**CRYPTOGRAPHY CODING ASSIGNMENT**

**SANSKRATI JAIN**

**CSB20047**

**Assignment 1:**

Implement Encryption and Decryption operations using OTP.

**Solution**

**Code for generating key:**

#include <stdio.h>

#include <unistd.h>

#include <fcntl.h>

#include <math.h>

int main()

{

    int i;

    unsigned long long int key\_size = pow(2, 64); //key size

    unsigned char buffer[8];

    int fd = open("/dev/urandom", O\_RDONLY);

    read(fd, buffer, key\_size);

    // buffer now contains the random data

    close(fd);

    //write file

    FILE \*fptr;

    fptr = fopen("key.txt", "w");

    for (i = 0; i < key\_size; ++i)

    {

        printf("%02X", buffer[i]);

        fprintf(fptr, "%02X", buffer[i]);

    }

    printf("\n");

    fclose(fptr);

    return 0;

}

**Code for OTP:**

import base64 # for converting into byte

# function for file handling

def FileHandling(filename, operation, message="read operaation"):

    with open(filename, operation) as f:

        # if operation is "write"

        if operation == "w":

            f.write(message)

            f.close()

            return

    # if operation is "read"

        elif operation == "r":

            text = f.read()

            f.close()

            return text

# function for encryption

def Enc(plaintext, key):

    r = '' # encrypted text

# xor byte by byte

    for i in range(len(plaintext)):

        c = ord(plaintext[i]) ^ key[i]

        r += chr(c)

    # convert string to bytes

    retval = base64.b64encode(r.encode())

    return retval

# function for decryption

def Dec(ciphertext, key):

    r = ''  # decrypted string

    # convert bytes to string

    med1 = base64.b64decode(ciphertext)

    med2 = med1.decode()

# xor byte by byte

    for i in range(len(ciphertext)):

        c = ord(med2[i]) ^ key[i]  # ascii value e.g 108

        r += chr(c)

    return r

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

    # plaintext

    plaintext = FileHandling("plaintext.txt", "r")

    length = len(plaintext)

    # key generation

    key = FileHandling("key.txt", "r")

# key conversion in byte format

    key = base64.b64encode(key.encode())

    # cipher text generation

    cipher\_text = Enc(plaintext, key)

    print("\nMessage Decrypted and stored in file\n")

    FileHandling("encrypt.txt", "w", str(cipher\_text))

    # decrypt text generation

    decrypted\_text = Dec(cipher\_text, key)

    print("\nMessage Encrypted and stored in file")

    FileHandling("decrypt.txt", "w", str(decrypted\_text))

**Output:**

