בס"ד

הסבר על הקוד

שלב 2:

```python
message = b'Hello World'
padded_message = pad(message, 16)

print(padded_message)
```

ניצור את ההודעה ונרפד אותה בעזרת הפונקציה pad ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
b'Hello World\x05\x05\x05\x05\x05'
```

שלב 3:

```python
message = b'Hello World'
padded = pad(message, 16)
key = b'poaisfun'
IV = b'\x00' * 8
cipher = DES.new(key, DES.MODE_CBC, IV)
encryptedBlocks = []
numOfBlocks = len(padded) // 8 #2
for i in range(numOfBlocks):
    start = i * 8
    end = (i + 1) * 8
    block = padded[start:end]
    encryptedBlock = cipher.encrypt(block)
    encryptedBlocks.append(encryptedBlock)

encrypted = b''.join(encryptedBlocks)
messHex = encrypted.hex()
for i in range(0, len(messHex), 2):
    print(messHex[i:i+2])
```

כעת נגדיר את המפתח poaisfun כמבוקש ואת IV להיות 8*00 כי DES בבלוקים בגודל שמונה ונבצע חלוקה לבלוקים לפי כמות בלוקים של אורך ההודעה חלקי 8 (בחילוק שלמים), ניצור את ה cipher שלנו בשורה 8 ונבצע הצפנה בלולאה על הבלוקים בשורות 11-16 ונוסיף למערך של הבלוקים המוצפנים שלנו, לבסוף נכניס הכל להודעה ארוכה ב18, נבצע המרה להקסאדצימלי ונדפיס כל פעם שני תווים כי בכל בית יש שני תווים הקסאדצימליים, נדפיס עם הקידומת 0x לקריאה נוחה יותר.

ואכן-

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
0x33
0xaa
0xa3
0x01
0x7e
0x45
0x33
0x7b
0xd3
0x63
0x42
0xb3
0x92
0x0b
0xe6
0x56
```

ההצפנה עבדה והבתים כמו שהיו אמורים לצאת.

בס״ד

שלב 4:

```python
message = b'Hello World'
padded = pad(message, 16)
key = b'poaisfun'
IV = b'\x00' * 8
cipher = DES.new(key, DES.MODE_CBC, IV)
encryptedBlocks = []
numOfBlocks = len(padded) // 8 #2
for i in range(numOfBlocks):
    start = i * 8
    end = (i + 1) * 8
    block = padded[start:end]
    encryptedBlock = cipher.encrypt(block)
    encryptedBlocks.append(encryptedBlock)

encrypted = b''.join(encryptedBlocks)
print(encrypted)
cipher = DES.new(key, DES.MODE_CBC, IV)
decrypted = cipher.decrypt(encrypted)
unPadded = unpad(decrypted, 16)
print(unPadded)
```

נבצע הצפנה כמו קודם, כעת ניצור cipher חדש לפיענוח כי אי אפשר באותו אחד, ונפענח, לבסוף נבטל את הריפוד ונדפיס לפני הפיענוח ואחרי ונקבל אכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
b'3\xaa\xa3\x01~E3{\xd3cB\xb3\x92\x0b\xe6V'
b'Hello World'
```

שלב 5:

```python
def xor(x, y, z):
    return x^y^z

print(xor(0,0,0))
print(xor(0,0,1))
print(xor(0,1,0))
print(xor(0,1,1))
print(xor(1,0,0))
print(xor(1,0,1))
print(xor(1,1,0))
print(xor(1,1,1))
```

על ידי פעולת xor המוגדרת, בbitwise ואכן ההדפסות כנדרש:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
0
1
1
0
1
0
0
1
```

שלב 6:

```python
def oracle(ciphertext, key, iv):
    try:
        cipher = DES.new(key, DES.MODE_CBC, iv)
        decrypted = cipher.decrypt(ciphertext)
        unPadded = unpad(decrypted, 16)
        print(unPadded)
        return True
    except:
        return False
```

באופן דומה לקודם, הפונקציה יוצרת cipher חדש עם הkey,iv ומבצעת פיענוח על הciphertext, במידה והפיענוח לא הצליח אז ניכשל, ואם זה ייכשל שם נגיע לexcept. נוסיף הדפסה בשביל שנוכל לראות אם זה תקין.

נבצע ניסיון:

בס"ד

```
result = oracle(b'3\xaa\xa3\x01~E3{\xd3cB\xb3\x92\x0b\xe6V', key, IV)
print(result)
print("--------------------------------------------------")
result = oracle(b'3\xaa\xa3\x01~E3{\xd2cB\xb3\x92\x0b\xe6V', key, IV)
print(result)
```

(הראשון זה עם הciphertext המוצפן מקודם, שהמקור זה Hello World ובשני שיניתי את ה\xd3 ל\xd2 ואמור לא לעבוד) ואכן:

```
b'Hello World'
True
--------------------------------------------------
False
```

שלב 7:

```
blockSize = 8
firstHalf = b'\x00' * blockSize
secondHalf = encrypted[blockSize:blockSize * 2]
c = firstHalf + secondHalf
printHex(c)
```

השמות פשוטות כמבוקש, כאשר printHex זאת הפונקציה עם הלולאה המוזכרת בשלב 3 שמדפיסה בית-בית בהקסאדצימלי.

ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0xd3
0x63
0x42
0xb3
0x92
0x0b
0xe6
0x56
```

כמבוקש.

שלב 8:

```
41  blockSize = 8
42  firstHalf = b'\x00' * blockSize
43  secondHalf = encrypted[blockSize:blockSize * 2]
44  c = firstHalf + secondHalf
45  while True:
46      eighth = c[7]
47      newEighth = (eighth + 1) % 256
48      c = c[:7] + bytes([newEighth]) + c[8:]
49      result = oracle(c, key, IV)
50      print(result)
51      if result:
52          printHex(c)
53          break
```

הוספתי כאן את הלולאה- בשורה 46 ניקח את הבית השמיני ונגדיל אותו בשורה לאחר מכן (יש שם מודולו 256 משום שיש רק 256 אופציות לבית 0-255), נעדכן את c בהתאם וננסה לפענח עד שהתוצאה תהיה

נכונה, נדפיס את התוצאה בכל פעם לבקרה ולבסוף את המקרה שעבד:

```
False
False
False
False
False
False
False
False
False
False
False
b'9\xe8\xd3E\xbfw\xdb\xdcA\xc6\xc7\x04{@6'
True
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x7f
0xd3
0x63
0x42
0xb3
0x92
0x0b
0xe6
0x56
```

אחרי רצף של הרבה falseים נקבל את הtrue (ויש את ההדפסה שנשארה בoracle משלב 6.

שלב 9:
המשוואה:

$$P_i[x] = P'_2[x] \oplus C_{i-1}[x] \oplus X_j[x]$$

לכן בשביל לקחת את $P_i[7]$ נעשה xor על כל הגורמים:

```python
encrypted = b''.join(encryptedBlocks)
blockSize = 8
firstHalf = b'\x00' * blockSize
firstHalfEncrypted = encrypted[0:blockSize]
secondHalf = encrypted[blockSize:blockSize * 2]
c = firstHalf + secondHalf
while True:
    eighth = c[7]
    newEighth = (eighth + 1) % 256
    c = c[:7] + bytes([newEighth]) + c[8:]
    result = oracle(c, key, IV)
    if result:
        break
p7 = hex(xor(0x01, firstHalfEncrypted[7], c[7]))
print(p7)
```

לאחר שבלולאה נמצא את התו שאיתו זה כן עובד, ניקח כל הגורמים בהתאם (הfirstHalfEncrypted כאן זה הבלוק הקודם).

ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
0x5
```

שלב 10:

```python
p7 = hex(xor(0x01, firstHalfEncrypted[7], c[7]))
forCheck = c[7] ^ 1
toChange = xor(c[7], 2, 1)
c = c[:7] + bytes([toChange]) + c[8:]
print(hex(forCheck ^ c[7]))
```

לאחר ש*p7* אכן פוענח כראוי, נשמור לבדיקה בהדפסה את *c[7]^1* (בשביל לפענח עם *xor* על כל הגורמים, ונשנה את *c[7]* להיות $c[7]\oplus 2\oplus 1$, משום שעל פי הנוסחה מהמצגת, משום ש2 זה המבוקש שלנו ו1 זה ה*plaintext* הקודם וככה נבטל אותו, ואכן בבדיקה עם $forCheck = c[7]\oplus 1$ נקבל $c[7]\oplus 1\oplus c[7]\oplus 1\oplus 2$ כנדרש.

ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
0x2
```

שלב 11:

נכניס ללולאה ונהפוך הכל לכללי יותר עם תלות במיקום:

```
45  answer = [0, 0, 0, 0, 0, 0, 0, 0]
46  for i in range(8):
47      while True:
48          current = c[7-i]
49          newCurrent = (current + 1) % 256
50          c = c[:7-i] + bytes([newCurrent]) + c[8-i:]
51          result = oracle(c, key, IV)
52          if result:
53              break
54      pi = xor(1 + i, firstHalfEncrypted[7-i], c[7-i])
55      for j in range(i + 1):
56          c = c[:7-i+j] + bytes([xor(c[7-i+j], i+2, i+1)]) + c[8-i+j:]
57      answer[7-i] = pi
58  print(bytes(answer))
```

בשורה 45 נאתחל מערך לשמירת התשובות, שורות 48-53 מוכרות לנו מקודם (רק השם eight השתנה לcurrent שכן עכשיו זה לא השמיני), ב54, הגורם הראשון בxor שונה לi+1 שכן אנחנו כל פעם משנים את הבית שיתפענח לplaintext אחר, וב55 בכל פעם נצטרך לשנות את כל הבתים האחרונים שיתאימו לאיטרציה הבאה ונשמור את התשובה שלנו.

לבסוף נדפיס ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
b'rld\x05\x05\x05\x05\x05'
```

שלב 12:

ראשית נראה כי אכן אם "נמיר" את זה לבלוק הראשון זה יעבוד גם עליו:

```
40  blockSize = 8
41  firstHalf = b'\x00' * blockSize
42  firstHalfEncrypted = IV
43  secondHalf = encrypted[0:blockSize]
44  c = firstHalf + secondHalf
45  answer = [0, 0, 0, 0, 0, 0, 0, 0]
46  for i in range(8):
47      while True:
48          current = c[7-i]
49          newCurrent = (current + 1) % 256
50          c = c[:7-i] + bytes([newCurrent]) + c[8-i:]
51          result = oracle(c, key, IV)
52          if result:
53              break
54      pi = xor(1 + i, firstHalfEncrypted[7-i], c[7-i])
55      for j in range(i + 1):
56          c = c[:7-i+j] + bytes([xor(c[7-i+j], i+2, i+1)]) + c[8-i+j:]
57      answer[7-i] = pi
58  print(bytes(answer))
```

רק נחליף את מה שמצד ימין לשוויון במה שנחליף ונשמור על השמות אותו הדבר:

בשורה 42: החצי הראשון של הencrypted יהיה הIV שלנו,

בשורה 43: ניקח את החצי הראשון של הencrypted, והשאר נפעיל אותו הדבר על הc החדש שלנו ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
b'Hello Wo'
```

איזה כיף 😊 !!

בס״ד

כעת נכניס את זה לפונקציה כללית:

```python
def findBlock(previousBlock, currentBlock):
    c = b'\x00' * 8 + currentBlock
    answer = [0, 0, 0, 0, 0, 0, 0, 0]
    for i in range(8):
        while True:
            current = c[7-i]
            newCurrent = (current + 1) % 256
            c = c[:7-i] + bytes([newCurrent]) + c[8-i:]
            result = oracle(c, key, IV)
            if result:
                break
        pi = xor(1 + i, previousBlock[7-i], c[7-i])
        for j in range(i + 1):
            c = c[:7-i+j] + bytes([xor(c[7-i+j], i+2, i+1)]) + c[8-i+j:]
        answer[7-i] = pi
print(bytes(answer))
```

ונבדוק:

```python
findBlock(IV, encrypted[0:blockSize])
findBlock(encrypted[0:blockSize],encrypted[blockSize:blockSize*2])
```

ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
b'Hello Wo'
b'rld\x05\x05\x05\x05\x05'
```

כעת נעבור לכמעט סופי:

```python
blockSize = 8
BlocksLen = len(encrypted) // blockSize
blocks = [IV]
whole = b''
for i in range(BlocksLen):
    blocks.append(encrypted[i * 8:(i + 1) * 8])

for i in range(BlocksLen):
    whole = whole + findBlock(blocks[i], blocks[i + 1])
print(unpad(whole, blockSize).decode())
```

ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py
Hello world
```

וכעת כל שנותר זה לקבל את הארגומנטים מהטרמינל:

```python
encrypted = bytes.fromhex(sys.argv[1])
key = sys.argv[2].encode()
IV = bytes.fromhex(sys.argv[3])
```

ובגלל שרשמתי את אותם השמות, מכאן זה כבר אותו הדבר, והידד:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py 83e10d51e6d122ca3faf089c7a924a7b mydeskey 0000000000000000
Hello World
```

בס״ד

נבדוק בשביל הכיף בדיקה נוספת:

```python
from Cryptodome.Cipher import DES
from Cryptodome.Util.Padding import pad, unpad

def printHex(bytes):...

message = b'Hey bro whatsapp?'
padded = pad(message, 8)
key = b'mydeskew'
IV = b'\x00' * 8
cipher = DES.new(key, DES.MODE_CBC, IV)
encryptedBlocks = []
numOfBlocks = len(padded) // 8  # 2
for i in range(numOfBlocks):
    start = i * 8
    end = (i + 1) * 8
    block = padded[start:end]
    encryptedBlock = cipher.encrypt(block)
    encryptedBlocks.append(encryptedBlock)

encrypted = b''.join(encryptedBlocks)
print(encrypted.hex())
```

נריץ בשביל לראות את ההצפנה של זה:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 crypt.py
df8f489d9d172d6b2590f38bf226f9510e26a09e2001a637
```

נעתיק ונדביק לקוד שלנו ואכן:

```
C:\Users\Roy\Desktop\CS\NetSec\ex1>python3 ex1.py df8f489d9d172d6b2590f38bf226f9510e26a09e2001a637 mydeskew 000000000000
0000
Hey bro whatsapp?
```

😊😊😊😊😊😊😊😊😊😊