RSA 中根据 (N, e, d) 求 (p, q)

转载

ayang1986 于 2021-01-16 16:42:50 发布 1867 收藏 5
分类专栏： 密码学
原文链接：https://www.40huo.cn/blog/get-pq-from-ned.html

湖湘杯有一道题是知道 (N,e,d) 求 (p,q)，当时用了 $e\cdot d-1=h\cdot\varphi(n)$ 这个公式，爆破 h，考虑 φ(n) 与 N 相差不大，可以认为位数相同，求出 φ(n) 之后再根据 $N=p\cdot q$ 和 $\varphi(n)=(p-1)(q-1)$ 联立一个方程。

$N=pq$

$\varphi(n)=(p-1)(q-1)$

$\Rightarrow N-\varphi(n)+1=p+q$

可得到一个一元二次方程

$X2-(N-\varphi(n)+1)X+N=(X-p)(X-q)$

根据求根公式即可解出 p 和 q。

```
# coding=utf-8
import gmpy2


def cal_bit(num):
    return len(bin(num)) - 2

d = 5
e = 884471203420353290772038018901751814412278435487123949154059830988049860742284919937163038613467133369
n = 0x009d70ebf2737cb43a7e0ef17b6ce467ab9a116efedbecf1ead94c83e5a082811009100708d690c43c3297b787426b926568a

k = e * d - 1

while k % 2 == 0:
    k /= 2
    if cal_bit(k) == cal_bit(n):
        print k
        break

a = 1
b = (n - k + 1)
c = n
p = (b + gmpy2.iroot(b**2-4*a*c, 2)[0])/2
q = n / p
print int(p)
print q
```

之后看到有人发了 writeup，用了 Calculate primes p and q from private exponent (d), public exponent (e) and the modulus (n) 链接里的算法。

The steps involved are:

1. Let k=de–1. If k is odd, then go to Step 4.
2. Write k as k=2t·r, where r is the largest odd integer dividing k, and t≥1. Or in simpler terms, divide k repeatedly by 2 until you reach an odd number.

   For i=1 to 100 do:

   1. Generate a random integer g in the range [0, n−1].
   2. Let y=grmodn
   3. If y=1 or y=n–1, then go to Step 3.1 (i.e. repeat this loop).

      For j=1 to t–1 do:

      1. Let x=y2modn
      2. If x=1, go to (outer) Step 5.
      3. If x=n–1, go to Step 3.1.
      4. Let y=x.

   5. Let x=y2modn
   6. If x=1, go to (outer) Step 5.
   7. Continue
4. Output “prime factors not found” and stop.
5. Let p=GCD(y–1,n) and let q=n/p
6. Output (p,q) as the prime factors.

原 writeup 里的代码是基于 sage 库的，而且有点看脸，没好好处理结果，改写了一个纯 Python 的。

```
# coding=utf-8
import random
import libnum

d = 5
e = 884471203420353290772038018901751814412278435487123949154059830988049860742284919937163038613467133369
n = 0x009d70ebf2737cb43a7e0ef17b6ce467ab9a116efedbecf1ead94c83e5a082811009100708d690c43c3297b787426b926568a

k = e * d - 1

r = k
t = 0
while True:
    r = r / 2
    t += 1
    if r % 2 == 1:
        break

success = False

for i in range(1, 101):
    g = random.randint(0, n)
    y = pow(g, r, n)
    if y == 1 or y == n - 1:
        continue

    for j in range(1, t):
        x = pow(y, 2, n)
        if x == 1:
            success = True
            break
        elif x == n - 1:
            continue
        else:
            y = x

    if success:
        break
    else:
        continue

if success:
    p = libnum.gcd(y - 1, n)
    q = n / p
    print 'P: ' + '%s' % p
    print 'Q: ' + '%s' % q
else:
    print 'Cannot compute P and Q'
```