Week3-6vv+

Crypto

LikiPrime

打开文件，注意到n巨大，无法直接分解，就安装下RsaCtfTool

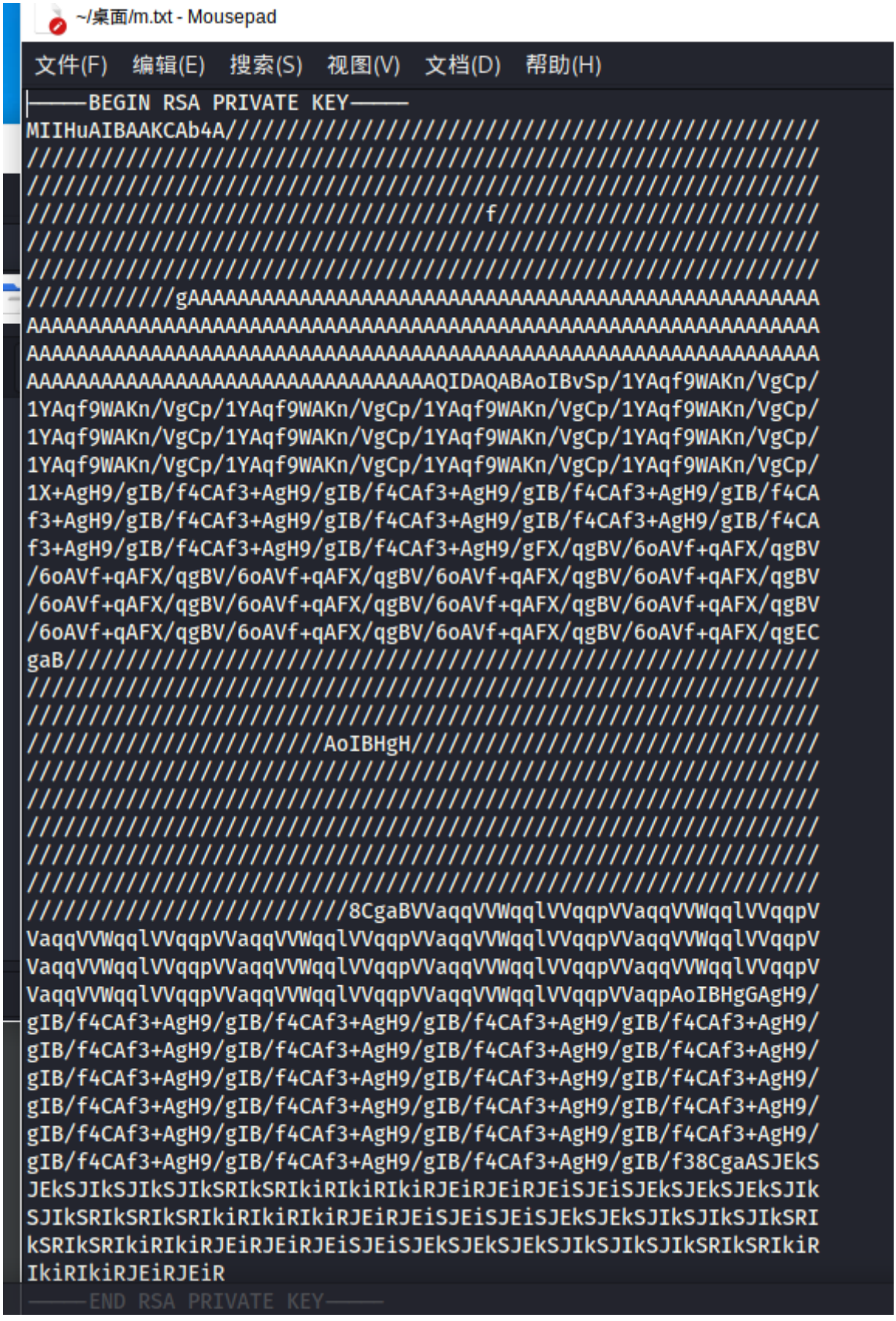
把n，e转换为pem格式

```
┌──(magnesium㉿kali)-[~/RsaCtfTool]
└─$ python RsaCtfTool.py --createpub -n 46428490480520354965036337048952936312177107210484806430930662734535390206065032629579799207254959806444743815824019876594838450877851589955835248019744116006600044695216060438284643983812683857418391061076791522852680527856935317377820783114124274163527296450205331019792978157951000350834018873192006629066168210285867895906847893354437627520436293420848347984596180904747532533554105610254721286750379186665231076449107407661050288263147305990184974954928620638007287786926005881958288820488298072121928363857655583851582466697806222889069190838082491113819347804421024619157144538425118702454989031134165715273862977767430005775189646427128196363586309432322645639174989268135957426553745861978616777135684343927102295460689896802360759649131947056870471710771817942549912018779512987877358337168678445758891750575795153116850268675866355329047836020447026922102080362331150225398142640335743967220632637309590996745246910198667848953208765106491886816469714977694486726982759929023769520531793453797252293580955739852760297841385296819079616874372929389547522464153 7 -e 65537
private argument is not set, the private key will not be displayed, even if recovered.
-----BEGIN PUBLIC KEY-----
MIIB3zANBgkqhkiG9w0BAQEFAAOCAcwAMIIBxwKCAb4A////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
/f////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////
////////////////////////////////////gAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAQID
AQAB
-----END PUBLIC KEY-----
```

然后将公钥复制进公钥文件里，根据公钥生成私钥

```
┌──(magnesium㉿kali)-[~/RsaCtfTool]
└─$ python RsaCtfTool.py --publickey /home/magnesium/桌面/n.key --private
```

复制生成私钥文件



这个时候我们得到了私钥的pem格式，这个也就是说包含了d的信息，就可以把pem文件转化为n，d的信息

```
└─$ python RsaCtfTool.py --dumpkey --key /home/magnesium/桌面/d.key \                1 ×
>
private argument is not set, the private key will not be displayed, even if recovered.
n: 46428490480520354965036337048952936312177107210484806430930662734535390206065032629579799207254959806444743
81582401987659483845087785158995583524801974411600660004469521606043828464398381268385741839106107679152285268
05278569353173778207831141242741635272964502053310197929781579510003508340188731920066290661682102858678959068
47893354437627520436293420848347984596180904747532533554105610254721286750379186665231076449107407661050288263
14730599018497495492862063800728778692600588195828882048829807212192836385765558385158246669780622288906919083
80824911138193478044210246191571445384251187024549890311341657152738629777674300057751896464271281963635863094
32322645639174989268135957426553745861978616777135684343927102295460689896802360759649131947056870471710771817
94254991201877951298787735833716867844575889175057579515311685026867586635553290478360204470269221020803623311
50225398142640335743967220632637309590996745246910198667848953208765106491886816469714977694486726982759929023
76952053179345379725229358095573985276029784138529681907961687437292938954752246415 37
e: 65537
d: 77077372541932261473609617024369126917083010581820146477337322512740138462545974794364742874244161724540154
83103671761865081440187238129586637891549122113953214344359429799007713763622745655131737981517990074183570153
71687876247391657674474391064746477832347190493921747024737718398589830285373667285704448235210839541392964991
53925217248483669125360089541789534523467391496887859540201795206546837195322979326496111065409307086534418289
79076010932458139368740309657395542622177950801839218971807935960287470287381376626795977651885315657288873409
86161397150246232036553519792079001163180694624897916603106343065963758602553926821994399899387763161456806939
85949532430688368845609972143076914453623533161988629961898590751920433141879209935086841628332272425616319744
31462022126923575182496881291602500901174882244733642076480036339767974046113854043621828870645647707245404828
31043005176387272197349168925019470641628096392062982237553509096120775879454378983657609664663657876869899429
82419516689446811068158249151452996133869194019246042109165740096158838245282333747 3
p: 10407932194664399081925240327364085538615262247266704805319112350403608059673360298012239441732324184842421
61395428100779138356624832346490813990660567732076292412950938922034577318334966158355047295942054768981121169
36771475484788669625013844382602917323488853111608285384165850282556046662248318909188018470682222031405210266
98435488732958028878050869736186900714720710555703168729087
q: 44608755718375842957115170640210180988620863241285990111199121996340468579282047336911254526900398902615324
```

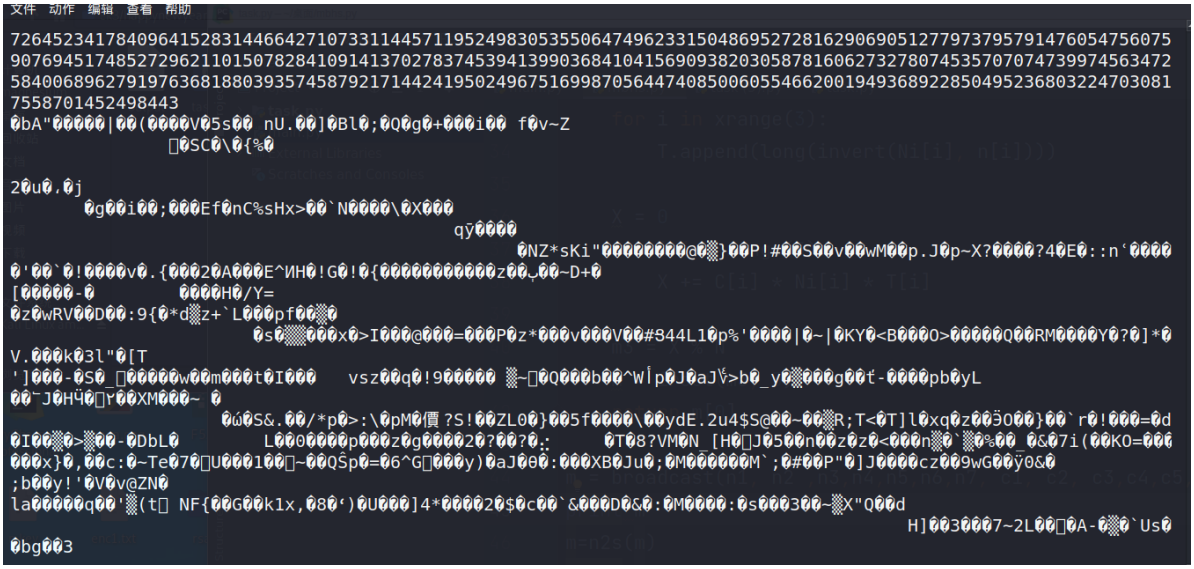
把d复制出来，就可以使用普通的rsa解密脚本了，这里有个小知识是，此时尽管我们得到了p和q，但是在产生n且生成公钥后，p和q就对解密没有作用了，只需n,e,d,m即可知三求四

exp

```
from libnum import *
import gmpy2
n=46428490480520354965036337048952936312177107210484806430930662734535390206065032629579799207254959806444743815824019876594838450877851589955835248019744116006600044695216060438284643983812683857418391061076791522852680527856935317377820783114124274163527296450205331019792978157951000350834018873192006629066168210285867895906847893354437627520436293420848347984596180904747532533554105610254721286750379186665231076449107407661050288263147305990184974954928620638007287786926005881958288820488298072121928363857655583851582466697806222889069190838082491113819347804421024619157144538425118702454989031134165715273862977767430005775189646427128196363586309432322645639174989268135957426553745861978616777135684343927102295460689896802360759649131947056870471710771817942549912018779512987877358337168678445758891750575795153116850268675866355532904783602044702692210208036233115022539814264033574396722063263730959099674524691019866784895320876510649188681646971497769448672698275992902376952053179345379725229358095573985276029784138529681907961687437292938954752246415374
e=65537
c=37096875518023866467977457857681578790270506093289775688856947317905847125794604654087583066134185674159628307957634541028502725678059090677424751714345308195313233767628444834251429785846705354445569102983111587852446743461850245867441904341667421254024474270246412112999107107004138300696578141672754250863666081865294889731589680244701282220423947221963763697427470120915123628559857965684731338342997048235067247153716222795565760849643190453293854721705024559392803903541382306559975167813746674109215629262484286416248866737236719770411192539220976662164358342165347759715599610231717408100011088752474114007984413321024183446683265646928676897812922482407883434225594652388550567533816485504668981321109411926531701826750072660695040799990642681677399721558032297007835281771363696279499537896513203796824429700942684936774195892656600728634199518759235135362071614292681609347362124450302551663741051915772732121368261798488842908757135648473263934844258817202643266593801439725574854213639998017170911451912793210107131617815394232933945106751037350138599495506
d=77077372541932261473609617024369126917083010581820146477337322512740138462545974794364742874244161724540154831036717618650814401872381295866378915491221139532143443594297990077137636227456551317379815179900741835701537168787624739165767447439106474647783234719049392174702473771839858983028537366728570444823521083954139296499153925217248483669125360089541789534523467391496887859540201795206546837195322979326496111065409307086534418289790760109324581393687403096573955426221779508018392189718079359602874702873813766267959776518853156572888734098616139715024623203655351979207900116318069462489791660310634306596375860255392682199439989938776316145680693985949532430688368845609972143076914453623533161988629961898590751920433141879209935086841628332272425616319744314620221269235751824968812916025009011748822447336420764800363397679740461138540436218288706456477072454048283104300517638727219734916892501947064162809639206298223755350909612077587945437898365760966466365787686989942982419516689446811068158249151452996133869194019246042109165740096158838245282333747 3

m=pow(c,d,n)
print(hex(m))
m=n2s(m)
print(m)
```

HappyNewYear!!

打开output发现是7组e,n,c，且e相同，可以推测是低加密指数广播攻击，用普通的脚本试了下发现是堆乱码



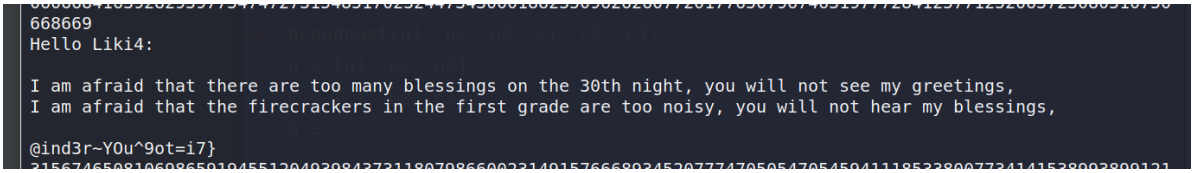
推测7组信息并非由同一个message得来，就加个循环，排列组合，每次用三组明文求密文

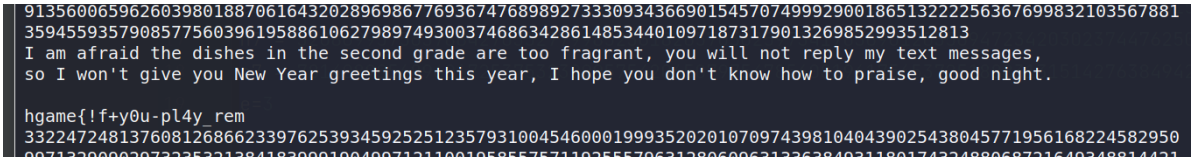
exp

```
import random
from gmpy2 import invert, iroot
from libnum import *

n1=...
c1=...
n2=...
c2=...
...
...     #篇幅过长省略，把output复制进来
e=3

def broadcast(n1, n2 ,n3, c1, c2, c3):
    n = [n1, n2, n3]
    C = [c1, c2, c3]
    N = 1
    for i in n:
        N *= i

    Ni = []
    for i in n:
        Ni.append(N / i)

    T = []
    for i in xrange(3):
        if(gcd(Ni[i],n[i]) == 1):
            T.append(long(invert(Ni[i], n[i])))
        else:
            return 0

    X = 0
    for i in xrange(3):
```

```
        X += C[i] * Ni[i] * T[i]

    m3 = X % N
    m = iroot(m3, 3)
    return m[0]

def main():
    e = 3
    n=[n1,n2,n3,n4,n5,n6,n7]
    c=[c1,c2,c3,c4,c5,c6,c7]
    for i in range(0,5):
        for j in range(i,6):
            for z in range(j,7):
                m = broadcast(n[i],n[j],n[z],c[i],c[j],c[z])
                print m
                m=n2s(m)
                print m

if __name__=="__main__":
    main()
```

然后就在一大堆乱码中得出一点点信息





`@ind3r~YOu^9ot=i7}`

`hgame{!f+y0u-pl4y_rem`

就得到了flag