RSA Algorithm

Ken Wais 10/6/11

The RSA algorithm is a numerical method in cryptology to encrypt private keys for PKI digital signing. As such it utilizes some of the principles of algebraic sets and their relations. I will try to explain in plain terms how one key is created.

The Modulus

First we must understand the modulus to grasp RSA. The modulus is a way to relate number sets to each other using a base number to map one set to another. An example should illustrate the idea.

The simplest example would be 1 = 3 mod 2. What this means is if we only have two numbers, we count with 0 and 1 to represent larger numbers we must put them in terms of 0 or 1. So, 3 is just going 1 beyond 2. What would 4 be? 4 would be just 2 twice so, it would be 0 again. How 'bout 5? 5 would be 2 twice and then 1 more so it would be 1. How bout 6? 6 would be 2 three times so it would be 0. Now let's put these examples together and show them below:

$1 = 3 \bmod 2$

$0 = 4 \bmod 2$

$1 = 5 \bmod 2$

$0 = 6 \bmod 2$

$1 = 7 \bmod 2$

$0 = 8 \bmod 2$

$1 = 9 \bmod 2$

Notice on the left side every time we increase the base 10 number it just keeps going between 0 and 1. That's because the only numbers we can use in base 2 are 0 and 1. If the base is 10 we can use 0 to 9 different numbers (0,1,2,3,4,5,6,7,8,9), and beyond that we could map small numbers to larger ones using the modulus again. As an example of a base 10 modulus we can make the base 12, and then every number we count will have to be a multiple of 12. This is what the 12 hour clock does. Again an example will illustrate this.

Midnight is $0 = 12 \bmod 12$

1 o'clock is $1 = 13 \bmod 12$

2 o'clock is $2 = 14 \bmod 12$

3 o'clock is $3 = 15 \bmod 12$

Here is a strange one. 1 =1/4 mod 5.

This equation is not integer division. It means if we divide 5/4 the remainder is 1. Mods are always integers. So one fourth of 5 is 1. You drop anything after the division. This is important because RSA uses a fraction in its algorithm. If this isn't clear now, when I give an actual numerical example below it should be.

The RSA Algorithm

First you choose two prime number p and q.

Then compute n=pq

Next form the factored function

F(x) = (p-1)(q-1). So far it's quite simple.

Next things get a little more complex. Now we have to choose an exponent *e* such that the following is true:

$1 < e <$ gcd (greatest common denominator)(e, p-1) =1

and

$1 < e <$ gcd (greatest common denominator)(e, q-1) =1

Actually the inequality is $1 < e <$ gcd F(x) =1, but this means it has to be true for each factored term of the F(x).

This means that e, (p-1), and (q-1) can have only one common denominator and that is 1. It is not important to know why p and q must have only common denominator to use RSA. It involves aspects of algebraic sets I won't discuss here. Just know they have to have this property. What this means is the two numbers p and q are co-prime. That is the have no factor between them, except 1.

There is a way to determine if the two numbers have only 1 as common denominator. It's called *Euclid's algorithm*. You take the two primes and divide them repeatedly until you get down to 1 as the remainder. If you don't get 1 as a remainder in this process then the two primes don't have 1 as their gcd. Here is an example with a table that shows how it works. On the top we have the dividend (the number to be divided), the divisor, the quotient and the remainder. We divide the two numbers across and move the divisor to the second row and repeat the process until we get 1 in a row. If we get 1, then the two numbers have only 1 as their GCD. We will use 140 and 21

Dividend	Divisor	Quotient	Remainder
140	21	6	14

21	6	3	3
6	3	2	0
3	2	1	1

In the first row we have 140/21 =6 (w/o the remainder). 6*21 =126-140 = 14 (remainder)

Moving the 21 to the 2nd row we have 21/6 =3 (w/o the remainder) 3*6 =18-21=3 (remainder)

Moving 6 to the 2nd row we have 6/3 = 2 and 0 remainder.

We repeat again and finally we get 1 as a remainder. This means 140, 21 have one gcd = 1

Now if we apply this procedure to two prime numbers we will quickly see they have only one gcd and it's 1. So, here is an example I found on the Net, (I was going to manually go thru several primes, so I looked up a numerical example of RSA. They are p=137 and q=131. Let's do it's Euclid table

137/131 = 1.0458. Drop the non-integer remainder and the only GCD they share is 1, so they are good primes to use. Now we are ready to compute the private key and the public key using RSA.

So, we go back to the above equations

P=137, Q=131

N = PQ = 137*131 =17947

F(x) =(137-1)(131-1) = (136)(130) =17680

Now we select prime *e* = 3. If we check GCD(3,136) = 1 it is. Look at the above equation to see why I plugged in that number

And GCD(3, 130) = 1 it is.

Now the hardest part comes Here, we form the modulus called d. It is

D =mod F(x)/e = mod 17680/3 = 11787. Now I must explain how this comes out like that. Alternately this is written D =3^{-1} mod F(x) = mod 17680/3 = 11787

The integer division 17680/3 = 5893. But remember this is not simple integer division. If we take 5893*3= 17679. 17680-17679 = 1.

Remember that is what (p-1) and (q-1) must equal, namely 1. So the modulus is saying take 5893, times and minus 17679 and it equals 1.

So, now we have everything to form our public key which is (n,e) and private key is (n,d).

The public key is the pair (17947, 3) and the private key is (17947, 11787). That would be the digital signature of the first sender. Notice 11787 is prime and hard to guess, that is why it's the private key. Of course in a real RSA algorithm, the prime numbers p and q would be about 125 digits long. But notice the public key is PQ and could be a non-prime number (it isn't in this case). But knowing that will never let you know the D, because D is the mod of two prime numbers, each minus 1 and only share a gcd of 1. That gets near impossible guess by brute force (trying combinations) the larger the prime numbers p and q and the exponent *e* becomes. To get the receiver's public and private key pair you reverse the process, but I won't go into that.