南京邮电大学攻防平台 逆向writeup

原创

Aslani 于 2017-01-27 11:02:41 发布 2027 收藏 1

分类专栏： 逆向工程 文章标签： 逆向-writeup

本文链接：https://blog.csdn.net/Aslani/article/details/54754898

南京邮电大学攻防平台 逆向writeup

看题目说使用IDA，用f5直接反编译可以得到源码

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  _BYTE v4[3]; // [sp+11h] [bp-7Fh]@2
  signed int v5; // [sp+75h] [bp-1Bh]@1
  signed int v6; // [sp+79h] [bp-17h]@1
  signed int v7; // [sp+7Dh] [bp-13h]@1
  signed int v8; // [sp+81h] [bp-Fh]@1
  signed int v9; // [sp+85h] [bp-Bh]@1
  signed int v10; // [sp+89h] [bp-7h]@1
  signed __int16 v11; // [sp+8Dh] [bp-3h]@1
  char v12; // [sp+8Fh] [bp-1h]@1

  __main();
  printf("请输入flag: ");
  v5 = 'galf';
  v6 = 'leW{';
  v7 = 'emoc';
  v8 = '_oT_';
  v9 = 'W_ER';
  v10 = 'dlro';
  v11 = '}!';
  v12 = 0;
  while ( scanf("%s", v4) != -1 && strcmp(v4, (const char *)&v5) )
    printf("flag错误。再试试？\n");
  printf("flag正确。\n");
  printf("如果是南邮16级新生并且感觉自己喜欢逆向的话记得加群\n");
  printf("群号在ctf.nuptsast.com的to 16级新生页面里\n");
  printf("很期待遇见喜欢re的新生23333\n");
  getchar();
  getchar();
  return 0;
}
```

题目中给给提示说R可以将数字转为字符，以上代码V5-V11我已经进行了转换。
代码很好懂，当拟输入的字符串与以V5开头的字符串进行比较，若相同，则输出。注意一直比较到V11，字符串以V12进行结尾。字符串在内存中以小端进行存储
给出EXP

```
flag = ''
v = ["galf","leW{","emoc","_oT_","W_ER","dlro","}!"]
for i in range(7):
    str1 = v[i]
    str2 = str1[::-1]
    flag += str2;
print(flag)
```

运行结果`flag{Welcome_To_RE_World!}`

ReadASM

题目就是读汇编代码，这是题目中C语言的主程序

```
int main(int argc, char const *argv[])
{
  char input[] = {0x0,  0x67, 0x6e, 0x62, 0x63, 0x7e, 0x74, 0x62, 0x69, 0x6d,
                  0x55, 0x6a, 0x7f, 0x60, 0x51, 0x66, 0x63, 0x4e, 0x66, 0x7b,
                  0x71, 0x4a, 0x74, 0x76, 0x6b, 0x70, 0x79, 0x66 , 0x1c};
  func(input, 28);
  printf("%s\n",input+1);
  return 0;
}
```

这是调用的汇编函数

```
00000000004004e6 <func>:
  4004e6: 55                    push   rbp-0x4          ; 主程序栈帧入栈
  4004e7: 48 89 e5              mov    rbp,rsp          ; 建立子程序栈帧
  4004ea: 48 89 7d e8           mov    QWORD PTR [rbp-0x18],rdi     ; 第一个参数
  4004ee: 89 75 e4              mov    DWORD PTR [rbp-0x1c],esi     ; 第二个参数
  4004f1: c7 45 fc 01 00 00 00  mov    DWORD PTR [rbp-0x4],0x1
  4004f8: eb 28                 jmp    400522 <func+0x3c>
  4004fa: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
  4004fd: 48 63 d0              movsxd rdx,eax
  400500: 48 8b 45 e8           mov    rax,QWORD PTR [rbp-0x18]
  400504: 48 01 d0              add    rax,rdx          ; rax = input[1];
  400507: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]
  40050a: 48 63 ca              movsxd rcx,edx
  40050d: 48 8b 55 e8           mov    rdx,QWORD PTR [rbp-0x18]
  400511: 48 01 ca              add    rdx,rcx          ; rdx = input[1];
  400514: 0f b6 0a              movzx  ecx,BYTE PTR [rdx]
  400517: 8b 55 fc              mov    edx,DWORD PTR [rbp-0x4]
  40051a: 31 ca                 xor    edx,ecx          ; input[ax] = input[ax]^ax
  40051c: 88 10                 mov    BYTE PTR [rax],dl
  40051e: 83 45 fc 01           add    DWORD PTR [rbp-0x4],0x1
  400522: 8b 45 fc              mov    eax,DWORD PTR [rbp-0x4]
  400525: 3b 45 e4              cmp    eax,DWORD PTR [rbp-0x1c]
  400528: 7e d0                 jle    4004fa <func+0x14>
  40052a: 90                    nop
  40052b: 5d                    pop    rbp
  40052c: c3                    ret
```

看懂，然后写EXP

```
varIn = [0x0,  0x67, 0x6e, 0x62, 0x63, 0x7e, 0x74, 0x62, 0x69, 0x6d,0x55, 0x6a, 0x7f, 0x60, 0x51, 0x66,
ax = 0x1
ex = 0x1
while ax<28:
    # dx = varIn[0+ax]
    varIn[ax] ^= ax
    ax+=1
for i in range(len(varIn)-1):
    print(chr(varIn[i+1]),end='')
```

运行结果

`flag{read_asm_is_the_basic}`