*RSA 2048 Encryption N=P*Q*
In Python or PyCharm

By Ricardo.gil@sbcglobal.net

02/24/2017

Abstract

The purpose of this paper is to provide algorithm that is 5 lines of code and that finds P & Q when N is given. It will work for RSA-2048 if the computer can float large numbers in PyCharm or Python. Also, the P&Q from Part I of the algorithm becomes the range for a for loop in Part II that returns and solves P*Q=N (True).

I. Given N below (N=617 digits). To find P and Q. First solve for P by multiplying A*A and then dividing it by A+A. Then continue by dividing by the square root of A that is divided by 2 and 1E308 is added to it. To solve for Q divide A by P. The algorithm will solve for P & Q and print (n,P,Q).

RSA-2048

```
import math

N=2519590847565789349402718324004839857142928212620403202777713783604366202070759555626401852588078440691829064124951508218929855914917618450
2808489120072844992687392807287776735971418347270261896375014971824691165077613379859095700097330459748808428401797429100642458691817195118746
1215151726546322822168699875491824224336372590851418654620435767984233871847744479207399342365848238242811981638150106748104516603773060562016
1967625613384414360383390441495263443219011465754445417842402092461651572335077870774981712577246796292638635637328991215483143816789988504044
5364023527381951378636564391212010397122822120720357
P=((N*N)/((N+N))/((math.sqrt(N)/2)+100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000) #617 digits/2=308))
Q=(N/P)
n=P*Q
print(n,P,Q)
```

**n=25195908475657890000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000**

**P=70235984124766200000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000**

**Q=35873219105039150000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000**

II. After P & Q are returned from Part I or the algorithm, it becomes the range in part II. In part II the range is multiplied in a for loop until P*Q= N (True).

```
for x in range(7023598412476620000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000 000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000,73747783310045100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000):

    for y in range(358732191050391500000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000,3946054101554306500000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000):
     print('%d * %d = %d' % (x, y, x * y))

  print((x * y)==25195908475657893494027183240048398571429282126204032027777137836043662020707595556264018525880784406918290641249515082189298559
149176184502808489120072844992687392807287776735971418347270261896375014971824691165077613379859095700097330459748808428401797429100642458691817195
1187461215151726546322822168699875491824224336372590851418654620435767984233871847744479207399342365848238242811981638150106748104516603773060562016196762561338441436038339044149526344321901146575444541784240209246165157233507787077498171257724679629263863563732899121548314381678998850404453640235273819513786365643912120103971228221207203
57)
break
```

III. The algorithm will return P*Q= N (True). Below is an estimation of the return.

7023598412476620000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000000000000000*35873219105039150000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000=25195908475657890000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000

(True)

IV. **In the actual algorithm when N=*25195908475657893494027183240048398571429282126204032027 77713783604 3662020707595556264 018525880784406 91829064124951508218929855 91 4917618450280848912 0 07284499268739280728777673597141834727026189637501497 18246911650776133798590957 00097330459748808428401797429 1 006424586 9181719511874612151517265463228221686998754918242243363725908514186 54620 43576798423387184 77444792073993423658482382428 11 9816381501 0674810451660377306056201619676256133844143603833904 4 1495263443219011465754445417842402 0924616 5157233507787077498171257724679629263863563732899121548314381678998850404453640235273819513786365643912120103971228221207 20357*, P&Q will be large primes.**

V. **Conclusion, If your computer can process P in Part I, one will get N=P*Q for RSA-2048 in Part II. Part II is a for loop with the range of P & Q which is an estimate from Part I. This algorithm is 5 lines long in Part I and can find P & Q when N is given in Part II and the range P&Q is taken from Part I and used in the for loop in Part II. It returns P*Q=N (True) in Part III, when the parameters are satisfied in the algorithm.**

VI. **References**

1). Gil, R. (2016).Cicada Rsa NPQ. viXra [v-1], 1-2).