ECE 404 Homework 1

Ranjan Behl (rbehl)

January 28, 2021

1. The Quote

Always go to other people's funerals, otherwise they won't go to yours.

- Yogi Berra

2. The Key

The key was the integer: 30053

3. Explanation of the Code

The code I used is shown below:

```
rom BitVector import *
# constants
BLOCKSIZE = 16
numbytes = BLOCKSIZE // 8
# main function


def main():
    for intKey in range(0, 65535):
        key_bv = BitVector(intVal= intKey, size=16)
        decryptedMsg = cryptBreak(sys.argv[1], key_bv)
        #print(decryptedMsg)
        if 'Yogi Berra' in decryptedMsg:
            print('Encrpytion Broken!')
            print('\n The key was was: ', intKey)
            print('\n The original quote was: ',decryptedMsg)
        else:
            #print('Not decrypted yet')
            pass


def cryptBreak(ciphertextFile, key_bv):
        encryptedFile = open(ciphertextFile.strip(),'r')
        encrypted_bv = BitVector(hexstring = encryptedFile.read())
        encryptedFile.close()  # close the file
        ### use the code from DecryptForFun ###
        PassPhrase = "Hopes and dreams of a million years"
        # Reduce the passphrase to a bit array of size BLOCKSIZE:
        bv_iv = BitVector(bitlist=[0]*BLOCKSIZE)
        for i in range(0, len(PassPhrase) // numbytes):
            textstr = PassPhrase[i*numbytes:(i+1)*numbytes]
```

```
        bv_iv ^= BitVector(textstring = textstr)
    # Create a bitvector for storing the decrypted plaintext bit array:
    msg_decrypted_bv = BitVector(size = 0)
    # Carry out differential XORing of bit blocks and decryption:
    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 text back to caller
    return outputtext


if __name__ == "__main__":
    main()
```

The code can be broken into two main parts the main function and cryptBreak. In the main function since it was known that the max size of the key is 16 bits, and I knew that since the key is a int the max int value could only be $65535(2^{16} - 1)$), minus one because python starts at 0. Due to the 16 bits that also meant that block-size had to be 16. The rest of the code in main is based on the usage of cryptBreak provided in the homework document. The second function is cryptBreak, in which the decryption happens. Most of the code here is from DecryptforFun file. However I had to change some things when compared to the original source. The first thing I did was use the strip() function because without I was getting some extra characters in the encrypted file. Another thing is I only had to convert the passphrase to a bitvector since the key bitvector is already passed in as a parameter of the function. The last change is that I used a return statement to return the decrypted message to main so main can check if the encryption was broken.