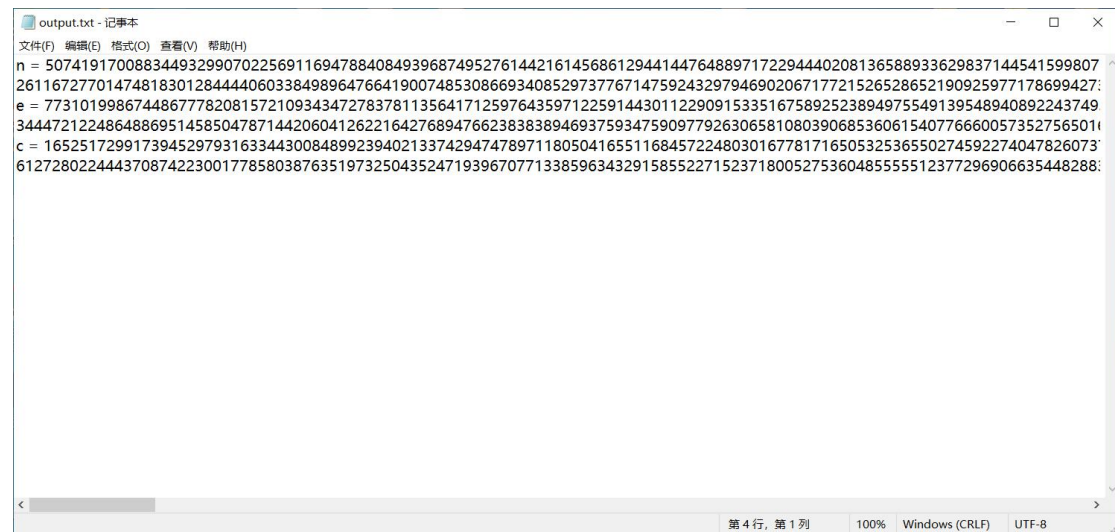


RSA Attack 3 题解

下载附件，发现加密指数 e 过大。



```
output.txt - 记事本
文件(F) 编辑(E) 格式(O) 查看(V) 帮助(H)
n = 5074191700883449329907022569116947884084939687495276144216145686129441447648897172294440208136588933629837144541599807
2611672770147481830128444406033849896476641900748530866934085297377671475924329794690206717721526528652190925977178699427
e = 7731019986744867778208157210934347278378113564171259764359712259144301122909153351675892523894975549139548940892243749
3444721224864886951458504787144206041262216427689476623838389469375934759097792630658108039068536061540776660057352756501
c = 1652517299173945297931633443008489923940213374294747897118050416551168457224803016778171650532536550274592274047826073
6127280224443708742230017785803876351973250435247193967077133859634329158552271523718005275360485555512377296906635448288:
```

这样会导致解密指数过小。

于是我们采用低解密指数攻击



```
1  # -*- coding: utf-8 -*-
2  """
3  Created on Fri Feb  4 13:40:13 2022
4
5  @author: chax
6  """
7  from libnum import n2s
8  import gmpy2
9  import libnum
10 # numerator(n):分子, denominator(d): 分母
11 def t_cf(n, d): # 将分数 x/y 转为连分数的形式
12     res = []
13     while d:
14         res.append(n // d)
15         n, d = d, n % d
16     return res
17
18
19 def cf(sub_res): # 得到渐近分数的分母和分子
20     n, d = 1, 0
21     for i in sub_res[::-1]: # 从后面往前循环
22         d, n = n, i * n + d
23     return d, n
24
25
26 def list_fraction(x, y): # 列出每个渐近分数
27     res = t_cf(x, y)
28     res = list(map(cf, (res[0:i] for i in range(1, len(res))))) # 将连分数的结果逐一截取以求渐近分数
29     return res
30
```

```

31
32 def get_pq(a, b, c): # 由p+q和pq的值通过维达定理来求解p和q(解二元一次方程)
33     par = gmpy2.isqrt(b * b - 4 * a * c) # 由上述可得, 开根号一定是整数, 因为无解
34     x1, x2 = (-b + par) // (2 * a), (-b - par) // (2 * a)
35     return x1, x2
36
37
38 def wienerAttack(e, n):
39     for (d, k) in list_fraction(e, n): # 用一个for循环来注意试探e/n的连续函数的渐近分数, 直到找到一个满足条件的渐近分数
40         if k == 0: # 可能会出现连分数的第一个为0的情况, 排除
41             continue
42         if (e * d - 1) % k != 0: # ed=1 (mod φ(n)) 因此如果找到了d的话, (ed-1)会整除φ(n), 也就是存在k使得(e*d-1)//k=φ(n)
43             continue
44
45         phi = (e * d - 1) // k # 这个结果就是 φ(n)
46
47         px, qy = get_pq(1, n - phi + 1, n)
48
49         if px * qy == n:
50             p, q = abs(int(px)), abs(int(qy)) # 可能会得到两个负数, 负负得正未尝不会出现
51             d = gmpy2.invert(e, (p - 1) * (q - 1)) # 求ed=1 (mod φ(n))的结果, 也就是e关于 φ(n)的乘法逆元d
52             return d
53     print("求解d失败")
54
55 n = 5074191700883449329907022569116947884084939687495276144216145686129441447648897172294440208136588933629837144541599807190263663613
56 e = 7731019986744867778208157210934347278378113564171259764359712259144301122909153351675892523894975549139548940892243749367025255097
57 c = 1652517299173945297931633443008489923940213374294747897118050416551168457224803016778171650532536550274592274047826073731074774197
58 d=wienerAttack(e,n)
59 m=pow(c,d,n)
60 print(n2s(int(m)))

```

得到 flag