LAB Manual

PART A

(PART A : TO BE REFFERED BY STUDENTS)

**Experiment No.01**

**A.1 Aim:**

To implement a simple substitution cipher and a transposition cipher using Socket Programming

**A.2 Prerequisite:**

Fundamentals of encryption and decryption process

**A.3 Outcome:**

**After successful completion of this experiment students will be able to** 1. Learn and appreciate the operation of classical ciphers

2. Understand the shortcomings of these ciphers

**A.4 Theory:**

The art and science of keeping messages secure is **cryptography.** A message is **plaintext (**sometimes called **cleartext).** The process of disguising a message in such a way as to hide its substance is **encryption.** An encrypted message is **ciphertext.** The process of turning ciphertext back into plaintext is **decryption.**

A **cryptographic algorithm**, also called a **cipher**, is the mathematical function used for encryption and decryption. Both the encryption and decryption operations use a key. The range of possible values of the key is called the **keyspace.**

* *EK(M) = C*
* *DK(C) = M*
* *DK(EK(M)) = M*

There are two basic types of classical ciphers:

* Transposition ciphers
* Substitution ciphers

A **substitution cipher** is one in which each character in the plaintext is substituted for another character in the ciphertext. The receiver inverts the substitution on the ciphertext to recover the plaintext.

For example:

**Caesar Cipher,** in which each plaintext character is replaced by the character three to the right modulo 26 ("A" is replaced by "D," "B" is replaced by "E,"..., "X“ is replaced by "A," "Y" is replaced by "B," and "Z" is replaced by "C") is a simple substitution cipher.

A **transposition cipher** is obtained by performing some sort of permutation on the plain text symbols. Here, the symbols remain the same but their order is shuffled around.

For example:

In a simple columnar transposition cipher, the plain text is written horizontally row by row in rectangle of fixed width. To form cipher, the text is read off vertically. The order of columns is kept secret and is the key. Decryption involves writing the ciphertext vertically of identical length and then reading the text horizontally. Let me illustrate this with the help of an example:

Consider the sentence: “IT IS RAINING TODAY”

Key value: 4 (rectangular width)

Encryption process:

|  |  |  |  |
| --- | --- | --- | --- |
| I | T | I | S |
| R | A | I | N |
| I | N | G | T |
| O | D | A | Y |

Ciphertext: IRIO TAND IIGA SNTY

PART B

(PART B : TO BE COMPLETED BY STUDENTS)

***(Students must submit the soft copy as per following segments within two hours of the practical. The soft copy must be uploaded on the Blackboard or emailed to the concerned lab in charge faculties at the end of the practical in case the there is no Black board access available)***

|  |  |
| --- | --- |
| Roll. No. C023 | Name: Nilay Gaitonde |
| Class: B | Batch: B1 |
| Date of Experiment: 31-08-2022 | Date of Submission: 31-08-2022 |
| Grade: | |

**B.1 Software Code written by student:**

**cipher.py**

import math

from pydoc import plain

class SubstitutionCipher:

def \_\_init\_\_(*self*):

pass

def encrypt(*self*,*key*,*plainText*):

cipherText = ""

for letter in plainText:

cipherText=cipherText+chr(ord(letter)+len(key))

transpositionCipher = TranspositionCipher()

cipherText = transpositionCipher.encrypt("four",cipherText)

return cipherText

def decrypt(*self*,*key*,*cipherText*):

plainText = ""

transpositionCipher = TranspositionCipher()

cipherText = transpositionCipher.decrypt("four",cipherText)

for letter in cipherText:

plainText=plainText+chr(ord(letter)-len(key))

return plainText

class TranspositionCipher:

def \_\_init\_\_(*self*):

pass

def encrypt(*self*,*key*,*plainText*):

cipherText = ""

ind = 0

orderedKey = sorted(list(key))

plainTextList = list(plainText)

columns = len(key)

rows = math.ceil(len(plainText)/columns)

plainTextList.extend("\_"\*(int((rows \* columns) - len(plainText))))

matrix = [plainTextList[i:i+columns] for i in range(0, len(plainTextList), columns)]

for \_ in range(columns):

i = key.index(orderedKey[ind])

cipherText += "".join([rows[i] for rows in matrix])

ind += 1

return cipherText

def decrypt(*self*,*key*,*cipherText*):

ind = 0

orderedKey = sorted(list(key))

index=0

cipherTextList = list(cipherText)

columns = len(key)

row = int(math.ceil(len(cipherText) / columns))

dec\_matrix = []

for \_ in range(row):

dec\_matrix += [[None] \* columns]

for \_ in range(columns):

curr\_idx = key.index(orderedKey[ind])

for j in range(row):

dec\_matrix[j][curr\_idx] = cipherTextList[index]

index += 1

ind += 1

try:

cipherText = ''.join(sum(dec\_matrix, []))

except TypeError:

raise TypeError("This program cannot",

"handle repeating words.")

null\_count = cipherText.count('\_')

if null\_count > 0:

return cipherText[: -null\_count]

return cipherText

if \_\_name\_\_=="\_\_main\_\_":

# substitutionCipher = TranspositionCipher()

substitutionCipher = SubstitutionCipher()

cipherText = substitutionCipher.encrypt("four","Hello world")

plainText = substitutionCipher.decrypt("four",cipherText)

print(plainText)

pass

**myclient.py**

import socket

import ciphers

class Client:

def \_\_init\_\_(*self*,*host*,*port*,*message*,*cipher*):

with socket.socket(socket.AF\_INET,socket.SOCK\_STREAM) as s:

s.connect((host,port))

s.sendall(message.encode())

data = s.recv(1024)

print(f"Received: {data}")

data = cipher.decrypt("four",data.decode())

print(f"Final message: {data}")

if \_\_name\_\_=='\_\_main\_\_':

cipher = ciphers.SubstitutionCipher()

c = Client('127.0.0.1',6666,cipher.encrypt("four",input("Enter message: ")),cipher)

**myserver.py**

import socket

import ciphers

substitutionCipher = ciphers.SubstitutionCipher()

# transpositionCipher = ciphers.TranspositionCipher()

HOST = '127.0.0.1'

PORT = 6666

with socket.socket(socket.AF\_INET,socket.SOCK\_STREAM) as s:

s.bind((HOST,PORT))

s.listen()

conn,addr = s.accept()

with conn:

while True:

data = conn.recv(1024)

if not data:

break

data = data.decode()

print(f"Received: {data}")

data = substitutionCipher.decrypt("four",data)

print(f"Message: {data}")

data = substitutionCipher.encrypt("four",data)

print(f"Sent: {data}")

conn.sendall(data.encode())

**B.2 Input and Output:**

**Input:**

1. **Plaintext/message: Hello world**
2. **Key value: “Four” (fixed)**

**Output:**

1. **Ciphertext :** Lipps${svph

A screenshot of a computer

Description automatically generated with medium confidence

**B.3 Observations and learning:**

***Learnt socket programming and encryption and decryption of text.***

**B.4 Conclusion:**

*Ceaser cipher is a string cipher that focuses on substitution*

*Transposition cipher is a string ci[her that focuses on confusion*

**B.5 Question of Curiosity**

***(To be answered by student based on the practical performed and learning/observations)***

Q1: Give the advantages and disadvantages of substitution and transposition ciphers.

Q2: Enlist various types of substitution and transposition ciphers and explain them in brief.