Global Information Assurance Certification Paper

RSA: Overview, Analysis, and Implications for the Future

By Kristi A. Aho

February 16, 2001

Introduction

As the amount of business conducted over the Internet continues to rise, the need for secure communications methods becomes more critical. The exchange of sensitive information, such as credit card numbers and other financial data, over the Internet is commonplace. We regularly rely on safe, private communications over an unsecure network. Public key cryptography, also referred to as asymmetric encryption, and digital signatures are integral technologies in the effort to keep private information out of unfriendly hands and to conduct reliable business transactions over the Internet.

The security of RSA is based on the difficulty of factoring large numbers that are the product of two prime numbers. This factoring problem has been studied for hundreds of years and still appears to be intractable. For this reason, people are confident of RSA's security, and it has become fundamental to information security. Another "warm and fuzzy" reason for RSA's popularity should not be overlooked. Contrary to some other cryptographic systems such as Elliptic Curve cryptography, it is relatively easy for executives, managers, and other decision-makers who do not have a technical or mathematical background to understand the difficulty of factoring a very large number, and thus feel confident in the security provided by the RSA algorithm.

As a result, RSA is widely used today in many of the encryption applications and protocols in use on the Internet, including Pretty Good Privacy (PGP), the Secure Sockets Layer (SSL), S/MIME, Secure Electronic Transactions (SET), and the Secure Shell (SSH). RSA is also included in major World Wide Web browsers such as Netscape and Microsoft Internet Explorer. This paper will provide an overview and analysis of RSA, including its history, its security against attack, and implications for its future use.

Overview of Public Key Cryptosystems

Using public key encryption, it is possible for two users who do not know one another to send secret messages back and forth without having to agree upon and exchange a secret key over an untrusted network. A pair of keys is used to make this communication possible. One key, the public key, is used for encryption and is distributed freely to other users. The other key, the private key, is not shared with other users and can be used for decryption. The keys are different, and one key cannot be determined from the other.

Public key encryption is several orders of magnitude slower than symmetric encryption, which uses only one secret key that is shared between sender and recipient. For this reason, a hybrid encryption scheme is used when implementing public key cryptosystems. The sender, Alice, generates a random key and uses it to encrypt the message using a symmetric encryption algorithm such as DES, triple-DES, IDEA or Rijndael (recently selected by the U.S. National Institute for Standards and Technology as the Advanced Encryption Standard (AES)). This random key, referred to as a session or message key, is then encrypted using the recipient, Bob's, public key. Bob then uses his private key to decrypt the session key, which is in turn used to decrypt the message.

In addition to key transfer as described above, RSA can also be used to ensure that a message which asserts that it came from Alice was not actually sent by an imposter. This is referred to as authentication and involves digitally "signing" a message. The same public-private key pair is used when signing a message. The procedure for signing a message is as follows: Alice takes a one-way hash or digest of the message using a hash function such as SHA-1 (the U.S. government's Secure Hash Algorithm), RIPEMD-160 (a European algorithm), or MD5. This creates a cryptographic checksum of Alice's message.

A signature can be generated for the hash by encrypting it using Alice's private key. The signature then is appended to the original message and sent to Bob. When Bob receives the message, he passes the message through the same hash and also decrypts the signature using Alice's public key. Bob's newly calculated hash and the decrypted hash calculated by Alice should match, thereby verifying that Alice was the one who sent the message. Because public key encryption computations are slow, as noted above, it is faster to sign the hash rather than the actual message. It should also be noted that some digital signature algorithms do not actually encrypt the hash. They perform a different, faster type of mathematical calculation on the hash based on the message and the private key. The same calculation is performed by the message recipient using the hash, the message and public key to verify the sender's authenticity.

There are several popular public key encryption algorithms in use today, including ElGamal, elliptic curves, and RSA. In addition, the Diffie-Hellman key exchange algorithm can be used for secure exchange of keys. There are also currently several popular public key algorithms that can be used for digital signatures, including the U.S. government's Digital Signature Algorithm (DSA), used in the Digital Signature Standard (DSS), ElGamal, elliptic curves, and RSA.

History of RSA

The possibility of public key cryptography was first published in 1976[1] by Whitfield Diffie and Martin Hellman, who at the time were researchers at Stanford University. Ralph Merkle, a graduate student at the University of California, Berkeley, was studying the concept at the same time, but his ideas were not published until public key cryptography was well known. In their classic paper, Diffie and Hellman proposed the idea of public key cryptography and its use for exchanging keys, but not a public key cryptosystem. Several public key cryptosystems were subsequently proposed, but many were deemed insecure. Some systems are secure but are not practical for routine use either because the key is too large or because the ciphertext is significantly larger than the plaintext.

The RSA algorithm for public key cryptography, based on the idea that factorization of integers into their prime factors is hard to do, was proposed by (then) MIT professors Ronald Rivest, Adi Shamir, and Leonard Adleman in 1977. RSA has become one of the most successful algorithms for public key encryption and digital signatures. Many people had suspected that a government cryptographic agency such as the U.S. National Security Agency (NSA) had studied the possibility of public key encryption years earlier, but any evidence to this effect was classified. However, in 1997 CESG, a British cryptographic agency, released previously classified documents which revealed that James Ellis had discovered public key cryptography in 1970 and Clifford Cocks had internally published a version of the RSA algorithm in 1973. Nonetheless, Rivest, Shamir and Adleman are credited with the invention of RSA, and a patent for the algorithm was issued to MIT in 1983. The RSA patent will be discussed in more detail below.

A public key cryptosystem is made up of several components. There is a set of all possible plaintext messages, called M. There is also a set of keys, K. For each key $k \in K$, there is an encryption function $encrypt_k$ and a decryption function $decrypt_k$. These components must satisfy the following requirements:

1. $encrypt_k(decrypt_k(M)) = M$ and $decrypt_k(encrypt_k(M)) = M$ for every $m \in M$ and every $k \in K$.

2. For every m and every k, the values of $encrypt_k(m)$ and $decrypt_k(m)$ are easy to compute.

3. For almost every $k \in K$, if someone knows only the function $encrypt_k$, it is not computationally feasible to find an algorithm to compute $decrypt_k$.

4. Given $k \in K$, it is easy to find the functions $encrypt_k$ and $decrypt_k$.

[1] "New directions in cryptography," IEEE Transactions in Information Theory, 22 (1976), 644-654.

In the following section, we will show that the RSA cryptosystem satisfies the four requirements listed above.

RSA Algorithm

As mentioned earlier, RSA is based on the idea that it is difficult to factor large numbers. This is what makes RSA secure, provided that the public key is sufficiently large (see "Breaking RSA" section below). The following is a description of the mathematics of sending an encrypted message from Alice to Bob using the RSA algorithm.

1. Alice will choose two large (e.g. 512 or 1024 bit) prime numbers, P and Q.

2. Alice will choose an encryption key E such that E is less than the product $N = PQ$ and such that E and $(P-1)(Q-1)$ are relatively prime; in other words, $gcd(E,(P-1)(Q-1)) = 1$. $(P-1)(Q-1)$ is referred to as $f(N)$, commonly called Euler's f function or Euler's Totient function. Also, gcd stands for greatest common divisor, which is defined as the largest factor that two numbers have in common.

3. Using the extended Eudidean algorithm, Alice will compute the decryption key D which has the property that $DE \equiv 1 \bmod f(N)$. This can also be written $D \equiv E^{-1} \bmod f(N)$. Mod is short for $modulo$; the $modulo$ function over two variables, a and b, written $a \bmod b$ is defined to be the remainder when a is divided by b.

4. The numbers P and Q are no longer needed and should be kept secret or discarded.

5. The numbers N and E are the public key and can be freely distributed. The number D is the private key and should be kept secret. Alice sends her public key to Bob.

6. Bob writes his message as a number M, which must be smaller than N. If M is larger than N, Bob breaks the message into blocks, each of which is less than N.

7. Bob calculates the ciphertext $C = encrypt(M) \equiv M^E \bmod N$ and sends C to Alice.

8. Alice receives the ciphertext C and decrypts it to find the original plaintext message using the function
$M = decrypt(C) \equiv C^D \bmod N$.

9. Note that the encryption and decryption functions can be "reversed;" i.e., Alice could have encrypted a message M using her private key D. She could then send the encrypted message C to Bob, who would use Alice's public key E to decrypt C.

The following table summarizes this method.

RSA Algorithm Summary (Encryption)

Public Key	$N = PQ$, where P and Q are prime numbers that are kept secret E where $gcd(E, \phi(N)) = 1$
Private Key	$D \equiv E^{-1} \bmod \phi(N)$
Encryption Function	$encrypt(M) \equiv M^E \bmod N = C$
Decryption Function	$decrypt(C) \equiv C^D \bmod N = M$

We now recall the four requirements for a public key cryptosystem as listed in the "History of RSA" section above and demonstrate that RSA meets these requirements. In RSA, a key k is actually three values, E, D and N, where $D \equiv E^{-1} \bmod \phi(N)$. The encryption and decryption functions are inverses of one another, so we have satisfied requirement 1. If you know k, then you can determine the encryption and decryption functions, which satisfies requirement 4, and they are easy to compute, which satisfies requirement 2. Finally, we need to satisfy requirement 3. Given the appropriate selection of the key k, if we know only the encryption function and k (E, D and N), it is presumed, though not proven, to be computationally infeasible to determine the decryption function. This topic will be explored in more detail in the "Breaking RSA" section below.

RSA can also be used for digital signatures. The following table summarizes this method.

RSA Algorithm Summary (Digital Signatures)

Public Key	$N = PQ$, where P and Q are prime numbers that are kept secret E where $gcd(E, \phi(N)) = 1$
Private Key	$D \equiv E^{-1} \bmod \phi(N)$
Signature Function	$sign(M) \equiv M^D \bmod N = S$
Verification Function	$verify(S) \equiv S^E \bmod N = V$ If $V = M$ then the signature is valid

Breaking RSA

As noted earlier, the strength and security of the RSA algorithm rely on the difficulty of factoring large numbers. It should be noted, however, that it has not been mathematically proven that the only way to determine the plaintext message M from the ciphertext and the public key is by factoring N. It is theoretically possible that some entirely new method will be devised to find M. If such a method were discovered, it could also be used as a factoring method, and since this mathematical problem has been studied for hundreds of years, the mathematical community is confident that the RSA algorithm is quite secure. In fact, RSA has withstood years of extensive cryptanalysis. Again, however, we should mention that if a governmental cryptographic agency such as the NSA were to successfully cryptanalyze RSA, it is unlikely that they would publish those results. Rather they would use the results to decrypt the electronic communications transmissions of their adversaries.

Factoring N is possible if the key length is small enough and if enough computing resources can be devoted to the task. There are a number of different factoring methods than can be employed, such as the Universal Exponent Factorization Method, the Exponent Factorization Method, Pollard's $p-1$ Factoring Algorithm, the Quadratic Sieve, and the Number Field Sieve. The Number Field Sieve was successfully used in 1999 to factor both a 140- and a 512-bit RSA key and, at present, is the most powerful factoring method known. The challenge of breaking the RSA keys was posed by RSA Laboratories. This research organization will be discussed in detail in the "RSA Patent" section below.

RSA Laboratories publishes a series of cryptographic challenges to the public. The goals of the RSA Factoring Challenges are to help to encourage research into computational number theory and factoring techniques and to assure the public of RSA's security. Cash prizes are awarded to successful participants, and the results of these challenges are available to the public and are used to help RSA users determine suitable key lengths for various levels of security. When referring to RSA key lengths, one is actually referring to the size of N, the product of the primes P and Q. Thus if P and Q are both 256-bit numbers, then N is a 512-bit RSA key. In fact, a 515-bit N is a 155-digit number. Factoring the 155-digit RSA Challenge number was accomplished on August 22, 1999, by an international group of researchers using the following computers located in 11 different sites around the world:

160	175-200 MHz SGI and Sun workstations
8	250 MHz SGI Origin 2000 processors
120	300-450 MHz Pentium II PCs
4	500 MHz Digital/Compaq boxes
1	Cray C916 supercomputer

The team required 5.2 months, plus an additional nine weeks for preliminary computations, to factor RSA-155. This translated to about 35.7 CPU years. In contrast, it took only 8.9 CPU years and 9 weeks of calendar time to factor RSA-140, the 140-bit RSA Challenge number. Increasing the key size dramatically increases the difficulty of the resulting factoring problem. As a result of the RSA Factoring Challenges, as well as other research, the current minimum recommended key size for RSA is 768 bits. Key sizes of 1024 bits or even 2048 bits are not uncommon. When choosing a key size one needs to consider a number of factors, including the importance of the data, how long the data will need to remain secure, and the resources available to an adversary. For example, a much larger RSA key would be used to protect nuclear secrets than would be used to protect routine email messages. Computing power will continue to improve, and factoring methods have made great strides and presumably will continue to do so, but these methods are still very slow. The RSA algorithm, with a sufficiently large key length, remains highly secure.

The security of the RSA algorithm can be undermined by a flawed implementation of the algorithm in an application or protocol. There are a number of known implementation flaws. For example, the private key or the primes P and Q could be inadequately protected. Other known defects in implementation include using a common N for all users but giving them different values for the exponents E and D. With this approach, an attacker could factor N. Thus, a successful implementation of RSA should not use the same N for more than one user. In addition, choosing a value for E that is too low can leave a door open to attackers. This problem can be counteracted by padding a message with independent random values. Finally, choosing a value for D that is too low may also make it possible to learn the value of D. A secure implementation should have a large value for D.

Another more disturbing method of attacking RSA is through the use of timing attacks. This method was discovered in 1995 by Paul Kocher while he was an undergraduate student at Stanford University. Using the fact that many implementations of cryptography do things at different speeds for different keys, he demonstrated that it is possible to determine the private key being used by taking careful measurements of the length of time it takes to accomplish a series of decryptions. It is possible to defend against timing attacks, and this is currently the topic of research.

RSA Patent

U.S. patent number 4,405,829, "Cryptographic Communications System And Method," was issued to MIT for the RSA algorithm on September 20, 1983. The patent was scheduled to expire on September 20, 2000. Shortly after the patent was awarded, it was licensed exclusively to RSA Data Security Inc., (RSA Security) a maker

of encryption software. This company initially was controlled by RSA inventors Rivest, Shamir and Adleman but is now a wholly-owned subsidiary of Security Dynamics Technologies, Inc. RSA Laboratories is a division of RSA Data Security, Inc., and acts as their research arm. RSA Laboratories devises the RSA Challenges. Between 1989 and 1994, a company called Public Key Partners, which was a joint venture between RSA Data Security Inc. and Cylink, a maker of encryption hardware, administered the RSA patent. The joint venture dissolved acrimoniously in late 1994.

The RSA Patent has been highly controversial for several reasons. First, it is classified as a "software patent," and there has been much debate about whether or not computer algorithms are legally patentable at all. Second, it has commonly been claimed that the RSA Patent covers most of the widely used public key encryption and digital signature techniques, which would imply that anyone in the U.S. who wishes to use any of these techniques must purchase a license from the patent owner. Finally, over the period of the patent, RSA Security maintained a highly aggressive litigation posture in enforcing the patent. They filed a number of lawsuits and threatened legal action in many other instances in order to protect their intellectual property. Because of the patent, RSA Security had a virtual monopoly on the development and sales of public key encryption and digital signature software in the United States.

Although the RSA patent would have expired on September 20, 2000, RSA Security released it into the public domain two weeks earlier, on September 6. Software developers will now be free to include public key security methods freely in their products throughout the United States. Industry observers predict that many more security products will soon be available in the United States, including security tool kits designed especially for performance or for specific markets. Baltimore Technologies, a leading competitor of RSA Security based in Dublin, Ireland, also predicts that the influx of new security tools will enhance security and trust on the Internet and lead to advances in electronic commerce.

Although they will begin to face more and more competition, RSA Security believes that they have a long, strong history of providing high-quality implementations of public key cryptography tools and that this history will serve them well in the competitive marketplace. In fact, it is possible that there is higher potential for the release of flawed implementations of the RSA algorithm as competition increases and new developers enter the security software field.

Conclusion

The RSA algorithm is the most popular and the most widely implemented public key cryptosystem worldwide. It is used in most major security software tools in use on the Internet today. Since the algorithm has been highly scrutinized and it seems very unlikely that a rapid factorization technique will be developed within the near future,

RSA is very secure. The weak link in encryption using this algorithm will be in the cryptographic implementation, including the proper choice of keys. Thus, it would be very foolish to feel that the security of RSA will, by default, provide for secure networks. Great care and attention must still be paid to the implementation of even the most secure of algorithms. Given strong keys and skilled implementation, there does not appear to be any reason why RSA will not continue to be widely trusted throughout the cryptographic community.

References

Baltimore Technologies. "Why the Patent Expiry is a Significant Event." RSA Patent Expiry. URL: http://dev.baltimore.com/patentexpiry/significance.html (Feb. 4, 2001).

Berinato, Scott. "Expiration of RSA patents opens up Net security." August 25, 2000. URL: http://www.zdnet.com/eweek/stories/general/0,11011,2620278,00.html (Feb. 4, 2001).

Chidi, George. "RSA releases computer security patent." September 7, 2000. URL: http://www.cnn.com/2000/TECH/computing/09/07/security.patent.release.idg/ (Feb. 4, 2001).

Flinn, Patrick J. and Jordan, James M. "Using the RSA Algorithm for Encryption and Digital Signatures: Can You Encrypt, Decrypt, Sign and Verify without Infringing the RSA Patent?" July 9, 1997. URL: http://www.cyberlaw.com/rsa.html (Feb. 4, 2001).

Garfinkel, Simson with Spafford, Gene. Web Security & Commerce, Cambridge: O'Reilly & Associates, Inc., 1997.

Litterio, Francis. "The Mathematical Guts of RSA Encryption." 1999-2001. URL: http://world.std.com/~franl/crypto/rsa-guts.html (Feb. 4, 2001).

RSA Laboratories. "RSA Factoring Challenge." URL: http://www.rsasecurity.com/rsalabs/challenges/factoring/factoring.html (Feb. 16, 2001).

RSA Laboratories. "Factorization of RSA-155." URL: http://www.rsasecurity.com/rsalabs/challenges/factoring/rsa155.html (Feb. 6, 2001)

RSA Laboratories. "Factorization of RSA-155 -- Frequently Asked Questions." URL: http://www.rsasecurity.com/rsalabs/challenges/factoring/rsa155_faq.html (Feb. 6, 2001).

RSA Security, Inc. "RSA Crypto Challenge Sets New Security Benchmark." News, Press Releases. August 26, 1999. URL: http://www.rsasecurity.com/news/pr/990826-2.html (Feb. 6, 2001).

Schneier, Bruce. Secrets and Lies, Digital Security in a Networked World, New York: John Wiley & Sons, 2000.

Schneier, Bruce. Applied Cryptography, Second Edition, New York: John Wiley & Sons, 1996.

Trappe, Wade and Washington, Lawrence C., Introduction to Cryptography, preprint, 2000.

Zhou, Tao. "Digital Signature Technology." February 1999. URL: http://www.winntmag.com/Articles/Index.cfm?ArticleID=4772. (Feb. 19, 2001).

Last Updated: July 23rd, 2017

Upcoming Training

SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
Community SANS Omaha SEC401*	Omaha, NE	Aug 14, 2017 - Aug 19, 2017	Community SANS
Community SANS Trenton SEC401	Trenton, NJ	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC401: Security Essentials Bootcamp Style	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Pasadena SEC401 @ NASA	Pasadena, CA	Aug 23, 2017 - Aug 30, 2017	Community SANS
Mentor Session - SEC401	Minneapolis, MN	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
Mentor Session - SEC401	Edmonton, AB	Sep 06, 2017 - Oct 18, 2017	Mentor
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor Session - SEC401	Ventura, CA	Sep 11, 2017 - Oct 12, 2017	Mentor
Community SANS Albany SEC401	Albany, NY	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Columbia SEC401	Columbia, MD	Sep 18, 2017 - Sep 23, 2017	Community SANS
Community SANS Dallas SEC401	Dallas, TX	Sep 18, 2017 - Sep 23, 2017	Community SANS
SANS Baltimore Fall 2017	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS Copenhagen 2017	Copenhagen, Denmark	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Boise SEC401	Boise, ID	Sep 25, 2017 - Sep 30, 2017	Community SANS
Baltimore Fall 2017 - SEC401: Security Essentials Bootcamp Style	Baltimore, MD	Sep 25, 2017 - Sep 30, 2017	vLive
Community SANS New York SEC401	New York, NY	Sep 25, 2017 - Sep 30, 2017	Community SANS
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS London September 2017	London, United Kingdom	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Sacramento SEC401	Sacramento, CA	Oct 02, 2017 - Oct 07, 2017	Community SANS
SANS DFIR Prague 2017	Prague, Czech Republic	Oct 02, 2017 - Oct 08, 2017	Live Event
Community SANS Charleston SEC401	Charleston, SC	Oct 02, 2017 - Oct 07, 2017	Community SANS
Mentor Session - SEC401	Arlington, VA	Oct 04, 2017 - Nov 15, 2017	Mentor