虎符 CTF 2020 Crypto GM writeup

原创

Slightwindsec 已于 2022-01-30 22:53:00 修改 1150 收藏 1

分类专栏： CTF 文章标签： 密码学 python

于 2020-04-19 20:13:12 首次发布

本文链接：https://blog.csdn.net/qq_41956187/article/details/105621086

博客：https://blog.slight-wind.com/

GM writeup

审代码发现生成的 N 和 phi 很特殊，有了 N 和 phi 显然可以算出来 p , q。

看主要加密部分是把明文转化成二进制流，对 bi 逐个加密：

```
r = random.randint(1, N)
if gcd(r, N) == 1:
    br = bin(r)[2:]
    c = (pow(x, int(br + bi, 2), N) * r ** 2) % N
    cipher.append(c)
    break
```

如果令 e = int(br + bi, 2) ，

$c=\quad x*r \mod N$ ，如果 bi 为 0 ，那么 e 就是偶数，可以表示为 e = 2*k ，那么 c 就是

c 是否为 mod N 下的平方剩余可以通过勒让德符号来判断，N 不是个素数所以要分解 N ，然后通过勒让德符号来判断

$$c \mod p$$

是否为模 p 下的平方剩余，逐个判断就得到了明文的二进制流。

先用二分法算出 p ：

```
phi = 943345166174941322591941459524332131176290203790885095479970339608386371864113650305321599557655800317124919296997286484079529878473055321041798371459376455758292743478491517763973199831089116868599924093740787177136997171351531363419874461607461086692409485467190033481035312744677860713715775192568024399071118090459884125566044321409184867437624516395377471711324620392824450903373418491300586583762013483114288071183225663479759077341383165973361572257483025749680141776033707348483817055449795303348713163497337114335750702773189940277716951677026421865648348704539315689483288562884385831667979320557234868882086483487045393156894832885628843858316679793205572348688820
N = 943345166174941322591941459524332131176290203790885095479970339608386371864113650305321599557655800317124919296997286484079529878473055321041798371459376455758292743478491517763973199831089116868599924093740787177136997171351531363419874461607461086692409485467190033481035312744677860713715775192568024399090552814107286416854451927989722449484920618426220213030582018756914805724773124365108425819400945993670290965544896969358980098726637824989115794026289855404724760504954999778351110737324846258731832315252496968472469031691876138715488249636776962692129909168837711893869307448632599530840323222828283975697

K=N-phi+1#p+q
L,R=1,K
while L<R:
    mid=(L+R)//2
    if K<=mid:
        break
    y=(K-mid)*mid
    if y>N:
        R=mid
    elif y<N:
        L=mid
    else:
        print(mid)
        break
#941305244949403565068759409019015068729846990336109288142693109780033763077305806672342096403094435645602674146306448617123315594406588532018045567817844933762844464263930748829429574468699255584221466777408544991533387620166945600337512668984373809028537024524089333725318464411474508329436122818256951097 1
```

exp:

```
from Crypto.Util.number import *
N = 943345166174941322591941459524332131176290203790885095479970339608386371864113650305321599557655800317124919
296997286484079529878473055321041798371459376455758292743478491517763973199831089116868599924093740787177136997171351531363419874461607461086692409485467190033481035312744677860713715775192568024399090552814107286416854451927989722449484920618426220213030582018756914805724773124365108425819400945993670290965544896969358980098726637824989115794026289855404724760504954999778351110737324846258731832315252496968472469031691876138715488249636776962692129909168837711893869307448632599530840323222828283997569 7
p = 941305244949403565068759409019015068729846990336109288142693109780033763077305806672342096403094435645602674146306448617123315594406588532018045567817844933762844464263930748829429574468699255584221466777740854499153338762016694560033751266898437380902853702452408933372531846441147450832943612281825695109 71

def Legrend(a, p):
    if a == 1:
        return 1
    if p % a == 0:
        return 0
    if a % 2 == 0:
        return Legrend(a // 2, p) * pow(-1, (pow(p, 2) - 1) // 8)
    return Legrend(p % a, a) * pow(-1, (a - 1)*(p - 1) // 4)

def read_file():
    f = open("output")
    fileread = f.read()
    cipher_list = []
    Number = "0123456789"
    temp = ''
    for i in fileread:
        if i not in Number and len(temp) != 0:
            cipher_list.append(int(temp))
            temp = ''
        elif i in Number:
            temp += i
    return cipher_list

cipher_list = read_file()
ans=''
for cip in cipher_list:
    if Legrend(cip,p)==1:
        ans+='0'
    elif Legrend(cip,p)==-1:
        ans+='1'
m=int(ans,2)
print(long_to_bytes(m))
```

flag{bd4f1790-f4a2-4904-b4d2-8db8b24fd864}