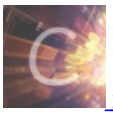
看雪CTF2019Q2第一题神秘的来信

原创

初识逆向 于 2019-08-09 20:58:31 发布 374 收藏

分类专栏： 逆向分析

本文链接：https://blog.csdn.net/w_g3366/article/details/98985074



逆向分析 专栏收录该内容

11 篇文章 1 订阅

订阅专栏

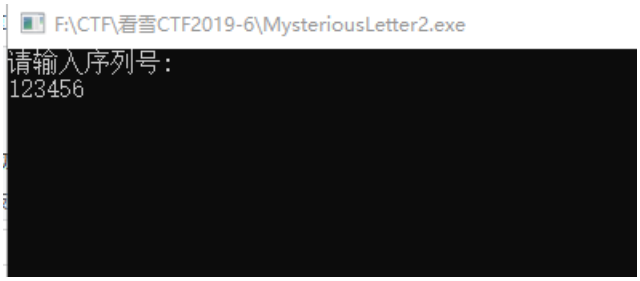
看雪CTF2019Q2第一题神秘的来信

环境和工具

Windows10+x32dbg

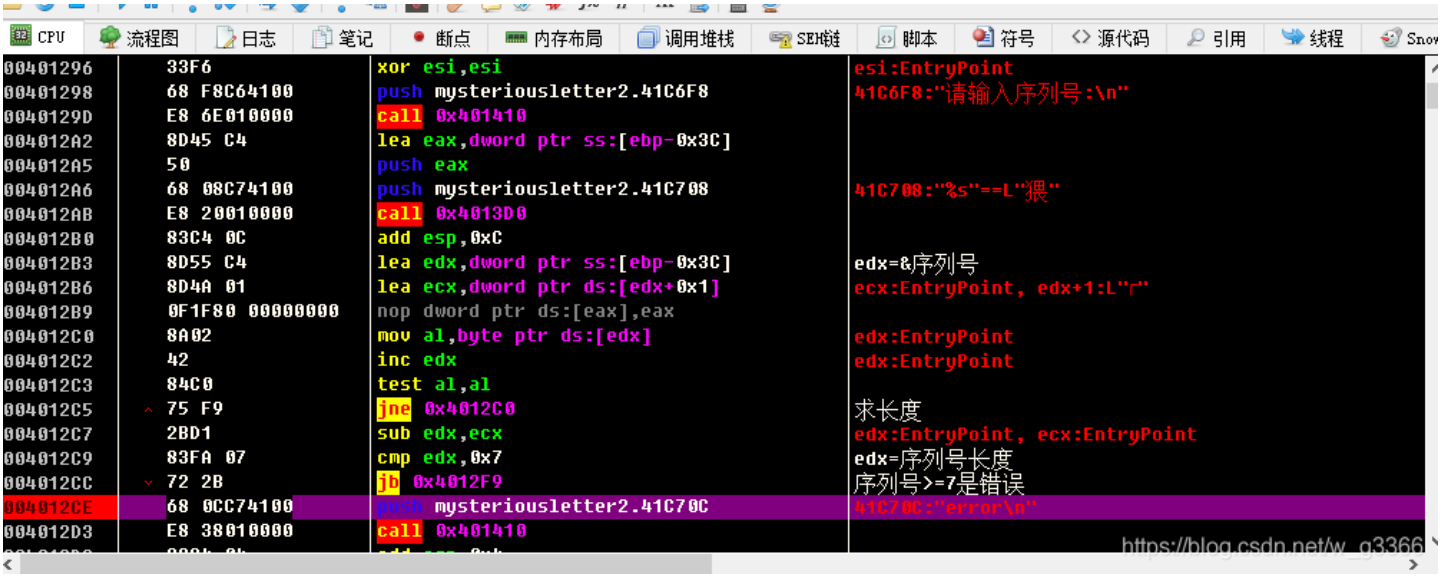
程序分析

运行程序，输入序列号，输错程序就直接退了。



没看到任何提示，先搜索字符串，看看有什么有用信息

定位error和success字符串位置，观察代码，发现代码不是很长，那么直接从输入字符串的地方开始分析



分析代码

```
00401298  | 68 F8C64100           | push mysteriousletter2.41C6F8          | 41C6F8:"请输入序列号:\n"
0040129D  | E8 6E010000           | call 0x401410                          |
004012A2  | 8D45 C4               | lea eax,dword ptr ss:[ebp-0x3C]        |
004012A5  | 50                    | push eax                               |
004012A6  | 68 08C74100           | push mysteriousletter2.41C708          | 41C708:"%s"==L"猥"
004012AB  | E8 20010000           | call 0x4013D0                          |
004012B0  | 83C4 0C               | add esp,0xC                            |
004012B3  | 8D55 C4               | lea edx,dword ptr ss:[ebp-0x3C]        | edx=&序列号
004012B6  | 8D4A 01               | lea ecx,dword ptr ds:[edx+0x1]         | ecx:EntryPoint, edx+1:L"Γ"
004012B9  | 0F1F80 00000000       | nop dword ptr ds:[eax],eax             |
004012C0  | 8A02                  | mov al,byte ptr ds:[edx]               | edx:EntryPoint
004012C2  | 42                    | inc edx                                | edx:EntryPoint
004012C3  | 84C0                  | test al,al                             |
004012C5  | 75 F9                 | jne 0x4012C0                           | 求长度
004012C7  | 2BD1                  | sub edx,ecx                            | edx:EntryPoint, ecx:EntryPoint
004012C9  | 83FA 07               | cmp edx,0x7                            | edx=序列号长度
004012CC  | 72 2B                 | jb 0x4012F9                            | 序列号>=7是错误
004012CE  | 68 0CC74100           | push mysteriousletter2.41C70C          | 41C70C:"error\n"
004012D3  | E8 38010000           | call 0x401410                          |
```

```
004012D3  | E8 38010000           | call 0x401410                                 |
004012D8  | 83C4 04               | add esp,0x4                                   |
004012DB  | 33C0                  | xor eax,eax                                   |
004012DD  | 8B4D F0               | mov ecx,dword ptr ss:[ebp-0x10]               | ecx:EntryPoint, [ebp-10]:@BaseThr
eadInitThunk@12+24
004012E0  | 64:890D 00000000      | mov dword ptr fs:[0],ecx                      | ecx:EntryPoint
004012E7  | 59                    | pop ecx                                       | ecx:EntryPoint
004012E8  | 5F                    | pop edi                                       | edi:EntryPoint
004012E9  | 5E                    | pop esi                                       | esi:EntryPoint
004012EA  | 5B                    | pop ebx                                       |
004012EB  | 8B4D E4               | mov ecx,dword ptr ss:[ebp-0x1C]               | ecx:EntryPoint
004012EE  | 33CD                  | xor ecx,ebp                                   | ecx:EntryPoint
004012F0  | E8 49010000           | call 0x40143E                                 |
004012F5  | 8BE5                  | mov esp,ebp                                   |
004012F7  | 5D                    | pop ebp                                       |
004012F8  | C3                    | ret                                           |
004012F9  | 807D C9 33            | cmp byte ptr ss:[ebp-0x37],0x33               | k[5]=3
004012FD  | 75 CF                 | jne 0x4012CE                                  |
004012FF  | 807D C8 35            | cmp byte ptr ss:[ebp-0x38],0x35               | k[4]=5
00401303  | 75 C9                 | jne 0x4012CE                                  |
00401305  | 807D C7 33            | cmp byte ptr ss:[ebp-0x39],0x33               | k[3]=3
00401309  | 75 C3                 | jne 0x4012CE                                  |
0040130B  | 0FB64D C4             | movzx ecx,byte ptr ss:[ebp-0x3C]              | ecx=k[0]
0040130F  | 0FB645 C5             | movzx eax,byte ptr ss:[ebp-0x3B]              | eax=k[1]
00401313  | 03C8                  | add ecx,eax                                   | ecx=k[0]+k[1]
00401315  | 0FB645 C6             | movzx eax,byte ptr ss:[ebp-0x3A]              | eax=k[2]
00401319  | 03C8                  | add ecx,eax                                   | ecx=k[0]+k[1]+k[2]
0040131B  | 81F9 95000000         | cmp ecx,0x95                                  | ecx:EntryPoint
00401321  | 75 AB                 | jne 0x4012CE                                  | 不相等就错了
00401323  | 33C9                  | xor ecx,ecx                                   | ecx=0
00401325  | 85D2                  | test edx,edx                                  | edx:EntryPoint
00401327  | 74 19                 | je 0x401342                                   | 判断输入是否为空?
00401329  | 0F1F80 00000000       | nop dword ptr ds:[eax],eax                    |
00401330  | 0FB6440D C4           | movzx eax,byte ptr ss:[ebp+ecx-0x3C]          | eax=k[ecx]
00401335  | C1E6 04               | shl esi,0x4                                   | esi:EntryPoint
00401338  | 83C6 D0               | add esi,0xFFFFFFD0                            | esi:EntryPoint
0040133B  | 03F0                  | add esi,eax                                   | esi=0x序列号
0040133D  | 41                    | inc ecx                                       | ecx:EntryPoint
0040133E  | 3BCA                  | cmp ecx,edx                                   | ecx:EntryPoint, edx:EntryPoint
00401340  | 72 EE                 | jb 0x401330                                   |
00401342  | C745 FC 00000000      | mov dword ptr ss:[ebp-0x4],0x0                |
00401349  | 85F6                  | test esi,esi                                  | esi:EntryPoint
0040134B  | 74 10                 | je 0x40135D                                   | esi=0 就错了
0040134D  | 50                    | push eax                                      |
0040134E  | E8 01000000           | call 0x401354                                 |
00401353  | EB 58                 | jmp 0x4013AD                                  |
00401355  | 83E8 00               | sub eax,0x0                                   |
00401358  | 2BF0                  | sub esi,eax                                   | esi:EntryPoint
0040135A  | F7F6                  | div esi                                       | esi:EntryPoint
0040135C  | 58                    | pop eax                                       |
```

分析到这，可以得出结论：序列号六位，前三位ASCII码相加等于0x95，后三位是353.
我输入了500353（满足条件）重新运行，最终将序列号转为一个地址：esi=0x500353，最终在这里卡死了

```
00401349  | 85F6                  | test esi,esi                                  | esi:EntryPoint
0040134B  | 74 10                 | je 0x40135D                                   | 这里相等肯定就是错了
```

这里相等肯定就失败了，不等的话，分析了一下，好像也直接就到打印错误的函数了，卡了老半天

```
00401342    C745 FC 00000000    mov dword ptr ss:[ebp-0x4],0x0
00401349    85F6                test esi,esi                    esi:EntryPoint
0040134B    74 10               je 0x40135D                     esi=0 就错了
0040134D    50                  push eax
0040134E    E8 01000000         call 0x401354
00401353    90                  nop                             去除花指令
00401354    58                  pop eax
00401355    83E8 00             sub eax,0x0
00401358    2BF0                sub esi,eax                     esi:EntryPoint
0040135A    F7F6                div esi                         esi:EntryPoint
0040135C    58                  pop eax
0040135D    90                  nop                             相等跳到这里
0040135E    90                  nop
0040135F    90                  nop
00401360    90                  nop
00401361    90                  nop
00401362    90                  nop
00401363    90                  nop
00401364    68 14C74100         push mysteriousletter2.41C714   41C714:"error!\n"
00401369    E8 A2000000         call 0x401410
```


分析的流程没错，那么这里肯定有办法直接跳到执行成功的代码，逐行分析，CALL POP其实就是eax=0x401353，这个值很有意思，看起来也是序列号,和输入的序列号相减，不等于0就错了，等于0就会引发除0异常

```
00401349    85F6                test esi,esi                    esi:EntryPoint
0040134B    74 10               je 0x40135D                     esi=0 就错了
0040134D    50                  push eax
0040134E    E8 01000000         call 0x401354
00401353    90                  nop                             去除花指令
00401354    58                  pop eax                         eax=0x401353  有点意思
00401355    83E8 00             sub eax,0x0
00401358    2BF0                sub esi,eax                     esi-eax
0040135A    F7F6                div esi                         除0异常行不行???
0040135C    58                  pop eax
0040135D    90                  nop                             相等跳到这里
```

因为刚开始学的就是OD，习惯了刚上来就往OD里扔，其实这里用IDA比较快速定位问题所在，找到main函数可以很容易看到一开始就在注册异常处理函数

```
.text:00401260 ; __unwind { // __except_handler4
.text:00401260                 push    ebp
.text:00401261                 mov     ebp, esp
.text:00401263                 push    0FFFFFFFEh
.text:00401265                 push    offset stru_41CC98
.text:0040126A                 push    offset __except_handler4
.text:0040126F                 mov     eax, large fs:0
.text:00401275                 push    eax
.text:00401276                 sub     esp, 2Ch
.text:00401279                 mov     eax, ___security_cookie
.text:0040127E                 xor     [ebp+ms_exc.registration.ScopeTable], eax
.text:00401281                 xor     eax, ebp
.text:00401283                 mov     [ebp+var_1C], eax
.text:00401286                 push    ebx
.text:00401287                 push    esi
.text:00401288                 push    edi
.text:00401289                 push    eax
.text:0040128A                 lea     eax, [ebp+ms_exc.registration]
.text:0040128D                 mov     large fs:0, eax
.text:00401293                 mov     [ebp+ms_exc.old_esp], esp
.text:00401296                 xor     esi, esi
.text:00401298                 push    offset asc_41C6F8 ; "请输入序列号:\n"
.text:0040129D                 call    sub_401410
.text:004012A2                 lea     eax, [ebp+var_3C]
.text:004012A5                 push    eax
.text:004012A6                 push    offset aS       ; "%s"
.text:004012AB                 call    sub_4013D0
.text:004012B0                 add     esp, 0Ch
```



```
.text:00402120 SEH_4152F0:
.text:00402120                 push    ebp
.text:00402121                 mov     ebp, esp
.text:00402123                 sub     esp, 1Ch
.text:00402126                 push    ebx
.text:00402127                 mov     ebx, [ebp+TargetFrame]
```

```
.text:0040212A                push    esi
.text:0040212B                push    edi
.text:0040212C                mov     [ebp+var_1], 0
.text:00402130                mov     eax, [ebx+8]
.text:00402133                lea     esi, [ebx+10h]
.text:00402136                xor     eax, ___security_cookie
.text:0040213C                push    esi
.text:0040213D                push    eax
.text:0040213E                mov     [ebp+var_C], 1
.text:00402145                mov     [ebp+var_10], esi
.text:00402148                mov     [ebp+var_8], eax
.text:0040214B                call    _ValidateLocalCookies
.text:00402150                mov     edi, [ebp+arg_8]
.text:00402153                push    edi
.text:00402154                call    nullsub_1
.text:00402159                mov     eax, [ebp+ExceptionRecord]
.text:0040215C                add     esp, 0Ch
.text:0040215F                test    byte ptr [eax+4], 66h
.text:00402163                jnz     loc_40221D
.text:00402169                mov     [ebp+var_1C], eax
.text:0040216C                lea     eax, [ebp+var_1C]
.text:0040216F                mov     [ebp+var_18], edi
.text:00402172                mov     edi, [ebx+0Ch]
.text:00402175                mov     [ebx-4], eax
.text:00402178                cmp     edi, 0FFFFFFFEh
.text:0040217B                jz      loc_402241
```


分析总结

序列号：401353

无壳的程序还是用IDA分析比较好，一键`F5`就可以了解程序的大致流程，分析也比较快速。在IDA`F5`失败或者具体细节不清楚的时候，在用OD动态分析。