pwnable-mistake

原创

galaxy3000 于 2022-02-02 11:04:08 发布 1236 收藏

分类专栏： # PWN 文章标签： pwnable CTF 网络安全 writeup 源代码审计

本文链接：https://blog.csdn.net/galaxy3000/article/details/122768259

PWN 专栏收录该内容

26 篇文章 0 订阅

订阅专栏

文章目录

概述

pwnable是一个经典的CTF中PWN方向练习的专业网站，本文记录的题目是mistake，主要考察的是C语言中操作符的优先级。

题目

题目描述

题目提示 **operator priority**，即操作符的优先级，题目连接信息为`ssh mistake@pwnable.kr -p2222 (pw:guest)`

连接信息

通过ssh登录靶机，查看家目录

```
mistake@pwnable:~$ ls -l
total 24
-r-------- 1 mistake_pwn root      51 Jul 29  2014 flag
-r-sr-x--- 1 mistake_pwn mistake 8934 Aug  1  2014 mistake
-rw-r--r-- 1 root        root     792 Aug  1  2014 mistake.c
-r-------- 1 mistake_pwn root      10 Jul 29  2014 password
```


基本信息

使用`file`查看基本信息

```
➜  pwnable file mistake
mistake: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, for
 GNU/Linux 2.6.24, BuildID[sha1]=ef56e67046843c3d794fda2e5842140e937dd7c6, not stripped
```

查看加固措施，发现Canary和NX启用

```
checksec mistake

    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    Canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

直接运行程序，发现能输入2次

```
mistake@pwnable:~$ ./mistake
do not bruteforce...
123

input password : 123
Wrong Password
```


查看源代码

mistake.c

```
#include <stdio.h>
#include <fcntl.h>

#define PW_LEN 10
#define XORKEY 1

void xor(char* s, int len){
    int i;
    for(i=0; i<len; i++){
        s[i] ^= XORKEY;
    }
}

int main(int argc, char* argv[]){

    int fd;
    if(fd=open("/home/mistake/password",O_RDONLY,0400) < 0){
        printf("can't open password %d\n", fd);
        return 0;
    }

    printf("do not bruteforce...\n");
    sleep(time(0)%20);

    char pw_buf[PW_LEN+1];
    int len;
    if(!(len=read(fd,pw_buf,PW_LEN) > 0)){
        printf("read error\n");
        close(fd);
        return 0;
    }

    char pw_buf2[PW_LEN+1];
    printf("input password : ");
    scanf("%10s", pw_buf2);

    // xor your input
    xor(pw_buf2, 10);

    if(!strncmp(pw_buf, pw_buf2, PW_LEN)){
        printf("Password OK\n");
        system("/bin/cat flag\n");
    }
    else{
        printf("Wrong Password\n");
    }

    close(fd);
    return 0;
}
```

源代码分析

pw_buf和pw_buf2长度均为10，程序本意是读取本地的password文件保存在变量pw_buf，输入的密码保存在pw_buf2，并通过xor()处理，然后pw_buf和pw_buf2比较，如果相等打印flag。

问题出在这句代码上

```
if(fd=open("/home/mistake/password",O_RDONLY,0400) < 0)
```

由于操作符优先级的问题，会优先执行`open("/home/mistake/password",O_RDONLY,0400) < 0`，然后再赋予变量`fd`。`open()`函数返回的结果大于0，因此`fd`的值为0，即`fd=0`为标准输入。

题目解法

因此，通过操作符优先级问题，通过标准输入控制pw_buf，如输入`0000000000`，也可以通过程序正常的输入控制pw_buf2，如输入`1111111111`，经过xor()处理后得到`0000000000`，使两者相等。

```
mistake@pwnable:~$ ./mistake
do not bruteforce...
0000000000
input password : 1111111111
Password OK
Mommy, the operator priority always confuses me :(
```