[BUUCTF]REVERSE——[ACTF新生赛2020]Oruga

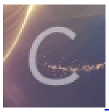
原创

Angel~Yan 于 2021-01-09 14:02:45 发布 134 收藏

分类专栏： BUUCTF刷题记录 REVERSE

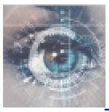
本文链接：https://blog.csdn.net/mcmuyanga/article/details/112390335

版权


BUUCTF刷题记录 同时被 2 个专栏收录

198 篇文章 14 订阅

订阅专栏


REVERSE

75 篇文章 1 订阅

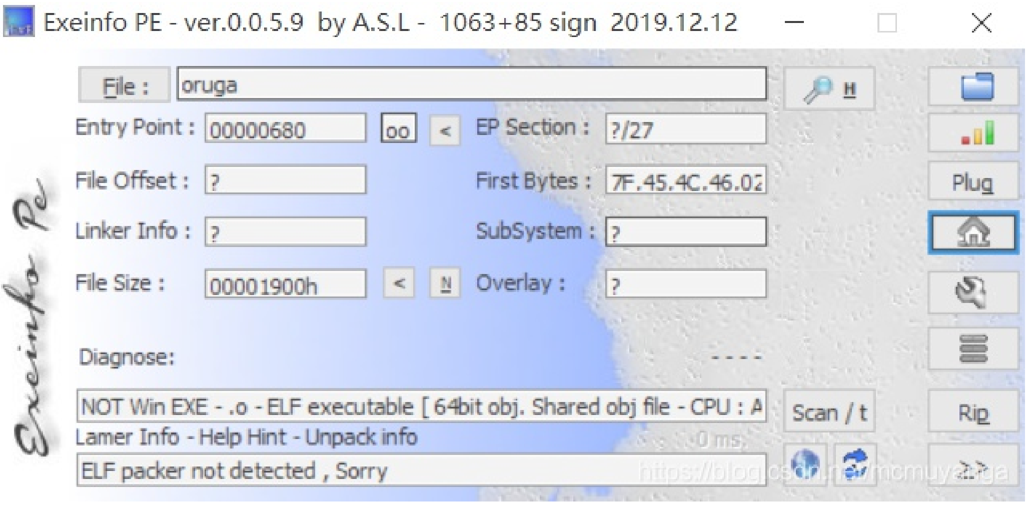
订阅专栏

[ACTF新生赛2020]Oruga

附件

步骤：

1. 例行检查，64位程序，无壳



Exeinfo PE - ver.0.0.5.9 by A.S.L - 1063+85 sign 2019.12.12

File : oruga

Entry Point : 00000680

EP Section : ?/27

File Offset : ?

First Bytes : 7F.45.4C.46.02

Linker Info : ?

SubSystem : ?

File Size : 00001900h

Overlay : ?

Diagnose:

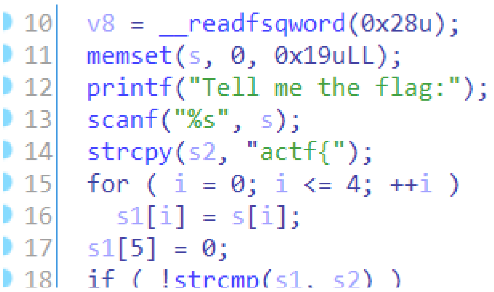
NOT Win EXE - .o - ELF executable [64bit obj. Shared obj file - CPU : A

Lamer Info - Help Hint - Unpack info

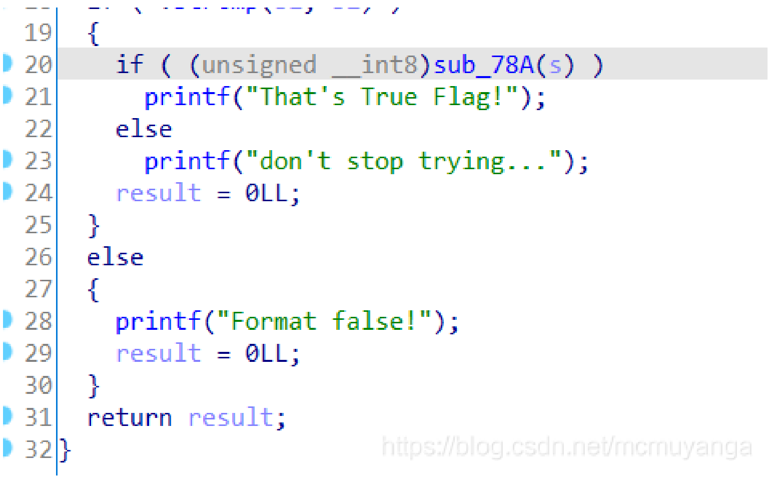
0 ms.

ELF packer not detected , Sorry

2. 64位ida载入，检索字符串，根据提示来到关键函数

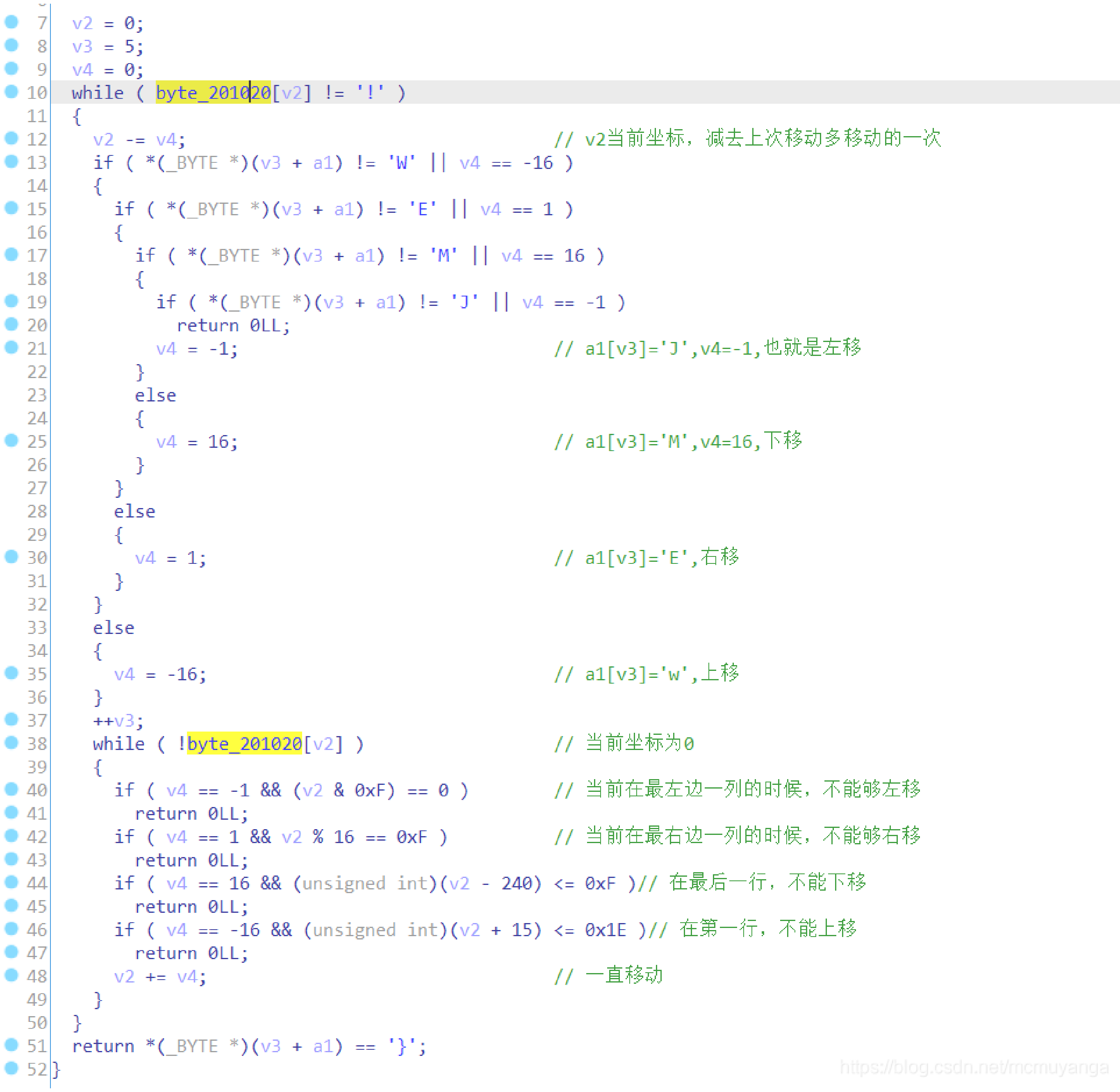


```
v8 = __readfsqword(0x28u);
memset(s, 0, 0x19uLL);
printf("Tell me the flag:");
scanf("%s", s);
strcpy(s2, "actf{");
for ( i = 0; i <= 4; ++i )
  s1[i] = s[i];
s1[5] = 0;
if ( !strcmp(s1, s2) )
```



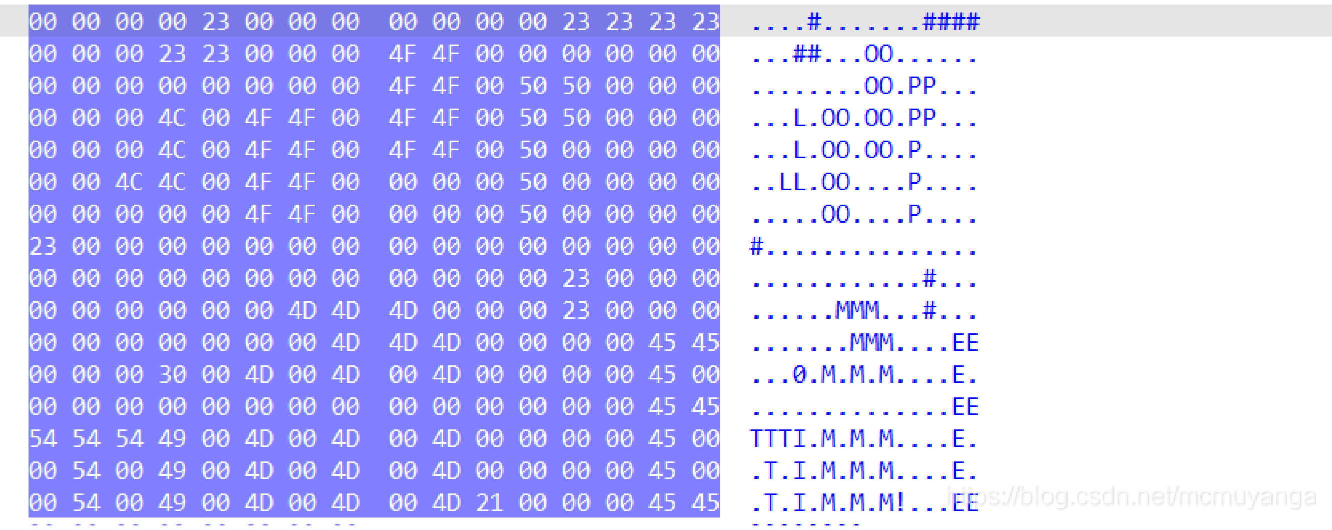
```
19  {
20    if ( (unsigned __int8)sub_78A(s) )
21      printf("That's True Flag!");
22    else
23      printf("don't stop trying...");
24    result = 0LL;
25  }
26  else
27  {
28    printf("Format false!");
29    result = 0LL;
30  }
31  return result;
32}
```

14行~18行就是让字符串的前5位是 actf{，sub_78A（）是关键函数，分析可知应该是迷宫



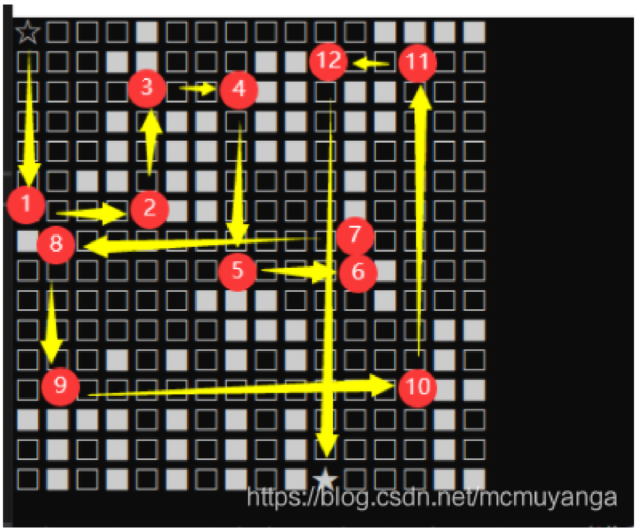
```
  v2 = 0;
  v3 = 5;
  v4 = 0;
  while ( byte_201020[v2] != '!' )
  {
    v2 -= v4;                                   // v2当前坐标，减去上次移动多移动的一次
    if ( *(_BYTE *)(v3 + a1) != 'W' || v4 == -16 )
    {
      if ( *(_BYTE *)(v3 + a1) != 'E' || v4 == 1 )
      {
        if ( *(_BYTE *)(v3 + a1) != 'M' || v4 == 16 )
        {
          if ( *(_BYTE *)(v3 + a1) != 'J' || v4 == -1 )
            return 0LL;
          v4 = -1;                              // a1[v3]='J',v4=-1,也就是左移
        }
        else
        {
          v4 = 16;                              // a1[v3]='M',v4=16,下移
        }
      }
      else
      {
        v4 = 1;                                 // a1[v3]='E',右移
      }
    }
    else
    {
      v4 = -16;                                 // a1[v3]='w',上移
    }
    ++v3;
    while ( !byte_201020[v2] )                  // 当前坐标为0
    {
      if ( v4 == -1 && (v2 & 0xF) == 0 )        // 当前在最左边一列的时候，不能够左移
        return 0LL;
      if ( v4 == 1 && v2 % 16 == 0xF )          // 当前在最右边一列的时候，不能够右移
        return 0LL;
      if ( v4 == 16 && (unsigned int)(v2 - 240) <= 0xF )// 在最后一行，不能下移
        return 0LL;
      if ( v4 == -16 && (unsigned int)(v2 + 15) <= 0x1E )// 在第一行，不能上移
        return 0LL;
      v2 += v4;                                 // 一直移动
    }
  }
  return *(_BYTE *)(v3 + a1) == '}';
}
```

byte_201020(推荐在16进制界面查看)，查看迷宫图形



```
00 00 00 00 23 00 00 00  00 00 00 00 23 23 23 23  ....#.......####
00 00 00 23 23 00 00 00  4F 4F 00 00 00 00 00 00  ...##...OO......
00 00 00 00 00 00 00 00  4F 4F 00 50 50 00 00 00  ........OO.PP...
00 00 00 4C 00 4F 4F 00  4F 4F 00 50 50 00 00 00  ...L.OO.OO.PP...
00 00 00 4C 00 4F 4F 00  4F 4F 00 50 00 00 00 00  ...L.OO.OO.P....
00 00 4C 4C 00 4F 4F 00  00 00 00 50 00 00 00 00  ..LL.OO....P....
00 00 00 00 00 4F 4F 00  00 00 00 50 00 00 00 00  .....OO....P....
23 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00  #...............
00 00 00 00 00 00 00 00  00 00 00 00 23 00 00 00  ............#...
00 00 00 00 00 00 4D 4D  4D 00 00 00 23 00 00 00  ......MMM...#...
00 00 00 00 00 00 00 4D  4D 4D 00 00 00 00 45 45  .......MMM....EE
00 00 00 30 00 4D 00 4D  00 4D 00 00 00 00 45 00  ...0.M.M.M....E.
00 00 00 00 00 00 00 00  00 00 00 00 00 00 45 45  ..............EE
54 54 54 49 00 4D 00 4D  00 4D 00 00 00 00 45 00  TTTI.M.M.M....E.
00 54 00 49 00 4D 00 4D  00 4D 00 00 00 00 45 00  .T.I.M.M.M....E.
00 54 00 49 00 4D 00 4D  00 4D 21 00 00 00 45 45  .T.I.M.M.M!...EE
```

这个移动方法有点意思，从左上角去往！，点代表路，其他符号是障碍物，点的时候会一直走，遇到障碍物才会停下，手动走一下



flag{MEWEMEWJMEWJM}