International Journal of
Emerging Research in Management &Technology
ISSN: 2278-9359 (Volume-2, Issue-3)

Research Article | **March 2013**

An overview and Cryptographic Challenges of RSA

Bhawana
Department of CSE,
Shanti Devi Institute of Technology & Management,
Israna, Haryana
India

ABSTRACT: With the introduction of the computer, the need for automated tools for protecting files and other information stored on the computer became evident. Security is always a major concern in the field of communication. Secure and efficient data transfer is essential in many business sectors. To ensure the security to the applications of business, the business sectors use Public Key Cryptographic Systems. The public key cryptography solves one of the most vexing problems of all prior cryptography: the necessity of establishing a secure channel for the exchange of the key. An RSA system generally belongs to the category of Public Key Cryptographic System. This paper has an introduction which includes encryption and decryption using RSA and Cryptographic challenges with issues such as key distribution and speed of RSA.

KEYWORDS: RSA, Encryption, Decryption, Plain text, Cipher text, Public key, Private Key.

I. INTRODUCTION

In a classic cryptosystem in order to make sure that nobody, except the intended recipient, deciphers the message, the people involved had to strive to keep the key secret [1]. RSA is one of the oldest and most widely used public key cryptographic systems [4]. It was invented by Ronald Rivest, Adi Shamir and Leonard Adleman in 1977[15]. The letters RSA are the initials of their surnames .The basic technique was first discovered in 1973 by Clifford Cocks of CESG but this was a secret until 1997[10]. RSA was known as an Asymmetric cryptographic system. This encryption technique is basically a one-way function, so a message encrypted with a public key can only be decrypted with the corresponding private key. This technique simplifies the problem of key distribution somewhat; since there is only one key to distribute and it can be publicly published [12]. It is the first algorithm known to be suitable for signing as well as encryption, and was widely used in electronic commerce protocols, and is believed to be secure given sufficiently long keys and the use of up-to-date implementations[2].The RSA algorithm can be used for both public key encryption and digital signatures. RSA was one of the first great advances in asymmetric cryptography and the algorithm is based on the integer factorization problem [13].

II. OPERATION

This section describes how to generate a pair of keys and use them in encryption and decryption respectively.
The RSA algorithm involves three steps: key generation, encryption and decryption.

Generating the keys

This task can be divided into five steps. The first step, generate large prime numbers, requires a cryptographically secure random number generator. These prime numbers for RSA are in the order of thousands of bits, the smallest recommended key size by NIST is 1024 bits [14]. A simple example is given with small numbers to show the principle of the key generation.

1. Choose (generate) two different large prime numbers, **p** and **q**

   $p = 7$
   $q = 19$

2. Let n, the modulus for both the public and the private key, be defined by:

   $n = p \cdot q$
   $n = 7 \cdot 19$
   $= 133$

3. Compute the totient

   $\Phi(n) = (p - 1)(q - 1)$
   $\Phi(n) = (7 - 1)(19 - 1)$
   $= 6 \cdot 18$
   $= 108$

4. Choose a small integer *e* which is a coprime to $\Phi(n)$ such that $1 < e < \Phi(n)$

The below shown describes the different attempts of finding e out of $\Phi(n)$. In this example e = 5 is a small number and a coprime to $\Phi(n)$ and thereby a solution.

Table1.1 Different attempts of finding e out of Φ(n)

e	GCD$(e, \Phi(n))$	Status
$e = 2$	$GCD(2, 108) = 2$	incorrect
$e = 3$	$GCD(3, 108) = 3$	incorrect
$e = 4$	$GCD(4, 108) = 4$	incorrect
$e = 5$	$GCD(5, 108) = 1$	correct

5. Compute *d* to satisfy the equation

$de \equiv 1 \bmod(\Phi(n))$

Above Equation can be written as de = 1 + k Φ(n) or d = (1+k Φ(n))/ e where k is any integer. The last equation gives following table.

Table 1.2 describes the procedure of finding d

k	$d = (1 + k\Phi(n))/e$	Status
$k = 0$	$d = (1/5) = 0.2$	incorrect
$k = 1$	$d = (109/5) = 21.8$	incorrect
$k = 2$	$d = (217/5) = 5.4$	incorrect
$k = 3$	$d = (325/5) = 65$	correct

The above table describes the procedure of finding d by letting the integer k increase from 0 until d becomes an integer. Note that this is a simple example and to achieve a high security level the smallest d may not be the optimal solution.

This was a basic example. The calculations are becoming more time consuming as the integers are expanding. A more efficient way of finding d, is to use Euclid's algorithm [10]. However, these are the basics of key generation in RSA. The integers e and d, chosen in step four respectively five, are the private and the public key. It is possible to choose which integer is to be public respectively private. The different sizes of e and d make them suitable for different calculations. The integer e is smaller than d, and by choosing e as the private key, the decryption process will be faster than the encryption process. When choosing e to be the public key, the encryption will be faster than the decryption. However, the integer d is hereafter described as the private key and e is the public key.

RSA encryption This computation requires the integers n and e and the specific Plaintext, m. The size of the message is limited by the size of *n*. This means that in RSA encryption the message must be divided into, k, blocks m = m0,m1...mk where each mi < n. In the example the message '6' is used which is less than n = 133. The encryption algorithm is done by:

$c = m^e \bmod(n)$

Encrypting the message '6' using the public key e = 5 results in the Cipher text:

$c = 6^5 \bmod(133)$

$= 7776 \bmod(133)$

$= 62$

Because of the expensive computations of RSA it is not recommended to split the message into several blocks. The solution is to encrypt the message with a symmetric encryption and then encrypt the secret key with asymmetric encryption [11]. The decryption in RSA is however not as easy as the RSA encryption.

RSA decryption The decryption is similar to the encryption but it involves a larger exponent. The decryption requires the Ciphertext, c, the integer n and the private integer d and is calculated by:

$m = c^d \bmod(n)$

Decrypting the Ciphertext '62' using the private d = 65 will hopefully result in message *m* = 6. The calculations are done by:

$m = 62^{65} \bmod(133)$

$= 62 \cdot 62^{64} \bmod(133)$

$= 62 \cdot (62^2)^{32} \bmod(133)$

$= 62 \cdot 3844^{32} \bmod(133)$

$= 62 \cdot (3844 \bmod(133))^{32} \bmod(133)$

$= 62 \cdot 120^{32} \bmod(133)$

The sequence of operations which reduced 62^{65} to 120^{32} is now repeated to reduce the exponent to 1.

$m = 62 \cdot 36^{16} \bmod(133)$
$= 62 \cdot 99^{8} \bmod(133)$
$= 62 \cdot 85^{2} \bmod(133)$
$= 62 \cdot 43 \bmod(133)$
$= 2666 \bmod(133)$
$= 6$

As shown in decryption there are a great number of operations, however this example is using small numbers. Another short example is p = 61 and q = 53. These prime numbers give n = 3233 and the totient $\Phi(n) = 3120$. Then chose e = 17 and compute d = 2753. The use of the public and the private key will be:

The **public key** is (n = 3120, e = 17)

$c = m^{e} \bmod(n)$
$= m^{17} \bmod(3120)$

The **private key** is (n = 3120, d = 2753)

$m = c^{d} \bmod(n)$
$= c^{2753} \bmod(3120)$

Another example

Thus the plaintext message "HELLOWORLD" would be represented by the set of integers m1, m2, ... (9,6,13,13,16,24,16,19,13,5)

Using our table above, we obtain cipher text integers c1, c2, ... (3,18,19,19,4,30,4,28,19,26)

Note that this example is no more secure than using a simple Caesar substitution cipher, but it serves to illustrate a simple example of the mechanics of RSA encryption. Remember that calculating m^{e} mod n is easy, but calculating the inverse c^{d} mod n is very difficult, well, for large n's anyway[12]. However, if we can factor n into its prime factors p and q, the solution becomes easy again, even for large n's. Obviously, if we can get hold of the secret exponent d, the solution is easy, too.

III. CRYPTOGRAPHIC CHALLENGES

There are several different asymmetric encryption schemes using different mathematical problems in order to make them irreversible. To be able to understand the better part of the algorithms, most of the mathematics terms are being described, though some are assumed to be well known. There are mainly three families of asymmetric cryptography. The most widely used are those based on the integer factorization, RSA in particular [13]. Various cryptographic challenges including the RSA Factoring Challenge served in the early days of commercial cryptography to measure the state of progress in practical cryptanalysis and reward researchers for the new knowledge they have brought to the community.

*Factoring Challenge***:** The RSA Factoring Challenge was a challenge put forward by RSA Laboratories on March 18, 1991 to encourage research into computational number theory and the practical difficulty of factoring large integers and cracking RSA keys used in cryptography.The RSA Factoring challenge was factoring effort, to learn about the actual difficulty of factoring large numbers of the type used in RSA keys. Posted here for historical interest is the set of eight challenge numbers, ranging in size from 576 bits (174 decimal digits) to 2048 bits (617 decimal digits) that made up the challenge. Each number is the product of two large primes, similar to the modulus of an RSA key pair [7].Factoring a number means representing it as the product of prime numbers. Prime numbers, such as 2, 3, 5, 7, 11, and 13, are those numbers that are not evenly divisible by any smaller number, except 1. A non-prime, or composite number, can be written as the product of smaller primes, known as its prime factors. 665, for example is the product of the primes 5, 7, and 19. A number is said to be factored when all of its prime factors are identified.

Example

Name: RSA-576
Digits: 174
Digit Sum: 785
188198812920607963838697239461650439807163563379417382700763356422988859715234665485319060606504743045317388011303396716199692321205734031879550656996221305168759307650257059

As the size of the number increases, the difficulty of factoring increases rapidly [7]. Factoring 100-digit numbers is easy with today's hardware and algorithms. No public effort has yet resulted in successful factoring of numbers of more than 200 digits.

*Secret-Key Challenge***:** The RSA Secret-Key Challenge consisted of a series of cryptographic contests organised by RSA Laboratories with the intent of helping to demonstrate the relative security of different encryption algorithms. The goal of secret-key challenges was to quantify the security offered by the data encryption standard (DES) and other secret-key ciphers with keys of various sizes [6]. The information obtained from these contests was of value to researchers and developers alike

as they estimated the strength of algorithms or applications against exhaustive key-search. It is widely agreed that 56-bit keys, such as those offered by the government's DES standard, offer marginal protection against a committed adversary [7]. The RSA Secret-Key Challenge consisted of one DES challenge and twelve contests based around the block cipher RC5. While DES has a fixed key of length 56 bits, RC5 is a fully parameterized block cipher. RSA posted twelve RC5 contests as well as having a variable key size, RC5 also has a variable block size and a variable number of rounds; however, all the RC5 contests posted as part of the RSA Secret-Key Challenge used 12-round RC5 with a 32-bit word size. The different RC5 contests involved secret keys of different lengths. The first RC5 contest consisted of some unknown plaintext encrypted using a 40-bit key; the second consisted of some unknown plaintext encrypted using a 48-bit key; and so forth to the twelfth contest, which consisted of some unknown plaintext message encrypted using a 128-bit key.

IV. SPEED AND KEY DISTRIBUTION

RSA is much slower than DES and other symmetric cryptosystems. In practice, Bob typically encrypts a secret message with a symmetric algorithm, encrypts the symmetric key with RSA, and transmits both the RSA-encrypted symmetric key and the symmetrically-encrypted message to Alice. This procedure raises additional security issues [5]. For instance, it is of utmost importance to use a strong random number generator for the symmetric key, because otherwise Eve could bypass RSA by guessing the symmetric key. Key distribution must be secured against a man-in-the-middle attack. Suppose Eve has some way to give Bob arbitrary keys and make him believe they belong to Alice. Suppose further that Eve can intercept transmissions between Alice and Bob. Eve sends Bob her own public key, which Bob believes to be Alice's .Eve can then intercept any cipher text sent by Bob, decrypt it with her own private key, keep a copy of the message, encrypt the message with Alice's public key, and send the new cipher text to Alice. In principle, neither Alice nor Bob would be able to detect Eve's presence [8]. Defenses against such attacks are often based on digital certificates or other components of a public key infrastructure.

V. CONCLUSION

This paper provides that RSA is a public key cryptographic system which is widely used for encryption [1]. It is slower as compared to DES and its security is based on the difficulty of factoring large integers, it is used for small encryption exponent such as If you use a small exponent like e=3 and send the same message to different recipients and just use the RSA algorithm without adding random padding to the message, then an eavesdropper could recover the plaintext [9]. It uses a same key for encryption and signing which declines the security. Key distribution and signing if an attacker can convince a key holder to sign an unformatted encrypted message using the same key then she gets the original.

REFERENCES

[1] Rivest R, Shamir A and Adelman L, "*A Method for Obtaining Digital Signature and Public Key Cryptosystems*", Communications of the ACM, 21, pp. 120-126, 1978.
[2] Diffie W and Hellman M, "*New Direction in Cryptography*, IEEE Transaction on Information Theory, IT- 22(6): 644-654, 1976.
[3] Okamoto T and Uchiyama S "*A New Public Key Cryptosystem as Secure as Factoring*", in Proceedings of Europcrypt'98, LNCS 1403, Springer Verlag, pp.308-318, 1998.
[4] Lam K. and Hui L, "*Efficiency of square- and-multiply exponentiation algorithms*", Electronics Letters, Vol. 30, Issue 25, pp.2115- 2116, 1994.
[5] Douglas R. Stinson "*Cryptography Theory and Practice*", Chapman & Hall/CRC Press, 3rd Edition, pp. 211-214, 2006.
[6] Pointcheval D "*New Public Key Cryptosystem Based on the Dependent-RSA Problem*", in proceedings of Euorcrypt'99, LNCS 1592, Springer Verlag, pp. 239-254, 1999.
[7] Rabin, M. "*Digitalized signature and Public Key Functions as intractable as factorization*", Technical Report, MIT/ LCS/ Tr, MIT Lab. Computer Science, Cambridge, Jan. 1979.
[8] Shimada M "*Another Practical Public Key Cryptosystem*", Electronic Letters, 5th November, Vol. 28, No. 23, 1992
[9] S. Yen, S. Kim, S. Lim and S. Moon, "*RSA Speedup with Chinese Remainder Theorem Immune against Hardware Fault Cryptanalysis*", IEEE Transaction on Computers, Vol. XX, No. Y, pp. 461-472, 2003.
[10] Alexander Bogomolny. Euclid's algorithm from interactive mathematics miscellany and puzzles. URL: http://www.cut-the-knot.org/blue/Euclid.shtml, 2007.
[11] William Stanley Jevons. *The Principles of Science: A Treatise on Logic and Scientific Method.* 1874, 2nd ed. 1877, 3rd ed. 1879.
[12] Douglas R. Stinson "*Cryptography Theory and Practice*", Chapman & Hall/CRC Press, 3rd Edition, pp. 211-214, 2006.
[13] RSA Laboratories. What is the factoring problem? URL: http://www.rsa.com/rsalabs/node.asp?id=2189, 2007.

[14] RSALaboratories. Rsa algorithm. URL: http://www.rsa.com/rsalabs/node.asp?id=2146, 1977.
[15] RSA Laboratories. Pkcs #1 v2.1: Rsa cryptography standard. URL: ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-1/pkcs-1v2-1.pdf, June 14, 2002.