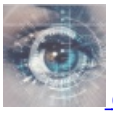
buuctf刷题记录23 [ACTF新生赛2020]Oruga

原创

ytj00 于 2020-08-08 21:52:44 发布 118 收藏

分类专栏： ctf 逆向

本文链接：https://blog.csdn.net/ytj00/article/details/107735105



ctf 同时被 2 个专栏收录

28 篇文章 0 订阅

订阅专栏

逆向

27 篇文章 0 订阅

订阅专栏

无壳，拖进ida，

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  __int64 result; // rax
  __int64 v4; // [rsp+0h] [rbp-40h]
  char v5; // [rsp+9h] [rbp-37h]
  char s2[4]; // [rsp+Ah] [rbp-36h]
  char s[40]; // [rsp+10h] [rbp-30h]
  unsigned __int64 v8; // [rsp+38h] [rbp-8h]

  v8 = __readfsqword(0x28u);
  memset(s, 0, 0x19uLL);
  printf("Tell me the flag:", 0LL);
  scanf("%s", s);
  strcpy(s2, "actf{");
  LODWORD(v4) = 0;
  while ( (signed int)v4 <= 4 )
  {
    *((_BYTE *)&v4 + (signed int)v4 + 4) = s[(signed int)v4];
    LODWORD(v4) = v4 + 1;
  }
  v5 = 0;
  if ( !strcmp((const char *)&v4 + 4, s2) )     // 比较flag的头部
  {
    if ( (unsigned __int8)sub_78A((__int64)s) ) // 关键代码
      printf("That's True Flag!", s2, v4);
    else
      printf("don't stop trying...", s2, v4);
    result = 0LL;
  }
  else
  {
    printf("Format false!", s2, v4);
    result = 0LL;
  }
  return result;
}
```

然后进入sub_78A这个关键函数里去看，有点迷宫的感觉，起点为（0,0）

```
int v2; // [rsp+Ch] [rbp-Ch]
signed int v3; // [rsp+10h] [rbp-8h]
signed int v4; // [rsp+14h] [rbp-4h]

v2 = 0;
v3 = 5;
v4 = 0;
while ( word_201020[v2] != '!' )              // word_201020大小为256，当取到！时，程序正常结束
{
  v2 -= v4;
  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"，向左一列
      }
      else
      {
        v4 = 16;                              // a1[v3]="M"，向下走一行
      }
    }
    else
    {
      v4 = 1;                                 // a1[v3]="E"，向右一列
    }
  }
  else
  {
    v4 = -16;                                 // a1[v3]="W",向上走一行
  }
```



```
  else
  {
    v4 = -16;                                 // a1[v3]="W",向上走一行
  }
  ++v3;
  while ( !word_201020[v2] )                  // 当走到0的时候，继续while循环里的内容
  {
    if ( v4 == -1 && !(v2 & 15) )             // 到最左边
      return 0LL;
    if ( v4 == 1 && v2 % 16 == 15 )           // 到最右边
      return 0LL;
    if ( v4 == 16 && (unsigned int)(v2 - 240) <= 15 )// 到最下面
      return 0LL;
    if ( v4 == -16 && (unsigned int)(v2 + 15) <= 30 )// 到最上面
      return 0LL;
    v2 += v4;                                 // 重点
  }
}
return *(_BYTE *)(v3 + a1) == 125;
```

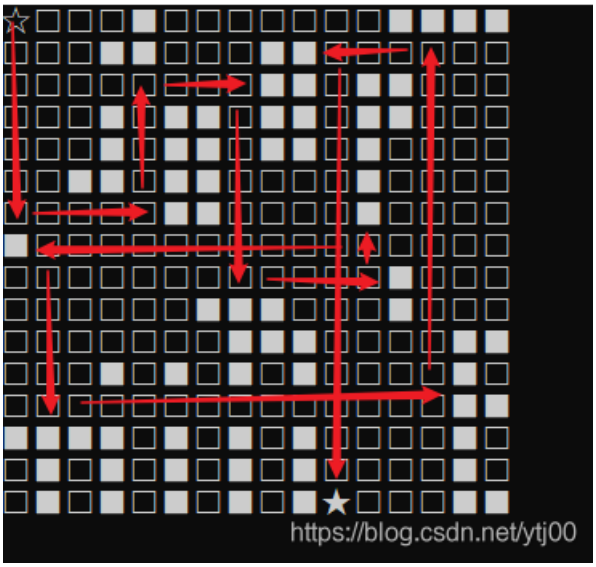

这里”0“是我们移动停止的条件，注意v2+=v4这一句，由于这是在while循环里的，所以他是一直以同一个方向移动的，直到碰到了非0的东西，很类似于象棋里面的车，但一次必须走到头

进入数据word_201020看

```
word_201020     dw 0                    ; DATA XREF: sub_78A+2
                                        ; sub_78A+DC↑o
                db    0
                db    0
                db  23h ; #
                db    0
                db    0
                db    0
                db    0
                db    0
                db    0
                db    0
                db  23h ; #
                db  23h ; #
                db  23h ; #
                db  23h ; #
                db    0
                db    0
                db    0
                db  23h ; #
                db  23h ; #
                db    0
                db    0
                db    0
                db  4Fh ; O
                db  4Fh ; O
                db    0
                db    0
                db    0
                db    0
                db    0
                db    0
                db    0
                db    0
```


然后构造脚本，画出来这个迷宫，

```c
#include <stdio.h>
char maze[256] = {
    0x00, 0x00, 0x00, 0x00, 0x23, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x23, 0x23, 0x23, 0x23,
    0x00, 0x00, 0x00, 0x23, 0x23, 0x00, 0x00, 0x00, 0x4F, 0x4F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x4F, 0x4F, 0x00, 0x50, 0x50, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x4C, 0x00, 0x4F, 0x4F, 0x00, 0x4F, 0x4F, 0x00, 0x50, 0x50, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x4C, 0x00, 0x4F, 0x4F, 0x00, 0x4F, 0x4F, 0x00, 0x50, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x4C, 0x4C, 0x00, 0x4F, 0x4F, 0x00, 0x00, 0x00, 0x00, 0x50, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x4F, 0x4F, 0x00, 0x00, 0x00, 0x00, 0x50, 0x00, 0x00, 0x00, 0x00,
    0x23, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x23, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x4D, 0x4D, 0x4D, 0x00, 0x00, 0x00, 0x23, 0x00, 0x00, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x4D, 0x4D, 0x4D, 0x00, 0x00, 0x00, 0x00, 0x45, 0x45,
    0x00, 0x00, 0x00, 0x30, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x00, 0x00, 0x00, 0x45, 0x00,
    0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x45, 0x45,
    0x54, 0x54, 0x54, 0x49, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x00, 0x00, 0x00, 0x45, 0x00,
    0x00, 0x54, 0x00, 0x49, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x00, 0x00, 0x00, 0x45, 0x00,
    0x00, 0x54, 0x00, 0x49, 0x00, 0x4D, 0x00, 0x4D, 0x00, 0x4D, 0x21, 0x00, 0x00, 0x00, 0x45, 0x45
};
int main(void)
{
    int i, j;
    for (i = 0; i < 16; i++)
    {
        for (j = 0; j < 16; j++)
        {
            if (i == 0 && j == 0)
                printf("☆");  //起点
            else if (maze[16 * i + j] == 0)
                printf("□");  //路
            else if (maze[16 * i + j] == 0x21)
                printf("★");  //终点
            else printf("■"); //障碍物
        }
        putchar('\n');
    }
}
```



得到路径为（找的老费劲了）

MEWEMEWJMEWJM

所以flag为： flag{MEWEMEWJMEWJM}