Obscure - Reversing

Author : Aidenpearce369

Lets download the challenge file and analyse it,

```
horus@moni:~/TamilCTFpyc$ls
reverseme
horus@moni:~/TamilCTF/pyc$ file reverseme
reverseme: python 2.7 byte-compiled
```

So it is a `byte-code` file from `python 2.7`,

Lets try decompiling it into its own source code by `uncompyle6`,

```
horus@moni:~/TamilCTF/pyc$ uncompyle6 -o . reverseme


# file reverseme
# path reverseme must point to a Python source that can be compiled, or Python
bytecode (.pyc, .pyo)

reverseme --
# decompile failed
```

So it is expecting for `.pyc` or `.pyo` file

Lets rename it,

```
horus@moni:~/TamilCTF/pyc$ ls
reverseme
horus@moni:~/TamilCTF/pyc$ mv reverseme reverseme.pyc
horus@moni:~/TamilCTF/pyc$ ls
reverseme.pyc
```

Lets decompile it now,

```
horus@moni:~/TamilCTF/pyc$ uncompyle6 -o . reverseme.pyc
reverseme.pyc --
# Successfully decompiled file
```

Lets view the decompiled python file,

```
horus@moni:~/TamilCTF/pyc$ cat reverseme.py
# uncompyle6 version 3.7.4
# Python bytecode 2.7 (62211)
# Decompiled from: Python 2.7.18 (default, Mar  8 2021, 13:02:45)
# [GCC 9.3.0]
# Embedded file name: reverseme.py
# Compiled at: 2021-09-04 14:51:21
import numpy as np
flag = 'TamilCTF{this_one_is_a_liability_dont_fall_for_it}'
np.random.seed(369)
data = np.array([ ord(c) for c in flag ])extra = np.random.randint(1, 5,
len(flag))
product = np.multiply(data, extra)
temp1 = [ x for x in data ]
temp2 = [ ord(x) for x in 'dondaVSclb' * 5 ]
c = [ temp1[i] ^ temp2[i] for i in range(len(temp1)) ]
flagdata = ('').join(hex(x)[2:].zfill(2) for x in c)
real_flag =
'300e030d0d1507251700361a3a0127662120093d551c311029330c53022e1d3028541315363c5e3d0
63d0b250a090c52021f'
```

So in this python script an obfuscated output is produced from the string

And our string is a fake flag

But the output is produced from each characters in the flag

And there is an obfuscated value for the real flag, we have to just reverse it to its characters

By checking the output for the string `TamilCTF{` it gives the obfuscated value as `300e030d0d15072517` which is also present in `real_flag`

All we have to do is perform the obfuscation mechanism for each character and compare it with the `real_flag` , which is moreover a bruteforce attack

Solve script for this challenge,

```
import numpy as np
import string

final_flag='300e030d0d1507251700361a3a0127662120093d551c311029330c53022e1d30285413
15363c5e3d063d0b250a090c52021f'
def encrypt(flag):
    np.random.seed(369)
    data=np.array([ord(c) for c in flag])
    extra=np.random.randint(1,5,len(flag))
    product=np.multiply(data,extra)
    temp1=[x for x in data]
    temp2=[ord(x) for x in "dondaVSclb"*5]
    c=[temp1[i]^temp2[i] for i in range(len(temp1))]
    flagdata=''.join(hex(x)[2:].zfill(2) for x in c)
    return flagdata

flag=list('TamilCTF{')
while(True):
    for character in string.printable:
        out=encrypt(''.join(flag)+character)
        if final_flag.startswith(out):
            flag.append(character)
            print("Found one character "+"".join(flag))
        if final_flag==out:
            exit()
```

By running our `solve.py` we will get our original flag

```
horus@moni:~/TamilCTF$ python3 solve.py
Found one character TamilCTF{b
Found one character TamilCTF{bR
Found one character TamilCTF{bRu
Found one character TamilCTF{bRuT
Found one character TamilCTF{bRuTe
Found one character TamilCTF{bRuTeF
Found one character TamilCTF{bRuTeF0
Found one character TamilCTF{bRuTeF0r
Found one character TamilCTF{bRuTeF0rC
Found one character TamilCTF{bRuTeF0rCe
Found one character TamilCTF{bRuTeF0rCe_
Found one character TamilCTF{bRuTeF0rCe_1
Found one character TamilCTF{bRuTeF0rCe_1s
Found one character TamilCTF{bRuTeF0rCe_1s_
Found one character TamilCTF{bRuTeF0rCe_1s_t
Found one character TamilCTF{bRuTeF0rCe_1s_tH
Found one character TamilCTF{bRuTeF0rCe_1s_tHe
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0n
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nL
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0r
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rC
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_b
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bR
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bRe
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReA
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk_
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk__
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk__1
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk__1n
Found one character TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk__1n}
```

Our final original flag for this challenge is `TamilCTF{bRuTeF0rCe_1s_tHe_0nLy_F0rCe_2_bReAk__1n}`