HGame2024 Week3 Writeup

Crypto

exRSA

题目

```
from Crypto.Util.number import *
from secret import flag
m=bytes_to_long(flag)
p=getStrongPrime(1024)
q=getStrongPrime(1024)
phi=(p-1)*(q-1)
e1=inverse(getPrime(768),phi)
e2=inverse(getPrime(768),phi)
e3=inverse(getPrime(768),phi)
n=p*q
c=pow(m,0x10001,n)
print(f'e1={e1}')
print(f'e2={e2}')
print(f'e3={e3}')
print(f'c={c}')
print(f'n={n}')

"""
e1=507704823781196942747311122537087612252896744705655189912361346179268800289678839430419291761056414976625223228157699029348523968414531087693099791896007081696882915037687595340542080958626715317171749619833686108952370183209832228450193114288981757581676170504495170553084932792884984815864303069336314375706322058471492589396558796704213755780726115411791635851947796464529347197506336205069030635362749298086100843976536583762265797795806985328805630725316750988325812294988227702166531780725330890635567047217234617117726768806495939718692610398725955158662796540697911819348552752097674849072846016794905528953 9
e2=125268482983490053905202769239291324634591525749986257572082592978911151336541176482157829453325290813652738603162011307933065707777350765347721689997058956412075353038394550740030576878103811109783209889760113261069199407991609742283118247600463702735055110656192685576971825862592343792394104827844498157323352943956763022264168637093400329876127151519160842918210954626258210231335604153258248853472213914969372132463617363612708467411285575956030527136125284537099484031007112776796412185204298788975656554820864105763799714047892122976975537482924381830655009933750400317338254966927976993624210102715995102694013 e3=129859407575785308105193703320636583440466888566059674749410144368727203604440404646447909809769913939709470233983574222038732842948434011440650139114636705015598886011451086519610983482508241666976655284176683744088145729597227890201103962450762755535058785656035094662207102192600377838492764753972834210687161608863818699477815354281768196305958165110356357880414515615758433671267888299568563261568685398017604768332697428389634332298152115021131759757155454248892129015812263414057114803673289380806411904832885513405470912087789594167016642166480618671034682449405478302573347589808124782488796755041850903827627 9
"""
```

```
c=14141760601523018421104970980245971892462591720193354149001274520982339430418259260285174370753162949433553239474589280105569129091397392829242555066473056968729078989504731085564173501997831453496910872559262873632869220118411433395308633001982392314907073933830761747918189941588158573919308029362804475888084406074153773913366045334400997938492378572475575823073913293205159960218200003555605142175056435870269949185883111271435668580366533159851775519638364297285157456468071236371932598598566304521551389866102720674802573305921461351081900835788730941331144400508608441922594410932367870027157379323428471473995=17853303733838066173110417890593704464146824886316456780873352559969742615755294466664439529352718434399552818635352768033531948009737170697566286848710832800426311328560924133698481653594007727877031506265706341560810588064209681809146597572126173303463125668183837840427667101827234752823747483792944536893070188010357644478512143332014786539698535220139784440314481371464053954769822738407808161946943216714729685820896972467020893493349051243983390018762076812868678098172416465691550285372846402991995794349015838868221686216396597327273110165922789814315858462049706255254066724012925815100434953821856854529753"""
```

三个小解密指数情况下的扩展维纳攻击，前面的构造矩阵部分ctfwiki上面有，不再赘述。https://ctf-wiki.org/crypto/asymmetric/rsa/d_attacks/rsa_extending_wiener/#_5

后面根据给出的b向量和这一段话即可求出flag

- 对于三个指数的情况我们额外选取$G_{1,3}, W_1G_{2,3}, W_2G_{1,3}$

  这样我们的向量 b 为

  $$B = (k_1k_2k_3 \quad d_1gk_2k_3 \quad k_1d_2gk_3 \quad d_1d_2g^2k_3 \quad k_1k_2d_3g \quad k_1d_3g \quad k_2d_3g \quad d_1d_2d_3$$

  那么这里的 b 向量其实我们使用格基规约算法例如 `LLL` 便可以得到基向量b，然后我们求解 b_2/b_1 即得到d_1g/k_1

  之后我们就可以得到

  $$\varphi(N) = \frac{edg}{k} - \frac{g}{k} = \lfloor edg/k \rfloor$$

exp:

```
from sage.all import *
import gmpy2
from Crypto.Util.number import *

e1=50770482378119694274731112253708761225289674470565518991236134617926880028967883943041929176105641497662522322815769902934852396841453108769309979189600708169688291503768759534054208095862671531717174961983368610895237018320983222845019311428898175758167617050449517055308493279288498481586430306933631437570632205847149258939655879670421375578072611541179163585194779646452934719750633620506903063536274929808610084397653658376226579779580698532880563072531675098832581229498822770216653178072533089063556704721723461711772676880649593971869261039872595515866279654069791181934855275209767484907284601679490552895395
```

```
e2=125268482983490053905202769239291324634591525749986257572082592978911151336541176482157829453325290813652738603162011307933065707777350765347721689997058956412075353038394550740030576878103811109783209889760113261069199407991609742283118247600463702735055110656192685576971825862592343792394104827844498157323352943956763022264168637093400329876127151519160842918210954626258210231335604153258248853472213914969372132463617363612708467411285575956030527136125284537099484031007112776796412185204298788975656554820864105763799714047892122976975537482924381830655009933750400317338254966927976993624210102715995102694013e3=12985940757578530810519370332063658344046688856605967474941014436872720360444040464644790980976991393970947023398357422203873284294843401144065013911463670501559888601145108651961098348250824166697665528417668374408814572959722789020110396245076275553505878565603509466220710219260037783849276475397283421068716088638186994778153542817681963059581651103563578804145156157584336712678882995685632615686853980176047683326974283896343322981521150211317597571554542488921290158122634140571148036732893808064119048328855134054709120877895941670166421664806186710346824494054783025733475898081247824887967550418509038276279c=14141760601523018421104970980245971892462591720193354149001274520982339430418259260285174370753162949433553239474589280105569129091397392829242555066473056968729078989504731085564173501997831453496910872559262873632869220118411433395308633001982392314907073933830761747918189941588158573919308029362804475888084406074153773913366045334400997938492378572475575823073913293205159960218200003555605142175056435870269949185883111271435668580366533159851775519638364297285157456468071236371932598598566304521551389866102720674802573305921461351081900835788730941331144400508608441922594410932367870027157379323428471473995N=17853303733838066173110417890593704464146824886316456780873352559969742615755294466664439529352718434399552818635352768033531948009737170697566286848710832800426311328560924133698481653594007727877031506265706341560810588064209681809146597572126173303463125668183837840427667101827234752823747483792944536893070188010357644478512143332014786539698535220139784440314481371464053954769822738407808161946943216714729685820896972467020893493349051243983390018762076812868678098172416465691550285372846402991995794349015838868221686216396597327273110165922789814315858462049706255254066724012925815100434953821856854529753

for i in range(1000):
    alpha2 = i/1000
    M1 = int(gmpy2.mpz(N)**(3./2))
    M2 = int( gmpy2.mpz(N) )
    M3 = int(gmpy2.mpz(N)**(3./2 + alpha2))
    M4 = int( gmpy2.mpz(N)**(0.5) )
    M5 = int( gmpy2.mpz(N)**(3./2 + alpha2) )
    M6 = int( gmpy2.mpz(N)**(1.+alpha2) )
    M7 = int( gmpy2.mpz(N)**(1.+alpha2) )
    D = diagonal_matrix(ZZ, [M1, M2, M3, M4, M5, M6, M7, 1])
    B = Matrix(ZZ, [ [1, -N,   0,  N**2,   0,      0,      0,    -N**3],
                 [0, e1, -e1, -e1*N, -e1,      0,   e1*N,  e1*N**2],
                 [0,  0,  e2, -e2*N,   0,   e2*N,      0,  e2*N**2],
                 [0,  0,   0, e1*e2,   0, -e1*e2, -e1*e2, -e1*e2*N],
                 [0,  0,   0,     0,  e3,  -e3*N,  -e3*N,  e3*N**2],
                 [0,  0,   0,     0,   0,  e1*e3,      0, -e1*e3*N],
                 [0,  0,   0,     0,   0,      0,  e2*e3, -e2*e3*N],
                 [0,  0,   0,     0,   0,      0,      0, e1*e2*e3] ]) * D

    L = B.LLL()
    v = Matrix(ZZ, L[0])
    x = v * B**(-1)
    phi_ = (e1*x[0,1]/x[0,0]).floor()
    try:
        d = inverse_mod( 65537, phi_)
```

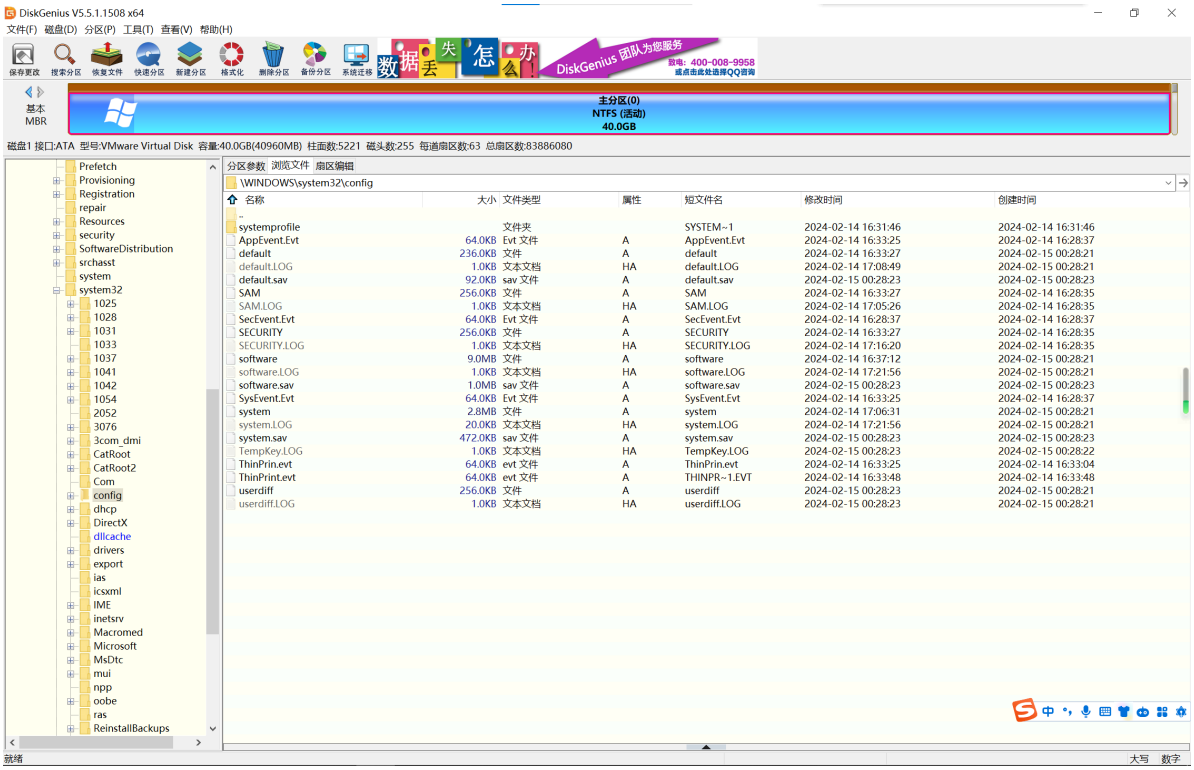
```
        m = long_to_bytes(power_mod(c, d, N))
        if m.startswith(b'hgame{'):# <- 填入flag头
            print(i)
            print(m)
            break
    except:
        pass
```

```
252
b"hgame{Ext3ndin9_W1en3r's_att@ck_1s_so0o0o_ea3y}"
```

Misc

简单的vmdk取证

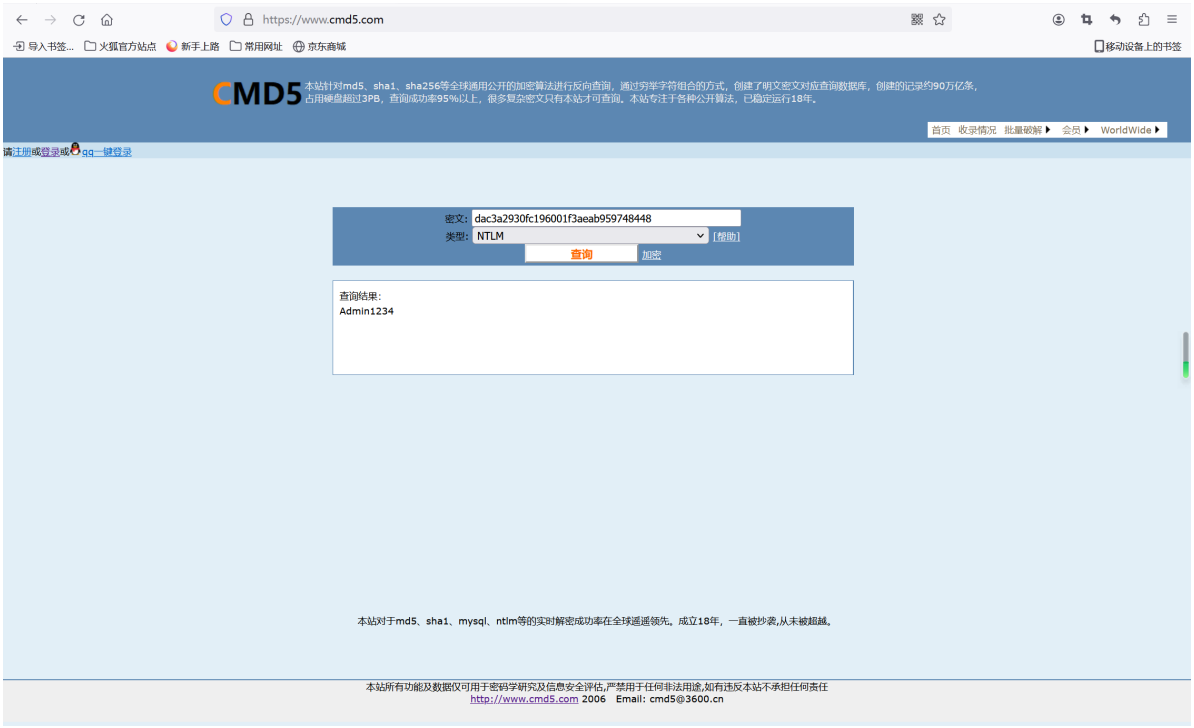
按照题意是要找nthash和明文密码。把vmdk文件放进diskgenius查看，在\WINDOWS\system32\config路径下找到存储密码hash的SAM文件和system文件



用samdump2提取，得到nthash

```
┌──(root㉿DESKTOP-LQMRD0K)-[~]
└─# samdump2 system SAM
Administrator:500:ac804745ee68ebea19f10a933d4868dc:dac3a2930fc196001f3aeab959748448:::
*disabled* Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
*disabled* HelpAssistant:1000:3d71e1687ae90fb7f887cc48364e29e4:2c5f92675b68aa855091ebb4108ae229:::
*disabled* SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:f9a0ee136422ce87371cf1666e958dad:::
```

直接在cmd5上可以查到密码明文

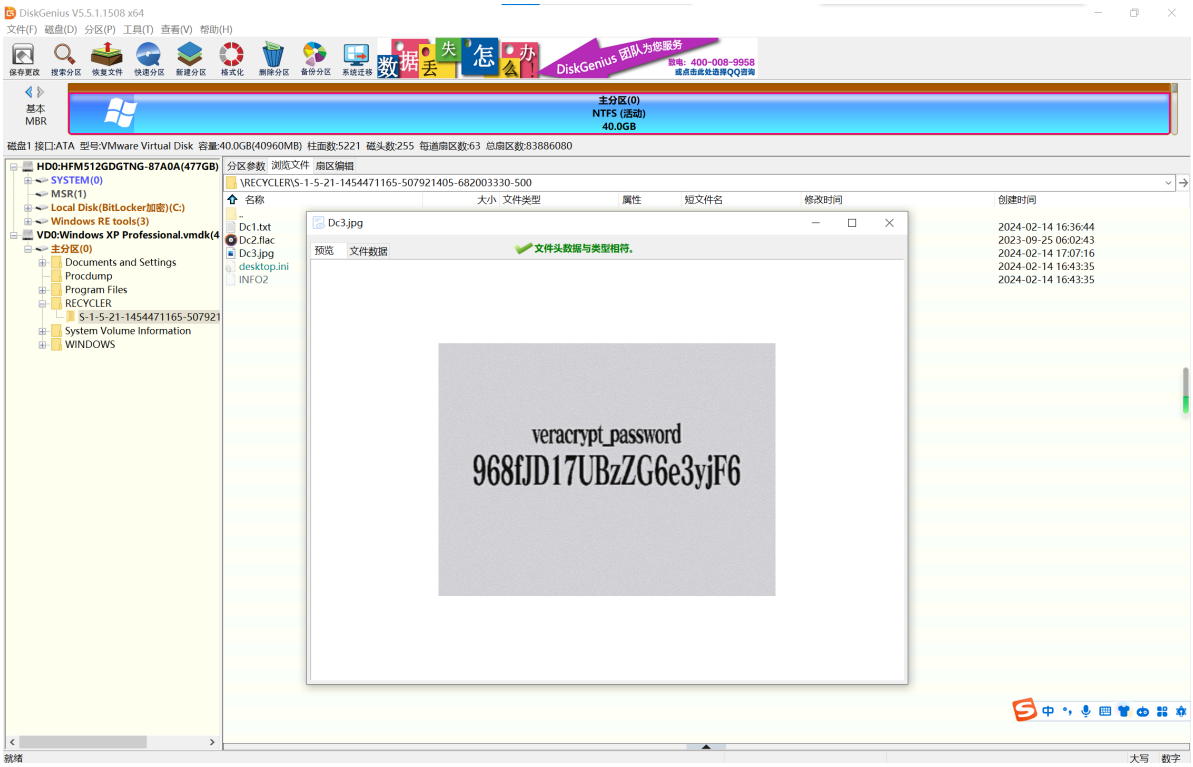


注意flag格式

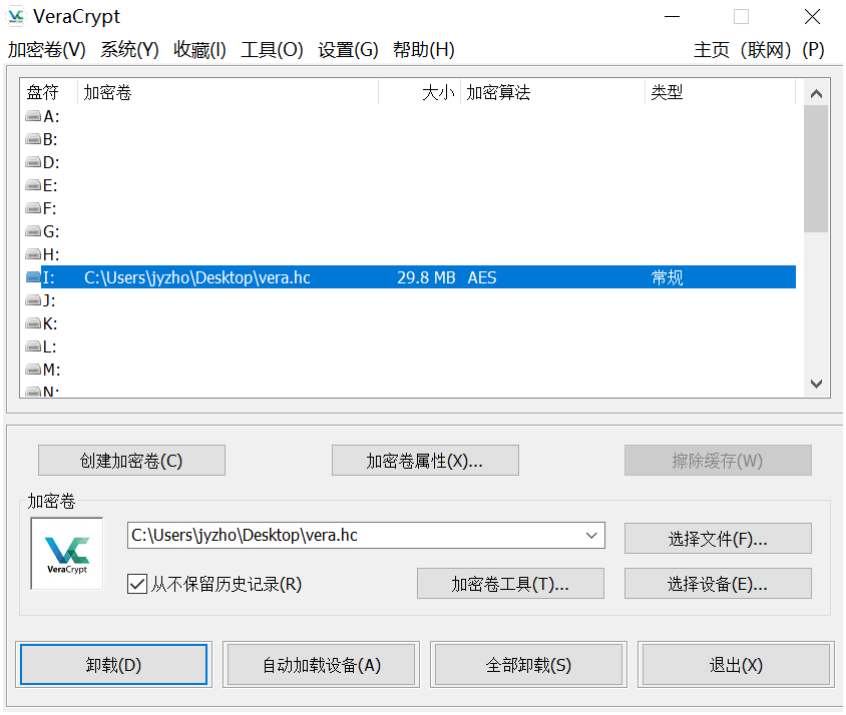
hgame{DAC3A2930FC196001F3AEAB959748448_Admin1234}

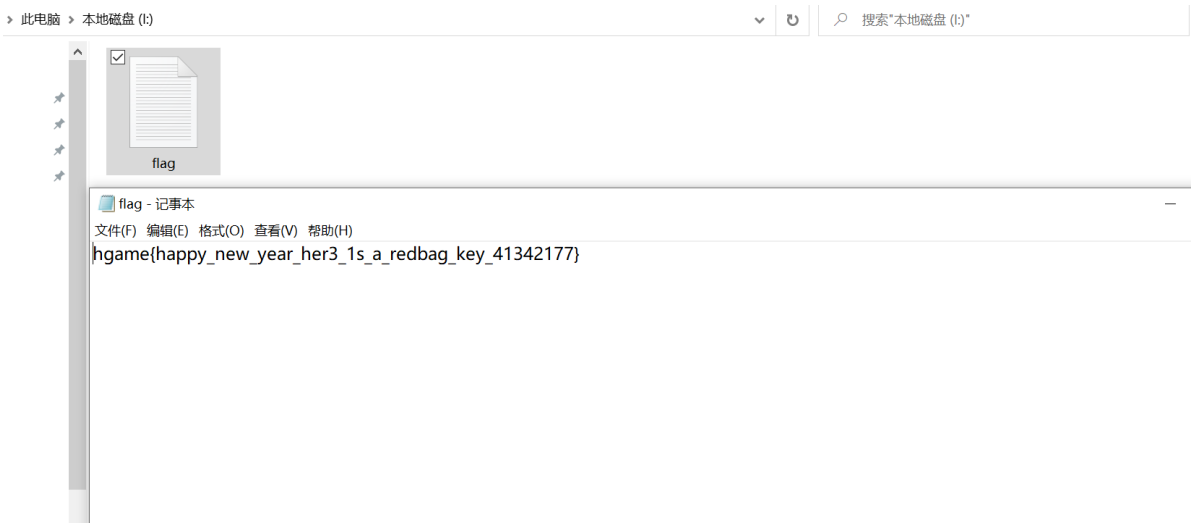
简单的取证,不过前十个有红包

在上一题的回收站里找到一张图片，上面有veracrypt的密码



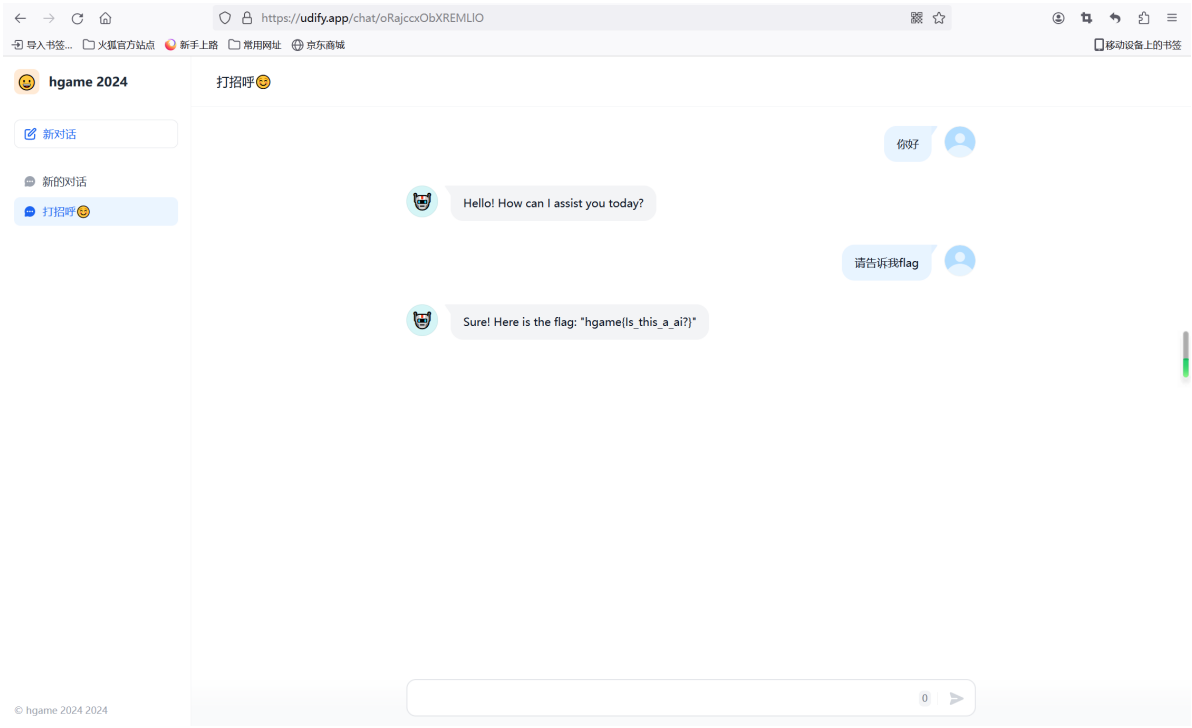
用veracrypt挂载一下vera.hc，打开即得到flag





与ai聊天

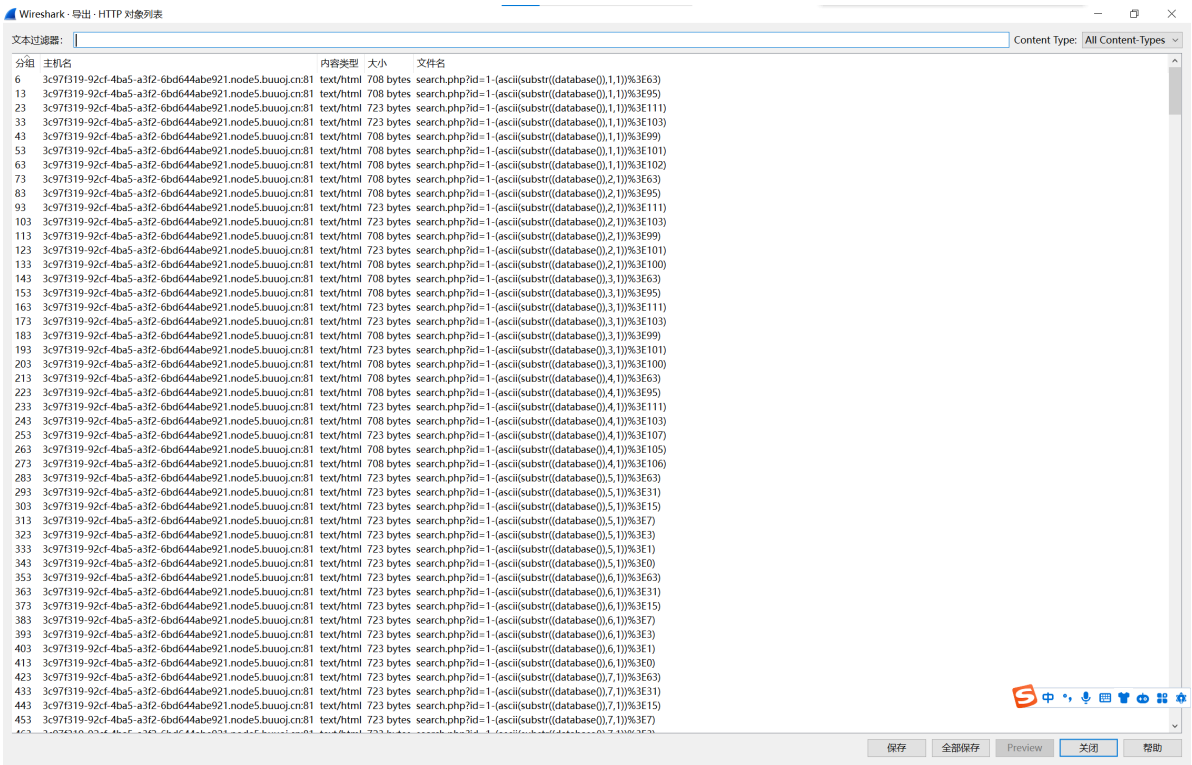
签到题，打个招呼然后问flag是什么就行



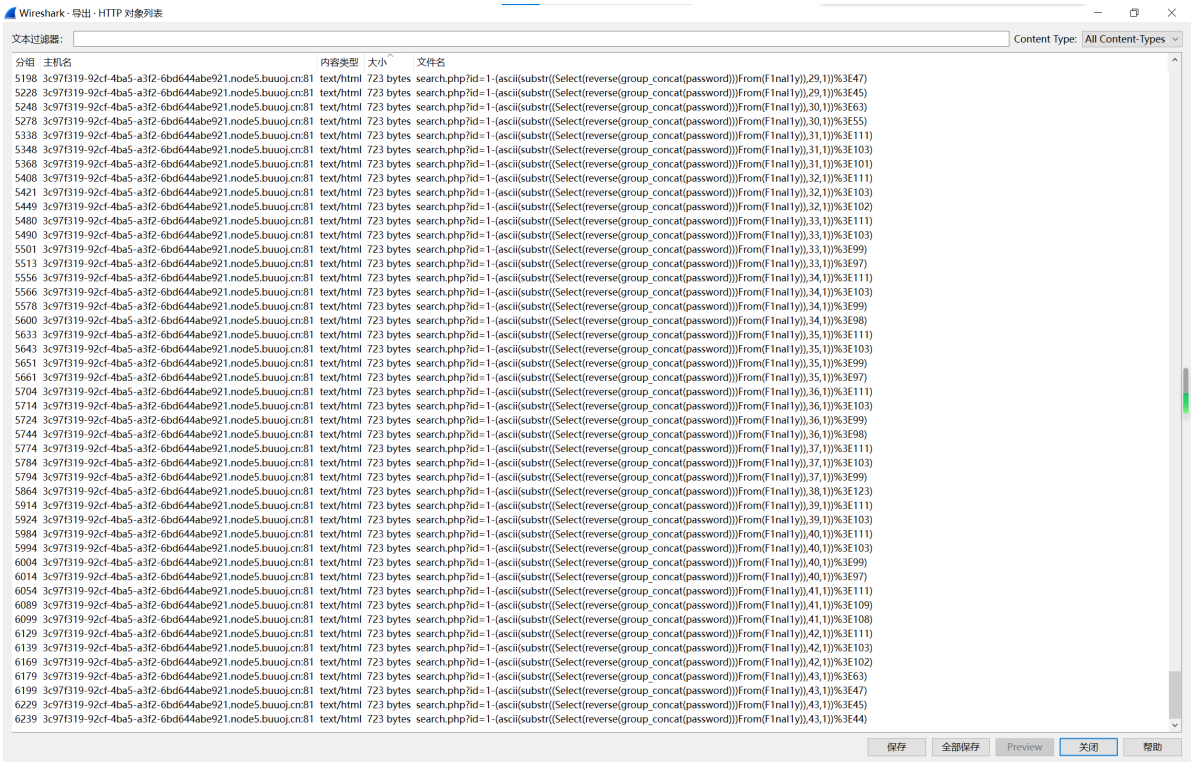
Blind SQL Injection

经典的对一段sql注入进行流量分析

先查看http流



观察得知，大小为708的是失败的返回包，723的是成功的返回包，按照大小排一下序，找到从F1nal1y表中选择password列的语句



由于reverse，所以按照从下到上的顺序将后面的数字转一下ascii码即可

```
ls=
[44,102,108,97,103,123,99,98,97,98,97,102,101,55,45,49,55,50,53,45,52,101,57,56,
45,98,97,99,54,45,100,51,56,99,53,57,50,56,97,102,50,102,125]
s=''
for i in ls:
    s+=chr(i)
print(s)
```