

2021美团高校挑战赛逆向randomWriteup

原创

bin cat 于 2021-12-12 17:29:36 发布 1823 收藏

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](https://creativecommons.org/licenses/by-sa/4.0/) 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/m0_58348028/article/details/121881200

版权

记录一次动调



我好没本领



查壳, 32位无壳, 用ida打开找到主函数

```
.text:004010E0
.text:004010E0 ; __unwind { // __except_handler4
.text:004010E0         push     ebp
.text:004010E1         mov     ebp, esp
.text:004010E3         push     0FFFFFFFh
.text:004010E5         push     offset stru_402518
.text:004010EA         push     offset __except_handler4
.text:004010EF         mov     eax, large fs:0
.text:004010F5         push     eax
.text:004010F6         sub     esp, 8
.text:004010F9         push     ebx
.text:004010FA         push     esi
.text:004010FB         push     edi
.text:004010FC         mov     eax, __security_cookie
.text:00401101         xor     [ebp+ms_exc.registration.ScopeTable], eax
.text:00401104         xor     eax, ebp
.text:00401106         push     eax
.text:00401107         lea     eax, [ebp+ms_exc.registration]
.text:0040110A         mov     large fs:0, eax
.text:00401110         mov     [ebp+ms_exc.old_esp], esp
```

000004E0 004010E0: main (Synchronized with Hex View-1)

CSDN @bin cat

通过伪代码发现主体逻辑在这里

```

00401110      mov     [ebp+ms_exc.old_esp], esp
00401113      cmp     dword_40336C, 0
0040111A      jnz    short loc_401130
0040111C      push  2Bh ; '+'
0040111E      push  offset input    ; Arglist
00401123      push  offset a5      ; "%s"
00401128      call  sub_4010B0
0040112D      add     esp, 0Ch
00401130
00401130 loc_401130: ; CODE XREF: _main+3A1j
00401130 ; __try { // __except at loc_40115D
00401130      mov     [ebp+ms_exc.registration.TryLevel], 0
00401137      call  ds:rand
0040113D      push  eax            ; Seed
0040113E      call  ds:srand
00401144      xor     eax, eax
00401146      div     eax
00401146 ; } // starts at 401130
00401148      mov     [ebp+ms_exc.registration.TryLevel], 0FFFFFFEh
0040114F      mov     edi, ds:rand
00401155      jmp     short loc_401179
00401157 ; -----

```

CSDN @bin cat

1处输入，2处生成伪随机数，3处种一个随机数种子，4处拿种子生成伪随机数

```

if ( !dword_40336C )
    sub_4010B0(v3, "%s", (char)input);
v4 = rand();
srand(v4);
v5 = dword_40336C;
input[v5] ^= rand();
if ( dword_40336C == 43 )
{
    dword_40336C = 0;
    v6 = 0;
    v7 = 0;
    v8 = 0;
    v9 = 0;
    do
    {
        if ( input[v9] != byte_402138[v8] )
            break;
        v8 = ++v6;
        v7 = v6;
        dword_40336C = v6;
    }
}

```

CSDN @bin cat

00000579 main:20 (401179)

显然关键在于v5，通过观察我们可以确定v5就是我们要找的flag

```

v9 = 0;
do
{
if ( input[v9] != byte_402138[v8] )
break;
v8 = ++v6;
v7 = v6;
dword_40336C = v6;
v9 = v6;
}
while ( v6 < 42 );
v10 = "fake input..\n";
if ( v7 == 42 )
v10 = "congratulation!\n";
sub_401050(v10);
result = 0;

```

CSDN @bin cat

发现有一个判断，意思应该是input[v5]^rand()之后按位和byte_402138比较

展开byte_402138，就是一些16进制数

```

!128 aFakeInput      db 'fake input..',0Ah,0 ; DA
!136                align 4
!138 ; char byte_402138[44]
!138 byte_402138    db 3Eh ; DA
!139                db 0CDh
!13A                db 0AAh
!13B                db 8Eh
!13C                db 96h
!13D                db 1Fh
!13E                db 89h
!13F                db 0CDh
!140                db 0DBh
!141                db 0F1h
!142                db 70h ; p
!143                db 0F2h
!144                db 0A9h
!145                db 9Ch
!146                db 0C2h
!147                db 8Bh
!148                db 0F2h
!149                db 0FEh

```

!138: .rdata:byte 402138 (Synchronized with Hex View-1) | CSDN @bin cat

静态分析找不到什么思路，动调一下

下断点让程序停在0004111C的地方

```

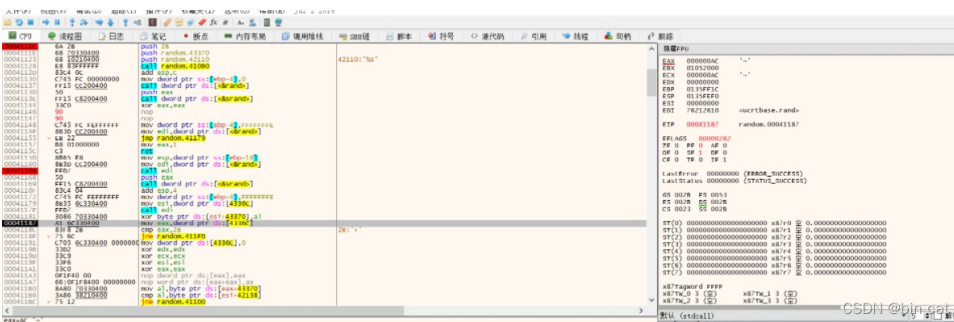
push 2B
push random.43370
push random.42110
call random.410B0 ← 1
add esp,c
mov dword ptr ss:[ebp-4],0
call dword ptr ds:[<&rand>]
push eax
call dword ptr ds:[<&srands>]
xor eax,eax
div eax
mov dword ptr ss:[ebp-4],FFFFFFFE
mov edi,dword ptr ds:[<&rand>]
jmp random.41179
mov eax,1
ret
mov esp,dword ptr ss:[ebp-18]
mov edi,dword ptr ds:[<&rand>]
call edi
push eax
call dword ptr ds:[<&srands>]
add esp,4
mov dword ptr ss:[ebp-4],FFFFFFFE
mov esi,dword ptr ds:[4336C]
call edi
xor byte ptr ds:[esi+43370],al
mov eax,dword ptr ds:[4336C]
cmp eax,2B ← 2
jne random.411FD
mov dword ptr ds:[4336C],0
xor edx,edx
xor ecx,ecx
xor esi,esi
xor eax,eax
nop dword ptr ds:[eax],eax
mov word ptr ds:[eax+eax],ax
mov al,byte ptr ds:[eax+43370]

```

1处有个输入，2处有个对比告诉我们eax寄存器里存的就是我们要对比的，开始动调



调到这里显示要跳转，继续跟下去



此时eax给出的是0xAC

这个显然不符合我们之前的判断要异或出f，eax最后两位应该是58

猜测可能是因为jmp


```

8B3D CC200400 mov edi,dword ptr ds:[<&rand>]
E8 01000000 jmp random.41179
B8 01000000 mov eax,1
C3 ret
8B65 E8 mov esp,dword ptr ss:[ebp-18]
8B3D CC200400 mov edi,dword ptr ds:[<&rand>]
FFD7 call edi
50 push eax
FF15 C8200400 call dword ptr ds:[<&srand>]
83C4 04 add esp,4
C745 FC FEFFFFFF mov dword ptr ss:[ebp-4],FFFFFFF
8B35 6C330400 mov esi,dword ptr ds:[4336C]
FFD7 call edi
3086 70330400 xor byte ptr ds:[esi+43370],al

```

这其中跳掉了

重新动调

这次直接改eip劫持程序到如图这里

成功

eax显示的是58符合猜想，确实第一位是f

在试试第二位，把程序劫持上去，劫持到00041137的地方

eax显示A1也符合对的猜想

然后按照我们推出的理论上一个脚本解题就可以了

也可以一位一位推直到“}”出现结束

```
int main()
{
    int i;
    char flag[43]={0x3E,0xCD,0xAA,0x8E,0x96,0x1F,0x89,0xCD,0xDB,0xF1,0x70,0xF2,0xA9,0x9C,0xC2,0x8B,0xF2,0xFE,0
    for(i=0;i<43;i++)
    {
        int x=rand();
        srand(x);
        int w=rand();
        srand(w);
        int v=rand();
        printf("0x%x 0x%x \n",x,v);
        flag[i]^=v;
    }
    printf("%s\n",flag);
    // char dd=getchar();
    return 0;
}
```

flag{3e625fe0-fb18-4f87-93c1-1ec217f86796}