Tamevic's Ctf-Crypto writeup@Crypto实验‘RSA’(2019强网杯’强网先锋-辅助‘)

原创

TameVic 于 2019-06-27 14:37:48 发布 1288 收藏

分类专栏： Crypto 文章标签： Crypto

本文链接：https://blog.csdn.net/qq_43264421/article/details/93874368

Tamevic’s Ctf-Web writeup@Crypto实验‘RSA’(2019强网杯’强网先锋-辅助‘)

> 还好在课上弥补一下当时比赛最后十分钟差点做出来的遗憾…

题目源码：

```
from Crypto.Util.number import getPrime,bytes_to_long,long_to_bytes
import sys
from fractions import gcd
# p=getPrime(1024)
# q=getPrime(1024)
# e=65537
# n=p*q
# m=bytes_to_long(flag)
# c=pow(m,e,n)
# print c,e,n

# p=getPrime(1024)
# e=65537
# n=p*q
# m=bytes_to_long("1"*32)
# c=pow(m,e,n)
# print c,e,n
```

分析

还是简单分析一下代码:

第一次加密**flag**的时候p和q都取得是随机素数，c1是加密后的密文，第二次加密是明文是32个1，但是这里只是重新取了p，却用了和加密flag一**样的q**。所以尝试以找到q为突破口。

而在我们这里有所有的密文和公钥。只需要对n1和n2取最大公因数就能找到q。

```
gcd(n1,n2)
```

知道q以后就能知道第一次的p。

```
n1/q
```

然后就可以按欧拉公式计算φn。

```
ϕn=(p-1)*(q-1)
```

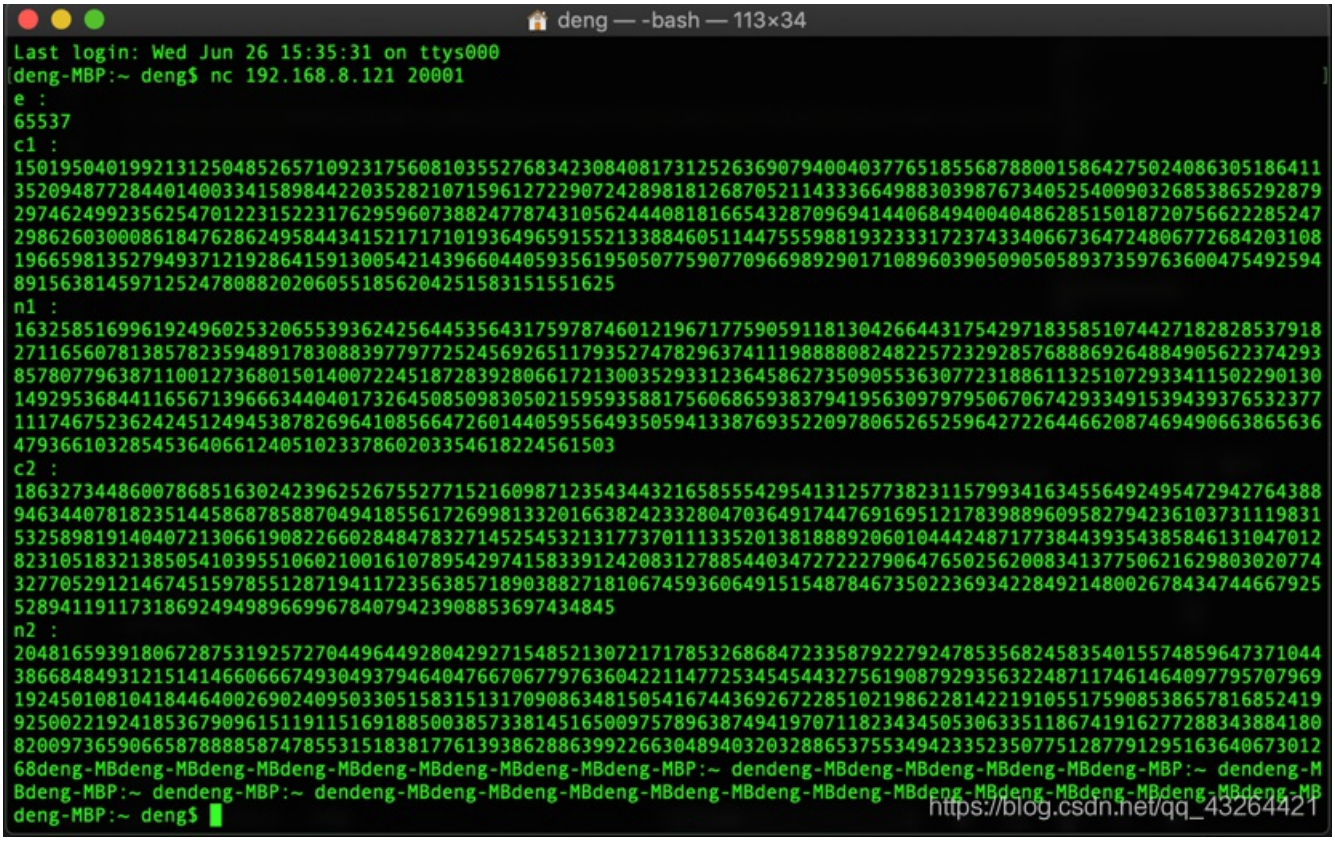
再用拓展欧几里得计算e模φn的乘法逆元即可以求得私钥d。

```
d=computeD(ϕn,e)
```

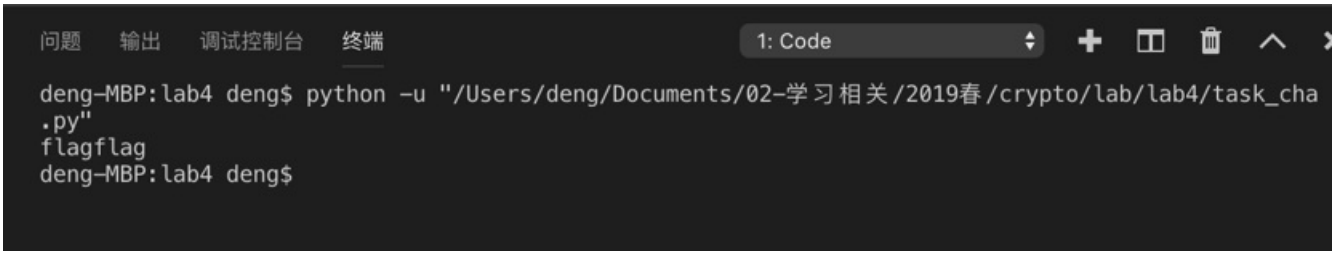
再对n1进行解密，将long型转成bytes型即可求得flag。

```
m1=pow(c1,d,n1)
```

结果



先交互一下，get到各个需要的参数



运行脚本 getflag

附上脚本：

```
# -*- coding: UTF-8 -*-
# flag=open("flag","rb").read()
```

```
from Crypto.Util.number import getPrime,bytes_to_long,long_to_bytes
import sys
from fractions import gcd
# p=getPrime(1024)
# q=getPrime(1024)
# e=65537
# n=p*q
# m=bytes_to_long(flag)
# c=pow(m,e,n)
# print c,e,n

# p=getPrime(1024)
# e=65537
# n=p*q
# m=bytes_to_long("1"*32)
# c=pow(m,e,n)
# print c,e,n

def computeD(fn, e):
    (x, y, r) = extendedGCD(fn, e)
    #y maybe < 0, so convert it
    if y < 0:
        return fn + y
    return y

def extendedGCD(a, b):
    #a*xi + b*yi = ri
    if b == 0:
        return (1, 0, a)
    #a*x1 + b*y1 = a
    x1 = 1
    y1 = 0
    #a*x2 + b*y2 = b
    x2 = 0
    y2 = 1
    while b != 0:
        q = a / b
        #ri = r(i-2) % r(i-1)
        r = a % b
        a = b
        b = r
        #xi = x(i-2) - q*x(i-1)
        x = x1 - q*x2
        x1 = x2
        x2 = x
        #yi = y(i-2) - q*y(i-1)
        y = y1 - q*y2
        y1 = y2
        y2 = y
    return(x1, y1, a)
e=65537

c1=150195040199213125048526571092317560810355276834230840817312526369079400403776518556878800158642750240863051864113520948772844014003341589844220352821071596127229072428981812687052114333664988303987673405254009032685386529287929746249923562547012231522317629596073882477874310562444081816654328709694144068494004048628515018720756622285247298626030008618476286249584434152171710193649659155213388460511447555988193233317237433406673647248067726842031081966598135279493712192864159130054214396604405935619505077590770966989290171089603905090505893735976360047549259489156381459712524780882020605518562042515831515516251
```

```
n1=16325851699619249602532065539362425644535643175978746012196717759059118130426644317542971835851074427182828537918271165607813857823594891783088397797725245692651179352747829637411198888082482257232928576888692648849056223742938578077963871100127368015014007224518728392806617213003529331236458627350905536307723188611325107293341150229013014929536844116567139666344040173264508509830502159593588175606865938379419563097979506706742933491539439376532377111746752362424512494538782696410856647260144059556493505941338769352209780652652596427226446620874694906638656364793661032854536406612405102337860203354618224561503

c2=18632734486007868516302423962526755277152160987123543443216585554295413125773823115799341634556492495472942764388946344078182351445868785887049418556172699813320166382423328047036491744769169512178398896095827942361037311198315325898191404072130661908226602848478327145254532131773701113352013818889206010444248717738443935438584613104701282310518321385054103955106021001610789542974158339124208312788544034727222790647650256200834137750621629803020774327705291214674515978551287194117235638571890388271810674593606491515487846735022369342284921480026784347446679255289411911731869249498966996784079423908853697434845

n2=20481659391806728753192572704496449280429271548521307217178532686847233587922792478535682458354015574859647371044386684849312151414660666749304937946404766706779763604221147725345454432756190879293563224871174614640977957079691924501081041844640026902409503305158315131709086348150541674436926722851021986228142219105517590853865781685241992500221924185367909615119115169188500385733814516500975789638749419707118234345053063351186741916277288343884180820097365906658788885874785531518381776139386288639922663048940320328865375534942335235077512877912951636406730126871407199553335885703047258995712973704426087569397

q=gcd(n1,n2)
p=n1/q
fn=(p-1)*(q-1)
d=computeD(fn,e)
m=pow(c1,d,n1)
print long_to_bytes(m)
```

End

这道题就是简单的一个小破绽，考的主要是对rsa基本概念的掌握情况。其实rsa看似简单实际上还有很多可以学习的攻击方式，如同样打比赛没做出来的rsa的高位攻击…任重而道远！