# Ciphertext CTF 2020

## Cryptography & Steganography

### Super_xor

**Description:**
before you encrypt something. make sure you can decrypt it. unless you want to lose it... we have encrypted the flag with this cipher, can you figure out how to decrypt it?

**Files:**

| | | |
|---|---|---|
| encrypted | Size: 0.06 KB | MD5: a205b0ca3b766c567d55fd86526ab506 |
| main | Size: 0.54 KB | MD5: 136b348184a63ddc4193850c47a95b71 |

**Hint!** Many people are stuck on deriving the key, although it should be the easiest step, the key you were dreaming about is "cnfdnf".

**Solution:**
First, lets figure out how to derive the key, to begin we must understand the whole encryption process, let's take a look at the $enc()$ function:

```
def enc(text,key):
    enc_flag = ''
    i = 0
    while i < len(text):
        x = ord(text[i])
        y = ord(key[i % len(key)])
        res = (x ^ y)
        y += i
        res = (res | y) - (res & y)
        y += i
        res = ~(res & y) & ~(~res & ~y)
        y += i
        res = (res & ~y) | (~res & y)
        y += i
        res = (res | y) & (~res | ~y)
        enc_flag += chr(res)
        enc_flag = enc_flag[::-1]
        i += 1
    return enc_flag
```

we can notice that all these Boolean expressions are other forms of XOR operation, so lets just substitute them to make it look nicer:

```
def enc(text,key):
    enc_flag = ''
    i = 0
    while i < len(text):
        x = ord(text[i])
        y = ord(key[i % len(key)])
        res = (x ^ y) ^ (y+i) ^ (y+2*i) ^ (y+3*i) ^ (y+4*i)
        enc_flag += chr(res)
        enc_flag = enc_flag[::-1]
        i += 1
    return enc_flag
```

We know that the flag starts with “CTCTF{“, but what is the position of these characters after encryption? Well as the length of the cipher text is 43, lets see how it is transposed:

```python
def transposition_test(text):
    enc_flag = ''
    i = 0
    while i < len(text):
        enc_flag += f' {i} '
        enc_flag = enc_flag[::-1]
        i += 1
    return enc_flag
print(transposition_test('0'*43))
```

output: “24 04 83 63 43 23 03 82 62 42 22 02 81 61 41 21 01 8 6 4 2 0 1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41”

The position of first 6 indexes are important for us to calculate the key, here we can see this correspondence:

| | | | | | |
|---|---|---|---|---|---|
| C | T | C | T | F | { |
| 21 | 22 | 19 | 23 | 18 | 24 |

Now we understand how the transposition is done, lets rearrange it (you can find the key without rearranging it, but we will need that function to make decryption easier):

```python
def rearrange(text):
    i = 0
    j = len(text) - 1
    rearranged_text = ''
    while i <= j:
        if i == j:
            rearranged_text += text[i]
            break
        rearranged_text += text[i]
        rearranged_text += text[j]
        i += 1
        j -= 1
    i = 0
    rearranged_text = rearranged_text[::-1]
    return rearranged_text
```

Hence, to get the key we brute force for each possible character of the key to see which character encrypts ‘C’ to get the 1st byte (which was 21st byte before rearrangement) of the cipher text. Do the same for the rest of “CTCTF{“ and you will get the key:

```python
CT = rearrange(open('encrypted','rb').read().decode().strip())
for z,x in enumerate('CTCTF{'):
    x = ord(x)
    for y in printable:
        y = ord(y)
        if z==0 and chr(x)=='C' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[0])):
             print(chr(y),end='')
        elif z==1 and chr(x)=='T' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[1])):
             print(chr(y),end='')
        elif z==2 and chr(x)=='C' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[2])):
             print(chr(y),end='')
        elif z==3 and chr(x)=='T' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[3])):
             print(chr(y),end='')
        elif z==4 and chr(x)=='F' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[4])):
             print(chr(y),end='')
        elif z==5 and chr(x)=='{' and ((x^y)^(y+z)^(y+2*z)^(y+3*z)^(y+4*z)==ord(CT[5])):
             print(chr(y),end='')
```

output: “cnfdnf”.

Now we have the key, what is left is just to decrypt it and Lo! There it is:

```python
def dec(text,key):
    dec_flag = ''
    i = 0
    while i < len(text):
        x = ord(text[i])
        y = ord(key[i % len(key)])
        res = (x^y)^(y+i)^(y+2*i)^(y+3*i)^(y+4*i)
        dec_flag += chr(res)
        i += 1
    return dec_flag

CT = rearrange(open('encrypted','rb').read().decode().strip())
print(dec(CT, 'cnfdnf'))
```

output: “CTCTF{xor_0p3ra71on_ha$_m@ny_b0ole@n_f0rms}”.