RRRRsa

套娃题，第一层跟巅峰极客tryRSA基本一致，第二层$modq1$稍微变换一下，消去p1之后和n2做一次gcd就好了，具体看代码实现

In [1]:
```
from Crypto.Util.number import *
from Crypto.Cipher import AES
import gmpy2
```

In [2]:
```
c=134923927174698178668834314754537709518374762413719897146837375583957697314165223008519178879579457661328641513828771
n1=750035573790802522195178259989901832266591170197707350805234095617572258836510408825475197481075887194982619228168611
c1=681119010920278130070996278938968385174269710828772040471104047878232792115081837834688914746613651399333259811915211
hint1=235520907163817694849907841168755588957155528969833134067640424163187100762561664724265535202402650239784499459711
hint2=527232296985307678979794339144708311532688270083723072396303871007522268507980233624444992119449967783638945287511
n2=114535923043375970380117920548097404729043079895540320742847840364455024050473125998926311644172960176471193602850411
c2=670542036669016911812152625874471809102254733391432601008311183135214710298893041762354341296322371169939103169780911
hint3=255909234167568135438805549638875769607073336073778894010337184193012788021572048810391163503218721621189779770611
hint4=104100726926923869566862741238876132366916970864374562947844669556403268955625670105641264367038885706425427864911
```

In [4]:
```
e1 = 202020
e2 = 212121
tt = (hint2 - e2) * inverse(2021, n1) * 2020 % n1
gcd(pow(tt, e1, n1) - hint1, n1)
```

Out[4]:
```
108665069130470381377409355667427082618324348566492530652986956525715829856056914276408994143308245251612797529248957
403772083185844841687225884182285399 43
```

In [6]:
```
p1 = 108665069130470381377409355667427082618324348566492530652986956525715829856056914276408994143308245251612797529248957
q1 = n1 // p1
assert p1 * q1 == n1
phi = (p1-1) * (q1-1)
d = inverse(65537, phi)
p = pow(c1, d, n1)
```

In [9]:
```
e3 = 202020
e4 = 212121
tt1 = pow(hint3, e2, n2) * inverse(pow(2020, e1*e2, n2), n2) * pow(2021, e1*e2, n2)
tt2 = pow(hint4, e1, n2)
gcd(tt2 - tt1, n2)
```

Out[9]:
```
967726984126262615401453880293246726749175339749044402409170233601784793481378296273846281596844525889887627788740490
197612323705703476785904241224980188 9
```

In [11]:
```
q2 = 967726984126262615401453880293246726749175339749044402409170233601784793481378296273846281596844525889887627788740490
p2 = n2 // q2
phi = (p2-1) * (q2-1)
d = inverse(65537, phi)
q = pow(c2, d, n2)
```

In [18]:
```
print(f'recover p = {p}')
print(f'recover q = {q}')
```

```
recover p = 104278778286079608288241968801899824472278731591440566541826034975538222259841122798493664641941347440258
369532335381204157861328497937489171065115474956 17
recover q = 809398095674643485648773574352244588290870037644747209197252178668414754896803709354429172169893103011413
601263137033738438605428179378897745677728538436 3
```

In [1]:
```
from Crypto.Util.number import *
from libnum import *
```

In [2]:
```
c = 134923927174698178668834314754537709518374762413719897146837375583957697314165223008519178879579457661328641513828771
p = 104278778286079608288241968801899824472278731591440566541826034975538222259841122798493664641941347440258369532335
q = 809398095674643485648773574352244588290870037644747209197252178668414754896803709354429172169893103011413601263137
```

In [3]:
```
n = int(p) * int(q)
phi = (p-1) * (q-1)
d = inverse(65537, phi)
m = pow(c, d, n)
print(long_to_bytes(m))
```

```
b'GKCTF{f64310b5-d5e6-45cb-ae69-c86600cdf8d8}'
```

```
from Crypto.Util.number import *
from Crypto.Cipher import AES
import gmpy2
c=134923927174698178668834314754537709518374762413719897146837375583957697314165223008519178879579457661328641513828771
4165223008519178879579457661328641513828774621420181298527034372405336846045083799502936432948777257736755059126222088134356251776966147816012164658075692013801516699426052084256452583721344655474523764674658330133870185429995562042758
```

```
n1=75003557379080252219517825998990183226659117019770735080523409561757225883651040882547519748107588719498261922816865626714101556207649929655822889945870341168644508079317582220034374613066751916750036253423990673764234066999306874078424803774652754587494762629397701664706287999727238636073466137405374927829
c1=68111901092027813007099627893896838517426971082877204047110404787823279211508183783468891474661365139933325981191524511345219830693064573462115529345012970089065201176142417462299650761299758078141504126185921304526414911455395289228444974516503526507906721378965227166653195076209418852399008741560796631569
hint1=23552090716381769484990784116875558895715552896983313406764042416318710076256166472426553520240265023978449945974218435787929202289208329156594838420190890104226497263852461928474756025539394996288951828172126419569993301524866753797584032740426259804002564701319538183190684075289055345581960776903740881951
hint2=52723229698530767897979433914470831153268827008372307239630387100752226850798023362444499211944996778363894528759290565718266340188582253307004810850030833752132728256929572703630431232622151200855160886614350000115704689605102500273815157636476901150408355565958834764444192860513855376978491299658773170270
n2=11453592304337597038011792054809740472904307989554032074284784036445502405047312599892631164417296017647119360285042760789919181061695302132474213749274615992128498214632017535639532589040770469701841245635086299084960620032308471735263028253915667063602592442586574119650647816392231289438428588984835524448 9
c2=67054203666901691181215262587447180910225473339143260100831118313521471029889304176235434129632237116993910316978096018724911531011857469325115308802162172965564951703583450817489247675458024801774590728726471567407812572210421642171456850352167810755440990035255967091145950569246426544351461548548423025004
hint3=25590923416756813543880554963887576960707333607377889401033718419301278802157204881039116350321872162118977797069089653428121479486603744700519830597186045931412652681572060953439655868476311798368015878628002547540835719870081007505735499581449077950263721606955524302365518362434928190394924399683131242077
hint4=10410072692692386956686274123887613236691697086437456294784466955640326895562567010564126436703888570642542786494139260159343730525829719811181922791545308179788956566227600312290113975515300221912636661102173606601674156223299804725333514167620337652174296536513359794366983807621044448545829624095166840251 3
e1 = 202020
e2 = 212121
tt = (hint2 - e2) * inverse(2021, n1) * 2020 % n1
gcd(pow(tt, e1, n1) - hint1, n1)
p1 = 10866506913047038137740935566742708261832434856649253065298695652571582985605691427640899414330824525161279752924895740377208318584484168722588418228539943
q1 = n1 // p1
assert p1 * q1 == n1
phi = (p1-1) * (q1-1)
d = inverse(65537, phi)
p = pow(c1, d, n1)

e3 = 202020
e4 = 212121
```

```
tt1 = pow(hint3, e2, n2) * inverse(pow(2020, e1*e2, n2), n2) * pow(2021, e1*e2, n2)
tt2 = pow(hint4, e1, n2)
gcd(tt2 - tt1, n2)
q2 = 967726984126262615401453880293246726749175339749044402409170233601784793481378296273846281596844525889887627788740490197612323705703476785904241224980 1889
p2 = n2 // q2
phi = (p2-1) * (q2-1)
d = inverse(65537, phi)
q = pow(c2, d, n2)

print(f'recover p = {p}')
print(f'recover q = {q}')

c = 134923927174698178668834314754537709518374762413719897146837375583957697314165223008519178879579457661328641513828774621420181298527034372405336846045083799502936432948777257736755059126222088134356251776966147816012164658075692013801516699426052084256452583721344655474523764674658330133870185429995 62042758
p = 104278778286079608288241968801899824472278731591440566541826034975538222259841122798493664641941347440258369532335381204157861328497937489171065115474 95617
q = 809398095674643485648773574352244588290870037644747209197252178668414754896803709354429172169893103011413601263137033738438605428179378897745677728538 4363

n = int(p) * int(q)
phi = (p-1) * (q-1)
d = inverse(65537, phi)
m = pow(c, d, n)
print(long_to_bytes(m))

# b'GKCTF{f6431Ob5-d5e6-45cb-ae69-c8660Ocdf8d8}'
```