

花指令专栏

花指令

张程思

注:本栏只去除花指令,后面的内容没继续进行

—,gdpctf

1.打开后f5编译不出来什么东西,猜测有花指令

```

r Lumina Options Windows Help
No debugger
Instruction Data Unexplored External symbol Lumina function
IDA View-A Hex View-1 Local Types Imports Exports
Seg
.te .text:000000014000194C call $+5
.te .text:0000000140001951
.te .text:0000000140001951 fake:
.te .text:0000000140001951 add [rsp+86h+var_86], 12h
.te .text:0000000140001956 ret
.te .text:0000000140001957
.te .text:0000000140001957
.te .text:0000000140001961 mov rax, 1400019A0h
.te .text:0000000140001961 jmp rax
.te .text:0000000140001963
.te .text:0000000140001963
.te .text:000000014000196A mov [rbp+var_30], 0
.te .text:0000000140001972 mov [rbp+hHandle], 0
.te .text:0000000140001976 lea rdx, [rbp+var_30]
.te .text:000000014000197A lea rax, [rbp+dwTlsIndex]
.te .text:000000014000197D mov rcx, rax
.te .text:0000000140001982 call InitAntiDebugSystem
.te .text:0000000140001982 mov [rbp+hHandle], rax
.te .text:0000000140001986 mov ecx, 3E8h ; dwMilliseconds
.te .text:0000000140001988 mov rax, cs: __imp_Sleep
.te .text:0000000140001992 call rax ; __imp_Sleep
.te .text:0000000140001994 xor eax, eax
.te .text:0000000140001996 jz short $+2
.te .text:0000000140001998
.te .text:0000000140001998 loc_140001998: ; CODE XREF: main+621j
.te .text:0000000140001998 xor eax, eax
.te .text:000000014000199A jz short loc_1400019A0
.te .text:000000014000199C add rsp, 11h
.te .text:00000001400019A0
.te .text:00000001400019A0 loc_1400019A0: ; CODE XREF: main+661j
.te .text:00000001400019A7 mov [rbp+var_60], 6A43DA9Dh
.te .text:00000001400019A7 mov [rbp+var_5C], 2B28BE89h
.te .text:00000001400019A7 mov [rbp+var_58], 0B2DEA67Ch
.te .text:00000001400019B5 mov [rbp+var_54], 64CF55F5h
.te .text:00000001400019B5 mov [rbp+var_50], 11F93DB3h
.te .text:00000001400019B5 mov [rbp+var_4C], 0A2D216E5h
.te .text:00000001400019CA mov [rbp+var_48], 8B69A237h
.te .text:00000001400019D1 mov [rbp+var_44], 0B080EF17h
.te .text:00000001400019D8 mov [rbp+var_40], 5A81C468h
.te .text:00000001400019DF mov [rbp+var_3C], 0CE0E998Eh
.te .text:00000001400019E6 mov [rbp+var_38], 24h ; '
.te .text:00000001400019ED lea rax, REAL_KEY+10h ; "welcome to GDPTCTF"
.te .text:00000001400019F4 mov rcx, rax ; Buffer
.te .text:00000001400019F7 call puts
.te .text:00000001400019FC mov ecx, 0 ; Time
.te .text:0000000140001A01 call __time64
00000FA0 00000001400019A0: main:loc_1400019A0 (Synchronized with Hex View-1)
-----
5, Apr 2 2024, 10:12:12] [MSC v.1938 64 bit (AMD64)]
APython Team <idapython@googlegroups.com>
-----
4 7-14
been propagated
```

IDA View-A Pseudocode-A Hex View-1

```
1 int __fastcall main(int argc, const char **argv, const char **envp)
2 {
3     _main();
4     return 0;
5 }
```

Instruction Data Unexplored External symbol Lum

IDA View-A

```
1 void __cdecl _main()
2 {
3     if ( !initialized )
4     {
5         initialized = 1;
6         _do_global_ctors();
7     }
8 }
```

2,断点调试,打上nop,根据f8的跳跃位置来nop

Hex View-1

00007FF76D0D18F0 00 00 48 8B 05 87 69 01 00 FF D0 48 89 45 F0 48 ..H..i.....E..

00007FF76D0D1900 83 7D F0 00 75 24 48 8B 45 10 8B 00 89 C1 48 8B .)....H.E.....

00007FF76D0D1910 05 CB 69 01 00 FF D0 48 8B 45 F8 48 89 C1 E8 5DE.H...]

00007FF76D0D1920 F9 00 00 B8 00 00 00 00 EB 04 48 8B 45 F0 48 83E...]

00007FF76D0D1930 64 40 5D C3 55 48 8B 45 48 81 FC 00 00 00 00 00 1 H

Stack view

00:0000 000000FA7AFFFCF0

01:0008 000000FA7AFFFCF8

02:0010 000000FA7AFFFD00

03:0018 000000FA7AFFFD08

04:0020 000000FA7AFFFD10

```

.text:00007FF76D0D1968 nop
.text:00007FF76D0D1969 nop
.text:00007FF76D0D196A mov     [rbp+hHandle], 0
.text:00007FF76D0D1972 lea     rdx, [rbp+var_30] ; _QWORD
.text:00007FF76D0D1976 lea     rax, [rbp+var_30+4]
.text:00007FF76D0D197A mov     rcx, rax ; p_dwTlsIndex
.text:00007FF76D0D197D call    InitAntiDebugSystem
.text:00007FF76D0D1982 mov     [rbp+hHandle], rax
.text:00007FF76D0D1986 mov     ecx, 3E8h ; dwMilliseconds
.text:00007FF76D0D1988 mov     rax, cs: __imp_Sleep
.text:00007FF76D0D1992 call    rax ; __imp_Sleep
.text:00007FF76D0D1994 xor     eax, eax
.text:00007FF76D0D1996 jz      short $+2
.text:00007FF76D0D1998
.text:00007FF76D0D1998 loc_7FF76D0D1998: ; CODE XREF: main+621j
.text:00007FF76D0D1998 xor     eax, eax
.text:00007FF76D0D199A jz      short loc_7FF76D0D19A0
.text:00007FF76D0D199C nop
.text:00007FF76D0D199D nop
.text:00007FF76D0D199E nop
.text:00007FF76D0D199F nop
.text:00007FF76D0D19A0
.text:00007FF76D0D19A0 loc_7FF76D0D19A0: ; CODE XREF: main+661j
.text:00007FF76D0D19A0 mov     dword ptr [rbp+var_60], 6A43DA9Dh
.text:00007FF76D0D19A7 mov     dword ptr [rbp+var_60+4], 2B28BE89h
.text:00007FF76D0D19AE mov     [rbp+var_58], 0B2DEA67Ch
.text:00007FF76D0D19B5 mov     [rbp+var_54], 64CF55F5h
.text:00007FF76D0D19BC mov     [rbp+var_50], 11F93DB3h
.text:00007FF76D0D19C3 mov     [rbp+var_4C], 0A2D216E5h
00000F7A 00007FF76D0D197A: main+46 (Synchronized with RIP)

```

3,nop掉之后p键创建main函数就可以f5查看反编译了

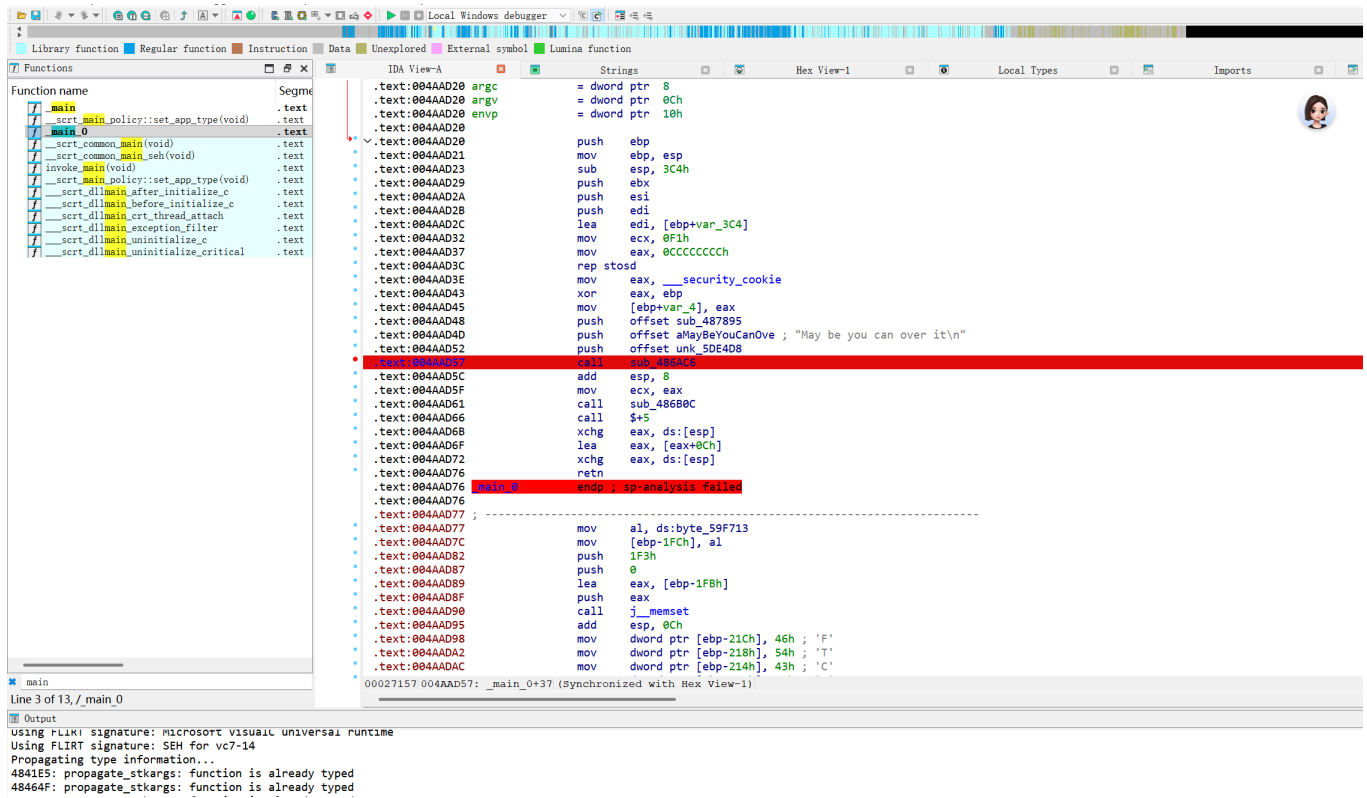
```

00007FF76D0D1934
00007FF76D0D1934 ; Attributes: bp-based frame
00007FF76D0D1934
00007FF76D0D1934 ; int __fastcall main(int argc, const char **argv, const char **envp)
00007FF76D0D1934 public main
00007FF76D0D1934 main proc near ; CODE XREF: __tmainCRTStartup+B41p
00007FF76D0D1934 ; DATA XREF: .pdata:00007FF76D0E506C10 ...
00007FF76D0D1934
00007FF76D0D1934 var_64= dword ptr -64h
00007FF76D0D1934 var_60= qword ptr -60h
00007FF76D0D1934 var_58= dword ptr -58h
00007FF76D0D1934 var_54= dword ptr -54h
00007FF76D0D1934 var_50= dword ptr -50h
00007FF76D0D1934 var_4C= dword ptr -4Ch
00007FF76D0D1934 var_48= dword ptr -48h
00007FF76D0D1934 var_44= dword ptr -44h
00007FF76D0D1934 var_40= dword ptr -40h
00007FF76D0D1934 var_3C= dword ptr -3Ch
00007FF76D0D1934 var_38= dword ptr -38h
00007FF76D0D1934 var_30= qword ptr -30h
00007FF76D0D1934 var_28= qword ptr -28h
00007FF76D0D1934 var_20= dword ptr -20h
00007FF76D0D1934 n10= qword ptr -1Ch
00007FF76D0D1934 var_14= dword ptr -14h
00007FF76D0D1934 hHandle= qword ptr -10h
00007FF76D0D1934 var_4= dword ptr -4
00007FF76D0D1934
00007FF76D0D1934 push     rbp
00007FF76D0D1935 mov     rbp, rsp
00007FF76D0D1934: main (Synchronized with RIP)

```

二,一杯花茶

1,ida中main无法正常编译,并且看到有call加几这样类似的东西,直接断点调试



The screenshot shows the IDA Pro interface with the main function selected. The Functions list on the left shows the main function at address 00401000. The main window displays the assembly code for the main function, starting with a call instruction at address 00401000. The instruction is highlighted in red, and the comment below it reads "00027157: 00401000: _main_0+37 (synchronized with Hex View-1)". The output window at the bottom shows the following messages:

```
Using FLIRT signature: Microsoft Visual C++ Runtime  
Using FLIRT signature: SEH for vc7-14  
Propagating type information...  
4841E5: propagate_stkargs: function is already typed  
48464F: propagate_stkargs: function is already typed
```

2,nop掉一个call和一个return再创建个函数就可以看到正确的伪代码了

```
.text:004AAD45 mov [ebp+var_4], eax
.text:004AAD48 push offset p_sub_487895 ; p_sub_487895
.text:004AAD4D push offset aMayBeYouCanOve ; "May be you can over it\n"
.text:004AAD52 push offset off_5DE4D8 ; _DWORD
.text:004AAD57 call sub_486AC6
.text:004AAD5C add esp, 8
.text:004AAD5F mov ecx, eax
.text:004AAD61 call sub_486B0C
.text:004AAD66 nop
.text:004AAD67 nop
.text:004AAD68 nop
.text:004AAD69 nop
.text:004AAD6A nop
.text:004AAD6B xchg eax, ds:[esp]
.text:004AAD6F lea eax, [eax+0Ch]
.text:004AAD72 xchg eax, ds:[esp]
.text:004AAD76 nop
.text:004AAD77 mov al, ds:byte_59F713
EIP .text:004AAD7C mov [ebp+Str], al
.text:004AAD82 push 1F3h ; Size
.text:004AAD87 push 0 ; Val
.text:004AAD89 lea eax, [ebp+var_1F8]
.text:004AAD8F push eax ; void *
.text:004AAD90 call j__memset
.text:004AAD95 add esp, 0Ch

IDA View-EIP Pseudocode-A
1 int __cdecl main_0(int argc, const char **argv, const char **envp)
2 {
3     _BYTE v5[12]; // [esp+0h] [ebp-3D0h] BYREF
4     size_t n23; // [esp+190h] [ebp-240h]
5     _DWORD v7[6]; // [esp+19Ch] [ebp-234h] BYREF
6     _DWORD v8[8]; // [esp+1B4h] [ebp-21Ch] BYREF
7     char Str[504]; // [esp+1D4h] [ebp-1FCh] BYREF
8
9     sub_486AC6(&off_5DE4D8, "May be you can over it\n");
10    _InterlockedExchange(MK_FP(__DS__, v5), _InterlockedExchange(MK_FP(__DS__, v5), sub_486B0C(sub_487895) + 12);
11    memset(Str, 0, 500);
12    v8[0] = 70;
13    v8[1] = 84;
14    v8[2] = 67;
15    v8[3] = 83;
16    v8[4] = 65;
17    v8[5] = 68;
18    v7[0] = -1161949172;
19    v7[1] = -98537090;
20    v7[2] = -1796902650;
21    v7[3] = 539168800;
22    j__puts("Give me your flag:");
23    sub_48531F(&dword_5DE430, Str);
24    n23 = j__strlen(Str);
25    if ( n23 == 23 )
26    {
27        sub_488123(Str, v8, v7);
28        if ( (unsigned __int8)sub_485C84(Str) )
29            sub_486AC6(&off_5DE4D8, "Good");
30    }
31}
00027120 _main_0:1 (4AAD20)
```

```
Hex View-1
004AAD50 59 00 68 D8 E4 5D 00 E8 6A BD FD FF 83 C4 08 8B Y.h..].....
004AAD60 C8 E8 A6 BD FD FF 90 90 90 90 3E 87 04 24 8D .....>..$.
004AAD70 40 0C 3E 87 04 24 90 A0 13 F7 59 00 88 85 04 FE @.>..$.
004AAD80 FF FF 68 F3 01 00 00 6A 00 8D 85 05 FE FF FF 50 ..h...j.....P
004AAD90 E8 50 A8 FD FF 83 C4 0C C7 85 E4 FD FF FF 46 00 ..h...T.....F.
004AADA0 00 00 C7 85 E8 FD FF FF 54 00 00 00 C7 85 EC FD ..h...T.....F.
004AADB0 FF FF 43 00 00 00 C7 85 F0 FD FF FF 53 00 00 00 ..C...h...S...
```

00:0000	0019F
01:0004	0019F
02:0008	0019F
03:000C	0019F
04:0010	0019F
05:0014	0019F
06:0018	0019F

≡,litectf

The screenshot displays the Immunity Debugger interface with the following components:

- Assembly Window:** Shows the assembly code for `sub_41114A`. The instruction `push offset sub_41114A` is highlighted. The comment `; DATA XREF: sub_413D30+2D↑o` is visible.
- Stack Trace:** Shows the current frame at `00401000` with a return address of `00401000`.
- Registers:** Shows the `EIP` register pointing to `00401000`.
- Disassembly:** Shows the instruction `push offset sub_41114A` at address `00401000`.
- Comments:** A comment `; DATA XREF: sub_413D30+2D↑o` is visible next to the instruction.

```

ext:00414FB5      mov     [ebp-4], eax
ext:00414FB8      mov     dword ptr [ebp-18h], 11223344h
ext:00414FBF      mov     dword ptr [ebp-14h], 55667788h
ext:00414FC6      mov     dword ptr [ebp-10h], 99AABBCCh
ext:00414FCD      mov     dword ptr [ebp-0Ch], 0DDEEFF11h
ext:00414FD4      mov     dword ptr [ebp-50h], 977457FEh
ext:00414FDB      mov     dword ptr [ebp-4Ch], 0DA3E1880h
ext:00414FE2      mov     dword ptr [ebp-48h], 0B8169108h
ext:00414FE9      mov     dword ptr [ebp-44h], 1E95285Ch
ext:00414FF0      mov     dword ptr [ebp-40h], 1FE7E6F2h
ext:00414FF7      mov     dword ptr [ebp-3Ch], 2BC5FC57h
ext:00414FFE      mov     dword ptr [ebp-38h], 0B28F0FA8h
ext:00415005      mov     dword ptr [ebp-34h], 8E0E0644h
ext:0041500C      mov     dword ptr [ebp-30h], 68454425h
ext:00415013      mov     dword ptr [ebp-2Ch], 0C57740D9h
ext:0041501A      xor     eax, eax
ext:0041501C      mov     [ebp-28h], eax
ext:0041501F      mov     [ebp-24h], eax
ext:00415022      mov     dword ptr [ebp-5Ch], 0
ext:00415029      push    64h ; 'd'
ext:0041502B      push    0
ext:0041502D      lea     eax, [ebp-0C8h]
ext:00415033      push    eax
ext:00415034      call    j_memset
ext:00415039      add     esp, 0Ch
ext:0041503C      push    offset unk_41DC74
ext:00415041      mov     eax, ds:cout@std@@@3V?$basic_ostream@DU?$char_traits@D@std@@@1@A ; std::ostream std::cout
ext:00415046      push    eax
ext:00415047      call    sub_41123A
ext:0041504C      add     esp, 8
ext:0041504F      lea     eax, [ebp-0C8h]
ext:00415055      push    eax
ext:00415056      mov     ecx, ds:cin@std@@@3V?$basic_istream@DU?$char_traits@D@std@@@1@A ; std::istream std::cin
ext:0041505C      push    ecx
ext:0041505D      call    sub_4110C8
ext:00415062      add     esp, 8
ext:00415065      jz      short near ptr loc_415069+1
ext:00415067      jnz     short near ptr loc_415069+1
ext:00415069      loc_415069:
ext:00415069      ; CODE XREF: .text:00415065↑j
ext:00415069      ; .text:00415067↑j
043B8 00414FB8: .text:00414FB8 (Synchronized with Hex View-1)

```

2,还是动态调试把花指令nop掉就行了