实验四 RSA中公开的模数N

实验目的

- 了解公钥加密方案的一般结构
- 深入理解RSA加密原语的密钥生成
- 编程实现对没有正确生成密钥的RSA的破解，提醒大家不要尝试自己随意实现加密原语

编程语言

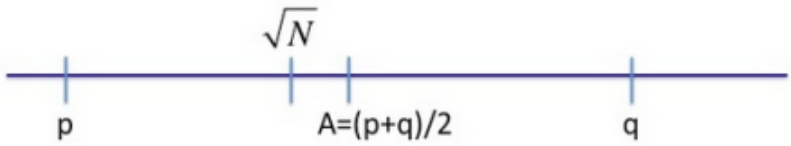
- Python（推荐）或者 C/C++或者Java

实验内容

- 本次实验是在公开的模数N没有被正确生成时破解RSA。这个实验是在提醒大家，千万不要自己轻易去实现一个加密原语。
- 通常，构成RSA模数N的素数p和q应该被独立地产生的。但是，假设一个开发者决定通过选择一个随机数R，并搜索其附近的两个素数作为p和q。那么，我们来证明这种方法得到的RSA的模数$N = pq$能被轻易的分解。（而RSA的安全基础就是假定模数不能被轻易分解！）假
- 设给定一个合数N并知道N是两个彼此很接近的素数p和q的乘积，即p和q满足：

$$|p - q| < 2N^{\frac{1}{4}} \qquad (*)$$

  你的任务是分解N。
- 令A是两个素数的算术平均值，即$A = \frac{p+q}{2}$。由于p和q都是奇数，所以A一定是一个整数。
- 为了分解N，首先需要观察，在条件(*)下$\sqrt{N}$是非常接近A的。具体来讲，有：$A - \sqrt{N} < 1$。由于A是一个整数，将$\sqrt{N}$凑成最接近的整数便能获取A的值。在代码中，形式大概是$A = ceil(sqrt(N))$，其中$ceil$是上取整函数。
- 更直观地，数字p、q、$\sqrt{N}$和A有如下关系：



- 由于A是p和q的中点，所以存在一个x使得$p = A - x$以及$q = A + x$。
- 又因为$N = pq = (A - x)(A + x) = A^2 - x^2$，因此$x = \sqrt{A^2 - N}$。
- 现在，根据x和A，你可以找到N的p和q，即分解出了N！
- 在接下来的任务中，需要使用上述的方法来分解给定的模数。本实验需要使用一个支持多精度算数平方根运算的环境。在Python中，可以使用$gmpy2$[1]模块；在C++中，可以使用GMP[2]。

任务一

- 模数N是两个素数p和q的乘积，满足$|p - q| < 2N^{\frac{1}{4}}$。（模数$N$请见附件task.txt）

任务二

- 模数N是两个素数p和q的乘积，满足$|p - q| < 2^{11}N^{\frac{1}{4}}$。（模数N请见附件task.txt）
- 提示：在$A - \sqrt{N} < 2^{20}$的情况下，尝试从$\sqrt{N}$向上搜索A，直到成功分解N。

实验时间与地点

- 4月23日晚6:30
- 思贤楼303机房

实验要求

- 在线提交源码和实验报告
- 实验报告需要包括实验结果（p,q的值以及一些中间值）、重要代码段解释以及本次实验总结
- 实验代码禁止抄袭，可以在网上进行参考，但是如果没有任何改变，将判定抄袭，本次实验记0分
- 鼓励大家思考新的求解方法和代码，对于能够用新思路求解出较好结果的同学本次实验加分的奖励
- 实验报告截止时间为4月30日
- 本次实验设立3个java小助教，2个C++小助教，4个python小助教，欢迎大家提前来找我们检查