An Introduction to RSA Public-Key Cryptography

David Boyhan

August 5, 2008

According to the U.S. Census Bureau, in the 1st quarter of 2008, approximately $33 billion worth of retail sales were conducted on the Internet.[1] In all likelihood, the vast majority of these were secured at some level by an encryption scheme known as public-key cryptography (“PKC”). Interestingly, relative to the Internet, PKC is quite old,¹ at the same time, relative to virtually all other types of “secret writing” or cryptography, PKC is extremely young.² It’s doubtful that without PKC either electronic commerce or the world wide web would have expanded as rapidly and as they have.

Before PKC

For the nearly 2000 years prior to PKC, all forms of ciphers were “symmetric.” That is, in order to read a secret encrypted message, a confidential key is needed. For example, in the case of the Caesar Shift Cipher, used by Julius Caesar, a plain-text message \( M \) is encrypted by shifting each letter in the message by three letters (e.g., all “A”s become “D”s, all “B”s become “E”s, etc.) [2] (page 10). As a result, the plain-text message \( M \):

\[
DM\ HURTS\ MY\ BRAIN
\]

is encrypted into the cipher-text message \( C \):

\[
GP\ KXUWV\ OB\ EUDLQ
\]

In this case, the number 3 is the key to the cipher, that must both be kept confidential from eavesdroppers, but must also be shared between both the sender and the recipient. A more complex cipher might have random letter substitutions, or use more than one alphabet. The German Enigma used during World War II relied on multiple mechanically rotated rotors to encrypt messages. The order and initial settings of the rotors represented the confidential keys that had to be distributed securely to all users of the Enigma. Every one of these ciphers, including the “Advanced Encryption Standard” (“AES”) approved by the U.S. National Institute of Standards and Technology in 2001, is therefore considered a “symmetric-key” cipher, in that the key to both “lock” and “unlock” the confidential message is identical and needs to protected from eavesdropping.³

Symmetric key encryption can also be thought of in terms of two functions, one for encrypting - \( E \), and one for decrypting - \( D \). In both cases, a single key \( k \) is used to both encrypt plain-text message \( M \), and to decrypt cipher-text message \( C \).

¹PKC is approximately 30 years old, while general public access to the Internet is only about 20 years old, and the World-Wide-Web is only about 15 years old.
²Most authorities date cryptography and secret writing to at least 400 B.C. [2][page 9].
³Technically, certain forms of secret writing, including invisible ink, stenography (“hidden writing” such as micro-dots) and one-time pads do not require “keys”. However, they all rely on some form of secure distribution that is vulnerable to interception.
Accordingly:

\[ E(M, k) \rightarrow C \]
\[ D(C, k) \rightarrow M \]

As a result of the requirement that symmetric keys be distributed securely, by the 1960’s and 1970’s the process of key distribution had become a serious problem for both businesses and governments. Major resources were committed to secure key distribution, in some cases requiring daily couriers between business and government agencies.[2](pages 251, 252).

**Diffie-Hellman Invet PKC**

However, in 1976 two mathematicians, Whitfield Diffie and Martin Hellman published a paper proposing a revolutionary new cryptographic approach. In their paper[3] Diffie and Hellman proposed a scheme where keys could be exchanged between senders without concern over interception. There would now be two keys, a “public-key” \( e \) that could be published to the world, and a corresponding “private-key” \( d \) that would be kept confidential. The public-key \( e \) would be used to encrypt message \( M \) into cipher-text \( C \) and the private-key \( d \) would be used to decrypt \( C \) back into plain-text \( M \). Even though the actual encryption algorithms could be published and available to eavesdroppers, it would be computationally infeasible to derive \( d \) from \( e \). In simple terms, the public-key \( e \) “locks” and the private key \( d \) “unlocks”. The public-key is published to the world and anyone who wishes to communicate securely with the key-owner uses the public-key. Accordingly:

\[ E(M, e) \rightarrow C \]
\[ D(C, d) \rightarrow M \]
\[ D(C, e) \not\rightarrow M \]

This concept would be the core of public-key cryptography and would eliminate any need for key distribution. Keys would now be “asymmetric” rather than symmetric.

In order to implement PKC, Diffie and Hellman proposed using “one-way” or “trapdoor” functions. These functions are so called because while they can be performed quite easily in one “direction”, it is significantly more complex and time-consuming to “reverse” the function. A simple example would be multiplication versus factoring. While it’s a simple matter to multiply 13,719 by 23,547 to get the result 323,041,293, it’s significantly more complex to factor 323,041,293 into its prime factors of \( 3^2 \times 17 \times 47 \times 167 \times 269 \). The group of one-way functions that Hellman, and another researcher, Ralph Merkle, proposed were based on “knapsack” problems. In simplest terms, knapsack functions require “packing” a knapsack with objects of various weights in order to perfectly fit within the confines of the knapsack. In cryptography, the problem could involve determining a subset \( (x_1, x_2, x_3) \) from a larger set of integers \( (x_1, x_2, x_3, x_4, x_5, x_6) \) such that the subset adds up exactly to a certain agreed upon value. In this case, the public-key would be the ordered set of weights and the encrypted message would be the product of the message text and the public-key. The private key would be the subset of the weights. Although calculating the sum is a very simple matter (“one-way”), as would be decrypting the message using the proper subset, determining what the private subset values are for a sufficiently large set of numbers \( x_n \) is computationally quite complex and is considered NP Complete. However, it was subsequently determined that although knapsack problems are NP Complete, there are special cases of the problems that are relatively easy to solve, rendering that particular trap-door cryptographically flawed.[5]

**RSA**

In 1978, Ronald Rivest, Adi Shamir and Leonard Adelman published a paper proposing an alternative PKC algorithm based upon prime number factorization and “multiplicative inverse” modular math.[4] Their approach survived significant cryptographic analysis and quickly became the de facto standard for PKC and has remained as such for the last 30 years.
In order to use RSA (named for the initials of the authors), a fictional protagonist, Alice, first needs to create her public and private keys. In RSA, a public-key actually consists of two numbers, \( n \) and \( e \) and the private key similarly consists of two numbers, \( n \) again and \( d \). The value \( n \) is generated by multiplying two very large prime numbers together. These two large prime numbers, \( p \) and \( q \) are kept secret.\(^4\)

Alice must then pick a large, random integer \( d \) that is relatively prime to \((p - 1) \cdot (q - 1)\). In other words, the greatest common divisor of \( d \) and \((p - 1) \cdot (q - 1)\) must be 1. Thus:

\[
gcd(d, (p - 1) \cdot (q - 1)) = 1
\]

Finally, Alice (or the software) must generate the public-key \( e \) from \( d \), \( p \) and \( q \). This step is the real heart of the RSA algorithm because the approach taken makes it computationally infeasible to determine \( d \) without \( p \) and \( q \). The integer \( e \) as the public-key encryption component is calculated as the multiplicative inverse of:

\[
d \mod ((p - 1) \cdot (q - 1))
\]

This is probably the most complex part of RSA. Traditionally, the multiplicative inverse of \( d \) would be \( d^{-1} \) such that \( d \cdot d^{-1} = 1 \). However, in this case, we are seeking the modular multiplicative inverse.\(^5\) A multiplicative inverse of \( d \mod ((p - 1) \cdot (q - 1)) \), is an integer \( e \) such that:

\[
d \cdot e = 1 \mod ((p - 1) \cdot (q - 1))
\]

In order for their to be a solution for \( e \), \( d \) and \((p - 1) \cdot (q - 1)\) must be relatively prime. To solve \( e \), we can use Euler’s Totient and Euler’s Generalization of Fermat’s Little Theorem. [4][6]

Euler’s Totient of \( n \), signified as \( \phi(n) \), is defined as the number of positive integers less than \( n \) that are relatively prime to \( n \). If \( n \) is prime then \( \phi(n) \) is simply \((n - 1) \). If both \( p \) and \( q \) and prime, then it’s easy to see that \( \phi(p \cdot q) \) will be \((p - 1) \cdot (q - 1)\). However, if \( n \) is not prime, such as, for example, 12, then the numbers that are less than \( n \) that are relatively prime to \( n \) would be 1, 5, 7, 11. Therefore \( \phi(12) \) is 4. Calculating Euler’s Totient can be accomplished in a number of ways, but the simplest method uses the unique prime factorization theorem in a manner similar to calculation of \( gcd \). If, in calculating \( \phi(n) \) we determine the prime factorization of \( n \) such that \( n = p_1^{k_1} \cdot p_2^{k_2} \) then, isolate the distinct primes and then compute \( \phi(n) \) using the following formula:

\[
n \prod_{p | n} \left(1 - \frac{1}{p}\right)
\]

In this formula, the range of \( p \mid n \) is only those primes identified as distinct in the prime factorization of \( n \). Therefore, for example, \( 12 = 2^2 \cdot 3^1 \) so the Totient would be \( 12 \cdot (1 - \frac{1}{2}) \cdot (1 - \frac{1}{3}) = 12 \cdot \frac{1}{2} \cdot \frac{2}{3} = 4 \). Similarly, because the prime factorization of 54 = \( 2^1 \cdot 3^3 \) then \( \phi(54) \) would be \( 54 \cdot \frac{1}{2} \cdot \frac{2}{3} = 18 \).

Euler’s generalization of Fermat’s Little Theorem states that: if \( gcd(d, z) = 1 \) then \( d^{\phi(z)} \mod z = 1 \). This is a condition we have already satisfied with \( d \) and \((p - 1) \cdot (q - 1)\).

Substituting \((p - 1) \cdot (q - 1)\) for \( z \) we can see, using Fermat’s Little Theorem that:

\[
1 = d^{\phi((p-1)\cdot(q-1))} \mod ((q-1)\cdot(p-1))
\]

We also know that based on the definition of multiplicative modular inverse that:

\(^4\)When we discuss “large” prime numbers, we are talking about prime numbers with hundreds, thousands or millions of digits.

\(^5\)It may clarify things somewhat to use Schneier’s [6] examples of modular multiplicative inverse that “\( a \cdot x \mod 7 \) . . . is equivalent to finding an \( x \) and \( k \) such that \( 4x = 7k + 1 \) where both \( x \) and \( k \) are integers. The general problem is finding an \( x \) such that \( 1 = (a \cdot x) \mod n \). This is also written as \( a^{-1} \equiv x \mod n \).”
Therefore:

\[ e \cdot d = 1 \pmod{(p-1) \cdot (q-1)} \]

Substituting in \( d^{\phi((p-1)(q-1))} \mod ((q-1)(p-1)) \) for \( 1 \pmod{(p-1)(q-1)} \), we get:

\[ e \cdot 1 = d^{\phi((p-1)(q-1)) - 1} \mod (p-1)(q-1) \]

The process to both encrypt and decrypt is now remarkably simple. To quote directly from the original RSA paper:

“... the result (the ciphertext \( C \)) is the remainder when \( M \) is divided by \( n \). To decrypt the ciphertext, raise it to another power \( d \) again modulo \( n \).”

As a result, the formulas for encrypting plain-text message \( M \) is \( E(M) \) and decrypting cipher-text message \( C \) is \( D(C) \) as follows:

\[ E(M) = M^e \pmod n = C \]
\[ D(C) = C^d \pmod n = M \]

We’ll now go through a simple example. We’ll start by creating Alice’s public-key. To create \( n \) we’ll use very small prime numbers: \( p = 5, q = 11 \). Therefore, the first part of the public encryption key

\[ n = p \cdot q = 5 \cdot 11 = 55 \]

We must now pick a random number that is relatively prime to \( (p-1)(q-1) \). We can easily see that 7 is relatively prime to 55, so we’ll use \( d = 7 \). Again, we would typically use much larger numbers.

We will also calculate \( e \) using \( d \) and Euler’s Generalization of Fermat’s Little Theorem. Accordingly:

\[ e = d^{\phi((p-1)(q-1)) - 1} \mod ((p-1)(q-1)) = 7^{\phi(40) - 1} \mod 40 \]

It’s easy to see that \( 40 = 2^3 \cdot 5^1 \)

Therefore, \( \phi(40) = 40 \cdot (1 - \frac{1}{2}) \cdot (1 - \frac{1}{5}) = 40 \cdot \frac{1}{2} \cdot \frac{4}{5} = 16 \)

Therefore, \( e = 7^{15} \mod 40 = 23 \)

Finally we’ll start with plain-text message \( M \):

\[ \text{DM HURTS MY BRAIN} \]

For the purposes of this example, we’ll use the integers 1-26 to represent the alphabet and remove all spaces. However, under normal circumstances, ASCII would be used instead. Therefore, \( M \) becomes:

\[ \text{DM HURTS MY BRAIN} \]
\[ 04 13 08 21 18 20 19 13 25 02 18 01 09 14 \]

\[ ^6\text{This was calculated using the OSS software Euler’s Math Toolbox available at:} \]
\[ \text{mathsrv.ku-eichstaett.de/MGF/homes/grothmann/euler/index.html} \]
Now, encryption is simply a matter of iteratively encrypting each block.\(^7\)

We can therefore see that the first encrypted block \(C_1 = 04^{23} \mod 5) = 09\)

The entire message is encrypted as:

\[09\ 52\ 17\ 21\ 02\ 25\ 39\ 52\ 05\ 08\ 02\ 01\ 14\ 49\]

And this would be decrypted using \(D(C) = C^d \mod n = M\).

So, \(M_1 = 09^7 \mod 5) = 04, M_2 = 52^7 \mod 5) = 13...M_{14} = 49^7 \mod 5) = 14\)

As a result of RSA, both encryption and decryption is accomplished quite rapidly. However, it’s relatively easy to see that in order to derive the secret key \(d\) from the public-key \(e\) and \(n\) an attacker would need to factor \(n\) into \(p\) and \(q\) order to determine \(\phi((p - 1) \cdot (q - 1))\). Since we would select \(p\) and \(q\) as very large prime numbers, the amount of time necessary to factor \(n\) becomes prohibitive and is currently believed to be NP Hard. It was estimated in the RSA paper that using the fastest known prime-factoring methods, if each step only required 1 microsecond, it would take an estimated \(4.2 \times 10^{25}\) years for a number \(n\) of 500 digits to be factored. [4](page 12).

### Digital Signatures & Certificates

In addition to its simplicity, the PKC scheme offers certain features that simplify electronic communications and electronic commerce. The two best examples of these are digital signatures and digital certificates. RSA has the ability to create unique “digital signatures” that are generated by a using a private key \(d\) but can be validated by using the public-key \(e\). As a result if Alice and Bob wish to execute a document without meeting in person, Bob can encrypt the document using Alice’s public-key \(e_a\). When Alice receives the document, she decrypts it and “signs” it using her digital signature generated from her private key \(d_a\). She then re-encrypts it using Bob’s public-key \(e_b\) and sends it to Bob. Bob decrypts the message using his private key \(d_b\) and views Alice’s digital signature. He can verify her digital signature using her public-key \(e_a\) verifying that it could only be signed by Alice. Similarly, “digital certificates” which are required for virtually all forms of electronic commerce are simply an enhanced form of the public-key \(e\). That is, the more advanced implementations of PKC allow users to generate “certificates” that contain not just their public-key, but unique information such as personal details and expiration dates. These can then be posted with, or generated by, so-called “trusted” certificate authorities such as Verisign. However, the underlying content is still essentially an RSA based public-key.

### PGP and a Cryptographic Crisis

Finally, it should be noted that although RSA public-key cryptography has been enormously successful and important to the growth of the Internet and e-commerce, there have been significant issues surrounding it. The most famous concerns the development of a software package called “PGP” (for “Pretty Good Privacy”). [2](pages 298-301) Although the RSA scheme had been published in 1977 in a widely available journal, the algorithm itself was patented by Rivest, Shamir and Adelman and a company was established to license the algorithm. This licensing was almost exclusively made to very large businesses and to the government. However, by the late 1980’s electronic communications tools were increasingly available to private individuals and privacy issues were stating to come to the forefront. In part to try to give private citizens access to improved privacy, an individual named Philip Zimmermann began developing a PC software package based on the RSA algorithms. His original intent was to obtain a license from RSA. However, he experienced difficulty in obtaining a license and 1991 he delivered a working copy of PGP, together with source-code to a friend to be placed on the Internet. As a result, individuals as far away as China and Russia suddenly had access to extremely strong cryptographic technology.

Two things immediately happened after the release of PGP. First, Mr. Zimmermann was sued by RSA for patent violation. Secondly and perhaps more interestingly, the US Federal government began inquiries
into Mr. Zimmerman’s actions, which included allegations that he had exported “munitions” outside of the United States without a license. At the time, the US Government categorized strong encryption technology as a munition comparable to a firearm or an explosive, requiring formal export approval. By placing PGP on the Internet, the government argued that Mr. Zimmermann had exported a munition. Ultimately, Mr. Zimmermann became the subject of an FBI grand jury investigation.

The US Government in fact became so concerned about the availability of strong encryption tools that they proposed an alternative encryption scheme, known as “Clipper” and “Capstone” which would be mandated on all electronic equipment (e.g., cell phones, modems, etc.). However, both Clipper and Capstone would “escrow” copies of all cryptographic keys with the US Government, the argument being that without mandatory key-escrow, the government would be unable to implement wire-taps and the like for legitimate law enforcement purposes. Not surprisingly, both privacy advocates and businesses reacted extremely negatively to the proposal. The privacy concerns are clear. However, businesses objected to the proposal because they recognized that no buyer outside of the United States would be interested in technology with a built-in back-door for the U.S. government.

By the end of the 1990’s, the FBI had dropped their grand jury investigation and Mr. Zimmermann obtained a license from RSA. More importantly, the government had also dropped their Clipper/Capstone proposal. Although this may have been a decision based upon public outcry, in all likelihood it is because the government recognized that it was not possible to control encryption technology. By the end of the 1990’s, PGP translations were available on the Internet in virtually every part of the world. It’s important to note that although the US has largely lifted any restrictions on strong encryption technology, there are many countries which have not. The current host of the 2008 Summer Olympics requires that all hotels install electronic communications interception technology for the government and outsiders may not enter that country with strongly-encrypted data without either a governmental license or provide an escrow copy of the keys to government.

References