IJCSE International Journal of Computer Science and Engineering Open Access
Research Paper Volume-2, Issue-6 E-ISSN: 2347-2693

Optimization of RSA Algorithm by using Binary Method

Raushan Kumar Singh[1] and Shobhit Kumar[2*]

[1] Dept. of Information Technology, MKRECIT Ambedkar Nagar, Uttar Pradesh, draushan004007@gmail,com
[2*]Dept. of Information Technology, MKRECIT Ambedkar Nagar, Uttar Pradesh, shobhit5786@gmail.com

www.ijcaonline.org

Received: 18/05/2014 Revised: 25/05/2014 Accepted: 17/06/2014 Published: 30/06/2014

Abstract— This paper proposes a calculation method for the improvement of RSA algorithm which reduces the issues of scalability, flexibility and performance. This paper will be based on the mathematical function used in the RSA algorithm and reduces the number of steps for calculating the value of M^d. The proposed method improves the calculation performance of the RSA to make it useful in providing security to every known application.

Index Term— RSA algorithm, Key generation Encryption, Decryption, Modular Exponentiation, Binary Method.

I. INTRODUCTION

With the regular and fast expansion of internet and wireless-based communications across open networks, transmitted data requires protection. RSA is an asymmetric cryptographic technology that is widely used for the provision of data protection services.

RSA was proposed by Rivest. Shamir, & Adleman [1] which is an asymmetric cryptographic system that uses modular exponentiation for encryption and decryption. The actual meaning of 'asymmetric cryptographic' is that the keys used for encryption is made publicly available and the decryption key is kept private in case of RSA. RSA is block cipher, in which every message is mapped to an integer. RSA consist of Public-Key and Private-Key. Once the data is encrypted with the Public-Key, it can be decrypted with the corresponding Private-Key only [2].

DESCRIPTION OF THE ALGORITHM

In the RSA algorithm plaintext is encrypted in blocks, with each block having a binary value less than some number 'n', i.e., block size must be less than or equal to $\log_2 n$; the block size is 'i' bits long, where $2^i<n<=2^{i+1}$ [3].
RSA algorithm consists of three steps:-

1. Key Generation.
2. Encryption.
3. Decryption.

KEY GENERATION

Before the data is encrypted, key generation should be done by this way-

- Select two prime numbers p, q where $p \neq q$.
- Calculate M, M= p×q.
- Calculate Ø(M) = (p-1) × (q-1).
  Or Ø(M) = (M-p-q+1).

Ø(M) is Euler totient function.

- Now choose an integer e such that 1 < e > Ø(M) and gcd(Ø(M),e)= 1. Now 'e' is used as Public-Key exponent.
- After that determine d as follows: $d = e^{-1} \bmod Ø(M)$ i.e., d is multiplicative inverse of e mod Ø(M).
- D is kept as private-Key Component.
-

Public Key : (e, m).
Private Key : (d, m).

II. RELATED WORK

USE OF SIEVE FUNCTION IN KEY GENERATION

By using sieve function [4] we detect the large percentage of composite numbers by trail modular computation for its fixed set before the random is fed for Selection of strong prime number test. The procedure of reliable Selection of strong prime number tests is time consuming and using the sieve function unit prior to it, decrease the number of Primility test iteration. The sieve function includes a set of prime numbers such that S (T) = $\{P_i | 2 < P_i \leq T\}$ and P_i is i^{th} prime.

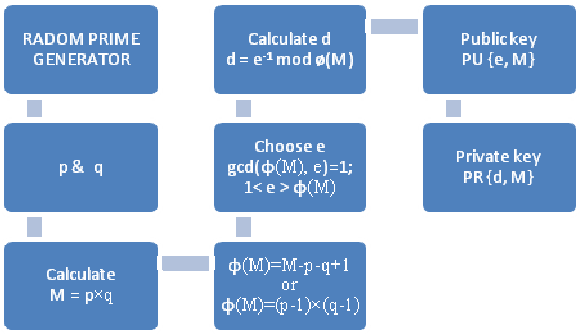

Fig.1 Flow Diagram For RSA Key Generator [4]

Corresponding Author: *Shobhit Kumar*

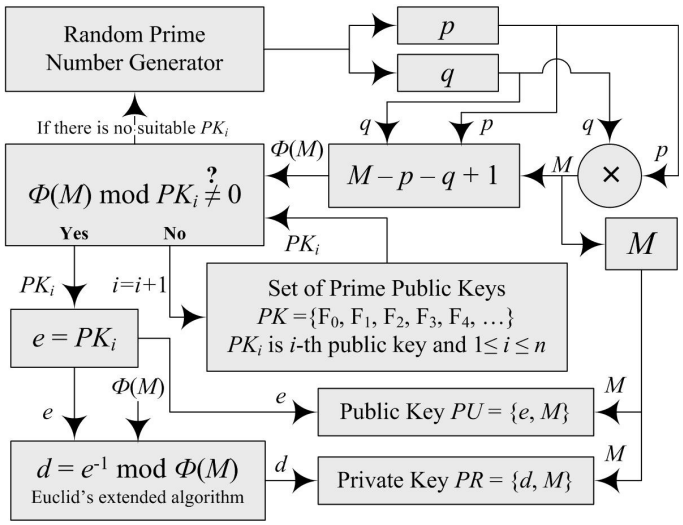

Fig.2 Flow Diagram of Key Generation

Mean time for generating RSA pair

Length of key pair	512bit	1024bit	2048bit
Mean time(sec)	2.85	6.82	44.78

With above diagram we extract the predefine set of public key to reduce the computation time of RSA algorithm. Also use random prime generator as a sieve function to decrease the numbers of iteration which is used for Selection of strong prime number test.

ENCRYPTION
To encrypt message simply computes

$$C = P^e \bmod M.$$

Where

C : Cipher text.
P : plain text.
e : encryption exponent.

DECRYPTION
To decrypt the cipher text C, use private exponent d and the modulus M.
$P = C^d \bmod M$.

PERFORMANCE IMPROVEMENT BY BINARY METHOD

This paper proposes an improvement of RSA algorithm by reducing the step of multiplication of exponentiation of any value of e and d. ($C = P^e \bmod M$)

The naïve [5] method would be prohibitive for large value of e and it requires (e-1) modular operations for calculation of P^e.

Suppose here e = 17 than required operations are (e-1) that is 16.

By using this method, compute all power of P until 17 –

$$P \rightarrow P^2 \rightarrow P^3 \rightarrow P^4 \rightarrow P^5 \rightarrow P^6 \text{ --------- } \rightarrow P^{17}$$

Number of operation is increase for large value of e.

For decreasing the number of operation and increasing the efficiency of RSA algorithm we use Binary method algorithm for calculation of modular exponentiation. These methods are also known as the square and multiply method.

From Binary method [6], it scans the bits of exponent either from right-to-left or left-to -right. Here we describe the left to right method.
The right-to-left algorithm requires one extra variable to keep the power of P [6].
By using binary method average number of multiplication is 3/2(k-1) where k is no. of bits of binary expansion of exponent.

PROBLEM DOMAIN
Binary method is easy and fast calculation method of power exponent value. But binary method apply only if given condition is satisfied-

$e_{k-1} = 1$ if $e_{k-1} = 0$ than binary method can not apply because if $e_{k-1} = 0$, than desire result does not come.

Example a:

If e = 17
Binary expansion = (10001). Here $e_{k-1} = 1$ is true and binary method apply. But if Binary expansion for predefine number of bits is 8.
Binary expansion = (00010001) = 17

Here $e_{k-1} = 0$ so pervious binary method can not apply for calculation so here we proposed a modified binary method for resolve this problem.

III. PROPOSED WORK

PROPOSED BINARY METHOD
For $C = P^e \bmod M$

1. Firstly compute the binary expansion of e-
   $\sum_{i=0}^{k-1} e_i 2^i$ = Binary expansion.
   Where k is number of bits of e.
   $k = 1 + \lfloor \log_2 e \rfloor$.
2. Find the greatest position of 1's in binary expansion. That is F, than
   j = F+1
3. If $e_{j-1} = 1$ then C = P else C = 1.
4. Run step a and b for i = j-1 down to 0

a. C = C.C (mod M).
b. If $e_i = 1$ then C = C.P (mod M).

5. Return C.

Example b:

$C = P^e \bmod M$
Let e = 17. Predefine number of bits = 8
So binary expansion of e = (00010001).
Position of bits 76543210

Greatest position of 1's in binary expansion
F = 4
So value of j
j = 5.

BINARY METHOD PROCEEDS AS FOLLOWS-
Initially assume $e_{j-1} = 1$
$e_{5-1} = e_4 = 1$.
Because $(e_7\ e_6\ e_5\ e_4\ e_3\ e_2\ e_1\ e_0) = (00010001)$.

i	e_i	Step a	Step b
3	0	$(P)^2 = P^2$	P^2
2	0	$(P^2)^2 = P^4$	P^4
1	0	$(P^4)^2 = P^8$	P^8
0	1	$(P^8)^2 = P^{16}$	$P^{16}P = P^{17}$

NUMBER OF MODULAR MULTIPLICATION REQUIRED BY THE BINARY METHOD

For given experiment total number of modular multiplication = 4+1 = 5.
This is 70-75% less than that of naïve method.

For any value of j number of multiplication

Step a: (j-1) number of operation.
Step b: (m-1) number of operation where m is number of 1's in binary expansion of exponent.

Now consider e > 0 thus total number of multiplication
Max: if j=m

$$(j-1) + (m-1 = (j-1) + (j-1) = 2(j-1)$$

For Min: if m=0

$$(j-1) + 0 = (j-1)$$

$$Avg = \frac{2(j-1)+(j-1)}{2}$$

$$= 3/2\ [(j-1)]$$

It means that number of multiplication Z lies between

$$2(j-1) \geq Z \geq (j-1).$$

IV. CONCLUSION

By using binary method for calculate the exponentiation value for encryption and decryption process of RSA algorithm are become time saving process for any large value of e and d.
In future we implemented the RSA algorithm by using the Binary Method and after that compare with other available model which show that how a binary method better than other method.

V. REFERENCES

[1]. "A method for obtaining Digital Signature and Public key crypto system" by R.Rivest, A.Shamir, and adleman.
[2]. "Data security in cloud computing using RSA Algorithm", International Journal of Research in Computer and Technology, IJRCCT, Parsi kalpna et. Al, ISSN 2278-5841, vol1, issue 4, September 2012.
[3]. "Cryptography and Network Security" principle and practices fourth edition, 2006, Page no. 306-366.
[4]. "A study on Improvement in RSA Algorithm and its implementation", International Journal of Research in Computer and Technology, IJRCCT, P Saveetha & S. Arumugam, ISSN volume-3 2012.
[5]. "High Speed RSA Implementation"Cetin Kaya Koc, RSA Laboratories, version-2.0 Nov 1994.
[6]. "The Art of computer programming: seminumerical Algorithms", D.E Knuth, volume-2 second Edition 1981.

