buu逆向[ACTF新生赛2020]Oruga wp

原创

43v3rY0unG 于 2020-07-22 23:09:03 发布 544 收藏

分类专栏： # RE

本文链接：https://blog.csdn.net/weixin_43876357/article/details/107525490

首先拿到题目，f5大法，找到主函数

```
  8  unsigned __int64 v8; // [rsp+38h] [rbp-8h]
  9
 10  v8 = __readfsqword(0x28u);
 11  memset(s, 0, 0x19uLL);
 12  printf("Tell me the flag:", 0LL);
 13  scanf("%s", s);
 14  strcpy(s2, "actf{");
 15  LODWORD(v4) = 0;
 16  while ( v4 <= 4 )
 17  {
 18    *(&v4 + v4 + 4) = s[v4];
 19    LODWORD(v4) = v4 + 1;
 20  }
 21  v5 = 0;
 22  if ( !strcmp(&v4 + 4, s2) )
 23  {
 24    if ( sub_78A(s) )
 25      printf("That's True Flag!", s2, v4);
 26    else
 27      printf("don't stop trying...", s2, v4);
 28    result = 0LL;
 29  }
 30  else
```

14-22行就是在做一件事：让前五位为actf{
24行发现关键函数：sub_78A()，点进去看看：

```
 7  v2 = 0;
 8  v3 = 5;
 9  v4 = 0;
10  while ( byte_201020[v2] != '!' )
11  {
12    v2 -= v4;
13    if ( *(v3 + a1) != 'W' || v4 == -16 )
14    {
15      if ( *(v3 + a1) != 'E' || v4 == 1 )
16      {
17        if ( *(v3 + a1) != 'M' || v4 == 16 )
18        {
19          if ( *(v3 + a1) != 'J' || v4 == -1 )
20            return 0LL;
21          v4 = -1;
22        }
23        else
24        {
25          v4 = 16;
26        }
27      }
28      else
29      {
30        v4 = 1;
31      }
32    }
33    else
34    {
35      v4 = -16;
36    }
37    ++v3;
38    while ( !byte_201020[v2] )
39    {
40      if ( v4 == -1 && !(v2 & 0xF) )
41        return 0LL;
42      if ( v4 == 1 && v2 % 16 == 15 )
43        return 0LL;
44      if ( v4 == 16 && (v2 - 240) <= 0xF )
45        return 0LL;
46      if ( v4 == -16 && (v2 + 15) <= 0x1E )
47        return 0LL;
48      v2 += v4;
49    }
50  }
51  return *(v3 + a1) == '}';
52}
```


看到函数整体结构，两个while循环嵌套，中间四个分支语句，盲猜迷宫...

正经来...先分析一波：
向下移动：M
向左移动：J
向上移动：E
向右移动：W（没有格数限制）

第二个嵌套：0可以移动（一直移动），到达非0位置停止，返回上一位置。左上角为起点，0x21为终点。
写脚本也行，手动解也行，因为数据不多：
（需要注意的是，这版迷宫只有遇到障碍物才会停）

```
................
....#.......####
...##...OO......
........OO.PP...
...L.OO.OO.PP...
...L.OO.OO.P....
..LL.OO....P....
.....OO....P....
#...............
............#...
......MMM...#...
.......MMM....EE
...0.M.M.M....E.
..............EE
TTTI.M.M.M....E.
.T.I.M.M.M....E.
.T.I.M.M.M!...EE
```

（.代表路，！代表终点，剩下都是障碍物）
我是手解的：MEWEMEWJMEWJM
get flag！

附上脚本代码供参考：

```
#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');
    }
}
```