XCTF 攻防世界 Reverse新手题（getit）

原创

sherlockjjobs 于 2020-12-13 16:24:05 发布 651 收藏 1

分类专栏： CTF 网络安全 渗透测试 文章标签： 反汇编 反编译 网络安全 渗透测试

本文链接：https://blog.csdn.net/weixin_43456810/article/details/111127072

XCTF 攻防世界 Reverse新手题（getit）

首先，判断文件有没有加壳，是多少位的，利用exeinfoPE可以得知，程序并没有加壳，并且是64位的elf程序

直接用IDA64打开，找到main函数，F5反编译一下：

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  char v3; // al
  __int64 v5; // [rsp+0h] [rbp-40h]
  int i; // [rsp+4h] [rbp-3Ch]
  FILE *stream; // [rsp+8h] [rbp-38h]
  char filename[8]; // [rsp+10h] [rbp-30h]
  unsigned __int64 v9; // [rsp+28h] [rbp-18h]

  v9 = __readfsqword(0x28u);
  LODWORD(v5) = 0;
  while ( (signed int)v5 < strlen(s) )
  {
    if ( v5 & 1 )
      v3 = 1;
    else
      v3 = -1;
    *(&t + (signed int)v5 + 10) = s[(signed int)v5] + v3;
    LODWORD(v5) = v5 + 1;
  }
  strcpy(filename, "/tmp/flag.txt");
  stream = fopen(filename, "w");
  fprintf(stream, "%s\n", u, v5);
  for ( i = 0; i < strlen(&t); ++i )
  {
    fseek(stream, p[i], 0);
    fputc(*(&t + p[i]), stream);
    fseek(stream, 0LL, 0);
    fprintf(stream, "%s\n", u);
  }
  fclose(stream);
  remove(filename);
  return 0;
}
```

简单分析一下代码，看到最后的for循环，猜测很有可能是输出flag的，所以先找到strlen()函数的地址400824，之后去汇编代码的界面，找到400824，确实是strlen函数：

```
.text:0000000000400824                call    _strlen
.text:0000000000400829                cmp     rbx, rax
.text:000000000040082C                jnb     loc_4008B5
.text:0000000000400832                mov     eax, [rbp+var_3C]
.text:0000000000400835                cdqe
```

回到伪代码，for循环的第一个函数是fseek()，因此jnb loc_4008B5很有可能就是fseek()，这一点确定之后，接下来便可以动态调试把400832设为断点进行尝试

```
pwndbg 文件名 //进入动态调试界面
b *0x400832 //设置断点的地址
r //运行
```

可以看到，运行一下直接就可以看到flag信息了：