

# buu-[ACTF新生赛2020]Oruga

原创

有点水啊 于 2022-03-13 18:51:00 发布 157 收藏

分类专栏: [buuctf-reserve](#) 文章标签: [安全](#)

版权声明: 本文为博主原创文章, 遵循 [CC 4.0 BY-SA](#) 版权协议, 转载请附上原文出处链接和本声明。

本文链接: <https://blog.csdn.net/qqq517384/article/details/123460051>

版权



[buuctf-reserve 专栏收录该内容](#)

59 篇文章 0 订阅

订阅专栏

64位



字符串略过

查看main函数

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
    __int64 result; // rax
    __int64 v4; // [rsp+0h] [rbp-40h]
    __int64 v5; // [rsp+0h] [rbp-40h]
    __int64 v6; // [rsp+0h] [rbp-40h]
    __int64 v7; // [rsp+0h] [rbp-40h]
    char v8; // [rsp+9h] [rbp-37h]
    char s2[4]; // [rsp+Ah] [rbp-36h]
    char s[40]; // [rsp+10h] [rbp-30h]
    unsigned __int64 v11; // [rsp+38h] [rbp-8h]

    v11 = __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;
    }
    v8 = 0;
    if ( !strcmp((const char *)&v4 + 4, s2) )
    {
        if ( (unsigned __int8)sub_78A(s, s2) )
            printf("That's True Flag!", v6);
        else
            printf("don't stop trying...", v7);
        result = 0LL;
    }
    else
    {
        printf("Format false!", s2, v5);
        result = 0LL;
    }
    return result;
}
```

跟进sub\_78A

```

_BOOL8 __fastcall sub_78A(__int64 a1)
{
    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 ( byte_201020[v2] != '!' )           // !号即为终点
    {
        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;                                // 输入J, 左移
                }
                else
                {
                    v4 = 16;                             // 输入M, 下移
                }
            }
            else
            {
                v4 = 1;                            // 输入E, 右移
            }
        }
        else
        {
            v4 = -16;                           // 输入W, 上移
        }
        ++v3;
    while ( !byte_201020[v2] )
    {
        if ( v4 == -1 && !(v2 & 0xF) )          // 最左边时不能左移
            return 0LL;
        if ( v4 == 1 && v2 % 16 == 15 )         // 最右边时不能右移
            return 0LL;
        if ( v4 == 16 && (unsigned int)(v2 - 240) <= 0xF )// 最下边时不能下移
            return 0LL;
        if ( v4 == -16 && (unsigned int)(v2 + 15) <= 0x1E )// 最上边时不能上移
            return 0LL;
        v2 += v4;                                //持续移动
    }
    return *(_BYTE *) (v3 + a1) == 125;
}

```

byte\_201020的值

```
; _BYTE byte_201020[256]
byte_201020    db 4 dup(0), '#', 7 dup(0), 4 dup('#'), 3 dup(0), 2 dup('#')
; DATA XREF: sub_78A+23↑o
; sub_78A+DC↑o
db 3 dup(0), 2 dup('0'), 0Eh dup(0), 2 dup('0'), 0, 2 dup('P')
db 6 dup(0), 'L', 0, 2 dup('0'), 0, 2 dup('0'), 0, 2 dup('P')
db 6 dup(0), 'L', 0, 2 dup('0'), 0, 2 dup('0'), 0, 'P'
db 6 dup(0), 2 dup('L'), 0, 2 dup('0'), 4 dup(0), 'P'
db 9 dup(0), 2 dup('0'), 4 dup(0), 'P', 4 dup(0), '#'
db 1Bh dup(0), '#', 9 dup(0), 3 dup('M'), 3 dup(0), '#'
db 0Ah dup(0), 3 dup('M'), 4 dup(0), 2 dup('E'), 3 dup(0)
db '0', 0, 'M', 0, 'M', 0, 'M', 4 dup(0), 'E', 0Fh dup(0)
db 2 dup('E'), 3 dup('T'), 'I', 0, 'M', 0, 'M', 0, 'M'
db 4 dup(0), 'E', 2 dup(0), 'T', 0, 'I', 0, 'M', 0, 'M'
db 0, 'M', 4 dup(0), 'E', 2 dup(0), 'T', 0, 'I', 0, 'M'
db 0, 'M', 0, 'M', '!', 3 dup(0), 2 dup('E')
_data
ends
```

CSDN @有点水啊

±1和±16既是移动也是边界

本来是想画迷宫的，但是突然发现16进制的显示刚刚好就是地图

从左上角移动到!

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 00 00 00 00 00	.....
00 00 00 00 23 00 00 00 00 00 00 00 00 00 00 00	00 00 00 00 23 23 23 23 23 23 23 23 23 23 23 23	.#.....###
00 00 00 23 23 00 00 00 00 00 00 00 00 00 00 00 00 00	4F 4F 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 00 00 00 00	4F 4F 00 50 50 00 00 00 00 00 00 00 00 00 00 00	....00.PP...
00 00 00 4C 00 4F 4F 00 00 00 00 00 00 00 00 00	4F 4F 00 50 50 00 00 00 00 00 00 00 00 00 00 00	...L.00.00.PP...
00 00 00 4C 00 4F 4F 00 00 00 00 00 00 00 00 00	4F 4F 00 50 00 00 00 00 00 00 00 00 00 00 00 00	...L.00.00.P....
00 00 4C 4C 00 4F 4F 00 00 00 00 00 00 00 00 00	00 00 00 50 00 00 00 00 00 00 00 00 00 00 00 00	..LL.00....P....
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	00 00 00 50 00 00 00 00 00 00 00 00 00 00 00 00	....00....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 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 23 00 00 00 00 00 00 00 00 00	.....#....
00 00 00 00 00 00 00 4D 4D 4D 00 00 00 23 00 00 00	00 00 00 4D 4D 4D 00 00 00 23 00 00 00 00 00 00	.....MMM...#...
00 00 00 00 00 00 00 4D 4D 4D 00 00 00 00 45 45	00 00 00 4D 4D 4D 00 00 00 00 00 45 45	.....MMM....EE
00 00 00 30 00 4D 00 4D 00 4D 00 00 00 00 45 00	00 00 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	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	00 4D 00 00 00 00 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	00 4D 00 00 00 00 00 00 00 00 45 00	.T.I.M.M.M....E.
00 54 00 49 00 4D 00 4D 00 4D 00 21 00 00 00 45 45	00 4D 21 00 00 00 00 00 00 00 45 45	.T.I.M.M.M!....EE
?? ?? ?? ?? ?? ?? ??	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	CSDN @有点水啊
?? ?? ?? ?? ?? ?? ??	00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	

那么路径就是MEWEMEWJMEWJM

flag{MEWEMEWJMEWJM}