Name: CHAARU MANJURAJ KOMALA

Student ID: 2406827

LM Network Security and Cryptography (34231) Exercise 1 (Formative Assessment)

Question – I

Decrypt the following cipher text encrypted with the columnar transposition cipher (you need to find the key by brute-force search):

AVUEVLETSEISBNACBOOLEOBTILBDLCOBOOE

A	L	I	C	E	L	O
V	E	S	B	O	B	B
U	T	B	O	B	D	O
E	S	N	O	T	L	O
V	E	A	L	I	C	E

The Decrypted Message is:

ALICE LOVES BOB BUT BOB DOES NOT LOVE ALICE

Question – II

What is the output of the first round of the DES algorithm when the plain-text and the key are both all zeros?

$L_n = R_{n-1}$

$R_n = L_{n-1} \oplus f(R_{n-1}, k_n)$

ROUND – I

$L_0 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$

$R_0 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$

$K_1 = 00000000\ 00000000\ 00000000\ 00000000\ 00000000\ 00000000$

$\boldsymbol{L_1 = R_0 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000}$

$\boldsymbol{R_1 = L_0 \oplus f(R_0, k_1) = 1101\ 1000\ 1101\ 1000\ 1101\ 1011\ 1011\ 1100}$

Question – III

Remember that it is desirable for good block ciphers that a change in one input bit affects many output bits, a property that is called diffusion or avalanche effect. We will try to get a feeling for the avalanche property of DES.

Let x be all zeros (0x0000000000000000) and y be all zeros except 1 in the 13th bit (0x0008 000000000000). Let the key be all zeros. After just one round, how many bits in the block are different when x is the input, compared to when y is the input? What about after two rounds? Three? Four?

X (0x0000000000000000)	Y (0x0008 000000000000)
Formula	Formula
$L_n = R_{n-1}$ $R_n = L_{n-1} \oplus f(R_{n-1}, k_n)$	$L_n = R_{n-1}$ $R_n = L_{n-1} \oplus f(R_{n-1}, k_n)$
Round – I	**Round – I**
$L_1 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$ $R_1 = 1101\ 1000\ 1101\ 1000\ 1101\ 1011\ 1011\ 1100$	$L_1 = 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000\ 0000$ $R_1 = 1101\ 0000\ 1101\ 1010\ 1101\ 0011\ 1011\ 1100$
Round – II	**Round – II**
$L_2 = 1101\ 1000\ 1101\ 1000\ 1101\ 1011\ 1011\ 1100$ $R_2 = 1110\ 0111\ 0011\ 1010\ 1110\ 1101\ 0100\ 1111$	$L_2 = 1101\ 0000\ 1101\ 1010\ 1101\ 0011\ 1011\ 1100$ $R_2 = 1001\ 0100\ 1000\ 1010\ 0101\ 1001\ 0001\ 1111$
Round – III	**Round – III**
$L_3 = 1110\ 0111\ 0011\ 1010\ 1110\ 1101\ 0100\ 1111$ $R_3 = 0101\ 1011\ 1111\ 1010\ 0110\ 1011\ 1010\ 0110$	$L_3 = 1001\ 0100\ 1000\ 1010\ 0101\ 1001\ 0001\ 1111$ $R_3 = 1001\ 1010\ 0101\ 1100\ 0001\ 1111\ 1111\ 1001$
Round – IV	**Round – IV**
$L_4 = 0101\ 1011\ 1111\ 1010\ 0110\ 1011\ 1010\ 0110$ $R_4 = 0101\ 1110\ 0100\ 0101\ 1110\ 0010\ 0101\ 0111$	$L_4 = 1001\ 1010\ 0101\ 1100\ 0001\ 1111\ 1111\ 1001$ $R_4 = 0011\ 1101\ 0001\ 0011\ 0100\ 1011\ 1010\ 1100$

Number of different bits in the block when x is the input, compared to when y is the input:

$L_1 = 0\ bits$	$L_2 = 3\ bits$	$L_3 = 1\ bits$	$L_4 = 17\ bits$
$R_1 = 3\ bits$	$R_2 = 14\ bits$	$R_3 = 17\ bits$	$R_4 = 19\ bits$

Question – IV

Consider AES with 128-bit keys. Assume that the principal key k is all-zeros. Then the initial round key (k0) is also all-zeros. What is the first round subkey (k1) and the second round subkey (k2)?

Message: 0123456789ABCDEFFEDCBA9876543210

Principal key (k): 00000000000000000000000000000000

Round Key in Round 0 (k_0)

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

Round Key in Round 1 (k_1)

62 63 63 63 62 63 63 63 62 63 63 63 62 63 63 63

Round Key in Round 2 (k_2)

9B 98 98 C9 F9 FB FB AA 9B 98 98 C9 F9 FB FB AA

Question – V

Programming exercise: Let MY24SHA a hash function which outputs the first 24 bits (6 nibbles) of SHA-1. For example, SHA-1 of "mark" is f1b5a91d4d6ad523f2610114591c007e75d15084 so the MY24SHA of "mark" is f1b5a9.

Find any collision for MY24SHA. Note: you should find two strings such that the unix command

echo -n str | sha1sum - | cut -c1-6

produces the same answer when str is replaced by each string. To enable me to verify your answer, please make sure the two strings are typable on a regular keyboard!

WORD – I: `abasedness`

WORD – II: `hydurilate`

MY24SHA of `abasedness` is `3fd8b0`

MY24SHA of `hydurilate` is `3fd8b0`

```python
import hashlib

# I named this function 'encrypt' instead of MY24SHA
# Because I liked it more. My apologies!
def encrypt(s):
    # Convert to b
    s_bin = bytes(s, 'utf-8')
    # Hashing
    full_hash = hashlib.sha1(s_bin).hexdigest()
    return str(full_hash[:6])

def compare():
    # The wordlist I used is available with this repo
    with open('words.txt', 'r') as wordlist:
        words = []
        sha = []
        # Creating 2 separate lists, trying to strip it along with
        # hashing didn't work and was taking too long
        for (index, line) in enumerate(wordlist):
            words.append(line.strip('\n'))
        for word in words:
            sha.append(encrypt(word))

        # Loops to linear comparision
        for i, item in enumerate(sha):
            for k in range(i+1, len(sha)):
                if((k) < len(sha)):
                    print(f'Comparing {sha[i]} and {sha[k]}')
                    if(sha[i] == sha[k]):
```

```
                print("\n\n-------Matched!-------\n")
                print(sha[i], i)
                print(sha[k], k+1)
                return (i, (k+1))
            else:
                k += 1
        i += 1

"""
print(encrypt('abasedness'))
print(encrypt('hydurilate'))
'''
result = compare()
index1, index2 = result
f = open('words.txt')
content = f.readlines()
print(f'WORD 1: {content[index1]}')
print(f'WORD 2: {content[index2]}')
f.close()
```

The code is available publicly on https://github.com/ChaaruManjuraj/MY24SHA