RSA public key encryption

Tahseen Rabbani

The RSA public key cryptosystem is a method of encryption which was made public in 1977. Named after its inventors, Ron Rivest, Adi Shamir, and Leonard Adleman, the method relies on the fact that large integers, on the order of hundreds of digits, require an extraordinary amount of time for a computer to factor. This paper explores the general procedure for RSA encryption, as well as a look into tests of primality.

1 Key creation and encryption

Suppose Hilbert wants to send Tahseen a message he wishes to be kept secret. There is a third party, Sasha, who wishes to intercept the message. Tahseen agrees to ensuring Hilbert's message be kept secure. The procedure is as follows:

1. Tahseen chooses two prime numbers p and q. For our purposes, the prime numbers will be kept small. Using the RandomPrime function in Mathematica, Tahseen generates $p = 1049$ and $q = 571$. Tahseen takes the product

$$pq = 1049 \cdot 571 = 598979 = N. \tag{1}$$

2. Tahseen computes g, such that

$$g = (p-1)(q-1) = (1048)(570) = 597360 \tag{2}$$

3. Tahseen chooses e, a positive integer relatively prime to g; that is, the greatest common divisor between g and e is 1. Let $e = 7$, then

$$GCD(e, g) = GCD(7, 597360) = 1 \tag{3}$$

4. Tahseen has now created his public keys, e and N. Tahseen sends Hilbert the public keys.

5. Hilbert encodes his plaintext message, using, for example, ASCII. Let the result of this conversion be the positive integer m=123456. m must satisfy $1 \leq m < N$.

6. Hilbert uses the public keys (N,e) to find k, such that

$$k \equiv m^e \pmod{N} \tag{4}$$

$$k \equiv 5195 \equiv 123456^7 \pmod{598979} \tag{5}$$

Calculated using Mod function in Mathematica.

7. Hilbert sends Tahseen the cyphertext, k= 5195.

8. Tahseen, the creator of the public keys, can easily decrypt the message. Tahseen must recover m knowing that Hilbert encrypted m as such:

$$k \equiv m^e \pmod{\text{N}} \tag{6}$$

By the following Theorem, this is not so difficult, as long as p and q are known [2] :
Let p and q be distinct primes, satisfying

$$GCD(e, (p-1)(q-1)) = GCD(e, g) = 1 \tag{7}$$

$$\textit{Then } e \textit{ has the inverse modulo } (p-1)(q-1) = g,\ de \equiv 1 \pmod{g} \tag{8}$$

$$\textit{Then } m^e \equiv k \pmod{N} \text{ has the solution } m \equiv k^d \pmod{N} \tag{9}$$

So, Tahseen begins by solving for d using Wolfram Alpha,

$$de \equiv 1 \pmod{g},\ 7 \cdot d \equiv 1 \pmod{597360} \Rightarrow d = 512023$$

Tahseen can now find m via the above theorem. Using the Mod function in Mathematica:

$$m \equiv k^d \pmod{N} \Leftrightarrow m \equiv 5195^{512023} \pmod{598979} \equiv 123456$$

Tahseen has decrypted Hilbert's message!

9. Note that no explicit information about decryption or encryption is sent between Hilbert and Tahseen [1]. Tahseen could release the public keys to Sasha, who wishes to intercept the message, but Sasha would not have much to work with unless he could factor N, which hypothetically should be hundreds of digits of long. What minimal information would Sasha need in order to figure out p and q? Sasha needs to solve for the congruency $m^e \equiv k$ mod N, and for that, he would need to know $(p-1)(q-1)$. However, we observe

$$(p-1)(q-1) = pq - p - q + 1 = N - (p+q) + 1 \tag{10}$$

So, if Sasha can figure out $p+q$, he can figure out $(p-1)(q-1)$, however, as Hoffstein, Pipher, and Silverman note [2], having the value of $(p+q)$ is as good as having the factors of N themselves, via solving the quadratic polynomial,

$$X^2 - (p+q)X + pq = (X-p)(X-q) = 0 \tag{11}$$

With p and q in hand, Sasha may readily solve for the decryption element, d.

2 Primality Tests

The first step in RSA encryption is choosing p and q, two large primes. It becomes evident that one would need a test to verify that both numbers are probability prime.

Fermat's Primality Test [3]

Fermat's Little Theorem states: *For a and p, relatively prime integers,* $1 \leq a < p$*, p is prime if*

$$a^{p-1} \equiv 1 \text{ (mod p)} \tag{12}$$

Any integer a for which $a^{p-1} \not\equiv 1$ is called a Fermat witness. The test is meant to tried with many values of a, and if many values do yield the congruency as desired, we can be confident, although never certain that p is probably prime. Any a which satisfies the congruency for a composite number is referred to as a Fermat Liar. Example:

-

Let us test whether 7 is a prime using Fermat's test via Mathematica's Mod function.

$2^6 \equiv 1 \pmod 7$

$3^6 \equiv 1 \pmod 7$

$4^6 \equiv 1 \pmod 7$

For a number that we weren't certain of its primality in the slightest, 3 iterations of test would not be enough. Let us try an interesting case, p=561:

$2^{560} \equiv 1 \pmod{561}$

$4^{560} \equiv 1 \pmod{561}$

$7^{560} \equiv 1 \pmod{561}$

All our choices of a above are Fermat Liars. $561 = 3 \cdot 11 \cdot 17$, so 561 is clearly not prime. In fact, all such values a relatively prime to 561 will be Fermat Liars [2]. This type of number is known as a Carmichael number. They're quite rare [4], although the fact that it has been proven that there is an infinite number of Carmichael numbers[2] is nevertheless unsettling, making Fermat's test not an ideal choice.

Miller-Rabin Test [2]

The disadvantage of Fermat's test was that there were composite numbers with no witnesses. In the Miller-Rabin test, any composite number will have a large number of witnesses, which increases our confidence for the probability of p being prime if there is a large lack of witnesses. In fact, for any given odd composite number n, the number of witnesses will be 75

Choose p and write $p - 1 = 2^k q$ such that q is odd. We call a, an integer relatively prime to p, a witness for p if these conditions are true:

i. $a^q \not\equiv 1 \pmod p$

ii. $a^{2^i q} \not\equiv -1$ For all $i = 0, 1, 2, ..., k - 1$

The Miller-Rabin test avoids the possibility of Carmichael numbers, making it an optimal choice to Fermat's primality test. Example test:
Let p=37. Then $37-1=36=2^2\cdot 9$, and choose a=7
i) $7^9\equiv 1 \pmod{37}$ So the first condition already fails suggesting that 37 is prime. This could be a Miller-Rabin liar, though.
Choose a=5
i) $5^9\not\equiv 1 \pmod{37}$ The first condition is true.
ii) $5^{2^1\cdot 9}\equiv 5^{18}\equiv -1 \pmod{37}$ So our second condition fails. We will adjourn the test here, but as with Fermat's test, many iterations would be needed so as to rule out any possible liars.

Solovay-Strassen Test [5]

The Solovay-Strassen algorithm is less accurate than the Miller-Rabin but like the Miller-Rabin, it is more accurate than Fermat's test because there are no Carmichael numbers. The procedure is as follows:
If $a^{(p-1)/2}\equiv \left(\frac{a}{p}\right) \pmod{p}$ for odd integer p and for all a such that $1<a<p$, then p is prime. $\left(\frac{a}{p}\right)$ is the Legendre symbol, that is

$$\left(\frac{a}{p}\right)=\begin{cases} 1 & \text{if } a \text{ is a quadratic residue } (\text{mod } p) \\ -1 & \text{if } a \text{ is not a quadratic residue } (\text{mod } p) \\ 0 & a\equiv 0 \ (\text{mod } p) \end{cases}$$

a is a quadratic residue if $a \pmod{p}$ is congruent to a perfect square. Any a which fails the conditions listed above is referred to as an Euler witness for the primality of p. Any a which passes the conditions for a composite number is referred to as an Euler liar.
Example: Let p=53, a=3
$3^{52/2}=3^{26}\equiv -1 \pmod{53}$
Using the Mathematica function, $JacobiSymbol[3,53]=-1$
Hence, $3^{26}\equiv \left(\frac{3}{53}\right) \pmod{p}$, but it could be a liar. We try another. Let a=5
$5^{52/2}=5^{26}\equiv -1 \pmod{53}$
$JacobiSymbol[5,53]=-1$
Hence, $5^{26}\equiv \left(\frac{5}{53}\right) \pmod{p}$. We would need to try many more to be sure, but note that test works how it's expected to a prime number (which 53 is).

Probabilistic versus deterministic [2]

The Solovay-Strassen, Miller-Rabin and Fermat tests are probabilistic tests of primality, that is, by repeated iterations of the test with a lack of witnesses, we can assert that the given integer n in question is "probably" prime. A deterministic test would ascertain whether n is prime with certainty. The most obvious deterministic method would be trying to divide n by every natural number up to $\sqrt{n}$, however this takes far too long.

Such an algorithm runs in "exponential-time," while several probabilistic tests such as the Miller-Rabin run in "polynomial-time" which is much faster. What would be desired, then, would be a deterministic primality test that would be able to run in polynomial time. Such a test arose with the advent of the AKS Primality Test, which will not be demonstrated in this paper. The AKS test runs slower than the Miller-Rabin, though, so it still leaves much to be desired in terms of efficiency.

3 Factorization

A third party who wishes to intercept a message being communicated via RSA encryption would be at a loss, if all they had to work with was N. This section explores various methods for factorization of numbers while reiterating why such methods are simply inefficient for numbers of large magnitude.

$p-1$ factorization .

The p-1 algorithm is used to find the prime factors of a composite integer n.

1. Choose a such that $1 < a < n$
2. Compute $a = a^{j!} \pmod{n}$ for positive integers up to a bound j.
3. Compute d=gcd($a-1$, n)
4. If $1 < d < n$, then this is a prime factor, p. The other factor, q can simply be found by n/p.

This method is not as simple as it would appear. Namely, it runs horribly slow for j too large, so when exactly would we find a result for j reasonably small? If $p-1$ or $q-1$ is a product of small primes upper bounded by j, also called j-smooth, then our algorithm will yield a positive result once such a j is tested. If one of the factors of $p-1$ or $q-1$ is quite large, then the algorithm is more or less, useless because of the run-time needed.
An example of $p-1$ factorization:
Test for $n = 13927189$ [2]

$$2^{12!} - 1 \equiv 6680550 \bmod 13927189,\ gcd(2^{12!} - 1, 13927189) = 1$$
$$2^{13!} - 1 \equiv 616077 \bmod 13927189,\ gcd(2^{13!} - 1, 13927189) = 1$$
$$2^{14!} - 1 \equiv 879290 \bmod 13927189,\ gcd(2^{14!} - 1, 13927189) = 3823$$

Notice that we reached a success at j=14. Upon inspection of $p-1 = 3822$ we find it factors into $2 \cdot 3 \cdot 7^2 \cdot 13 = 3822$, a series of small prime factors bounded by 14. As for the other factor $q = n/p = 13927189/3823 = 3643$, we find that $q-1$ factors into

$2 \cdot 3 \cdot 607 = 3642$. We would have needed to plug in $j \geq 607$ to reach a positive with the algorithm, which would be entirely unfeasible time-wise.

As an application to RSA encryption, this provides a new consideration when choosing p and q: if $p-1$ and $q-1$ factor into a series of small primes, then a third party can easily use the $p-1$ algorithm to find p and q.

Difference of squares [2]

This method of factorization is based on identity

$$X^2 - Y^2 = (X+Y)(X-Y) \tag{13}$$

We choose an n we are interested in factoring and proceed as follows,
Choose some b such that
$n + b^2 = a^2 \Leftrightarrow n = a^2 - b^2 = (a+b)(a-b)$

Here is an example where we factor 1073 into its distinct prime factors:

$$\begin{aligned} 1073 + 1^2 &= 1074 \text{ Perfect square? No} \\ 1073 + 2^2 &= 1077 \text{ Perfect square? No} \\ 1073 + 3^2 &= 1082 \text{ Perfect square? No} \\ 1073 + 4^2 &= 1089 = 33^2 \text{ Perfect square? Yes} \\ 1073 &= 33^2 - 4^2 = (33+4)(33-4) = 29 \cdot 33 \\ &\text{So our prime factors are 29 and 33.} \end{aligned} \tag{14}$$

What should one do if many values of b yield negative results? The next step would be observing that for some factor r, we write $rn = a^2 - b^2 = (a+b)(a-b)$, there is a chance that n could have nontrivial factors in common with a and b, if the multiple of n does.
Example:
Suppose we wish to factor 203299, but with r=1, we have no results after trying many values of b. We can try scaling n by 3 and try the difference of squares method again.

$$\begin{aligned} 3 \cdot 203299 + 6^2 &= 609933 \text{ Perfect square? No} \\ 3 \cdot 203299 + 7^2 &= 609946 \text{ Perfect square? No} \\ 3 \cdot 203299 + 8^2 &= 609961 = 781^2 \text{ Perfect square? Yes} \end{aligned}$$

So, we have $3 \cdot 203299 = 781^2 - 8^2 = (781+8)(781-8) = 789 \cdot 773$,
$gcd(203299, 789) = \mathbf{263}$, $gcd(203299, 773) = \mathbf{773}$, both of which are prime, so we have found our distinct prime factors.

Searching for a and b aimlessly is a fruitless endeavor dependent on luck, so we should refine what a and b should look like in terms of modular arithmetic. Using the scaling modification for difference of squares, we would have that for all multiples of n,

$$\textit{If } a^2 - b^2 = rn \textit{ then } a^2 - b^2 = 0 \text{ mod n} \Rightarrow a^2 \equiv b^2 \text{ (mod n)} \tag{15}$$

Modern factorization centers around the last congruency in (17), in the form of this process,

1. Find integers a_1,a_2,a_3,...,a_r with the property $c_i \equiv a_i^2 \pmod{n}$ is j-smooth for j small, that is, c_i is a product of small primes bounded by j.
2. Take a subset of the c_i's and multiply them together such that $c_{i1} \cdot c_{i2} \cdots c_{is} \equiv b^2 \pmod{n}$, a perfect square.
3. Let $a = a_{i1} \cdot a_{i2} \cdots a_{is}$, then we have $a^2 = (a_{i1} \cdot a_{i2} \cdots a_{is})^2 \equiv a_{i1}^2 \cdot a_{i2}^2 \cdots a_{is}^2 \equiv c_{i1} \cdot c_{i2} \cdots c_{is} \equiv b^2 \pmod{n}$ So, if we take $d = gcd(n, a - b)$ for a and b we have created, according to the scaled difference of squares method, we could possibly find a nontrivial factor this way.

Closing Remarks

The strength of RSA public key encryption lies in the fact that we have yet to come up with an efficient method for factoring large integers. The P versus NP problem, is an important, unsolved question within the field of computer science that informally asks: if we can verify a solution in polynomial time, can we solve it in polynomial time? This is very pertinent to RSA encryption. Verifying that a given sequence of numbers are indeed the factors of a large number is not a difficult task at all, but we have no fast algorithms which can factor larger numbers in a reasonable amount of time. If P = NP, then the essence of the security behind RSA, and many other cryptosystems would collapse.

References

[1] D. Curtis, *RSA Encrpytion* available at http://www.cwu.edu/ curtiswd/Math330-Spr12/Mathematica

[2] J. Hoffstein, J. Pipher, and J. Silverman, *An Introduction to Mathematical Cryptography*, Springer. New York, NY, 2008.

[3] K. Kedlaya, *Is this number prime?*, available at http://mathcircle.berkeley.edu/BMC5/docpspdf/is-prime.pdf

[4] Z. McGregor-Dorsey, *Methods of Primality Testing*, available at http://www-math.mit.edu/phase2/UJM/vol1/DORSEY-F.PDF

[5] B Rosenberg, *The Solovay-Strassen Primality Test*, Amer. Math: Cryptography. **609/597** (2000) available at http://www.cs.miami.edu/ burt/learning/Csc609.011/jacobi.pdf