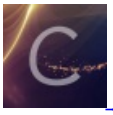
CTF writeup之pwn2own warehouse

原创

retme 于 2017-11-23 21:25:37 发布 409 收藏
分类专栏： ctf writeup pwn 文章标签： ctf writeup pwn



近来练习CTF，主要是PWN，有点小心得。分享一系列的writeup。

这题是pwn2own的一道pwn题--warehous。

一、题目分析

elf文件打开了NX和canary。

```
sean@ubuntu:~/ctf/warehouse$ checksec warehouse
[*] '/home/sean/ctf/warehouse/warehouse'
    Arch:     i386-32-little
    RELRO:    No RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
```


分析一下题目功能

题目还是比较简单的。在main函数里能调用store函数，并实现任意地址写。即可以跳过canary覆盖返回地址。并

二、ROP构造

1、储存“sh”字符串

2、计算并跳到system地址

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  int value; // ST14_4
  int index; // [esp+8h] [ebp-120h]
  char s2; // [esp+Ch] [ebp-11Ch]
  char start; // [esp+1Ch] [ebp-10Ch]
  unsigned int v8; // [esp+11Ch] [ebp-Ch]

  v8 = __readgsdword(0x14u);
  memset(&start, 0, 0x100u);
  while ( fgets(&s2, 16, _bss_start) )
  {
    if ( !strcmp(".\n", &s2) )
      break;
    index = atol(&s2);
    fgets(&s2, 16, _bss_start);
    if ( !strcmp(".\n", &s2) )
      break;
    value = atol(&s2);
    store((int)&start, index, value);
  }
  return 0;
}
```

```
int __cdecl store(int start, int index, int value)
{
  int result; // eax

  result = value;
  *(_DWORD *)(start + 4 * index) = value;
  return result;
}
```

储存sh字符串。有store函数，还是比较容易实现的。这里把"sh"存到一个可写地址。例如.dynamic段中,(用reade

这里碰到过问题，但是经过同事提点解决掉。

main函数的栈被写成如下。即可实现把sh写入目的地址。

……canary, store, 返回地址（EIP），para1, para2, para3，available1, available2, available3....

然而，store执行完后，会跳到返回地址中。而此时para1,para2,para3已经占住了栈，用普通的gatdget没法实现

要调用libc的system。这里又遇到一个问题。题目中居然一个put, write之类的函数都没有。没法常规地leak。

解决办法，利用ROPgadget居然找到一个极品的gadget。0x08048537 : add eax, dword ptr [eax + ebx*2] ; ret

只要实现eax是offset(如system和atol的offset), eax+ebx*2是 atol的GOT地址。则运算完后eax就是system的实际

划出最终的栈数据

addr_2_save_sh

call_eax

add_eax_and_[eax_2ebx]_ret

(atolGOT-offset)/2

pop_ebx_ret

offset_of_system_and_atol_in_libc(简称offset)

pop_eax_ret

string_of_sh

0

address_2_save_sh

pop_pop_pop_ret

store

canary

三、收获

1、收获一，控制EIP跳到函数的话与单纯的构造ROP有点不一样，返回地址在传参的低地址位置上。此时可把返

2、没有leak手段的话，可以用add 寄存器1，[寄存器2] 形式的gadget来实现

3、找可写地址，用readelf找一个能写的段，写进去。