[BUUCTF] [Reverse]不一样的flag

原创

flagorz 于 2021-10-05 21:08:42 发布 59 收藏
分类专栏： # BUUCTF 文章标签： 算法

本文链接：https://blog.csdn.net/inuseonline/article/details/120618298

前言

好多题解writeup直接说这个题目是个迷宫题，说的好像一眼就能看出来似的。但是实际上不提前看答案或者研究透彻这个题目的话，根本想不出来这是个迷宫。
下面说一下我的思路吧。

题目

不一样的flag

是不是做习惯了常规的逆向题目？试试这道题，看你在能不能在程序中找到真正的flag！注意：flag并非是flag{XXX}形式，就是一个'字符串'，考验眼力的时候到了！ 注意：得到的 flag 请包上 flag{} 提交

思路

首先是反编译：

```
int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
{
  _BYTE v_5x5_4[29]; // [esp+17h] [ebp-35h] BYREF
  int v4; // [esp+34h] [ebp-18h]
  int input; // [esp+38h] [ebp-14h] BYREF
  int i; // [esp+3Ch] [ebp-10h]
  _BYTE v7[12]; // [esp+40h] [ebp-Ch] BYREF

  __main();
  v_5x5_4[26] = 0;
  *(_WORD *)&v_5x5_4[27] = 0;
  v4 = 0;
  strcpy(v_5x5_4, "*11110100001010000101111#");
  while ( 1 )
  {
    puts("you can choose one action to execute");
    puts("1 up");
    puts("2 down");
    puts("3 left");
    printf("4 right\n:");
    scanf("%d", &input);
    if ( input == 2 )                           // 下
    {
      ++*(_DWORD *)&v_5x5_4[25];
    }
```

```
    else if ( input > 2 )
    {
      if ( input == 3 )                         // 左
      {
        --v4;
      }
      else
      {
        if ( input != 4 )
LABEL_13:
          exit(1);
        ++v4;                                   // 右
      }
    }
    else
    {
      if ( input != 1 )
        goto LABEL_13;
      --*(_DWORD *)&v_5x5_4[25];                // 上
    }
    for ( i = 0; i <= 1; ++i )
    {
      if ( *(int *)&v_5x5_4[4 * i + 25] < 0 || *(int *)&v_5x5_4[4 * i + 25] > 4 )// 超出边界退出。25是上下位置
        exit(1);
    }
    if ( v7[5 * *(_DWORD *)&v_5x5_4[25] - 41 + v4] == '1' )
      exit(1);
    if ( v7[5 * *(_DWORD *)&v_5x5_4[25] - 41 + v4] == '#' )
    {
      puts("\nok, the order you enter is the flag!");
      exit(0);
    }
  }
}
```

我在代码上加上注释了。

这个代码的核心之处就是，迷宫是个_BYTE v_5x5_4[29]变量，前面25位是迷宫地图，第26位是上下坐标的位置，v4变量是左右坐标的位置。左上角*号为(0,0)坐标，向右向下为第一象限。

下面一个2次的for循环，利用指针越界判断上下左右坐标。25是上下位置，29超出数组了，指针指向到v4的位置，为左右位置，超出迷宫范围则越界。

最后一段，就是碰墙退出，碰#号出迷宫。迷宫地图为：

```
"*1111"
"01000"
"01010"
"00010"
"1111#"
```

所以，最后的答案才是迷宫的路径，“下下下右右上上右右下下下”，即flag{222441144222}