題目: nani

1. 助教提示說會有 anti-Debug

所以裝 ScyllaHide 來反制:

https://github.com/x64dbg/ScyllaHide/releases

安裝:

ScyllaHide\x64dbg\x64\plugins 下面的東西 copy 到 \x64dbg\release\x64\plugins

開啟 x64dbg\release\x64\x64dbg.exe

就可以在上面的外掛程式那邊看到 ScyllaHide

2. 助教提示會有 anti-vm:

   因為 anti-vm 會比對 register 的內容,而內容會比對 VMwareVMware 這樣的字串, shift+F12 收尋 VMwareVMware,在按 x 找到 ref 這個的 code,就可以找到以下 function

```
v9 = -1;
v13 = 0x40000000;
v12 = 0;
Str2[0] = "KVMKVMKVM";
Str2[1] = "Microsoft Hv";
Str2[2] = "VMwareVMware";
Str2[3] = "XenVMMXenVMM";
Str2[4] = "prl hyperv  ";
Str2[5] = "VBoxVBoxVBox";
v11 = 6;
_RAX = 0x40000000i64;
__asm { cpuid }
v10[0] = _RBX;
v10[1] = _RCX;
v10[2] = _RDX;
memset(Str1, 0, sizeof(Str1));
memcpy(Str1, v10, 0xCui64);
for ( i = 0; ; ++i )
{
  if ( i >= v11 )
    sub_4017DF();
  v5 = strcmp(Str1, Str2[i]);
  v12 = v5 == 0;
  if ( !v5 )
    break;
}
sub_401550((int)"[!] %s\n");
sub_401550((int)"You use VM ... bad reverse engineer :(((\n");
return 1i64;
}
```

   可以看到他在比對,如果有符合的字串就跳出 for , 如果都沒有就進入 sub_4017DF,看起來這個 function 就是拿 flag 的 function

3. 我們可以發現裡面有一個 sub_4A0D40 裡面會有 function 會丟出 exception 所以我們要避免進入這個 function,並且助教有提示 anti-disassembly,我們觀察 sub_4A0D40 附近的 code

```
.text:000000000040180C                  mov     r8, cs:off_4A9B90
.text:0000000000401813                  lea     rdx, off_4AC810
.text:000000000040181A                  mov     rcx, rbx
.text:000000000040181D                  call    sub_4A0D40
.text:000000000040181D sub_4017DF       endp
.text:000000000040181D
.text:0000000000401822 ; ---------------------------------------------------------------------------
.text:0000000000401822                  mov     rsi, rax
.text:0000000000401825                  mov     rcx, rbx
.text:0000000000401828                  call    sub_4A0910
.text:000000000040182D                  mov     rax, rsi
.text:0000000000401830                  jmp     short $+2
.text:0000000000401832 ; ---------------------------------------------------------------------------
.text:0000000000401832
.text:0000000000401832 loc_401832:                              ; CODE XREF: .text:000000
.text:0000000000401832                  mov     rcx, rax
.text:0000000000401835                  call    sub_4A0640
.text:000000000040183A                  lea     rcx, unk_4A5002
.text:0000000000401841                  call    sub_401550
.text:0000000000401846                  call    sub_4A0860      ; exit the program
.text:000000000040184B                  jmp     short loc_401861
.text:000000000040184D ; ---------------------------------------------------------------------------
```

可以發現 401830 的 jmp 可能造成 disassemble 沒有成功,所以下面的 code 看起來是沒發生 exception 會執行的,所以我們開 x64dbg,斷點放在,40181D,f9 到了之後改 rip 到 401822,繞過 exception,接著執行到 401846 發現這邊也會造成程式結束,我們 f7 進去看,trace 到一個地方長這樣

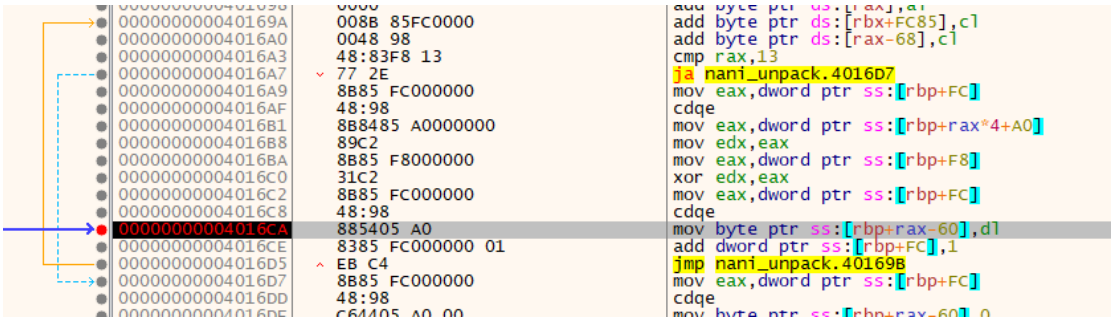
```
00000000004015B7    48:8DAC24 80000000          lea rbp,qword ptr ss:[rsp+80]
00000000004015BF    C785 A0000000 E8000000      mov dword ptr ss:[rbp+A0],E8
00000000004015C9    C785 A4000000 E2000000      mov dword ptr ss:[rbp+A4],E2
00000000004015D3    C785 A8000000 EF000000      mov dword ptr ss:[rbp+A8],EF
00000000004015DD    C785 AC000000 E9000000      mov dword ptr ss:[rbp+AC],E9
00000000004015E7    C785 B0000000 D5000000      mov dword ptr ss:[rbp+B0],D5
00000000004015F1    C785 B4000000 DC000000      mov dword ptr ss:[rbp+B4],DC
00000000004015FB    C785 B8000000 9D000000      mov dword ptr ss:[rbp+B8],9D
0000000000401605    C785 BC000000 D8000000      mov dword ptr ss:[rbp+BC],D8
000000000040160F    C785 C0000000 9D000000      mov dword ptr ss:[rbp+C0],9D
0000000000401619    C785 C4000000 DC000000      mov dword ptr ss:[rbp+C4],DC
0000000000401623    C785 C8000000 DD000000      mov dword ptr ss:[rbp+C8],DD
000000000040162D    C785 CC000000 9D000000      mov dword ptr ss:[rbp+CC],9D
0000000000401637    C785 D0000000 F1000000      mov dword ptr ss:[rbp+D0],F1
0000000000401641    C785 D4000000 E3000000      mov dword ptr ss:[rbp+D4],E3
000000000040164B    C785 D8000000 CF000000      mov dword ptr ss:[rbp+D8],CF
0000000000401655    C785 DC000000 9B000000      mov dword ptr ss:[rbp+DC],9B
000000000040165F    C785 E0000000 FA000000      mov dword ptr ss:[rbp+E0],FA
0000000000401669    C785 E4000000 9D000000      mov dword ptr ss:[rbp+E4],9D
0000000000401673    C785 E8000000 FC000000      mov dword ptr ss:[rbp+E8],FC
000000000040167D    C785 EC000000 D3000000      mov dword ptr ss:[rbp+EC],D3
0000000000401687    C785 F8000000 AE000000      mov dword ptr ss:[rbp+F8],AE
0000000000401691    C785 FC000000 00000000      mov dword ptr ss:[rbp+FC],0
000000000040169B    8B85 FC000000               mov eax,dword ptr ss:[rbp+FC]
00000000004016A1    48:98                       cdqe
```

看起來非常像在算 flag,並且可以看到這個位置對應到的地方如下圖

```
.text:00000000004015AE ; ---------------------------------------------------------------------------
.text:00000000004015AF byte_4015AF      db 0D2h                 ; DATA XREF: sub_4016FB+18↓o
.text:00000000004015AF                                          ; sub_4016FB:loc_4017C2↓o
.text:00000000004015B0                  dq 0CF873786076B06CFh, 4087878707A32B0Ah, 87876F8787872702h
.text:00000000004015B0                  dq 6587878723024087h, 87872F0240878787h, 2B02408787876887h
.text:00000000004015B0                  dq 408787876E878787h, 8787528787873702h, 5B87878733024087h
.text:00000000004015B0                  dq 87873F0240878787h, 3B02408787871A87h, 408787875F878787h
.text:00000000004015B0                  dq 87871A8787874702h, 5B87878743024087h, 87874F0240878787h
.text:00000000004015B0                  dq 4B02408787875A87h, 408787871A878787h, 8787768787875702h
.text:00000000004015B0                  dq 6487878753024087h, 87875F0240878787h, 5B02408787874887h
.text:00000000004015B0                  dq 408787871C878787h, 87877D8787876702h, 1A87878763024087h
.text:00000000004015B0                  dq 87876F0240878787h, 6B02408787877B87h, 4087878754878787h
.text:00000000004015B0                  dq 8787298787877F02h, 878787877B024087h, 87877B020C878787h
.text:00000000004015B0                  dq 0F0947F04CF1FCF87h, 488787877B020CA9h, 0A085848B98h
.text:00000000004015B0                  dq 0F8858BC289h, 0FC858BC231h, 8583A00554889848h, 8BC4EB01000000FCh
.text:00000000004015B0                  dq 0C69848000000FC85h, 0A0458D4800A00544h, 0FFFFFEB4E8C18948h
.text:00000000004015B0                  dq 5C16E800000001B9h
.text:00000000004016F8 ; ---------------------------------------------------------------------------
```

可能也是透過 anti-diassemble 讓他解析不出 code

下面有段 code 用到迴圈



看到這段 code 把 rbp 裡面的拿出來算並且放到另一個地方,我們讓這段 for 執行幾個 iter 之後我們去看 rbp+rax-60,就可以看到 flag 在那邊了