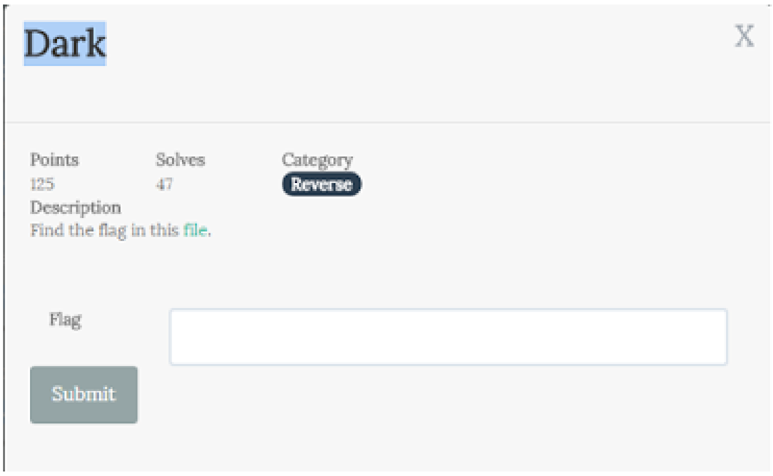
11th May 2015

ASIS 2015 Quals. Dark, reversing 125 points

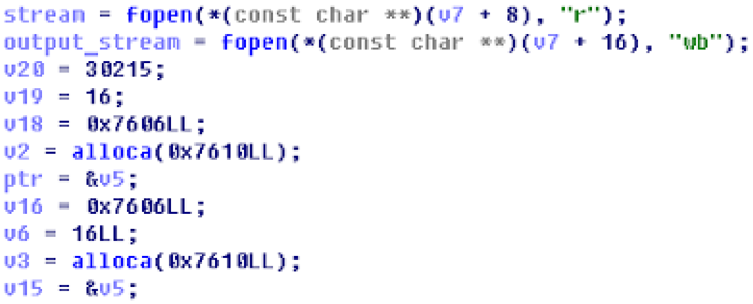
Category: Reversing
Points: 125
Solves: 47
Description:



Points 125
Solves 47
Category Reverse
Description
Find the flag in this file.

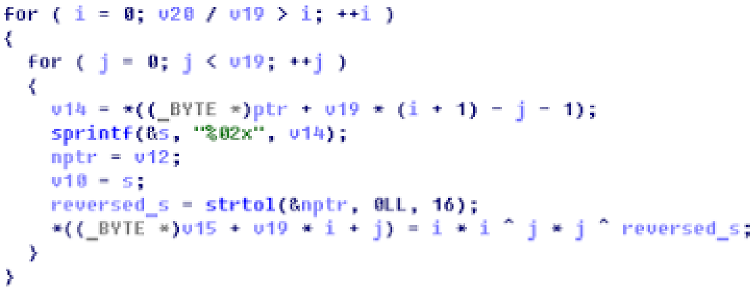
[http://3.bp.blogspot.com/-bTU4wy3P8Bc/VVDIILNu8DI/AAAAAAAAAz8/on0I-Zs2TPo/s1600/1.png]

Good task for morning work-out. There are ELF file as encryptor and encrypted flag.enc. Size of the last one is 30215 bytes.
Binary takes 2 parameters - input file to encrypt and output file.
So look at the code:



```
stream = fopen(*(const char **)(v7 + 8), "r");
output_stream = fopen(*(const char **)(v7 + 16), "wb");
v20 = 30215;
v19 = 16;
v18 = 0x7606LL;
v2 = alloca(0x7610LL);
ptr = &v5;
v16 = 0x7606LL;
v6 = 16LL;
v3 = alloca(0x7610LL);
v15 = &v5;
```

[http://4.bp.blogspot.com/-YPLc1yJb50Y/VVDJG_7yiXI/AAAAAAAAA0E/e-voduMTjOg/s1600/2.png]
Just open file and read it to array of 30215 bytes size. It means if we will enrypt file of length < 30215 it will think it has 30215 length with zero padding.
Next piece of code shows us the algorithm:

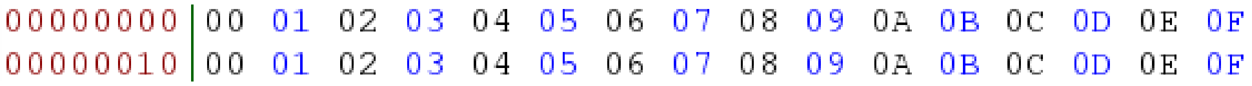


```
for ( i = 0; v20 / v19 > i; ++i )
{
  for ( j = 0; j < v19; ++j )
  {
    v14 = *((_BYTE *)ptr + v19 * (i + 1) - j - 1);
    sprintf(&s, "%02x", v14);
    nptr = v12;
    v10 = s;
    reversed_s = strtol(&nptr, 0LL, 16);
    *((_BYTE *)v15 + v19 * i + j) = i * i ^ j * j ^ reversed_s;
  }
}
```

[http://2.bp.blogspot.com/-OmaoZdKoL08/VVDJ2YgOonI/AAAAAAAAA0M/KR864bmzMUY/s1600/3.png]
Oh.
In pseudoalgorithm it looks like:
for every 16 byte length row:

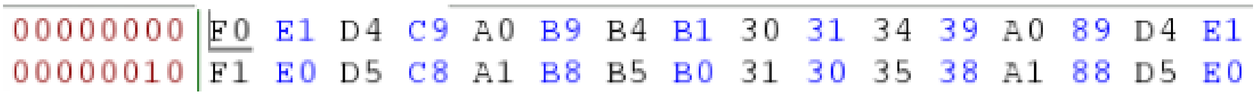
```
take string and reverse it
in every byte of reversed string:
    reverse byte and xor it with row_number*row_number and with col_number*col_number
```

So what we have two 16 byte strings 0x00 .. 0x0F as input?



```
00000000|00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
00000010|00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F
```

[http://2.bp.blogspot.com/-uWczLBga-II/VVDLgnQ_QzI/AAAAAAAAA0c/Ov73GOp3dBE/s1600/4.png]

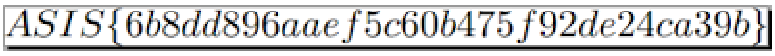


```
00000000|F0 E1 D4 C9 A0 B9 B4 B1 30 31 34 39 A0 89 D4 E1
00000010|F1 E0 D5 C8 A1 B8 B5 B0 31 30 35 38 A1 88 D5 E0
```

[http://1.bp.blogspot.com/-3_YHksfJsrw/VVDLgjFjumI/AAAAAAAAA0Y/JjMZ1dnaYAQ/s1600/5.png]

Algorithm is certain simple for reverse engineer to reverse it back. Take python for it :)
(See script in the end)

Decrypted file is PDF which contains the flag:



ASIS{6b8dd896aaef5c60b475f92de24ca39b}

[http://3.bp.blogspot.com/-ZSZJ3VnL7HU/VVDM-7rrjyI/AAAAAAAAA0s/tvn38VICCbY/s1600/6.png]

Python script solve.py

```
def rev(byte):
 s = "%02x" % byte
 s2 = s[1] + s[0]
 return int(s2, 16)

f = open('flag.enc','rb')
enc = [0] * 30215
for i in range (30215):
 enc[i] = ord(f.read(1))
dec = [0] * 30215
f.close

for i in range(30215/16):
 for j in range(16):
  print " [*] starting with byte", hex(enc[16*i + j])
  r_byte = (i*i ^ j*j ^ enc[16*i + j]) & 0xff
  print " [+] got reversed byte", hex(r_byte)
  n_byte = rev(r_byte)
  print " [+] result byte is", hex(n_byte), "\n"
  dec[16*(i+1) - j - 1] = n_byte

f = open('test.bin.enc.dec','wb')
for i in range(30215):
 f.write(chr(dec[i]))
f.close()
```

Опубликовано 11th May 2015 пользователем Кирилл Шипулин