ISSN 2278-7763

A Mathematical Attack Based Algorithm to Challenge the Security of RSA Cryptosystem*

Nitin R. Mise[1], and H. C. Srinivasaiah[2]

[1]Department of Telecommunication Engineering, Dayananda Sagar College of Engineering, Bangalore, India, [2]Department of Telecommunication Engineering, Dayananda Sagar College of Engineering, Bangalore, India.
Email: [1]*nitinmise@gmail.com*, and [2] *hcsrinivas@dayanandasagar.edu*

ABSTRACT

The ubiquity of RSA (named after its inventors Ron Rivest, Adi Shamir, and Leonard Adleman) has placed it at the heart of modern information security. It would not be an overstatement to say that Internet security relies heavily on the security properties of the RSA cryptosystem. While no devastating attack has ever been found, years of cryptanalysis of RSA have given us a broad insight into its properties and provided us with valuable guidelines for proper use and its implementation. RSA is an algorithm for public key cryptography, based on presumed difficulty of factoring large integers. Access to powerful cluster based parallel computing machines for adversaries, may lead to breaking RSA security through many of the attacking techniques highlighted in this paper. Based on binomial theorem and AKS algorithms, we present a mathematical attack based algorithm that challenges the security of RSA cryptosystem, necessitating improvement over existing cryptosystems, against any type of possible attacks.

Keywords : Cryptosystem, Cryptanalysis, Binomial theorem, Prime factorization**,** RSA algorithm, and AKS algorithms.

1 INTRODUCTION

The RSA cryptosystem, first publically described algorithm in 1977, is commonly used in securing ecommerce and *e*-mail, implementing virtual private networks and providing authenticity of electronic documents. It is also used to provide both secrecy and digital signatures and its security is based on the intractability of the integer factorization. Integer factorization is one of the oldest problems in mathematics. The nature of the problem of factoring is based on time complexity of the algorithm. The RSA scheme is block ciphering in which the plaintext and cipher text are integers between *0* and *n-1* for some *n*. A typical size for *n* is 1024 bits, or 309 decimal digits, it means *n* is less than 2^{1024} but the boundary or limit has been changing with time with the availability of more powerful computer particularly large parallel systems, have altered the scenario considerably. Factoring the public key is seen as the best way to go about cracking RSA.

RSA is being used for encrypting the secret data and authentication of user using the concept of digital signatures. The RSA algorithm security can be attacked by brute force attack, mathematical attacks, cryptanalysis attacks and timing attacks. The first attack is through factoring modulus which results in deciphering of messages, tractable. The present paper attempts to factorize modulus into prime numbers using parallel algorithm and well known number theory results.

1.1 Public Key Cryptography

Public key cryptosystems work on the principle of one-way functions which means mathematics functions which are easy to compute but their inverse function is rather difficult to compute.

The concept of public-key cryptography was proposed in 1976 by Whitfield Diffie and Martin Hellman. This concept minimizes the need for secure key distribution channels and supplies the equivalent of a written signature. The concept of public key cryptography has been of significant value for the purpose of transferring confidential data from one person to another. Two separate keys are used in public key cryptography, Public key and Private key. One is used to encrypt plain text to cipher text and other key is used to decrypt cipher text to get plain text. Since public key is available to everyone and private key is secret, there is no problem in distribution of keys.

1.2 RSA Public key Cryptography

RSA works on the following number theoretic result of Euler: For any number '*n*', number of co-primes between 1 and n is designated as Euler's totient function $\phi(n)$. If 'a' is co-prime then

$$a^{\phi(n)} \equiv 1 \pmod{n}$$

RSA scheme refined the idea and made it practicable. They chose *n* to be a product of two large primes, say *p* and *q*. Then number theoretic argument shows that:

$$\emptyset(n) = (p-1) \times (q-1)$$

This also shows that if message *x* lies from 1 to *p*-1, then for each *x*, one has

$$x^{\phi(n)} = 1 \pmod{n}$$

RSA scheme chose two number *e* and *d* such that

ISSN 2278-7763

$$ed = 1 mod \emptyset(n)$$

Now it is clear that if $1 \leq x \leq p-1$ (or $q-1$) whichever is smaller,

$$(x^e)^d = x^{1(\mathrm{mod}\ \phi(n))} = x$$

Thus x^e becomes the encrypted message a and a^d becomes the decrypted message x. This is the essence of RSA.

It is also clear that modulo n depends on two large prime p and q, which should be reasonably close for better margin of message space. Thus the factorization plays a big role because n and e lies in the public domain and d can be relative easily deciphered if p and q are known in

$$e \times d = 1 mod((p-1) \times (q-1))$$

2 RSA ALGORITHM

RSA is an algorithm for public key cryptography that is based on the presumed difficulty of factoring large integers. Its security is based on the intractability of the integer factorization. The RSA scheme is a block cipher in which plaintext and cipher text are integers between *0* and *n-1* for some *n*. Typically the minimum size for *n* is 1024 bits, or 309 decimal digits.

2.1 Main idea of the Algorithm

The scheme makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number *n*. That is, the block size must be less than or equal to $\log_2(n)$ in practice, the block size i bits, where $2^i < n < 2^{i+1}$. Encryption and decryption are of the following form, for some plaintext block M and cipher text block C

Cipher text $C = M^e mod\ n$,

Message $M = C^d mod\ n$.

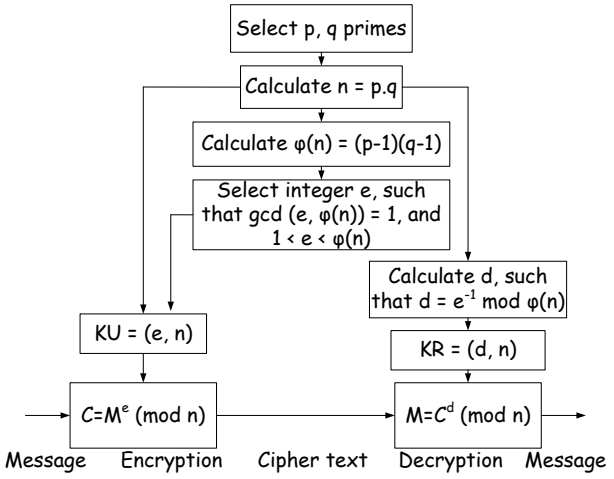

Fig. 1: Flow chart of RSA Algorithm

Both sender and receiver must know the value of *n*. The sender knows the value of *e*, and only the receiver knows the value of *d*. Thus, this is a public-key encryption algorithm with a public key, KU = {*e*, *n*} and a private key, KR = {*d*, *n*}. For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:

1. It should be possible to find values of *e*, *d*, and *n* such that M^{ed} mod *n* = M for all *M*<*n*.
2. It should be relatively easy to calculate $M^e mod n$ and $C^d mod\ n$, for all values of *M*<*n*.
3. It should be infeasible to determine *d* given *e* and *n*.
4. We need to find a relationship of the form $M^{ed}\ mod\ n = M$.

The preceding relationship holds if *e* and *d* are multiplicative inverses modulo $\phi(n)$, where $\phi(n)$ is the Euler totient function. For *p*, *q* prime, $\phi(p,q) = (p-1)\times(q-1)$. The relationship between *e* and *d* can be expressed as $edmod\ \phi(n)=1$

This is equivalent to saying

$$e \times d = 1 mod \emptyset(n)$$

$$d = e^{-1} mod \emptyset(n)$$

That is, *e* and *d* are multiplicative inverses mod $\phi(n)$ and relatively prime to $\phi(n)$; equivalently, $gcd(\emptyset(n), d) = 1$

2.2 Implementation of RSA algorithm

The RSA algorithm involves 3 steps

1. Key Generation
2. Encryption
3. Decryption

2.2.1 Key Generation Algorithm

RSA involves a public key and a private key. The Public key can be known to everyone and is used for encrypting messages. Messages encrypted with public key can only be decrypted in a reasonable amount of time using the private key. The keys for the RSA Algorithm are generated the following way.

1. Choose two large prime numbers *p* and *q* at random, and should be of similar bit-length (*p* and *q* both primes, and $p \neq q$).

2. Compute $n = p \times q$; *n* is used as the modulus for both public and private keys. Its length, usually expressed in bits, is the key length.

3. Compute $\phi(n) = (p-1)\times(q-1)$, where $\phi(n)$ is Eulers's totient function.

   Euler's totient function ($\phi(n)$) is an arithmetic function that counts the number of positive integers less than or equal to *n* that are relatively prime to *n*. That is if *n* is a positive integer, then $\phi(n)$ is the number of integers *k* in the range $1<k<n$ for which gcd (*n*, *k*) = 1.

4. Choose an integer *e* such that $\gcd(\phi(n),e)=1$ and $1<e<\phi(n)$, i.e. *e* and $\phi(n)$ are co-prime, *e* is released as the public key exponent.

ISSN 2278-7763

5. Determine d as $d=e^{-1}\text{mod}(\phi(n))$ i.e. d is the multiplicative inverse of $e^{-1}\text{mod}(\phi(n))$, d is kept as private key exponent.

Public key KU = {e, n}

Private key KR = {d, n}

User 'A' can send an encrypted message to user 'B' without any prior exchange of secret keys. 'A' just use's 'B''s public key to encrypt the message and 'B' decrypts it using its own private key, which only he knows. The RSA can also be used to sign a message, so 'A' can sign a message using its private key and 'B' can verify it using 'A''s public key.

2.2.2 Encryption

Sender 'A' does the following:

1. Obtains the recipients B's public key (e,n)
2. Represents the plaintext message as a positive integer M, 1<M<n
3. Computes the ciphertext C = $M^e \bmod n$
4. Sends the ciphertext C to B.

2.2.3 Decryption

Recipient 'B' does the following:

1. Uses his private key (d,n) to compute $M = c^d \bmod n$.
2. Extracts plaintext from the message representative m.

The RSA algorithm is also used for generating digital signatures, which can provide authenticity and non-repudiation of electronic legal documents.

3 SYSTEM ARCHITECTURE

The proposed approach to mathematical attack on the security of RSA cryptosystem requires very powerful parallel computing machine. We have succeeded in prime factorization of large integers using a novel technique. The algorithm developed based on this technique is discussed in later section. In order to run this algorithm we have used Flosolver (i.e. floating point operation solver) MK8 architecture based parallel machine. This architecture is cluster based, with cluster configuration as shown in Fig. 2. Flosolver MK8 architecture is organized as many interconnected clusters, where each cluster is made up of a four server boards. Flosolver MK8 has 1024 processors organized as clusters and its being developed to have a performance of 10 TFLOPS. Each cluster consists of 8 processors, FPGA based Flo-switch for inter and intra cluster communication and PCI card. Clusters are replicated and are interconnected using optical modules.

4 AKS ALGORITHM

The AKS primality test is a deterministic primality proving Algorithm (named after its inventors Manindra Agrawal, Neeraj Kayal, and Nitin Saxena) in a paper entitled "PRIMES is in P" [1]. The Algorithm determines whether a number is prime or composite within polynomial time. AKS is the first primality proving algorithm to be general, polynomial, deterministic, and unconditional. Previous algorithms had been developed for centuries but achieved three of these properties at most, but not all above four merits of AKS algorithm. The maximum running time of the algorithm can be expressed as a polynomial over the number of digits in the target number (n). The algorithm is guaranteed to distinguish deterministically, whether the target number n is prime or composite. The correctness of AKS is not conditional on any subsidiary unproven hypothesis [2].

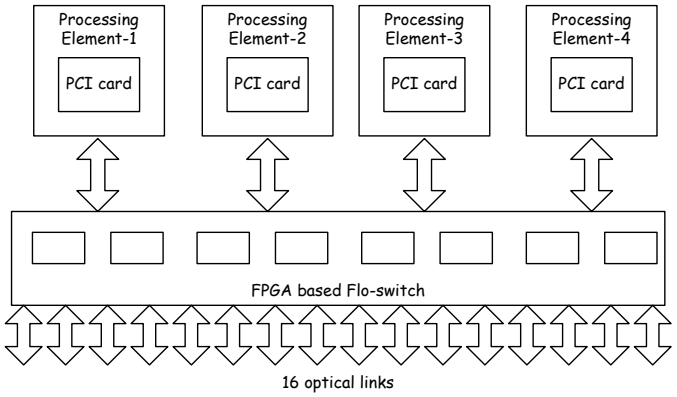

Fig. 2: Block diagram of single cluster configuration of Flosolver MK8.

The algorithm is as follows:

Input: integer $n > 1$.

1. If $n = a^b$ for integers $a > 1$ and $b > 1$, output *composite*.
2. Find the smallest r such that $O_r(n) > (\log n)^2$.
3. If $1 < \gcd(a,n) < n$ for some $a \le r$, output *composite*.
4. If $n \le r$, output *prime*.
5. For $a = 1$ to $\sqrt{\varphi(r)}\log(n)$ do
   If $(x+a)^n \neq (x^n+a) \pmod{x^r-1,n}$, output *composite*;
6. Output *prime*.

Here $O_r(n)$ is the multiplicative order [1] equal to '*n modulo r*', *log* is the binary logarithm, and $\phi(r)$ is Euler's totient function of r.

If n is a prime number, the algorithm will always return *prime*, since n is prime, steps 1 and 3 will never return *composite*. Step 5 will also never return *composite*, because 2 is true for all prime numbers n. The algorithm will return *prime* either in step 4 or in step 6.

Conversely, if n is composite, the algorithm will always return *composite*, if the algorithm returns *prime*, then this will occur in either step 4 or step 6. In the first case, since $n \le r$, n has a factor $a \le r$ such that $1 < gcd(a, n) < n$, which will return *composite*. The remaining possibility is that the algorithm returns *prime* in step 6.

4.1 An Approach to attack RSA

This approach to attack RSA is based on the result of AKS algorithm which states that a number n is prime if

$$((x + a)^n \,! \equiv (x^n + a) \ (mod\ n))$$

ISSN 2278-7763

This concept is used to find out the position k in Pascal triangle for which modules of sum of k binomial coefficients with n is coming out to be greater than one since $\binom{n}{0} (\text{mod } n) = 1$.

Greatest Common Divisor (gcd) of k with n will give us the value of p.

The Algorithm is as follows

- find k which satisfy the relation $\sum_{i=0}^{k} \binom{n}{i} (\text{mod } n) > 1$
- $p=gcd\ (n,k)$ and $q=n/gcd\ (n,k)$
- gcd of n and k can be find out using Euclidean Algorithm with complexity in the order of log n.

Since there is no direct approach to find value of $\sum_{i=0}^{k} \binom{n}{i}$, it is very difficult to implement this algorithm for factoring large composite numbers. But once a computationally better way of solving the above equation is found out, there might be a hope for implementing this proposed algorithm. Nevertheless this algorithm is tested to verify the integer factorization of fairly large numbers into prime factors, and its results are presented in next section.

5 RSA FACTORIZATION

The security of the RSA cryptosystem relies on the very well known problem of factoring large integers, which is widely believed to be intractable [3, 4, 5]. If an adversary can factor n, he can easily calculate the private exponent by solving the congruence $e \times d \equiv 1 \ mod(\Phi(n))$, and thus invert the RSA function.

1. Key (e, n) is in the public domain and (d, n) will be in private domain.
2. Modulus-n is factored into two primes p and q.
3. Euler's totient function can be calculated by
   $\Phi(n) = (p-1)\times(q-1)$, and
4. d can be easily deciphered if p and q are known in:
   $e \times d \equiv 1 \ mod\ ((p-1)\times(q-1))$

Accordingly, security of RSA will be compromised once the private key (d, n) is deciphered. The main objective of this research is to find the prime factors of large integers thereby challenging the security of the RSA cryptosystem.

6 RESULTS

The modulus-n is factored into two primes 'p' and 'q', using the algorithm of section 4.1 on supercomputer: Flosolver MK8. Some results to highlight the essence of our algorithm are as follows:

1. Number, n = 35 ($n = p \times q$)
   p=7
   q=5
2. Number, n=782507347
   p=953
   q=821099
3. Number, n=1888362961
   p=98573
   q=19157

7 CONCLUSIONS

RSA is the most secure Algorithm known till date and many efforts are in practice to attack its security. Binomial theorem and AKS algorithm shows a pathway to attack RSA but large calculations involved in manipulating binomial coefficients makes it very difficult to break it. With advancements in the theory of Binomial theorem, one is always moving one step ahead on the pathway to break it. So it is simultaneously necessary to work on development of some new cryptographic algorithm to live in a world free from the fear of data insecurity.

ACKNOWLEDGEMENT

Authors would like to thank Dr. U. N. Sinha, distinguished scientist CSIR – NAL/CMMACS for his valuable guidance and timely suggestions. Authors also acknowledge management, Dayananda Sagar Group of Institutions (DSI), Bangalore for all its support in pursuing this research in Research center, Department of Telecommunication Engineering, Dayananda Sagar College of Engineering, Bangalore, India. Our special thanks are due to Dr. Premachandra Sagar, Vice-chairman, DSI for all his encouragement for research.

REFERENCES

[1] M. Agrawal, N. kayal, and N. Saxena, "PRIMES is in P", *Annals of Mathematics*, Second Series, Vol. 160, No. 2 (Sep. 2004), pp.781-793.

[2] Hua Li, "*The Analysis and Implementation of the AKS Algorithm and Its Improvement Algorithms*", available at http://opus.bath.ac.uk/16757/1/CSBU-2007-09.pdf.

[3] Adi Shamir Eran Tromer, "*On the Cost of Factoring RSA1024*", available athttp://www.cs.tau.ac.il/~tromer/papers/cbtwirl.pdf.

[4] Divesh Aggarwal and Ueli Maurer, "*Breaking RSA generically is Equivalent to Factoring*" available at ftp://129.132.85.9/pub/crypto/publications/AggMau09.pdf.

[5] Sattar J Aboud and Mohammad A AL-Fayoumi, "*Efficent method for breaking RSA scheme*", available at http://www.ubicc.org/files/pdf/ubicc_manuscript_template1%5B1%5D_275.pdf.

