

【逆向学习】 maze writeup

原创

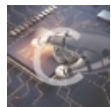
charlie_heng 于 2017-12-19 15:34:59 发布 614 收藏

分类专栏： [二进制-逆向工程](#)

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/charlie_heng/article/details/78843307

版权



[二进制-逆向工程 专栏](#)收录该内容

34 篇文章 3 订阅

订阅专栏

这题是16年的国赛逆向题目

main函数如下

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
    if ( argc == 2 )
    {
        memset(byte_410AA8, -1, 0x1E4u);
        sub_4010B0();
        sub_4010B0();
        if ( sub_401150(argv[1]) )
        {
            sub_401290((void *)argv[1]);
            return 0;
        }
        printf("play again!");
    }
    return 0;                                http://blog.csdn.net/charlie_heng
}
```

首先调用了两个函数来初始化，这里应该是初始化了一个22x22的矩阵

看下面的验证函数

```
14 int v12; // [esp+4h] [ebp-4h]
15
16 v1 = a1;
17 if ( strlen(a1) & 1 )
18     return 0;
19 v3 = *a1;
20 v4 = 0;
21 v5 = 0;
22 v6 = 0;
23 v7 = -1;
24 v12 = 0;
25 if ( !*a1 )
26     goto LABEL_21;
27 v8 = 0;
28 do
29 {
30     if ( v3 < 97 || v3 > 100 )
31         goto LABEL_17;
32     v9 = v1[1] - 'e';
33     if ( v9 > 0x15 )
34     {
35         v1 = a1;
36 LABEL_17:
37         ++v4;
38         goto LABEL_18;
39     }
40     v10 = v7;
41     switch ( v3 )
42     {
43         case 'a':
44             v7 = 0;
45             v6 = (signed int)(v6 - v9) % 22;
46             v8 = byte_410AA8[22 * v5 + v6];
47             break;
48         case 'b':
49             v7 = 0;
50             v6 = (signed int)(v9 + v6) % 22;
51             goto LABEL_14;
52         case 'c':  
http://blog.csdn.net/charlie_heng
```

```
59         v5 = v11 % 22;
60 LABEL_14:
61     v8 = byte_410AA8[22 * v5 + v6];
62     break;
63     default:
64     break;
65 }
66 byte_410AA8[22 * v5 + v6] = 0;
67 v4 = v12 + (v8 ^ 1) + (v10 == v7);
68 v1 = a1;
69 LABEL_18:
70     v3 = v1[2];
71     v1 += 2;
72     v12 = v4;
73     a1 = v1;
74 }
75 while ( v3 );
76 if ( v5 != 21 || v6 != 21 )
77 LABEL_21:
78     ++v4;
79     result = v4 < 0;
80     LOBYTE(result) = v4 <= 0;
81     return result;  
http://blog.csdn.net/charlie_heng
```

看了下，这里应该是一个走迷宫，从0, 0走到21, 21，一次只能往上下左右走，而且这一次向左右走了，下次只能往上下走
那么卡个断点，把矩阵给扒出来，之后跑个dfs只能跑出来答案

下面是dfs的代码

