密码学第三次实验报告

**一. 实验目的**：

请提交一个满足RSA算法要求的模数N，要求：使用一台PC机运行Sage（或相类似软件）对N进行分解，所需时间大概为60分钟。误差可在正负5分钟之内。实验报告中请描述选择N的依据，生成的方法，及相关的检测数据。

**二. 实验过程**：

1. 模数N的生成，要生成模数N，先随机找出两个大素数p与q，通过米勒拉宾算法检验p与q是否为素数，并根据算法生成特定位数的随机数，从而得到特定位数的大素数。
2. 米勒拉宾算法的实现是使用python程序设计语言实现，原因是python该语言的大数据系统可以方便地解决大数之间的运算。
3. 把生成的N放到Sage上运行，直接在Sage的官网进行在线测试，使用factor函数对N进行分解并计算时间，结果分析p和q为134bits左右时，即N大概为268bits的时候需要耗费一个小时进行对N的分解。

三. **米勒拉宾算法代码**：

import random

# (a \* b) % c

def mod\_pro(a, b, n):

return a\*b % n

# a ^ b % c

def mod(a, b, c):

if b == 1:

return a % c

a %= c

ret = 1

while b:

if b&1:

ret = mod\_pro(ret,a,c)

a = mod\_pro(a,a,c)

b >>= 1

return ret

def check (a, n, m, k):

ret = mod(a, m, n)

last = ret

for i in range(0, k):

ret = mod\_pro(ret, ret, n);

if (ret == 1 and last != 1 and last != n-1): return True

last = ret

if (ret != 1): return True

return False

def is\_prime(n, t):

if n < 2:

return False

if n == 2:

return True

k = 0

m = n - 1

while m&1 == 0:

m >>=1

k += 1

for i in range(0, t):

a = random.randint(0, n)

if(check(a,n,m,k)):

return False

return True

l = input()

while 1:

p = random.randint(2\*\*l, 2\*\*(l + 1))

if is\_prime(p,20):

print p

break

while 1:

q = random.randint(2\*\*l, 2\*\*(l + 1))

if is\_prime(q,20):

print q

break

print p \* q