【CTF WriteUp】2020天翼杯Crypto题解

原创

零食商人 于 2020-08-01 15:35:17 发布 1782 收藏 6

（开会没赶上，赛后做的，部分内容搬运自其他大佬）

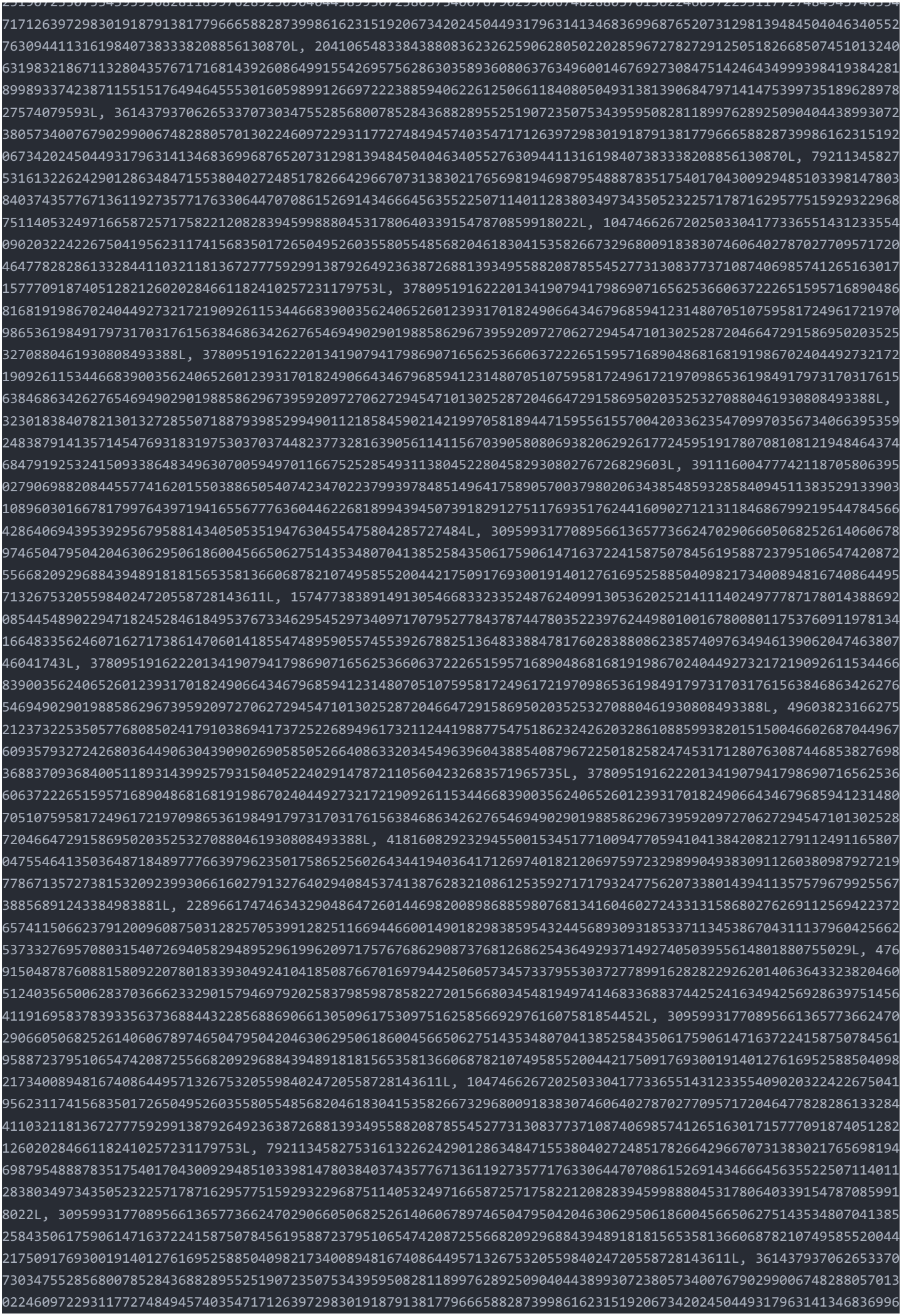
Crypto

easyRSA

根据题目，e<20000，加密为逐字符加密，所以可以通过爆破得到e，然后再依次解密。

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
import gmpy2
from libnum import n2s, s2n

n = 538684126342330450903691534377474128789754259920407545763467545966203473507844229175437598979366846466631508
934429988697637980067299799975645876808751759953096358770310738981923801281345099768890054087687343742160636399 0
22776903085059191782726151911631146459168672498278567513498518143465475056224714839499 37
c = [361437937062653370730347552856800785284368828955251907235075343959508281189976289250904044389930723805734007
767902990067482880570130224609722931177274849457403547171263972983019187913817796665882873998616231519206734202 4
504493179631413468369968765207312981394845040463405527630944113161984073833382088561308 70L, 304852893174891225 13
627424533258350515279207179552977911285197688270281245828407217991247923748564161409175076285380310289577344106 1
252201287023737041598012655099695448407937369654648784149145374837012342286290646393741763948920037478940006491 4
279802215768343706259661857458276155478728437251122600355026937 1L, 323018384078213013272855071887939852994901121
858459021421997058189447159556155700420336235470997035673406639535924838791413571454769318319753037037448237732 8
163905611411567039058080693820629261772459519178070810812194846437468479192532415093386483496307005949701166752 5
285493113804522804582930802767268296 03L, 257104308941580793118307148897185809741715819110021619897351539740743 19
352353008048142133029367435263698163394834768816086041815027231628825846495139208493868114079735178838159083476 4
406136880966157015996469047403031219608996278877991126058874849227718853970563594745735995474936700218314212818 7
1544848617729L, 197329688947001820051703229338535447339185565072186108791721013307911352587600481035747840928736
043139328294116875187817968701125288164568291312364476102384305345131414392553530778938324759690718237844149026 8
815567765608194884841287224449405777657024458589558535300152711388775204963460731062473137905422597 3L, 496038231
662752123732253505776808502417910386941737252268949617321124419887754751862324262032861088599382015150046602687 0
449676093579327242680364490630439090269058505266408633203454963960438854087967225018258247453171280763087446853 8
276983688370936840051189314399257931504052240291478721105604232683571965735L, 323018384078213013272855071887939 8
529949011218584590214219970581894471595561557004203362354709970356734066395359248387914135714547693183197530370 3
744823773281639056114115670390580806938206292617724595191780708108121948464374684791925324150933864834963070059 4
970116675252854931138045228045829308027672682960 3L, 361437937062653370730347552856800785284368828955251907235075
343959508281189976289250904044389930723805734007679029900674828805701302246097229311772748494574035471712639729 8
301918791381779666588287399861623151920673420245044931796314134683699687652073129813948450404634055276309441131 6
198407383338208856130870L, 309599317708956613657736624702906605068252614060678974650479504204630629506186004566 5
062751435348070413852584350617590614716372241587507845619588723795106547420872556682092968843948918181565358136 6
068782107495855200442175091769300191401276169525885040982173400894816740864495713267532055984024720558728143611 L
, 104746626720250330417733655143123355409020322422675041956231174156835017265049526035580554856820461830415358 26
673296800918383074606402787027709571720464778282861332844110321181367277759299138792649236387268813934955882087 8
554527731308377371087406985741265163017157770918740512821260202846611824102572311797 53L, 391116004777421187058 06
395027906988208445577416201550388650540742347022379939784851496417589057003798020634385485932858409451138352913 3
903108960301667817997643971941655677763604462268189943945073918291275117693517624416090271213118468679921954478 4
566428640694395392956795881434050535194763045547580428572748 4L, 361437937062653370730347552856800785284368828955
```



```
876520731298139484504046340552763094411316198407383338208856130870L, 176825152324298552720359019067727938254877213643447443628544664532453096325332444235383973215296851986648972524670801398060663552783015765559014127500481734840943982860140825226698285782587617017641082609318072626911388895416390026604815207283861133331278244441933073765212943774278703841021285005169606849 47L, 476915048787608815809220780183393049241041850876670169794425060573457337955303727789916282822926201406364332382046051240356500628370366623329015794697920258379859878582272015668034548194974146833688374425241634942569286397514564119169583783933563736884432285688690661305096175309751625856692976160758185 4452L, 309599317708956613657736624702906605068252614060678974650479504204630629506186004566506275143534807041385258435061759061471637224158750784561958872379510654742087255668209296884394891818156535813660687821074958552004421750917693001914012761695258850409821734008948167408644957132675320559840247205587281436 11L, 3383753296078297100979107779973876489685864958307296319951722730208139362112230640495544760599632083947014531499803342566111506880157011132572761327703515257577498287614945031308201282412874321800014925255783674428091167472468400356133778621439359709032906044910492040028941146731924851172241523533242380431 1L, 3003030331585013454198362376279364894057686324387540834638333674000201497791586716659648371683756509089978193778644466701160325589864501457956816338707524946273965505863586427119270436154977433745660750602696454966176542791708334415133342331074241601534371614287941800087416453451025510062610624464060720674 1L]
e = 0
for tmpe in range(2, 20000):
    if(tmpe%1000==0):
        print tmpe
    for i in range(256):
        if pow(i, tmpe, n) == c[0]:
            print i, tmpe
            e = tmpe
            break
    if e!=0:
        break

target = []
for i in range(256):
    target.append(pow(i, e, n))

m = ''
for i in c:
    m += chr(target.index(i))
print m
```

flag{caf94ef5f8400ae920c0bd79489f3791}

HardRSA

本题为Coppersmith算法中，已知d的部分低位情况的变种。根据题目，有

```
e * d = 1 mod ((p-1)*(q-1)*(r-1))
e * d = 1 + k*(p-1)*(q-1)*(r-1)
```

两边模2^540，得：

```
e * d0 = 1 + k*(p-1)*(q-1)*(r-1)  mod 2^540
```

两边乘以q

```
(e * d0 - 1) * q = k*(p-1)*(q-1)*(r-1) * q  mod 2^540
(e*d0-1) * q = k*(p-1)*(qr-q-r+1)*q
(e*d0-1) * q = k*(p-1)*q*r*q - k*(p-1)*q*q - k*(p-1)*r*q + k*(p-1)*q
(e*d0-k*(p-1)*(q*r)-k*(p-1))*q + k*(p-1)*q^2 + k*(p-1)*(q*r) = q
e*d0*q-k*(p-1)*(q*r+-q+1)*q + k*(p-1)*(q*r) = q
```

上式可视为关于q的一元二次方程，其中的每一项系数均可求，所以可以通过修改原始Coppersmith代码来分解q*r。此处step1求解后得到的q已经满足条件，所以不需要进行step2。完整代码如下：

```
from sage.all import *

def find_p(d0, kbits, e, n, p):
    X = var('X')
    for k in xrange(1, e+1):
        results = solve_mod([e*d0*X - k*(p-1)*X*(n-X+1) + k*(p-1)*n == X], 2^kbits)
        for x in results:
            q = ZZ(x[0])
            if n % q == 0:
                return q
    return None

if __name__ == '__main__':
    # n = q * r
    n = 0x5c44cf1cc8b3c0ef4bd76abbb5236f772edf728c59ea29ee8af3fe9e2d3a44a004c2694a398f02be6480c2b418fcf0c8e21805
77318fa9eeb80a6c11d1a7ee6c6945912eaac23322f6c1a0609bddc7e012e58163ed6eb3e13a88e880ee1e77e7aa0aac4b2e364b2df5609a
4ece180c539126f60c0fff7ff4f325e49a0caad4f
    p = 0x3bfd387720814dd8952e73745e96cb270964a35f7d5c6a0b23b200d7e64bcf95e3c8a2850dd3842dcc90f6fc236332b1b85492
aca5f9373d12b75e9aaf867271
    e = 7
    c = 0xec76baaf15a5c04bf857155374da8408ff0c142c79b7bf1a9359244e8f3a1b046543dd2e74be16746cc7fde06c876380618d87
d3027ebc0345deb34d884bb24ba8536ca981433d4d3ed66285b155820bd5e1a626f5ba4eda57ffbdd2cbc7422a9baa2c5b486bdacd9abd1b
fd5452110b41d0859585924feb57890dfb0a18cfd649b203a40f361da4604e6c01663c1e7fd81db9033e07cf327b5bb2e09bd23a5bc1bf49
ea21e79fa2ecce7f57035d9f9ebd795d5bb11a7e2c2a16abf6937884
    tempd = 0x414946b9c40728f9801e61e98ec6d17525cbe4163a5ffb8367b65c652ae4cc3abce62e70afbfb84fcf937b3119953b4892
2be19ef4312c4f3a88313368ca6c9b1d658b7
    kbits = 540
    d0 = tempd & (2^kbits-1)

    q = find_p(d0, kbits, e, n, p)
    print q
```

解得

q =

171768549986489619777590703727368301353369354721362744223435149769846660378741614519308553110035365
586738468174821871754688091892099253667583637507147331 7

然后就可解得明文

```
import gmpy2
from libnum import n2s, s2n

qr = 40495840300029705456640808238481295829384110345567926009143802294615467917723278944378449746945628043223200
98597200821632361860404029749610290918822378230751309535193434791281064861182569710224863795884931806318447090299
93268295512145732129977899760223793782378464837106476863556580599315817718632913480369 43
p = 31418863481129883391748654321792064129425883902281696451622939204701888824478552087832208997528876202210598 6
14673480593340308733505719794623638346152310 89
c = 86967712720155137368878433956123616473142972877815076091969363541832116553647446841643007105836254738729427 1
20633095076514963148003930094804218869261374037592284218584148334299800599030493118370144490933659116031081583 52
87185167745725605851082250915740770321086631747289458608755400115854095178716769916172049119867484852609364413 17
09561995379565643716017359731201935855247285498574923656191121426618306186331615280461405913653781283860760013 21
5603007314688132
q = 17176854998648961977759070372736830135336935472136274422343514976984666037874161451930855311003536558673846 8
17482187175468809189209925366758363750714733 17
r = qr//q
d = gmpy2.invert(7, (p-1)*(q-1)*(r-1))
print n2s(pow(c, d, p*q*r))
```

flag{6809781d08e120627e623dcdafe26b8a}

polycrypto

本题为NTRUEncrypt，参照wiki
https://en.wikipedia.org/wiki/NTRUEncrypt
看了看基本没啥区别，尝试照着做
第一步计算 a = e（密文）* f（私钥）

```
from sage.all import *
import random

def randomseq(N):
    return [randint(-1, 1) for _ in range(N)]

N = 61
p = 11
q = 5039
Ip = Integers(p)
Iq = Integers(q)
Rp = PolynomialRing(Ip, "x")
Rq = PolynomialRing(Iq, "y")
xp = Rp.gen()
xq = Rq.gen()
Sp = Rp.quotient(xp ^ N - 1, "x")
Sq = Rq.quotient(xq ^ N - 1, "y")

pubkey = Sq([2002, 3371, 2744, 363, 4617, 4767, 4447, 490, 2348, 909, 4739, 4766, 1417, 4776, 1606, 4383, 699, 3
445, 1989, 986, 2450, 971, 2890, 1050, 3887, 2624, 3083, 329, 1917, 1014, 2189, 4843, 4982, 3708, 4094, 535, 294
2, 1807, 4643, 3876, 2191, 476, 844, 1352, 4079, 637, 402, 4037, 2576, 2803, 1949, 3704, 4828, 2502, 526, 2032,
1145, 976, 2078, 3566, 2447])
prikey = Sq([12, 0, 5028, 0, 11, 0, 5028, 11, 5028, 11, 11, 5028, 0, 5028, 11, 0, 0, 0, 0, 5028, 5028, 0, 5028,
11, 0, 0, 0, 11, 0, 5028, 0, 0, 5028, 0, 11, 0, 5028, 5028, 0, 0, 0, 11, 0, 11, 0, 0, 5028, 0, 11, 5028, 0, 0, 0
, 11, 0, 5028, 11, 0, 0, 5028, 5028])
ciphers = [[221, 4761, 791, 4565, 163, 2959, 2174, 2402, 3083, 4464, 340, 752, 4898, 709, 3085, 4393, 156, 3680,
 600, 2074, 3058, 4331, 1621, 2977, 2420, 891, 429, 4304, 3811, 2672, 3615, 3592, 1069, 561, 3183, 1197, 931, 16
25, 89, 4118, 1214, 2147, 3542, 3305, 1192, 2768, 4519, 2672, 1548, 2598, 4075, 1667, 577, 924, 810, 1239, 4033,
 2198, 2742, 4197, 2069], [1249, 2762, 483, 3324, 449, 1647, 1024, 3292, 285, 83, 1144, 1818, 2323, 2215, 1157,
2370, 3735, 1712, 3278, 1118, 3073, 4803, 783, 4765, 3120, 4709, 3304, 199, 987, 4939, 3969, 1749, 1045, 450, 40
, 1644, 1143, 2642, 1464, 3462, 1990, 1235, 4756, 2240, 2808, 4728, 3874, 3746, 2106, 926, 778, 1071, 3488, 3469
, 926, 4497, 2220, 1425, 4549, 3130, 4937], [4287, 3659, 4672, 521, 3387, 1631, 674, 1739, 1927, 1655, 4653, 477
2, 245, 216, 845, 3161, 2756, 1752, 2562, 2893, 443, 1195, 2430, 2438, 152, 4567, 965, 1200, 1709, 1109, 3349, 2
988, 147, 3214, 4391, 2726, 1531, 251, 2685, 1624, 2963, 2385, 4992, 4995, 3296, 3262, 4337, 1002, 63, 959, 2631
, 315, 4114, 1292, 3588, 1867, 3349, 3928, 3886, 1258, 2783], [834, 3120, 1420, 3534, 3926, 2862, 1840, 1083, 32
58, 3146, 1080, 145, 119, 3348, 4115, 1551, 3426, 2077, 3493, 4002, 3008, 1321, 2182, 4197, 3047, 3349, 1555, 70
9, 2803, 63, 2792, 538, 4943, 1081, 2204, 2993, 4398, 1007, 3623, 4243, 2276, 2332, 2287, 363, 3920, 2229, 3078,
 1438, 1152, 297, 3425, 4381, 3651, 3818, 2970, 159, 3726, 1190, 1645, 632, 2697]]

for cipher in ciphers:
    e = Sq(cipher)
    a = prikey * e
    print list(a)
```

这样得到四个a，然后进行第二步，计算a%p。注意这里如果大于一半（2519）的话，需要先转换为负数（减5039）再约。代码不连续，见谅

```
b1 = list(Sp([-121, -22, 133, 45, 10, -22, 122, -67, -67, -32, 32, 121, 100, -66, 43, 23, 166, -67, 65, 67, 33,
78, -45, -78, 89, 66, -33, -23, 32, 154, -34, -65, -43, -21, -22, -44, 22, -98, -43, -1, -1, 21, 175, 54, -45, 1
0, 76, 87, -144, -12, -34, -111, 32, 21, 65, 54, 21, 21, -78, 142, 32]))
b2 = list(Sp([-1, 164, -78, 21, -21, -1, 44, 165, 21, 12, -43, -1, -131, -32, 110, 43, 22, 12, 54, 34, 111, 34,
-44, -99, 67, 67, 21, -122, 78, 10, -12, -22, 76, 34, 12, -44, -143, 34, 78, -56, -23, -111, 32, 65, 43, -34, -1
00, 21, 21, -45, 65, 54, 43, -100, -67, -34, -1, 10, 21, -100, 76]))
b3 = list(Sp([144, -10, 34, -34, 78, -78, -100, -55, 66, -12, 34, 110, 22, -67, 78, 10, -10, 56, 45, -11, -89, 8
8, -133, -56, 23, 153, -54, 76, 111, -77, -165, -12, -65, 11, -44, 132, -98, 1, 100, -1, -1, 21, 109, -12, 21, 2
1, -89, -111, -23, -12, 65, 87, 98, -12, -56, 43, -144, 65, 43, 76, 21]))
b4 = list(Sp([32, -32, 54, 44, -23, -66, -23, 89, 23, -45, 89, 89, 23, -55, 45, 109, -11, 44, -22, -55, 21, 65,
-1, 56, 98, -34, 65, 110, 54, -76, 11, -34, -22, 32, 11, 66, -89, 32, 76, 55, -34, 43, -34, -67, -1, -12, 109, -
56, 10, -45, 120, 43, -12, 32, 87, -1, -111, 76, 120, -23, -89]))
```

得到的结果同上，大于一半（5）的转负数（减11）

```
b01 = [0, 0, 1, 1, -1, 0, 1, -1, -1, 1, -1, 0, 1, 0, -1, 1, 1, -1, -1, 1, 0, 1, -1, -1, 1, 0, 0, -1, -1, 0, -1,
1, 1, 1, 0, 0, 0, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1]
b02 = [-1, -1, -1, -1, 1, -1, 0, 0, -1, 1, 1, -1, 1, 1, 0, -1, 0, 1, -1, 1, 1, 1, 0, 0, 1, 1, -1, -1, 1, -1, -1,
 0, -1, 1, 1, 0, 0, 1, 1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1
]
b03 = [1, 1, 1, -1, 1, -1, -1, 0, 0, -1, 1, 0, 0, -1, 1, -1, 1, 1, 1, 0, -1, 0, -1, -1, 1, -1, 1, -1, 1, 0, 0, -
1, 1, 0, 0, 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]
b04 = [-1, 1, -1, 0, -1, 0, -1, 1, 1, -1, 1, 1, 1, 0, 1, -1, 0, 0, 0, 0, -1, -1, -1, 1, -1, -1, -1, 0, -1, 1, 0,
 -1, 0, -1, 0, 0, -1, -1, -1, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1,
 -1]
```

然后花了一上午的时间去找最后一步乘以Fp（说到底还是没能理解算法），结果发现不用乘，这个就是最终解（WHY?）

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-
from libnum import n2s

bs = [[0, 0, 1, 1, -1, 0, 1, -1, -1, 1, -1, 0, 1, 0, -1, 1, 1, -1, -1, 1, 0, 1, -1, -1, 1, 0, 0, -1, -1, 0, -1,
1, 1, 1, 0, 0, 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, 0, 0, -1, 1, 1, -1, 1, 1, 0, -1, 0, 1, -1, 1, 1, 1, 0, 0, 1, 1, -1, -1, 1, -1, -1, 0, -1
, 1, 1, 0, 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, 0, 0, -1, 1, 0, 0, -1, 1, -1, 1, 1, 1, 0, -1, 0, -1, -1, 1, -1, 1, -1, 1, 0, 0, -1, 1,
0, 0, 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, 0, -1, 0, -1, 1, 1, -1, 1, 1, 1, 0, 1, -1, 0, 0, 0, 0, -1, -1, -1, 1, -1, -1, -1, 0, -1, 1, 0, -1, 0
, -1, 0, 0, -1, -1, -1, 0, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1, -1]]

m = ''
for b in range(len(bs)):
    count = 0
    for i in range(len(bs[b])):
        count += ((bs[b][i]+1)%3)*3**i
    m += n2s(count)
print m

# 59d34a385e1b59c977eea74e92e0d9dc
```

再看一下这个哈希值，嗯，md5(md5(ntru))，应该就是没跑了
flag{59d34a385e1b59c977eea74e92e0d9dc}

alicehomework

这题还没看呢，队友就把答案翻出来了，看了大佬解答，嗯。。。我应该做不出来
这里直接搬运了，源地址
https://0xdktb.top/2020/07/31/WriteUp-中国电信2020天翼杯-Crypto/

再一次深感与大佬差距不可以道理计，匿了