CRYPTO

MidRSA & revenge

直接 Coppersmith

```
In [13]: from Crypto.Util.number import *
         def phase2(high_m, n, c,e):
             R.<x> = PolynomialRing(Zmod(n), implementation='NTL')
             m = high_m + x
             M = m((m^e - c).small_roots()[0])
             return M

         e=3
         n=12083877842125286780879930260397282142527468245626174902901647223493487626661726634639990970574286245897057563766405918961361895688043007877
         c=11896154725446528260312891012636901107224805731765381111074661134801613736138301792146539576697712960143550859000659975574081807130392922757
         m0=13292147408567087351580732082961640130543313742210409432471625281702327748963274496942276607
         m0=m0<<208
         M=phase2(m0,n,c,e)
         print(long_to_bytes(M))

b'hgame{0ther_cas3s_0f_c0ppr3smith}\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff\xff
\xff\xff\xff\xff\xff'
```

```
In [14]: e=5
         n=27814334728135671995890378154778822687713875269624843122353458059697288888640572922486287556431241786461159513236128914176680497775619694684
         c=45622131411586708863820720303449463624470661111162172357784872909606923006795813266301862566144713150175868450263938320833284468193969812445
         m0=9999900281003357773420310681169330823266532533803905637
         m0=m0<<128
         M=phase2(m0,n,c,e)
         print(long_to_bytes(M))

b'hgame{c0ppr3smith_St3re0typed_m3ssag3s}'
```

Backpack & revenge

构造格解方程

```
In [1]: from Crypto.Util.number import *
        from gmpy2 import *
        pubkey=[74763079510261699126345525979, 51725049470068950810478487507, 47190309269514609005045330671, 64955989640650139818348214927, 68559937238
        c=1202548196826013899006527314947
        L = matrix(QQ,len(pubkey) + 1,len(pubkey) + 1)
        for i in range(len(pubkey)):
            L[i,i] = 1
            L[i,len(pubkey)] = pubkey[i]

        for i in range(len(pubkey)):
            L[len(pubkey),i]=1/2

        L[-1,-1] = c
        L = L.LLL()
        print(L)

[ -1/2  1/2  1/2  1/2  1/2 -1/2  1/2  1/2 -1/2  1/2  1/2  1/2 -1/2 -1/2 -1/2  1/2  1/2  1/2 -1/2 -1/2  1/2  1/2 -1/2
1/2 -1/2  1/2  1/2  1/2 -1/2  1/2 -1/2  1/2 -1/2  1/2 -1/2 -1/2  1/2 -1/2  1/2  1/2  1/2  1/2 -1/2  1/2 -1/2 -1/2 -
1/2 -1/2    0]
[   0   -2   -1    1    0   -3    2    1    0    0    1   -1    2   -1    1    0    1    1    1   -2    0    2   -2
-1   -1    0    1    1    0    1   -1   -2    0    0    1    2    1   -1    0    0    1   -1   -2   -1    1    0
0    0    0]
[   1   -1    2   -1    2   -3    1    0    1    0    1   -1    1    0    1    1    2    2   -1    0   -1   -1    0
-1   -1    2    1   -1   -2    0    1   -1    0    0    3    0    0   -1   -1   -1    1   -1   -2   -1   -1    0
0    0    0]
[   0    3    0    1    0    0   -1   -1    0    2   -1    1   -1   -2    1    0    2   -2   -2   -1   -1    0    3
1   -1    2    0    2   -2    0   -1    1    1   -1   -1    0    0    1   -1   -1   -1   -2    1    1   -1    0
0    0   -1]
[   1   -1    0   -1    0   -1    1    1   -1   -3    1    4    0    2   -1    2   -1    0    1    0    0    0   -2
-1    1   -1   -1   -1    1   -2    1   -1    0   -1    2   -2   -1    1   -1    1    0    1    1    0    0    0
0    0   -1]
[   0   -3   -3    3    1    0   -1    1   -1   -1    1    1    0    1    0   -1    2    0    0    2    1   -1   -3
-1    1   -1    0   -1    3    0    0   -1    2   -1    0   -1   -2    0    1    2    0    0   -1    0    1    0
0    0    0]
[  1/2 -3/2 -3/2  1/2 -3/2 -3/2  5/2 -5/2 -1/2  5/2  3/2  3/2  1/2  3/2  1/2 -1/2  3/2  1/2 -1/2  3/2  5/2  1/2 -7/2
```

第一行的相反向量就是方程解

```
from Crypto.Util.number import *

import hashlib

a=[-1/2, 1/2, 1/2, 1/2, 1/2,-1/2, 1/2, 1/2,-1/2, 1/2, 1/2, 1/2,-1/2,-1/2,-1/2, 1/2,
```

```
1/2, 1/2,-1/2,-1/2, 1/2, 1/2,-1/2, 1/2,-1/2, 1/2, 1/2, 1/2,-1/2, 1/2,-1/2, 1/2,-1/2,
1/2,-1/2,-1/2, 1/2,-1/2, 1/2, 1/2, 1/2, 1/2,-1/2, 1/2,-1/2,-1/2,-1/2,-1/2]
p=""
for i in a:
   if i>0:
      p="0"+p
   elif i<0:
      p="1"+p
p="0b"+p
a = [747630795102616991263455259 79, 51725049470068950810478487507,
471903092695146090050453306 71,
       649559896406501398183482149 27, 685599372386236236191140659 17,
723113391701121854014968670 01,
       708173360642547816402733540 39, 705381088265397857743616053 09,
437825309424818656212933810 23,
       582343281865780362910570662 37, 688082712654788585701269169 49,
616602004709381538360454838 87,
       632707269818515446203592313 07, 429047764866976916696399292 29,
415456372017875316374276033 39,
       740128390556498913971728708 91, 569437947956412606749536768 27,
517373919021877591880786874 53,
       492643689995616599861828839 07, 600442212373871040545978619 73,
638470463502605207610436878 17,
       621281466995821807790139835 61, 651093134232128526479302999 81,
668256358698317310926840393 51,
       677632651477912720837807523 27, 611678440839991796697026016 47,
551160159278687568590079619 43,
       523444885180556720822803775 51, 523758778919423123200318039 19,
696590359415641192916404047 91,
       525632820851786467678143828 89, 568106273122864204941091920 29,
497558777990068890638825665 49,
       438589016724517567544748451 93, 679237436151549832911456245 23,
516894555147285474239951626 37,
       674801311517071556725275833 21, 593962122483305800721846480 71,
634105288752204897994752492 07,
       480114092885508802292805781 49, 625619692603911329568182859 37,
448261586642837794103306159 71,
       704462187599762399477511620 51, 565098473798366000335019425 37,
501542879711798313550684431 53,
       490605071160958611749714671 49, 542368482942996246321605210 71,
641866264289749761084671968 69]
bag = 120254819682601389900652731 4947
sum=0
p=int(p,2)
```

```
flag='hgame{'+hashlib.sha256(str(p).encode()).hexdigest()+'}'
print(flag)
#hgame{04b1d0b0fb805a70cda94348ec5a33f900d4fd5e9c45e765161c434fa0a49991}
```

BabyRSA

先解出 e

```
p=14213355454944773291
c=10500213872246694649593663865603821400004347575163902508525511396508874927246190689258661625026492234819249659798645278628115115643622957406519396542284
1
gift=9751789326354522940
phi0=p-1
E=0x10001
D=inverse(E,phi0)
e=pow(gift,D,p)-114514
print(e)
#73561
print(GCD(e,phi))
#73561
```

发现 e 和 phi 不互素，故求不了 d

查得用 AMM 开根

```
import random
from Crypto.Util.number import bytes_to_long,long_to_bytes
p = 0
#设置模数
def GF(a):
    global p
    p = a
#乘法取模
def g(a,b):
    global p
    return pow(a,b,p)
def check(m):
    if 'hgame' in m:
        print(m)
        return True
    else:
        return False

x =
10500213872246694649593663865603821400004347575163902508525511396508874927246190689258
```

```
661625026492234819249659798645278628115115643622957406519396542284 1
e = 73561
p =
252395126560905375387770476333223213025135495780692042262472892345711692954868635690049187042705087220115020955644371201480080957031506006230262490543001 7
phi=252395126560905375370012873115532765884115104666470081919030062593024775474310975066652216794544351991373966513618043028516159397425063118953380527335664 0
GF(p)
y = random.randint(1, phi)
while g(y, phi//e) == 1:
    y = random.randint(1, phi)
    print(y)
print("find")
#p-1 = e^t*s
t = 1
s = 0
while p % e == 0:
    t += 1
    print(t)
s = phi // (e**t)
print('e',e)
print('p',p)
print('s',s)
print('t',t)
# s|ralpha-1
k = 1
while((s * k + 1) % e != 0):
    k += 1
alpha = (s * k + 1) // e
#计算 a = y^s b = x^s h =1
#h 为 e 次非剩余部分的积
a = g(y, (e ** (t - 1) ) * s)
b = g(x, e * alpha - 1)
c = g(y, s)
h = 1
#
root = (g(x, alpha * h)) % p
roots = set()
for i in range(e):
    mp2 = root * g(a,i) %p
    assert(g(mp2, e) == x)
    roots.add(mp2)
for mpp in roots:
        solution = str(long_to_bytes(mpp))
```

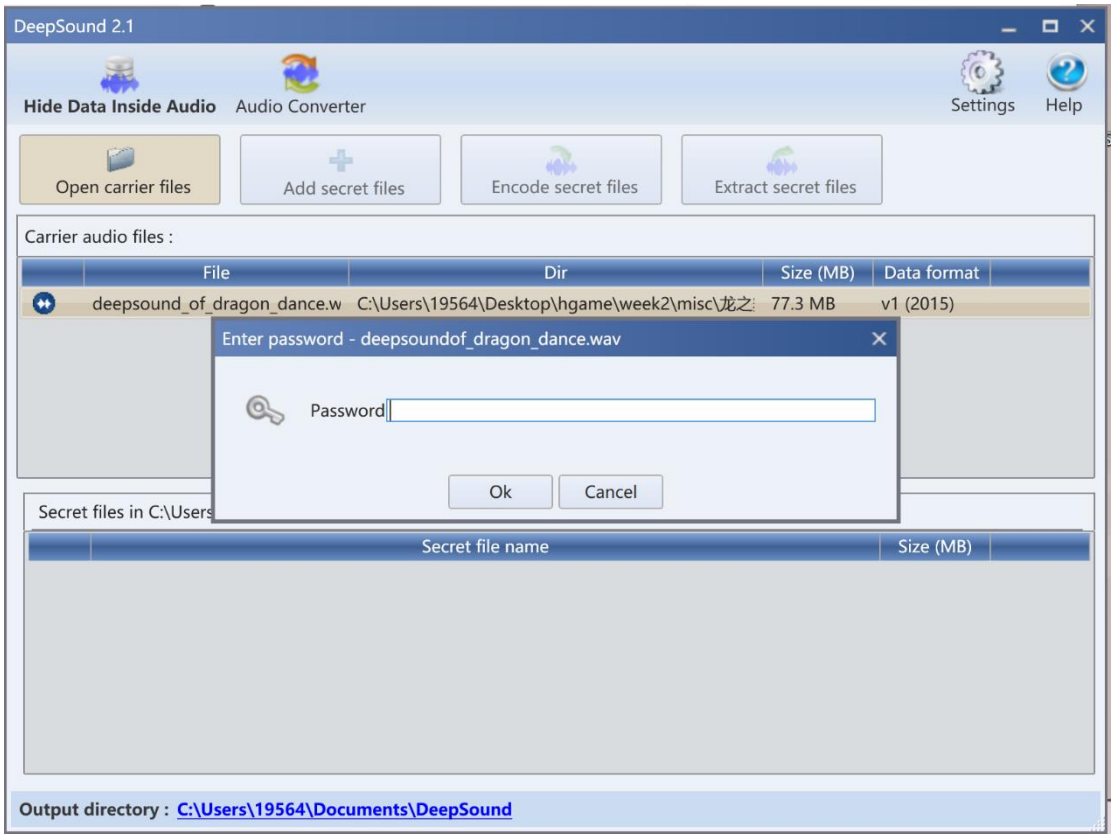
```
      if check(solution):
         print(solution)
'''
find
e 73561
p
252395126560905375387770476333223213025135495780692042262472892345711692954868635690049187042705087220115020955644371201480080957031506006230262490543001 7
s
343109972078826246747614732148193697589911916187205288018148288621721802958511949357203160362888421842245165935234761665170619482368460351209717822 40
t 1
b'hgame{Ad1eman_Mand3r_Mi11er_M3th0d}'
b'hgame{Ad1eman_Mand3r_Mi11er_M3th0d}'
'''
```

MISC

流量包内 http 提取出一张照片即可

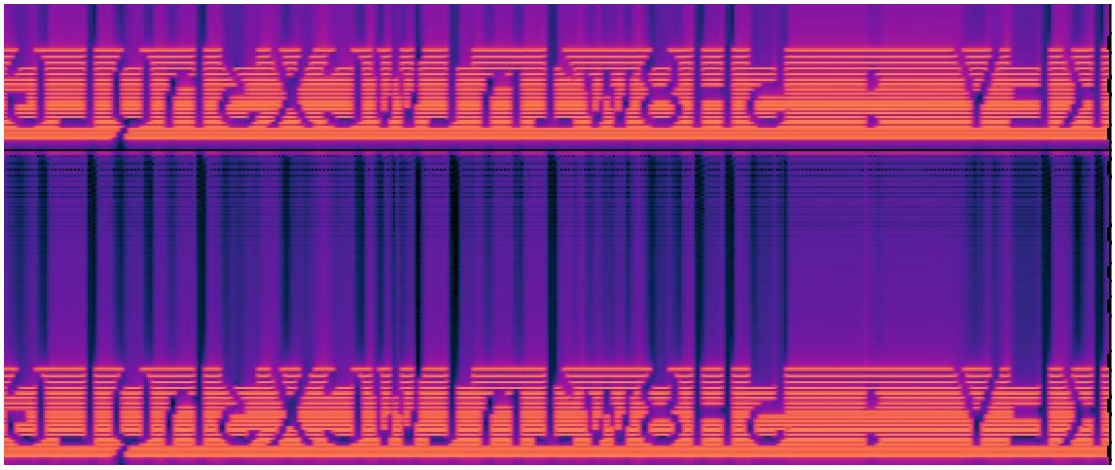
龙之舞

先注意到文件名前面的 deepsound，搜索知这是一种加密音频的软件



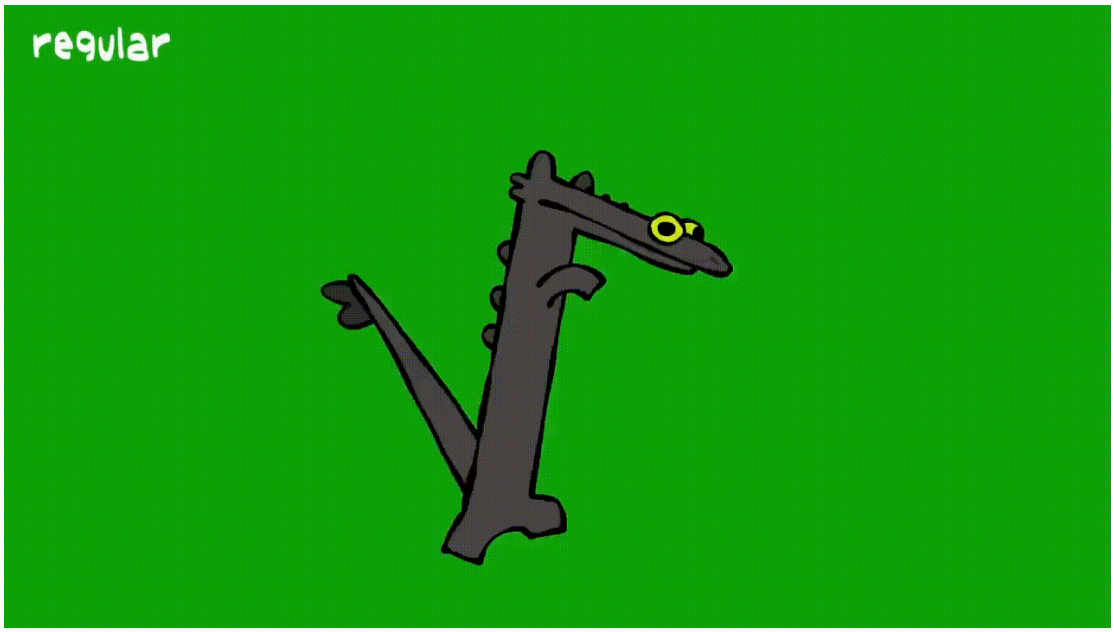
下载打开发现需要密钥

听音频发现开头不对劲，用 Audacity 查看频谱发现是 key



旋转再翻转下

得到一个 gif



帧分解再 ps 得到二维码

但是扫不出来，不正确

出题师傅告诉一个网站 Qrazybox

在里面打开再暴力破解即得 flag