week3

matrix_equation

造格子

$$\begin{pmatrix} p & q & r \end{pmatrix} \times \begin{pmatrix} 1 & 0 & 0 & 2^{256} \\ 0 & 1 & 0 & k1 \\ 0 & 0 & 1 & k2 \end{pmatrix} = \begin{pmatrix} p & q & r & temp \end{pmatrix}$$

```
k1=737153298772153401459512383432471562821657053960747864832566998176512557096718
k2=613619706622698697382703285238977654084439071983136324100684542237178242768378

m=matrix([[1,0,0,2^256],[0,1,0,k1],[0,0,1,k2]])
short=m.LLL()
print(short)
```

```
[-140124951574954439598312018   93963243579505738889945998   15154059265021257630097517   -58511170749450817230624778]
[ 19032620393921771901444797 -33281308486653930151733737   40662930488236217849932508 -65301243525031258000907285]
[   80823399368365632266190178   46397424257679676851556254 -57263293525378453480844839 -45515665156713370235083473]
```

第一行就是p,q,r,temp。temp位数刚好是82位，接下来就没什么了，刚开始格子造错了，位数没到，以为还要遍历一遍看哪个位数是对的（所以先走一步再走一步是指LLL吗）

```
import hashlib
hint=83

ans=[-140124951574954439598312018, 939632435795057388899459978
,151540592650212576300975178 ,-58511170749450817230624778]
print(len(bin(-ans[-1])[2:]))
p=-ans[0]
q=-ans[1]
r=-ans[2]
flag='hgame{'+hashlib.sha256(str(p+q+r).encode()).hexdigest()+'}'
print(flag)
```

```
83
hgame{3633c16b1e439d8db5accc9f602f2e821a66e6d80a412e45eb3e1048dffbb0e2}
```

exRSA

扩展维纳攻击，三个小解密指数，构造

$$b = (k_1k_2k_3, d_1gk_2k_3, k_1d_2gk_3, d_1d_2g^2k_3, \\ k_1k_2d_3g, k_1d_3g, k_2d_3g, d_1d_2d_3g^3),$$

by reducing the rows of the following lattice:

$$L_3 = \begin{pmatrix} 1 & -N & 0 & N^2 & 0 & 0 & 0 & -N^3 \\ & e_1 & -e_1 & -e_1N & -e_1 & 0 & e_1N & e_1N^2 \\ & & e_2 & -e_2N & 0 & e_2N & 0 & e_2N^2 \\ & & & e_1e_2 & 0 & -e_1e_2 & -e_1e_2 & -e_1e_2N \\ & & & & e_3 & -e_3N & -e_3N & e_3N^2 \\ & & & & & e_1e_3 & 0 & -e_1e_3N \\ & & & & & & e_2e_3 & -e_2e_3N \\ & & & & & & & e_1e_2e_3 \end{pmatrix} \times D,$$

，把L3 格基约

where D is the diagonal matrix

$$\mathrm{diag}(N^{3/2}, N, N^{(3/2)+\alpha_3}, N^{1/2}, N^{(3/2)+\alpha_3}, N^{1+\alpha_3}, N^{1+\alpha_3}, 1)$$

之后再乘L3^(-1)就是b向量了。

因为 $e_1d_1g - k_1(p-1)(q-1) = g$ 所以 $\varphi = \dfrac{(e1d1-1)g}{k1}$

又因为 $\dfrac{b[1]}{b[0]} = \dfrac{k1k2k3}{d1gk2k3} = \dfrac{k1}{d1g}$ 所以 $\varphi = \left[\dfrac{b0e}{b1}\right]$

得到phi之后，就ok了。

```
from Crypto.Util.number import *
from gmpy2 import invert
e1=
507704823781196942747311122537087612252896744705655189912361346179268800289678839430419291761056414976625223228157699029348523968414531087693099791896007081696882915037687595340542080958626715317171749619833686108952370183209832228450193114288981757581676170504495170553084932792884984815864303069336314375706322058471492589396558796704213755780726115411791635851947796464529347197506336205069030635362749298086100843976536583762265797795806985328805630725316750988325812294988227702166531780725330890635567047217234617117726768806495939718692610398725955158662796540697911819348552752097674849072846016794905528953 9
e2=125268482983490053905202769239291324634591525749986257572082592978911151336541176482157829453325290813652738603162011307933065707777350765347721689997058956412075353038394550740030576878103811109783209889760113261069199407991609742283118247600463702735055110656192685576971825862592343792394104827844981573233529439567630222641686370934003298761271515191608429182109546262582102313356041532582488534722139149693721324636173636127084674112855759560305271361252845370994840310071127767964121852042987889756565548208641057637997140478921229769755374829243818306550099337504003173382549669279769936242101027159951026940 1
```

```
e3=12985940757578530810519370332063658344046688856605967474941014436872720360444
040464644790980976991393970947023398357422203873284294843401144065013911463670505
155988860114510865196109834825082416669766552841766837440881457295972278902011033
962450762755535058785656035094662207102192600377838492764753972834210687160886381
869947781535428176819630595816511035635788041451561575843367126788829956856326156
568685398017604768332697428389634332298152115021131759757155454248892129015812263
414057114803673289380806411904832885513405470912087789594167016642166480618671034
682449405478302573347589808124782488796755041850903827627 9
c=
141417606015230184211049709802459718924625917201933541490012745209823394304182592
602851743707531629494335532394745892801055691290913973928292425550664730569687290
789895047310855641735019978314534969108725592628736328692201184114333953086330019
823923149070739338307617479181899415881585739193080293628044758880844060741537739
133660453344009979384923785724755758230739132932051599602182000035556051421750564
358702699491858831112714356685803665331598517755196383642972851574564680712363719
325985985663045215513898661027206748025733059214613510819008357887309413311444005
086084419225944109323678700271573793234284714739 9
n=
178533037338380661731104178905937044641468248863164567808733525599697426157552944
666644395293527184343995528186353527680335319480097371706975662868487108328004263
113285609241336984816535940077278770315062657063415608105880642096818091465975721
261733034631256681838378404276671018272347528237474837929445368930701880103576444
785121433320147865396985352201397844403144813714640539547698227384078081619469432
167147296858208969724670208934933490512439833900187620768128686780981724164656915
502853728464029919957943490158388682216862163965973272731101659227898143158584620
497062552540667240129258151004349538218568545297 53
L=matrix(ZZ,[[1,-n,0,n^2,0,0,0,-n^3],
             [0,e1,-e1,-n*e1,-e1,0,n*e1,n^2*e1],
             [0,0,e2,-n*e2,0,n*e2,0,n^2*e2],
             [0,0,0,e1*e2,0,-e1*e2,-e1*e2,-n*e1*e2],
             [0,0,0,0,e3,-n*e3,-n*e3,n^2*e3],
             [0,0,0,0,0,e1*e3,0,-n*e1*e3],
             [0,0,0,0,0,0,e2*e3,-n*e2*e3],
             [0,0,0,0,0,0,0,e1*e2*e3]])
a=0.375 #768/2048
e=0x10001
D=diagonal_matrix(ZZ,[n^1.5,n,n^(a+1.5),n^0.5,n^(a+1.5),n^(a+1),n^(a+1),1])
B=L*D
L=B.LLL()
v=vector(ZZ,L[0])
x=v*B^(-1)
phi=int(x[1]/x[0]*e1)
print(phi)
print(len(bin(phi)[2:]))
d=pow(e,-1,phi)
m=int(pow(c,d,n))
print(long_to_bytes(m))
```

最后检查一下位数对的，就ok 了。

```
178533037338380661731104178905937044641468248863164567808733525599697426
157552944666644395293527184343995528186353527680335319480097371706975662
868487108328004263113285609241336984816535940077278770315062657063415608
105880642096818091465975721261733034631256681838378404276671018272347528
237474837929445368928029270701817714906863187360128785923435183666162300
031565326172796507698012407884570824707982850983097873132340150030295460
693899839176463394642132703079730839201972454643232657746031300483849074
239103584404363219674145536351192203034270882979641353211329514783904544
247777825148677789278559501692519462706 28
2048
b"hgame{Ext3ndin9_W1en3r's_att@ck_1s_so0o0o_ea3y}"
```