【CSICTF】Vietnam WriteUp

原创

古月浪子 于 2020-07-23 16:02:54 发布 102 收藏

文章标签： CTF

本文链接：https://blog.csdn.net/tqydyqt/article/details/107466074

Vietnam

489

reversing

The Viet Cong in transmitting a secret message. They built a password checker so that only a selected few can view the secret message. We've recovered the binary, we need you to find out what they're trying to say.

nc chall.csivit.com 30814

vietnam

比较str是否等于HELLO\n，相等则获得flag

在伪代码中可以看到我们没有办法直接修改str

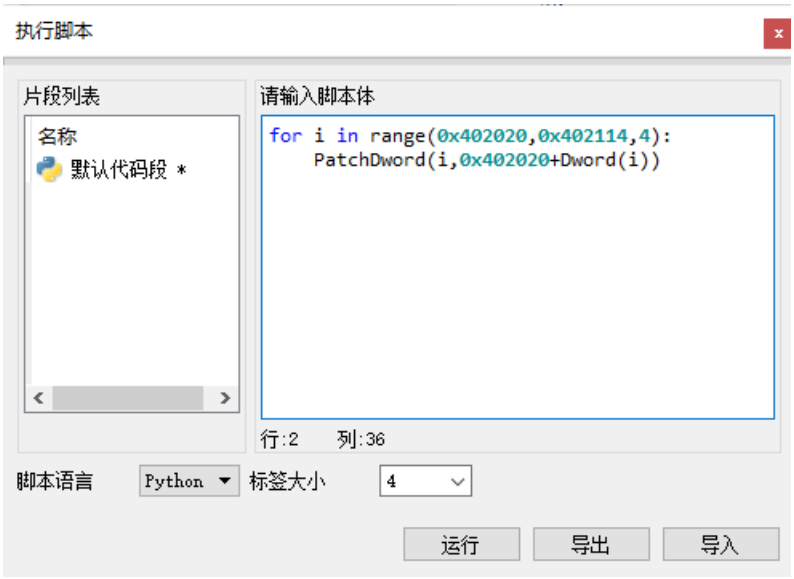
```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  unsigned int v3; // eax
  char *s; // [rsp+18h] [rbp-8h]

  s = malloc(0x400uLL);
  fgets(s, 1024, stdin);
  setbuf(_bss_start, 0LL);
  while ( *s )
  {
    v3 = *s - 33;
    if ( v3 <= 60 )
      JUMPOUT(__CS__, dword_402020 + dword_402020[v3]);
    ++s;
  }
  str = &STR;
  if ( !strcmp(&STR, "HELLO\n") )
  {
    puts(str);
    system("cat flag.txt");
  }
  else
  {
    puts("Failed.");
  }
  return 0;
}
```

这个jmp有点意思

```
.rodata:0000000000402020 dword_402020    dd 0FFFFF289h, 2 dup(0FFFFF390h), 0FFFFF1ECh, 6 dup(0FFFFF390h)
.rodata:0000000000402020                                         ; DATA XREF: main+6E↑o
.rodata:0000000000402020                                         ; main+7A↑o
.rodata:0000000000402020                 dd 0FFFFF20Dh, 0FFFFF2C6h, 0FFFFF24Bh, 0FFFFF2DBh, 2Ch dup(0FFFFF390h)
.rodata:0000000000402020                 dd 0FFFFF2FEh, 0FFFFF390h, 0FFFFF349h
```

这样肉眼看起来不太直观，写IDAPython脚本处理一下



这样就能直接从反汇编窗口中读出要跳转的地址了

```
.rodata:0000000000402020 dword_402020    dd 4012A9h, 2 dup(4013B0h), 40120Ch, 6 dup(4013B0h), 40122Dh
.rodata:0000000000402020                                         ; DATA XREF: main+6E↑o
.rodata:0000000000402020                                         ; main+7A↑o
.rodata:0000000000402020                 dd 4012E6h, 40126Bh, 4012FBh, 2Ch dup(4013B0h), 40131Eh
.rodata:0000000000402020                 dd 4013B0h, 401369h
```

可以看到，程序将输入的字符减去33然后当作下标索引，跳转到对应地址
其中0x4013b0重复出现了很多次，大概没啥用，直接忽略
分析一下其他地址是什么作用

```
.text:00000000004012E6 ; ---------------------------------------------------------------
.text:00000000004012E6                 call    _getchar
.text:00000000004012EB                 mov     edx, eax
.text:00000000004012ED                 mov     rax, cs:sa
.text:00000000004012F4                 mov     [rax], dl
.text:00000000004012F6                 jmp     loc_4013B7
.text:00000000004012FB ; ---------------------------------------------------------------
.text:00000000004012FB                 mov     rdx, cs:sa
.text:0000000000401302                 mov     rax, cs:str
.text:0000000000401309                 lea     rcx, [rax+1]
.text:000000000040130D                 mov     cs:str, rcx
.text:0000000000401314                 movzx   edx, byte ptr [rdx]
.text:0000000000401317                 mov     [rax], dl
.text:0000000000401319                 jmp     loc_4013B7
.text:000000000040131E ; ---------------------------------------------------------------
```

这里我本来汇总了一下有用的字符，但是好像分析了2个就觉得可以打穿这道题了
其他的字符对应的代码我大致看了一下，没看懂有什么作用

0x4012a9 ! 不知道
0x40120c $ 不知道
0x40122d + 不知道
0x4012e6 , 读取输入字符到sa
0x40126b - 不知道
0x4012fb . 把sa写入str，str指向下一位
0x40131e [不知道
0x401369] 不知道

因为我们要输入6个字符，因此读取6次，写入6次即可