguish filled spaces from blank ones**
7. **check the direction of flow, reverse the direction if we've just filled the top or bottom rail**
8. **Now to get the cipher text just append the alphabets row by row.**
9. **Now take the cipher text that has been encrypted twice and decrypt using rail fence cipher**
10. **We just need to append the alphabets stored in matrix in zigzag manner**
11. **Now take the decrypted text and again apply caesar cipher decryption to get original plaintext.**

Example of Product Cipher:

In [1]:

```
def encrypt(plaintext,key):
    encryption_str = ''
    for i in plaintext:
        if i.isupper():
            temp = 65 + ((ord(i) - 65 + key) % 26)
            encryption_str = encryption_str + chr(temp)
        elif i.islower():
            temp = 97 + ((ord(i) - 97 + key) % 26)
            encryption_str = encryption_str + chr(temp)
        else:
            encryption_str = encryption_str + i
    return encryption_str

def decrypt(ciphertext,key):
    decryption_str = ''
    for i in ciphertext:
        if i.isupper():
            if ((ord(i) - 65 - key) < 0):
                temp = 65 + ((ord(i) - 65 - key + 26) % 26)
            else:
                temp = 65 + ((ord(i) - 65 - key) % 26)
            decryption_str = decryption_str + chr(temp)
        elif i.islower():
            if ((ord(i) - 97 - key) < 0):
                temp = 97 + ((ord(i) - 97 - key + 26) % 26)
            else:
                temp = 97 + ((ord(i) - 97 - key) % 26)
            decryption_str = decryption_str + chr(temp)
        else:
            decryption_str = decryption_str + i
    return decryption_str
```

In [2]:

```
def encryptRailFence(text, key):
    rail = [['\n' for i in range(len(text))]
    for j in range(key)]
    dir_down = False
    row, col = 0, 0
    for i in range(len(text)):
        if (row == 0) or (row == key - 1):
            dir_down = not dir_down
        rail[row][col] = text[i]
        col += 1
        if dir_down:
            row += 1
        else:
            row -= 1
    result = []
    for i in range(key):
        for j in range(len(text)):
            if rail[i][j] != '\n':
                result.append(rail[i][j])
    return("" . join(result))

def decryptRailFence(cipher, key):
    rail = [['\n' for i in range(len(cipher))] for j in range(key)]
    dir_down = None
    row, col = 0, 0
    for i in range(len(cipher)):
        if row == 0:
            dir_down = True
        if row == key - 1:
            dir_down = False
        rail[row][col] = '*'
        col += 1
        if dir_down:
            row += 1
        else:
            row -= 1
    index = 0
    for i in range(key):
        for j in range(len(cipher)):
            if ((rail[i][j] == '*') and (index < len(cipher))):
                rail[i][j] = cipher[index]
                index += 1
    result = []
    row, col = 0, 0
    for i in range(len(cipher)):
        if row == 0:
            dir_down = True
        if row == key-1:
            dir_down = False
        if (rail[row][col] != '*'):
            result.append(rail[row][col])
            col += 1
        if dir_down:
            row += 1
        else:
            row -= 1
    return("".join(result))
```

Practical & Real-Time Application

In [5]:

```
word=input("Enter the text to be ciphered: ")
key=int(input("Enter the key:"))
encrypted = encrypt(word,key)
print(encrypted)
encryptRail=encryptRailFence(encrypted,key)
print(encryptRail)
print("The text deciphered:")
decryptRail=decryptRailFence(encryptRail,key)
print(decryptRail)
decrypted=decrypt(decryptRail,key)
print(decrypted)
```

```
Enter the text to be ciphered: Warren
Enter the key:3
Zduuhq
Zhduqu
The text deciphered:
Zduuhq
Warren
```

Conclusion:

The program was tested for different sets of inputs.
Program is working SATISFACTORY

Post Lab :

1. Explain why Modular arithmetic has been used in cryptography.

One major reason is that modular arithmetic allows us to easily create groups, rings and fields which are fundamental building blocks of most modern public-key cryptosystems. For example, Diffie-Hellman uses the multiplicative group of integers modulo a prime p.

2. To break the Caesar cipher using brute force attack, how many attempts are needed?

We need 25 attempts to crack Ceaser cipher by brute force as the key value ranges from 0 to 25 in this method . Therefore we can apply keys from 0 to 25 and decipher

the ciphertext and the text that makes sense is taken as plaintext.

3. Compare Substitution and Transposition techniques.

In substitution Cipher Technique, plain text characters are replaced with other characters, numbers and symbols.	In transposition Cipher Technique, plain text characters are rearranged with respect to the position.
Substitution Cipher's forms are: Mono alphabetic substitution cipher and poly alphabetic substitution cipher.	Transposition Cipher's forms are: Key-less transposition cipher and keyed transposition cipher.
In substitution Cipher Technique, character's identity is changed while its position remains unchanged.	While in transposition Cipher Technique, The position of the character is changed but character's identity is not changed.
In substitution Cipher Technique, The letter with low frequency can detect plain text.	While in transposition Cipher Technique, The Keys which are nearer to correct key can disclose plain text.