buuctf easyRE

原创

菜逼的ctf之路 于 2020-10-18 10:37:22 发布 592 收藏

buuctf easyRE[2019红帽杯]

简单总结一下，一般说是easy的都不简单
加载ida查看字符串，找到主要函数，

```
97  v52 = 76;
98  v53 = 64;
99  v54 = 69;
00  v55 = 67;
01  memset(v56, 0, sizeof(v56));
02  v57 = 0;
03  v58 = 0;
04  v3 = v56;
05  sub_4406E0(0LL, (__int64)v56);
06  v58 = 0;
07  v4 = v56;
08  LODWORD(v5) = sub_424BA0((const __m128i *)v56);
09  if ( v5 == 36 )
10  {
11    for ( i = 0; ; ++i )
12    {
13      v4 = v56;
14      LODWORD(v7) = sub_424BA0((const __m128i *)v56);
15      if ( i >= v7 )
16        break;
17      if ( (unsigned __int8)(v56[i] ^ i) != *(&v20 + i) )
18      {
19        result = -2;
20        goto LABEL_13;
21      }
22    }
23    sub_410CC0((const __m128i *)"continue!");
24    memset(&v59, 0, 0x40uLL);
25    v61 = 0;
26    v3 = &v59;
27    sub_4406E0(0LL, (__int64)&v59);
28    v60 = 0;
29    v4 = &v59;
30    LODWORD(v8) = sub_424BA0((const __m128i *)&v59);
31    if ( v8 == 39 )
```

00000AC1 main:97 (400AC1)

```
LODWORD(v8) = sub_424BA0((const __m128i *)&v59);
if ( v8 == 39 )
{
  v9 = (const __m128i *)sub_400E44((const __m128i *)&v59);
  v10 = (const __m128i *)sub_400E44(v9);
  v11 = (const __m128i *)sub_400E44(v10);
  v12 = (const __m128i *)sub_400E44(v11);
  v13 = (const __m128i *)sub_400E44(v12);
  v14 = (const __m128i *)sub_400E44(v13);
  v15 = (const __m128i *)sub_400E44(v14);
  v16 = (const __m128i *)sub_400E44(v15);
  v17 = (const __m128i *)sub_400E44(v16);
  v18 = sub_400E44(v17);
  v3 = off_6CC090;
  v4 = (char *)v18;
  if ( !(unsigned int)sub_400360(v18, (__int64)off_6CC090) )
  {
    sub_410CC0((const __m128i *)"You found me!!!");
    v4 = "bye bye~";
    sub_410CC0((const __m128i *)"bye bye~");
  }
  result = 0;
}
else
{
  result = -3;
}
```


第一组数据解出

Info:The first four chars are flag

然后看下面，打开函数发现是base64加密，加密十次，然后比较，把字符串解密十次，发现给了一个网址(不好意思，网址给关了，不想再解密10次了，就口头描述一下)，打开网址，发现是...

然后看大佬博客，才知道是个迷惑我们的函数，真正的核心函数在上图所示函数的下一个(wdnmd)口吐芬芳

```
__int64 v2; // rdi
__int64 result; // rax
unsigned __int64 v4; // rt1
unsigned int v5; // [rsp+Ch] [rbp-24h]
signed int i; // [rsp+10h] [rbp-20h]
signed int j; // [rsp+14h] [rbp-1Ch]
unsigned int v8; // [rsp+24h] [rbp-Ch]
unsigned __int64 v9; // [rsp+28h] [rbp-8h]

v9 = __readfsqword(0x28u);
v2 = 0LL;
v5 = sub_43FD20(0LL) - qword_6CEE38;
for ( i = 0; i <= 1233; ++i )
{
  v2 = v5;
  sub_40F790(v5);
  sub_40FE60();
  sub_40FE60();
  v5 = (unsigned __int64)sub_40FE60() ^ 0x98765432;
}
v8 = v5;
if ( ((unsigned __int8)v5 ^ byte_6CC0A0[0]) == 102 && (HIBYTE(v8) ^ (unsigned __int8)byte_6CC0A3) == 103 )
{
  for ( j = 0; j <= 24; ++j )
  {
    v2 = (unsigned __int8)(byte_6CC0A0[j] ^ *((_BYTE *)&v8 + j % 4));
    sub_410E90(v2);
  }
}
v4 = __readfsqword(0x28u);
result = v4 ^ v9;
if ( v4 != v9 )
  sub_444020(v2, a2);
return result;
}
```


然后利用"flag"字符串把v5解出来，然后把给定字符串依次异或就行了

最后贴上脚本

```
s='flag'
a=[
    0x40, 0x35, 0x20, 0x56, 0x5D, 0x18, 0x22,
    0x45, 0x17, 0x2F, 0x24, 0x6E, 0x62, 0x3C,
    0x27, 0x54, 0x48, 0x6C, 0x24, 0x6E, 0x72,
    0x3C, 0x32, 0x45, 0x5B, 0x00
]
key=''
flag=''
for i in range(4):
    key+=chr(ord(s[i])^a[i])

for i in range(len(a)):
    flag+=chr(ord(key[i%4])^a[i])
print(flag)
```

最后的结果各位自己来吧！我就不写了
结束