BUUCTF Reverse/[ACTF新生赛2020]SoulLike

原创

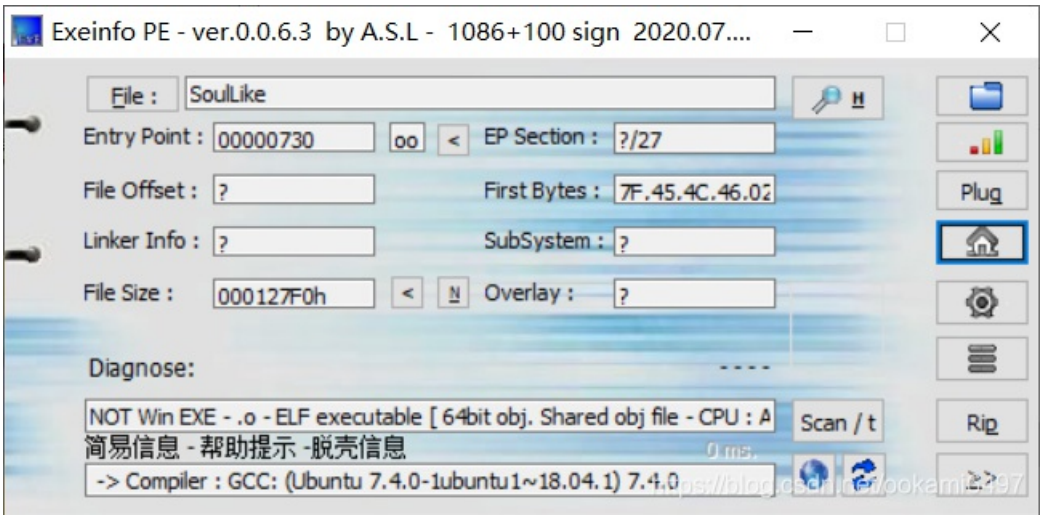
这就是强者的世界么 于 2021-08-16 21:07:51 发布 81 收藏

分类专栏： # BUUCTF Reverse

本文链接：https://blog.csdn.net/ookami6497/article/details/119739530

BUUCTF Reverse/[ACTF新生赛2020]SoulLike

> 看下文件信息，没有加壳



IDA64位打开，还是字符串比较题目

```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  __int64 result; // rax
  char v5; // [rsp+7h] [rbp-B9h]
  int i; // [rsp+8h] [rbp-B8h]
  int j; // [rsp+Ch] [rbp-B4h]
  int v8[14]; // [rsp+10h] [rbp-B0h] BYREF
  char v9[110]; // [rsp+4Ah] [rbp-76h] BYREF
  unsigned __int64 v10; // [rsp+B8h] [rbp-8h]

  v10 = __readfsqword(0x28u);
  printf("input flag:");
  scanf("%s", &v9[6]);
  strcpy(v9, "actf{");
  v5 = 1;
  for ( i = 0; i <= 4; ++i )
  {
    if ( v9[i] != v9[i + 6] )
    {
      v5 = 0;
      goto LABEL_6;
    }
  }
  if ( !v5 )
    goto LABEL_16;
LABEL_6:
  for ( j = 0; j <= 11; ++j )
    v8[j] = v9[j + 11];
  if ( (unsigned __int8)sub_83A(v8) && v9[23] == 125 )
  {
    printf("That's true! flag is %s", &v9[6]);
    result = 0LL;
  }
  else
  {
LABEL_16:
    printf("Try another time...");
    result = 0LL;
  }
  return result;
}
```

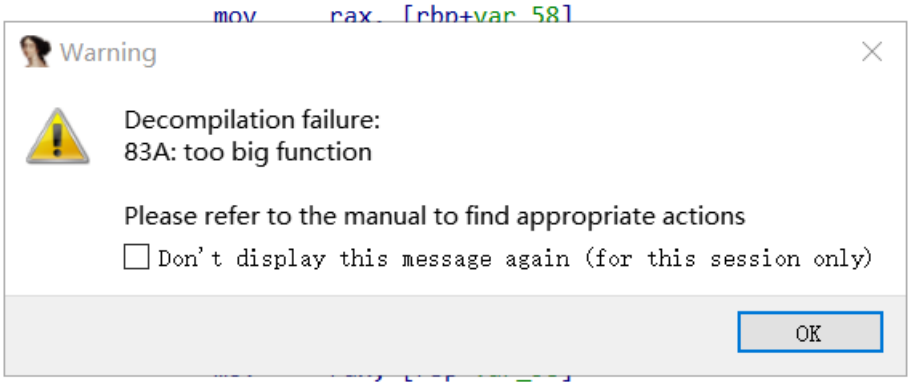
> 逻辑很简单，是以 actf{} 包裹的flag，且括号里面的字符为12位

> 这里就是输出flag的条件了

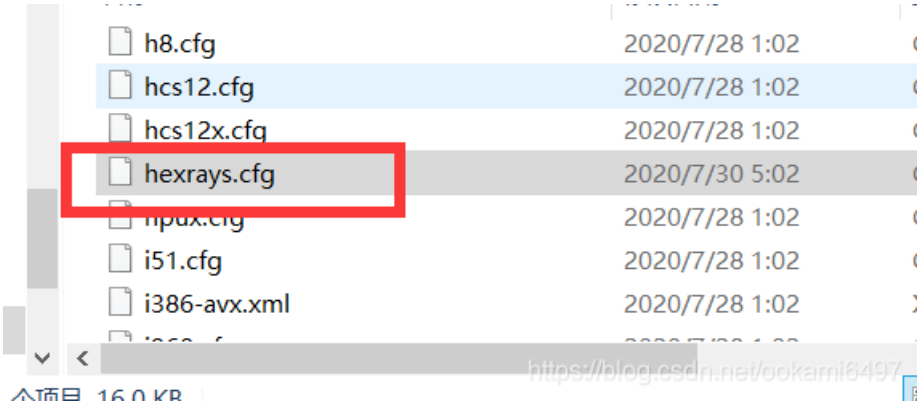
```
if ( (unsigned __int8)sub_83A(v8) && v9[23] == 125 )
  {
    printf("That's true! flag is %s", &v9[6]);
    result = 0LL;
  }
```

> `sub_83A(v8)` 对括号里面的12位字符进行变换，但在跟进的时候出了个小问题，说啥太大了

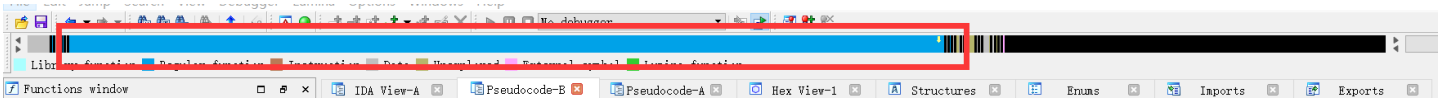
```
79F4                mov     edx, [rax]
79F6                mov     rax, [rbp+var_58]
79FA
79FE
7A01
7A03
7A07
7A0B
7A0D
7A11
7A15
7A18
7A1A
7A1E                add     rax, 10h
7A22                mov     ecx, [rax]
7A24                mov     rax, [rbp+var_58]
```

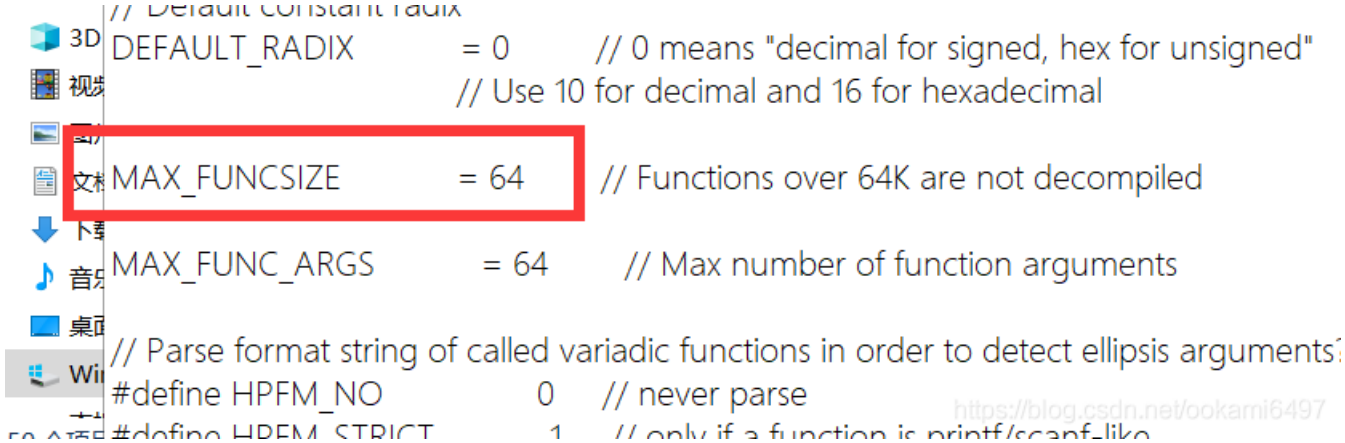


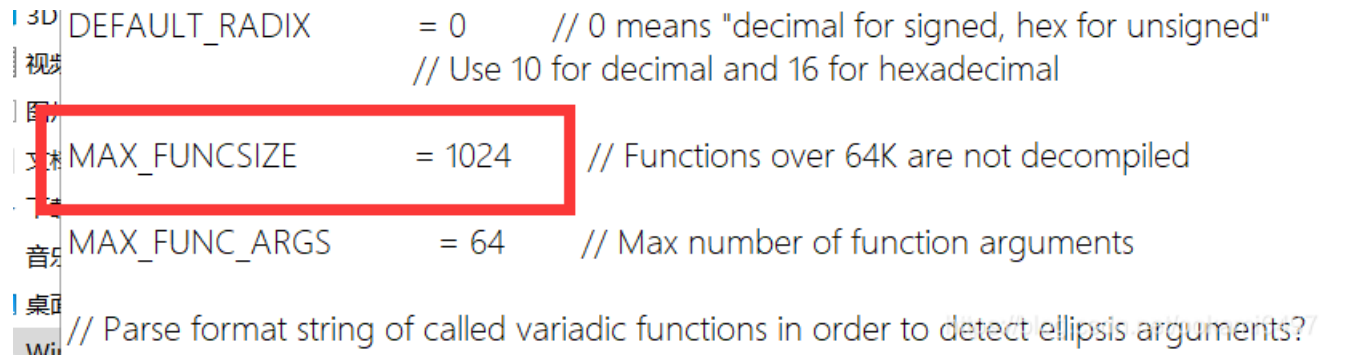
百度了一下，可以修改配置文件IDA 7.0\cfg\hexrays.cfg来解决



可编译的最大函数大小为64k，然后改成1024（这个随意，，不过不知道具体要改大多少，这函数确实挺大的，，一整个蓝条都是）







修改后就能正常反编译了，，，函数是在是太太太长了，完整的就不贴了，贴个开头和结尾意思意思。

```
__int64 __fastcall sub_83A(_DWORD *a1)
{
  int i; // [rsp+1Ch] [rbp-44h]
  int v3[14]; // [rsp+20h] [rbp-40h]
  unsigned __int64 v4; // [rsp+58h] [rbp-8h]

  v4 = __readfsqword(0x28u);
  *a1 ^= 0x2Bu;
  a1[1] ^= 0x6Cu;
  a1[2] ^= 0x7Eu;
  a1[3] ^= 0x56u;
  a1[4] ^= 0x39u;
  a1[5] ^= 3u;
  a1[6] ^= 0x2Du;
  a1[7] ^= 0x28u;
  a1[8] ^= 8u;
  ++a1[9];
  a1[10] ^= 0x2Fu;
  a1[11] ^= 0xAu;
  ++*a1;
  a1[1] ^= 0xDu;
  a1[2] ^= 0x73u;
  a1[3] ^= a1[2];
  a1[4] ^= 0x37u;
  ++a1[5];
  a1[6] ^= 0x69u;
  a1[7] ^= 0x59u;
  a1[8] ^= 0xCu;
  a1[9] ^= 0x70u;
  ++a1[10];
  a1[11] ^= 0x1Fu;
  ++*a1;
```

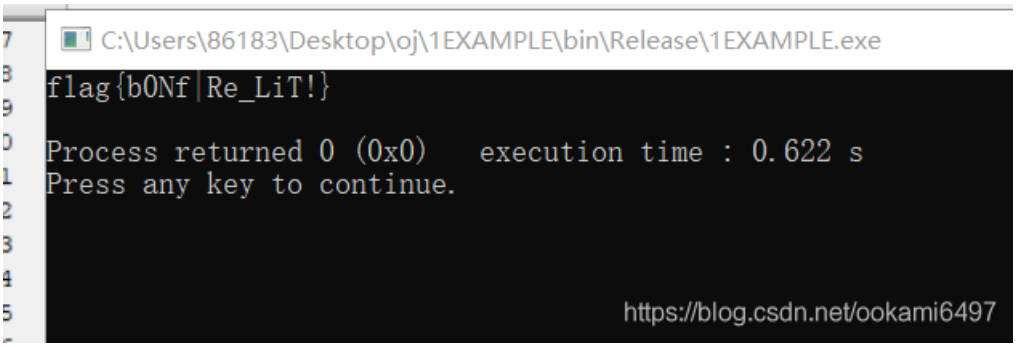
结尾这里，变换后的字符串要等于v3，不然输出wrong

```
  a1[10] ^= 0x29u;
  a1[11] ^= 0x3Bu;
  v3[0] = 126;
  v3[1] = 50;
  v3[2] = 37;
  v3[3] = 88;
  v3[4] = 89;
  v3[5] = 107;
  v3[6] = 53;
  v3[7] = 110;
  v3[8] = 0;
  v3[9] = 19;
  v3[10] = 30;
  v3[11] = 56;
  for ( i = 0; i <= 11; ++i )
  {
    if ( v3[i] != a1[i] )
    {
      printf("wrong on #%d\n", (unsigned int)i);
      return 0LL;
    }
  }
  return 1LL;
}
00010FA2 sub 83A:3007 (10FA2)
```


显而易见只能爆破了，太长了，只贴主要的

```c
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
int v[]={126,50,37,88,89,107,53,110,0,19,30,56};
int f(char *a1,int i)
{
  *a1 ^= 0x2Bu;
  a1[1] ^= 0x6Cu;
  a1[2] ^= 0x7Eu;
  a1[3] ^= 0x56u;
  ...
  ...
  ...
   a1[9] ^= 0x70u;
  a1[10] ^= 0x29u;
  a1[11] ^= 0x3Bu;
  if(a1[i] == v[i])
     return 1;
  else
    return 0;
}
int main()
{
    int i,j,k;
    char flag[13] = "";
    char temp[13] = "";
    for(i = 0; i < 12 ; i++)
    {
        for(k = 33; k <= 126 ; k++)
        {
            strcpy(flag,temp);
            flag[i] = k;
            if(f(flag,i))
            {
                temp[i] = k;
                break;
            }
        }
    }
    printf("flag{%s}\n",temp);
    return 0;
}
```

运行结果



flag{b0Nf|Re_LiT!}