实验四 RSA中公开的模数N

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：

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

你的任务是分解N。

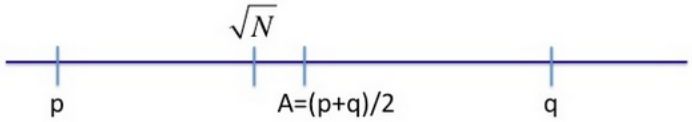
令A是两个素数的算术平均值，即$A = \frac{p+q}{2}$。由于p和q都是奇数，所以A一定是一个整数。

为了分解N，你首先需要观察，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$A = \text{ceil}(\text{sqrt}(N))$，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/C++中，可以使用**GMP**[2]。

任务#1

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

任务#2（选做）

模数N是两个素数p和q的乘积，满足$|p - q| < 2^{11}N^{1/4}$。（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，直到成功地分解N。

实验要求

- 请在线提交源码和实验报告。
- 实验报告需包括实验结果（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。

[1]http://readthedocs.org/docs/gmpy2/en/latest/mpz.html#mpz-methods
[2]http://gmplib.org/

补充说明

为了保持完整性，我们看一下为什么有$A-\sqrt{N}<1$。

首先，可以看到：
$A^2-N=(\frac{p+q}{2})^2-N=\frac{p^2+2N+q^2}{4}-N=\frac{p^2-2N+q^2}{4}=(p-q)^2/4$。

现在，由于对于所有的x,y：$(x-y)(x+y)=x^2-y^2$，我们可以得到：
$A-\sqrt{N}=(A-\sqrt{N})\frac{A+\sqrt{N}}{A+\sqrt{N}}=\frac{A^2-N}{A+\sqrt{N}}=\frac{(p-q)^2/4}{A+\sqrt{N}}$。

由于$\sqrt{N}\leq A$，那么$A-\sqrt{N}\leq\frac{(p-q)^2/4}{2\sqrt{N}}=\frac{(p-q)^2}{8\sqrt{N}}$。

由假设(*)可知$(p-q)^2<4\sqrt{N}$，因此$A-\sqrt{N}\leq\frac{4\sqrt{N}}{8\sqrt{N}}=1/2$。