攻防世界 Reverse高手进阶区 2分题 re2-cpp-is-awesome

原创

思源湖的鱼 于 2020-12-16 13:57:39 发布 212 收藏

分类专栏： ctf 文章标签： ctf 攻防世界 reverse

本文链接：https://blog.csdn.net/weixin_44604541/article/details/111264462

ctf 专栏收录该内容

200 篇文章 23 订阅

订阅专栏

前言

继续ctf的旅程

攻防世界Reverse高手进阶区的2分题

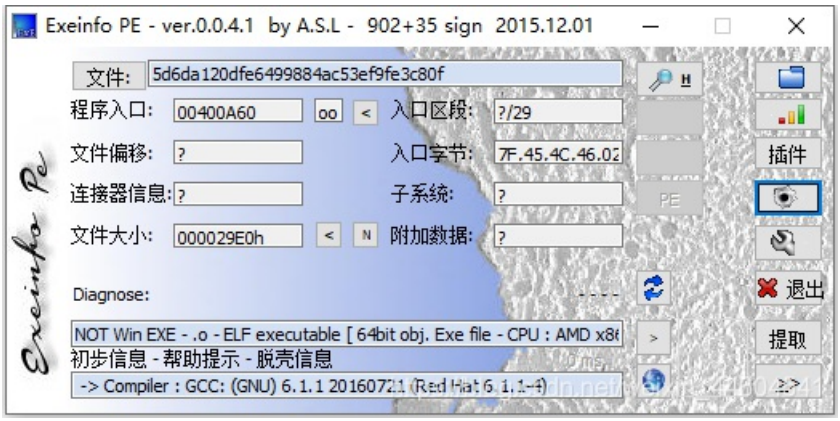
本篇是re2-cpp-is-awesome的writeup

发现攻防世界的题目分数是动态的

就仅以做题时的分数为准了

解题过程

PE查壳



扔进IDA

```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  char *v3; // rbx
  __int64 v4; // rax
  __int64 v5; // rdx
  __int64 v6; // rax
  __int64 v7; // rdx
  _BYTE *v8; // rax
  __int64 i; // [rsp+10h] [rbp-60h]
  char v11; // [rsp+20h] [rbp-50h]
  char v12; // [rsp+4Fh] [rbp-21h]
  __int64 v13; // [rsp+50h] [rbp-20h]
  int v14; // [rsp+5Ch] [rbp-14h]

  if ( a1 != 2 )
  {
    v3 = *a2;
    v4 = std::operator<<<std::char_traits<char>>(&std::cout, "Usage: ", a3);
    v6 = std::operator<<<std::char_traits<char>>(v4, v3, v5);
    std::operator<<<std::char_traits<char>>(v6, " flag\n", v7);
    exit(0);
  }
  std::allocator<char>::allocator(&v12, a2, a3);
  std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::basic_string(&v11, a2[1], &v12);
  std::allocator<char>::~allocator(&v12);
  v14 = 0;
  for ( i = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::begin(&v11); ; sub_400D7A(&i) )
  {
    v13 = std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::end(&v11);
    if ( !(unsigned __int8)sub_400D3D(&i, &v13) )
      break;
    v8 = (_BYTE *)sub_400D9A(&i);
    if ( *v8 != off_6020A0[dword_6020C0[v14]] )
      sub_400B56();
    ++v14;
  }
  sub_400B73();
  std::__cxx11::basic_string<char,std::char_traits<char>,std::allocator<char>>::~basic_string(&v11);
  return 0LL;
}
```


瞅着有点眼花缭乱

仔细看就是for循环里有东西

```
_QWORD *__fastcall sub_400D7A(_QWORD *a1)
{
  ++*a1;
  return a1;
}
```

```
.data:00000000006020A0 off_6020A0      dq offset aL3tMeT3llY0uS0
.data:00000000006020A0                                         ; DATA XREF: main+D1↑r
.data:00000000006020A0                                         ; "L3t_ME_T3ll_Y0u_S0m3th1ng_1mp0rtant_A_{"...
.data:00000000006020A8                 align 20h
.data:00000000006020C0 ; int dword_6020C0[]
.data:00000000006020C0 dword_6020C0    dd 24h                  ; DATA XREF: main+DD↑r
```

```
.data:00000000006020C0 dword_6020C0    dd 24h                  ; DATA XREF: main+DD↑r
.data:00000000006020C4                 align 8
.data:00000000006020C8                 db    5
.data:00000000006020C9                 db    0
.data:00000000006020CA                 db    0
.data:00000000006020CB                 db    0
.data:00000000006020CC                 db  36h ; 6
.data:00000000006020CD                 db    0
.data:00000000006020CE                 db    0
.data:00000000006020CF                 db    0
.data:00000000006020D0                 db  65h ; e
.data:00000000006020D1                 db    0
.data:00000000006020D2                 db    0
.data:00000000006020D3                 db    0
.data:00000000006020D4                 db    7
.data:00000000006020D5                 db    0
.data:00000000006020D6                 db    0
.data:00000000006020D7                 db    0
.data:00000000006020D8                 db  27h ; '
.data:00000000006020D9                 db    0
.data:00000000006020DA                 db    0
.data:00000000006020DB                 db    0
.data:00000000006020DC                 db  26h ; &
.data:00000000006020DD                 db    0
.data:00000000006020DE                 db    0
.data:00000000006020DF                 db    0
.data:00000000006020E0                 db  2Dh ; -
.data:00000000006020E1                 db    0
.data:00000000006020E2                 db    0
.data:00000000006020E3                 db    0
.data:00000000006020E4                 db    1
.data:00000000006020E5                 db    0
.data:00000000006020E6                 db    0
.data:00000000006020E7                 db    0
.data:00000000006020E8                 db    3
.data:00000000006020E9                 db    0
.data:00000000006020EA                 db    0
.data:00000000006020EB                 db    0
.data:00000000006020EC                 db    0
.data:00000000006020ED                 db    0
.data:00000000006020EE                 db    0
.data:00000000006020EF                 db    0
.data:00000000006020F0                 db  0Dh
.data:00000000006020F1                 db    0
.data:00000000006020F2                 db    0
.data:00000000006020F3                 db    0
.data:00000000006020F4                 db  56h ; V
```


看了看就是

一个循环

提取off_6020A0字符串中以dword_6020C0为索引的字符

那就简单了

```
S = 'L3t_ME_T3ll_Y0u_S0m3th1ng_1mp0rtant_A_{FL4G}_W0nt_b3_3X4ctly_th4t_345y_t0_c4ptur3_H0wev3r_1T_w1ll_b3_C00l_'

N = [0x24,0x0,0x5,0x36,0x65,0x7,0x27,0x26,0x2d,0x1,0x3,0x0,0x0d,0x56,0x1,0x3,0x65,0x3,0x2d,0x16,0x2,0x15,0x3,0x65,0x0,0x29,0x44,0x44,0x1,0x44,0x2b]

flag = ''

for i in N:
    flag  += S[i]

print(flag)
```

得到flag：`ALEXCTF{W3_L0v3_C_W1th_CL45535}`

结语

cpp的尝试