Homework Number: 01

Name: Yuan Liu

ECN Login: liu1827

Due Date: January.23 2020

1 The Recovered Plaintext Quote

It is my belief that nearly any invented quotation, played with confidence, stands a good chance to deceive.
  - Mark Twain

2 The encryption key

The key is: 25202

3 Explanation

As described in the document of homework 1, the ciphertext is encrypted by *EncryptForFun.py* from Prof. Kak. Therefore, the overall thought of *cryptBreak.py* is to reverse the process of *EncryptForFun.py* by testing every possible key from 0 to 2^{16}, (since the BLOCKSIZE is set to 16). The function cryptBreak will divide the bitvector of ciphertext by BLOCKSIZE, which is 16, and xor these divisions with its previous division to decrypt the message (The first one will xor with a null bitvector).

4 Code for cryptBreak

The code below is modified from DecryptForFun.py from Prof. Kak's ECE404 lecture

```
# Arguments:
# ciphertextFile: String containing file name of the ciphertext (e.g. encrypt
# key_bv: 16-bit BitVector of the key used to try to decrypt the ciphertext.

# Function Description:
# Attempts to decrypt ciphertext contained in ciphertextFile using key_bv and
# the original plaintext as a string

from BitVector import *


def cryptBreak(ciphertextFile, key_bv):
    BLOCKSIZE = 16

    # Create a null bitvector:
    bv_iv = BitVector(bitlist = [0]*BLOCKSIZE)

    # Create a bitvector from the ciphertext hex string:
    FILEIN = open(ciphertextFile)
    encrypted_bv = BitVector(hexstring=FILEIN.read())

    # Create a bitvector for storing the decrypted plaintext bit array:
    msg_decrypted_bv = BitVector(size=0)

    previous_decrypted_block = bv_iv
    for i in range(0, len(encrypted_bv) // BLOCKSIZE):
        bv = encrypted_bv[i * BLOCKSIZE:(i + 1) * BLOCKSIZE]
        temp = bv.deep_copy()
        bv ^= previous_decrypted_block
        previous_decrypted_block = temp
        bv ^= key_bv
        msg_decrypted_bv += bv

    # Extract plaintext from the decrypted bitvector:
    outputtext = msg_decrypted_bv.get_text_from_bitvector()

    return outputtext
```

5 Temporary Main for testing

```
if __name__ == '__main__':
    for key in range(0, 2 ** 16):
        someRandomInteger = key  # Arbitrary integer for creating a BitVector
        key_bv = BitVector(intVal=someRandomInteger, size=16)
        decryptedMessage = cryptBreak('encrypted.txt', key_bv)
        if 'Mark Twain' in decryptedMessage:
            print('Encryption Broken!')
            print('The message is: {}', decryptedMessage)
            print('The key is: {}', key)
            break
```