# 寻找产生子弹的CALL

首先我们要找到这个记录场上子弹数量的地址



通过CE过滤子弹数量, 当子弹数为1时搜1, 为2时搜2.......最终可以得到记录当前场上子弹数量的地址

记录当前子弹数量	22U2E358	4 Bytes	99

# 找到子弹数量之后 寻找哪些地址改变了这个值

- 5 00436B91 01 47 10 add [edi+10],eax
- 4 0043420C·FF 8F F0000000·dec [edi+000000F] 一个是减少的 一个是增加的 我们当然 选择增加

## 的去观察了啊

# 上X32Dbg 观察函数调用了

地址	赵則到	<b>巡問日</b>	人小	<b>注</b> 釋	Л
0019F93C	00422E9F	00436BC9	C	plantsvszombies.00436BC9	用户模块 用户模块
0019F948	0048B5A1	00422E9F	4C	plantsvszombies.00422E9F	用户模块
0019F994	00488E11	0048B5A1	34	plantsvszombies.0048B5A1	用户模块 用户模块
0019F9C8	00487282	00488E11	98	plantsvszombies.00488E11	用户模块
0019FA60	0042BF05	00487282	4	plantsvszombies.00487282	用户模块
0019FA64	025F4678	0042BF05	28C4A5	plantsvszombies.0042BF05	用户模块
0042BF09	858B0000	025F4678	4	025F4678	用户模块 用户模块 用户模块
0042BF0D	00000000	858B0000		858B0000	用户模块

# 看调用堆栈

# 如何确定是哪个CALL?

### 有这么多调用 哪个才是我们要找的呢?

**看参数**,产生子弹至少需要哪两个参数?,答案呼之欲出:坐标,子弹的X坐标和Y坐标,于是我们的这个函数至少有2个参数

```
int3
push esi
push edi
mov edi,eax
add edi,E0
call plantsvszombies.436ADO
mov ecx,dword ptr ss:[esp+14]
mov edx,dword ptr ss:[esp+10]
mov eax,dword ptr ss:[esp+10]
mov eax,dword ptr ss:[esp+10]
 00422E8F
00422E90
00422E91
                                     CC
56
57
                                      57
8BF8
81C7 E0000000
E8 313C0100
8B4C24 14
8B5424 10
 00422E92
00422E9A
                                                                                                                                                                                                                                                                                     子弹增加call
 00422E9F
00422EA3
                                      8BF0
8B4424 1C
                                                                                                                push eax
mov eax,dword ptr ss:[esp+10]
push ecx
push edx
push eax
mov eax,dword ptr ss:[esp+28]
push esi
call plantsvszombies.491840
pop edi
mov eax,esi
pop esi
 00422EAD
                                      50
8B4424 10
 00422EAE
00422EB2
00422EB3
                                      51
52
50
 00422EB4
                                      8B4424 28
                                      56
E8 81E90600
  00422EBA
 00422EBF
                                     8BC6
5E
C2 1400
                                                                                                                  pop est
```

## 这个call 很有可能



### 这个call 也有可能 那么到底是哪个呢? 使用工具测试



仿照代码调用 函数发现功能正常且多出来一颗子弹

```
push 0
push 0
mov eax,0x00049EAD

dec eax
push eax
mov eax,0x1DC7CCD0
push 0x5b
push 0x6C
call 0x422e90
```

## 上一层的call又是什么呢?

```
1 push 0
2 push 1
3 push 0
4 push 1DCDC4B8;这个比较奇怪 不知道是什么 经过分析发现是植物结构体地址
5 call 48b190
```

这样调用同样可以生成子弹

# 寻找产生子弹的CALL 总结: call 48b190 和 call 0x422e90 都是可以的

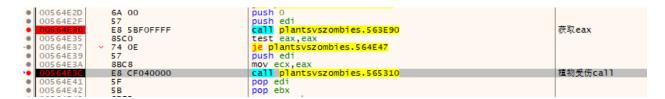
# 让我们来找找植物数组

为什么要找到植物数组,前面我们找到的call有无法分析的参数 ,猜测可能是植物结构体地址,所以我们找植物数组进行验证

### 坚果强受伤法

```
005654C0 - add dword ptr [esi+40],-04
esi = ecx = eax
```

一路向上跟 找到关键函数



# 可以看到 是一个函数生成植物的结构体地址

# 进去

```
8B45 08
8B50 04
8D7424 0C
E8 0113EDFF
84C0
74 6D
8B4D 08
8B51 1C
8B7424 0C
3B56 1C
75 DE
8D4424 10
                                                                                                                 mov eax, dword ptr
mov edx, dword ptr
lea esi, dword ptr
call plantsvszombi
test al, al
                                                                                                                                                                                                                                                                                     edx:"0Zr", [eax+4]:"0Zr"
                                                                                                                                                                                                                                                                                      生成植物地址
                                                                                                                 je plantsyszombie:
mov ecx,dword ptr
mov edx,dword ptr
mov esi,dword ptr
cmp edx,dword ptr
jne plantsyszombie
lea eax,dword ptr
                                                                                                                                                                                                                                                                                     获取植物数组
判断是不是我们的植物
                                                                                                                                          push edi
mov edi,FFFF0000
mov eax,dword ptr ds:[esi]
test eax,eax
jne plantsvszombies.4351E4
mov eax,dword ptr ds:[edx+C4]
jmp plantsvszombies.4351E9
add eax,14C
mov ecx,dword ptr ds:[edx+C8]
imul ecx,ecx,14C
add ecx,dword ptr ds:[edx+C4]
cmp eax,ecx
jae
plantsvszombies.435211
nop
test dword ptr ds:[eax+148],ed
004351D0
004351D1
004351D6
                                              57
BF 0000FFFF
8B06
85C0
                                                                                                                                                                                                                                                                                                                                                     植物数组guanjiancall
004351D8
004351DA
                                               75 08
                                             75 08
8B82 C400000
EB 05
05 4C010000
8B8A C8000000
69C9 4C010000
038A C4000000
3BC1
73 12
004351DA
004351DC
004351E2
004351E4
004351E9
004351EF
004351F5
004351FB
004351FD
 004351FF
                                             90
8588 48010000
75 13
05 4C010000
3BC1
72 EF
C706 FFFFFFF
32C0
5F
                                               90
                                                                                                                                           test dword ptr ds:[eax+148],edi
jne plantsvszombies.43521B
add eax,14C
00435200
00435206
00435208
                                                                                                                                          plantsvszombies.435200
mov dword ptr ds:[esi],FFFFFFFF
xor al,al
pop edi
ret
 0043520D
0043520F
00435211
00435217
00435219
                                                                                                                                          ret
mov dword ptr ds:[esi],eax
cmp byte ptr ds:[eax+141],0
ine plantsvszombies.4351D6
mov al,1
pop edi
0043521B
                                               8906
                                               80B8 41010000 00
75 B0
B0 01
                                                                                                                                           pop
ret
                                               5F
00435229
                                               C3
```

#### 可以看到跟僵尸的很相似

植物数组长度为14c 首地址在edx+c4处存储

#### 植物结构体

0x8 x坐标 相差0x4f

0xc v坐标 相差0x64

# 让我们来找找植物数组 总结:通过找到植物数组 经过比较 我们发现 call 48b190的第一个参数就是植物结构体

# 植物分身

我们试试拦截调用call 修改坐标相关寄存器看看会不会向其他方向发射子弹

```
    0048B593
    52
    push edx push ebx 分子并有数

    0048B594
    53
    push ebx dec eax

    0048B595
    48
    dec eax

    0048B596
    50
    push eax

    0048B597
    8B45 04
    mov eax_dword ptr ss:[ebp+4]

    0048B59A
    57
    push edi
    坐标У

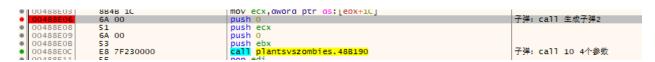
    0048B59B
    56
    push esi
    坐标X

    0048B59C
    E8 EF78F9FF
    call plantsvszombies.422E90
    子弹增加2call 14 5个参数
```

记录下寄存器的值 使用代码注入工具

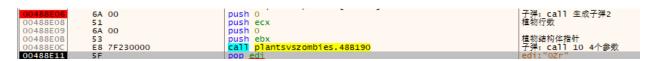


如此我们只要在发射子弹的合适地方进行HOOK循环调用发射子弹 修改y坐标的值 就可以做到子弹分身



我们选用上一层植物发射子弹的call 进行hook 可以直接获取植物数组的地址

当然 我们已经找到了植物数组的基址 也可以进行判断过后(是否可以发射子弹)进行自动发射子弹 也可以直接Hook这个当前这个函数 就是需要更多的处理



发现调用完毕 ebx的值还是不变 同样指向发射子弹的植物结构体指针 我们在这个地方 HOOK

```
1 //植物子弹分身
 2 static DWORD g_dwCall = ZHIWU_ZIDAN_FASHE_CALL_BASE; //发射子弹call
 3 static bool g_bIsZiDanFengSheng = false;//控制是否开始
 5 #define ZHIWU_YPOSITON_FIRST 0x50 //第一个Y坐标
6 #define ZHIWU_YPOSITON_OFFSET 0x64 //Y坐标之间的偏移
7 void __declspec(naked) m_fZiDanFengSheng(void)
9
       __asm
      {
10
11
         push ebp;
12
          mov ebp, esp;
13
          sub esp, 0x100;
      }
14
      PDWORD pdwZhiWu;
15
16
       __asm
17
      {
18
          mov pdwZhiWu, ebx;
19
      }
20
      if (g_bIsZiDanFengSheng == false)
21
      {
22
          goto exit;
23
       }
24
25
      DWORD dwPositionYOld;
26
      DWORD dwPositionYNew;
27
      __asm
28
      {
29
          mov dword ptr dwPositionYNew, ZHIWU_YPOSITON_FIRST;
30
      }
31
      dwPositionYOld = *PDWORD((DWORD)pdwZhiWu + JIANGSHI_YPOSITON_OFFSET);//保
32
      for (int i = 0; i < 5; i++, dwPositionYNew += ZHIWU_YPOSITON_OFFSET)</pre>
33
34
      {
35
          if (dwPositionYNew != dwPositionYOld)//同行不再发射子弹了
          {
36
               *PDWORD((DWORD)pdwZhiWu + JIANGSHI_YPOSITON_OFFSET) = dwPosition\
37
38
39
               __asm
              {
40
41
                   push 0
42
                   push i
43
                   push 0
```

```
push pdwZhiWu
44
                  call g_dwCall
45
              }//发射子弹
46
          }
47
48
       }
      *PDWORD((DWORD)pdwZhiWu + JIANGSHI_YPOSITON_OFFSET) = dwPositionYOld; //x
49
50 exit:
      __asm
51
52
      {
53
          leave;
         ret;
54
55
      }
56 }
```