Indian Journal of Science and Technology, *Vol 8(16), DOI: 10.17485/ijst/2015/v8i16/69809, July 2015*

ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Analysis and Enhancement of Speed in Public Key Cryptography using Message Encoding Algorithm

D. I. George Amalarethinam[1*], J. Sai Geetha[2] and K. Mani[2]

[1]Department of MCA, Jamal Mohamed College, Trichy, Tamilnadu, India; di_george@ymail.com
[2]Department of Computer Science, Nehru Memorial College, Trichy, Tamilnadu, India; jsaigeetha99@gmail.com

Abstract

In the world of internet, Cryptography places a vital role as an effective measure to maintain information security. Whenever the security of public key cryptography algorithm increases, the execution speed decreases since both are inversely proportionate. It is mandatory to improve the security without affecting the speed. The main objective of this paper focuses on speed enhancement. Generally the encryption and decryption process is done on each characters of the plain text separately. It is a time consuming process. In this paper, the plain text is being decomposed into blocks to achieve better performance. The characters are to be grouped (Block Cipher mode) into blocks. For grouping of blocks, the plain text is replaced by the binary form of ASCII code. The binary plain text is divided into blocks with fixed size. If the size of block increases, then the execution speed is also increased. The crucial factor is to fix the block size. In some instance, the process of obtaining the plain text (decryption) is affected due to inappropriate block size. To overcome this issue, precise block size is fixed by using the proposed methodology namely Message Encoding Algorithm (MsgEncA). This algorithm is implemented in Java. As a result of proposed algorithm, the appropriate block size is fixed as double of the key size and less than modulus value. The execution speed is also analyzed with different key sizes and file sizes using RSA algorithm. The speed is compared with the existing algorithm namely Short Range of Natural Numbers (SRNN). Even when applied with different key size, the MsgEncA produces better performance with same level of security. An experimental analysis and comparison proves that the execution speed of MsgEncA is better than the existing method. In the aspect of speed, the proposed algorithm can be applied in any public key algorithms.

Keywords: Block Cipher, Decryption, Encryption, Message Encoding, Public Key Cryptography

1. Introduction

Data security is vital in communication aspect which also supports the confidentiality, authentication and integrity. In the real world, internet is being used for various purposes including information distribution. Data transmitted over the Internet passes through many servers and routers and there are many opportunities for intruders to intercept that transmission. Secure data transmission is really difficult because of the hackers. Cryptography is the technique for providing security to confidential data transmitted through unsecured channel[1].

Encryption and decryption have need of some secret values, usually referred to as a key. The same key might be used for both encryption and decryption in the case of symmetric mechanism. On the other hand, different keys are used for encryption and decryption for asymmetric mechanisms. Encryption can provide strong security for data to give sensitive data with high level of security. The goal of encryption is to make data indecipherable to unauthorized readers and extremely difficult to decipher when attacked. The security[2] of encrypted data depends on several factors such as algorithm used, key size and the process of implementation.

**Author for correspondence*

In this work, the security can be achieved on the basis of key size. However, higher key size increases the processing time for encryption and decryption. The higher key size is to be maintained in view of security. To overcome this issue, the message is divided into fixed size of blocks. The blocks[3] of message helps to improve the execution speed of cryptosystem. The optimum block size is revealed after an extensive analysis of various key sizes with different file sizes. The block size is decided by implementation with RSA algorithm of key sizes 128 to 2048.

2. Related Work

In addition to security, the speed is also important in communication network. The various techniques are introduced for speed enhancement. Gayatri Kulkarni, et al.[4] provides a technique in which Armstrong number is used for encryption of message. Color is important in authentication process as it acts as password. Using this technique, message is hidden from unauthorized people and accessible to an authorized individual when required. Ankita Nag and Vinay Kumar JainBit[5] are suggested a bit stuffing as a logic to be used instead of increasing the number of bits in RSA. Since larger bit numbers will require more time and effort for calculation, bit stuffing saves time and effort. This idea is implemented in hardware. Same security as with larger bit number (1024 bits) obtained almost same time even for lesser bit numbers (512 bits) with lesser band width requirement. Snehal Javheri and Rahul Kulkarni[6] proposed a secured symmetric key generation scheme which generates primary cipher and this primary cipher is then converted into final cipher using DNA sequences, so as to make it again more complicated in reading. Many research works is going on to make the computational process more complex to the unauthorized user.

A. E. Amin and A. Abd Elbadea[7] introduced cryptography real time data transmission framework based on concepts of artificial intelligence techniques. The proposed framework is integrated between concepts of genetic algorithm and generation of unique genetic key. This key is used to formulate patterns table. Then the secret message is divided into blocks with same size according to length of key. Pattern recognition is used for matching of blocks by weighted Euclidean distance. Nentawe Y. Goshwe[8] presented a design of data encryption and decryption in a network environment using RSA algorithm with specific message block size. The algorithm allows the message sender to generate a public key to encrypt the message. The plain text is encoded before encryption. The receiver decrypts the message and then decodes the plain text. An incorrect private key will still decrypt the encrypted message but to a form different from the original message. Mr. Anil Hingmire proposed a new algorithm that includes a phase of Substitution, Position and Zigzag encryption[9]. It generates only single key and takes a key level from the users which is used in Substitution and Position method. Since it undergoes three phases, the overall complexity increases and the algorithm becomes quite immune to attack. In this survey, Block cipher is the key method in symmetric cryptosystem. Now, this method is applied into asymmetric cryptosystem in the aspect of speed.

3. Proposed Methodology

3.1 Message Encoding

The cryptosystem is the mathematical process. It needs the plain text in numerical form. Before beginning the encryption process, the plain text is to be converted into numerical value by using the ASCII code.

There are two ways to generate the cipher text

- Block ciphers[10], which encrypt block of data of fixed size, and
- Stream ciphers, which encrypt continuous streams of data.

In public key cryptography, encrypting stream of data is a time consuming process. Modularization and exponentiation are the critical component of encryption and decryption in asymmetric cryptosystem. It increases the time taken by the mathematical calculation when using key with large size. It is not possible to the change steps by the cryptographic functions and reduces the key size. In the aspect of security, the size of the key places an important role. One of the ways to improve the speed is to change the format of plain text before encryption by applying MsgEncA for message encoding. It produces block of plain text and then encrypts each block separately. The result is improving the speed considerably. For Example, the plain text consists of 100 characters, the encryption process repeated 100 times for each character separately. In block ciphers[11], the plain text is divided into 3 blocks; the encryption is applied into each block. Hence the encryption process takes place only for 3 times.

Though the process takes place only 3 times, the encryption speed has not improved much on account of key size.

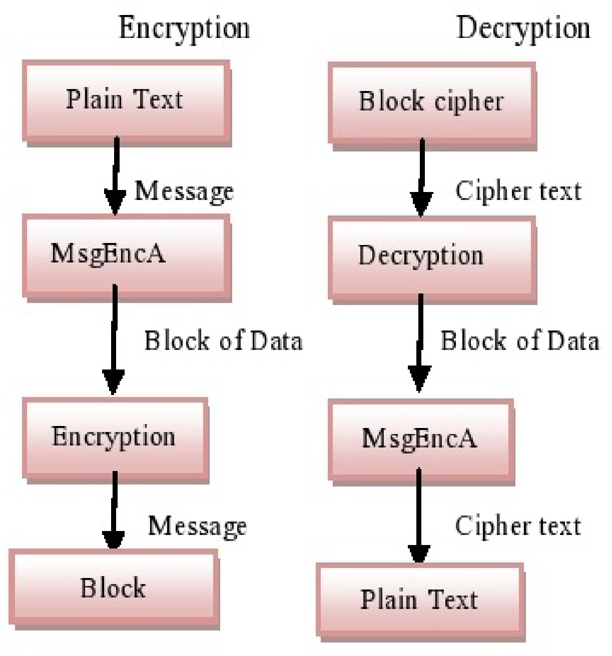

Figure 1. Frame work of MsgEncA method.

To overcome this issue, determination of exact blocks size to be found. The frame work of MsgEncA is given in Figure 1.

In MsgEncA, the plaintext is converted into binary string and fragmented into fixed size of blocks after being padded. This process is iterated for the entire message. The size of the block is fixed based on the key size. Finally, the resultant fixed length blocks are applied for the encryption and decryption process. The various size blocks are suitable for high speed encryption. The pseudo code of MsgEncA is given in the Figure 2.

In message encoding, the plain text 'M' is being segregated as blocks (B1,B2,…Bn). The first step of this process is to divide the plaintext into characters (C1,C2…Cm) which are interpreted into binary form of ASCII code. The blocks B1, B2..Bn of fixed size X are to be generated from the binary text obtained earlier. After the encryption process, the plain text is to be translated into block cipher Ch1,Ch2,…Chn. The same steps are to be reversed on message decoding process. Selection of suitable block size helps to improve the speed of encryption and decryption process and the higher key size enhances the security. Thus the block size plays an important role in performance enhancement.

```
Input: Plain text(M)
Output: Blocks of plain data(Bi where i=1,2,3…n)

Step 1: Read the plain text M.
Step 2: Convert into characters
            $M=C_i$  $C_i \in M$ i=1,2,3,…m
Step 3: Replace the characters by ASCII value
       $C_i=A_j$  $A_j \in$ ASCII value

Step 4: Convert ASCII value into 8 bits
          $A_j=b_0,b_1,b_2…b_7$ (8 bits)
Step 5: Fix the size of the block (X KB)
Step 6: Repeat
      Plain text bits into blocks of size X KB
       count=count +1
       Until no more bits in plain text
           n=count;
Step 7: Produce the block of data ($B_1,B_2,….B_n$)
Step 8: Encryption
Step 9: Produce Block cipher $Ch_i$ i=1,2,3,n
```

```
Input: Block Cipher ($Ch_i$) i=1,2,3..n
Output: Plain text(M)

Step 1: Read the block cipher $Ch_i$,  i=1,2,3,…n;
Step 2:  for i         1 to n
          Decryption of Block cipher $Ch_i$
          Convert  $B_i$ into bits
 Step 3: Repeat
          Split Block of data ($B_1,B_2…B_n$) into 8 bits
          Until no more bits
Step 3: Convert 8 bits into ASCII value
           $A_j=b_0,b_1,b_2…b_7$ (8 bits)  $A_i \in$ ASCII value
Step 5:  Replace Aj into characters.
           $C_i=A_j$  $A_j \in$ ASCII value
Step 7:  produce the plain text (M)
            $M=C_i$  $C_i \in M$ i=1,2,3,…m
```

Figure 2. Pseudo code of MsgEncA (Message encoding and Message decoding)

4. Results and Discussion

The security and performance are the two important parameters of public key cryptography. The proposed algorithm MsgEncA measures the block size based on the parameters such as execution time, file size and key size by using RSA Algorithm. The proposed algorithm MsgEncA has been implemented in Java.

The speed of the process is analyzed by using a file contains plain text with different size of blocks. In this research work, the file size and key size are fixed as 10KB and 128 bits respectively. The experimental results are tabulated in the Table 1 and graphically represented in Figure 3.

In this analysis, whenever the block size is increased, the performance of encryption and decryption process is also improved. Here, the block size is purely depending on the key size. The maximum size of the block is less than twice the value of the key size. If the block size exceeds the maximum size, the public and private keys are not suitable for encryption and decryption. It can be proved by the following experiment of key size with 32 bits and text file as 10 KB.

key size =32 bits

Encryption key=2865670243 Decryption key=8316262398734230927

Table 1. Comparison of various block size with speed

Filesize (Bits)	KeySize (Bits)	Block size(Bits)	No of blocks	Encryption Time(ms)	Decryption Time(ms)	Total Time(ms)
83264	128	16	5204	1383	2516	3899
83264	128	32	2602	660	1288	1948
83264	128	64	1301	359	680	1039
83264	128	128	651	172	344	516
83264	128	255	327	94	156	250

Table 2. Analysis of unsuitable block size with encryption and decryption process.

Block size(bits)	Block value(plain text)	Encrypted value	Decrypted value
65	36892544770435498496	11215373768679083684	5997296654191991350
64	11649164189284468767	3189697606957614345	4852055993351188286

Table 3. Analysis of suitable block size with encryption and decryption process.

Block size(bits)	Block value(plain text)	Encrypted value	Decrypted value
64	11649164189284468767	3069194259024954015	11649164189284468767
63	9223136192608874624	10613002358204603063	9223136192608874624

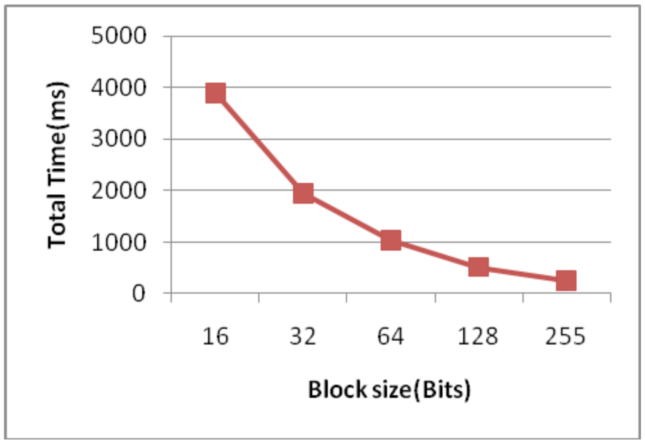

Figure 3. Comparison of various block size with speed.

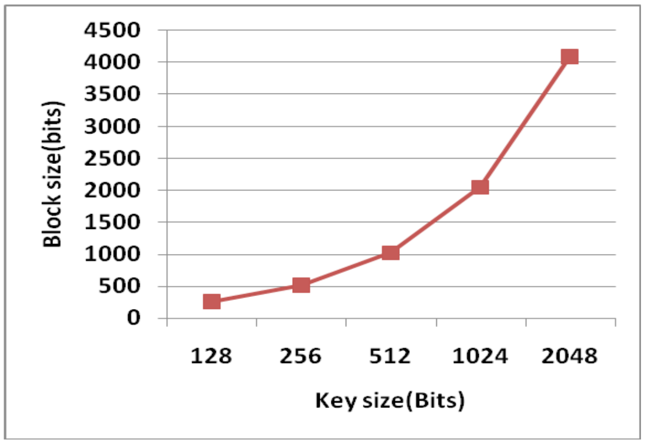

Figure 4. Comparison of key size with block size.

In Table 2, the decrypted value is not equivalent to the block value of the plain text. The value produced by the encryption is unable to produce the exact plain text by the decryption process.

While applying the 64 bits block produces the exact plain text as in Table 3. However, it could not be done where the block value exceeds the modular value. Hence it is proved that the result of encryption and decryption process depends on the values of block, key and modular. It is also exhibits that the 64 bits block also is not adequate to produce the desired output.

At the end of the analysis, it is revealed that 63 bits block size is ideal for any different key values of 32 bit keys. For any change in key size, appropriate block size can be obtained by applying the eq (1)

Table 4. Comparison of Key size and block size

Key Size(Bits)	Block Size(bits)
128	255
256	511
512	1023
1024	2047
2048	4095

Encryption function is: $E(M) = M^{2865670243} \pmod{15447624058121753573}$

Decryption function is: $D(M) = M^{8316262398734230927} \pmod{15447624058121753573}$

$1 \leq \text{Block size}(M) \leq (2^*\text{key size})-1$

The key size also affects the execution time for encryption and decryption process. The number of blocks depends on the file size. The execution time is also increased whenever the number of blocks increases. Hence it is derived that the block size is less than twice the value of key size.

$$\text{Block size}=(2^*\text{key size})-1 \quad (1)$$

The performance enhancement is achieved only by decomposing the plain text into blocks. The block size is decided by the key size. The block size is directly propositional to the key size as illustrated in Table 4 and Figure 4.

In the existing system Short Range Natural Number[12] (SRNN), though the security increases by increasing the modulus, it results in the decrement of speed. It is suggested that the increment of block size of original message 'M' increases both security and speed. It is also recommended to fix the block size depend on modulus value.

The results of the proposed methodology are proved by the comparison with the existing methodology. The speed of the process is analyzed by using the same file with different size of blocks.

Block size is to be fixed in two ways

i. Size of modulus value as per the existing system.

ii. Size of key as per the proposed methodology.

The comparison is done with 10 KB of file size with different key sizes. The results are tabulated in the Table 5.

Table 5. Comparison of Execution time with file size (10 KB) between SRNN and MsgEncA

KeySize (Bits)	Modulus Size (Bits)	Block size(Bits)		No of blocks		Total Time(ms)	
		SRNN	MsgEncA	SRNN	MsgEncA	SRNN	MsgEncA
128	256	128	255	651	327	522	250
256	512	256	511	325	163	1617	781
512	1024	512	1023	163	82	5503	2671
1024	2048	1024	2047	81	41	20316	9875
2048	4096	2048	4095	41	21	80078	38750

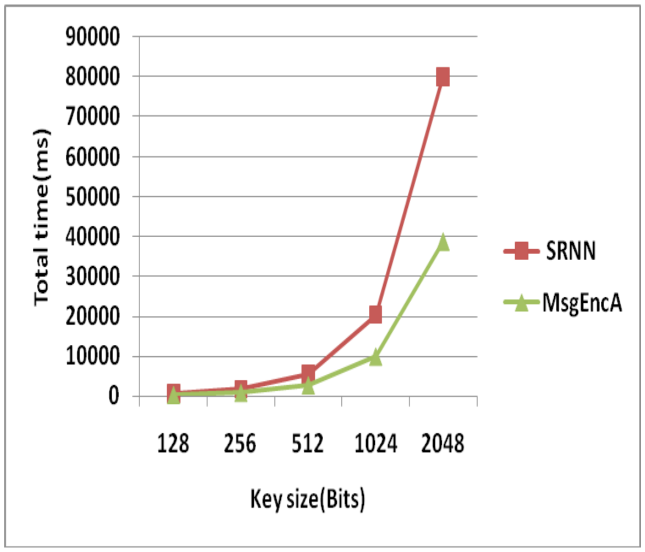

Figure 5. Comparison of execution speed between existing method and MsgEncA.

In the results of Table 5, the execution speed purely depends on the block size. It is substantiated that the time taken for the cryptosystem drastically reduced by using MsgEncA. The graphical representation of the above is given in Figure 5. An identical comparison is also done with increased file size of 20 KB with different key sizes. Finally, it proves that the MsgEncA provides better performance than SRNN.

In the analysis from the entire above tables, the performance of the public key cryptosystem is enhanced irrespective of the key size and file size. It is obtained by applying the appropriate block size.

In the aspect of security, the key size plays an important role. It is not possible to compromise the key size. Even the selection of higher key bits is not affecting the execution speed. The MsgEncA provides better performance than the existing method.

5. Conclusion

In public key cryptography, the algorithms are highly secured. Usage of separate keys with higher key size is the main reason for security. However it increases the execution speed needed for cryptosystem. On applying the proposed algorithm MsgEncA, the speed of execution is increased without changing the key size. It is proved by comparison with existing method SRNN. The block ciphers increase the security and as well as the speed. Whenever the block size increases, the execution speed is also increased. At one stage, the block size is not suitable for encryption and decryption process. As a result, the block size is fixed to less than the twice of key size. Thus, the proposed algorithm helps to decompose the plain text into specific fixed size and also raises the security and processing speed. It is also evidenced that the proposed algorithm can be applied in all the algorithms currently available in public key cryptography.

6. References

1. Kahate A. Cryptography and Network Security. 2nd ed. New Delhi: Tata McGraw Hill publishing company limited; 2003.

2. Vincent PMDR, Iqbal SA, Bhagat K, Kushwaha KK. Cryptography: A mathematical approach. Indian Journal of Science and Technology. 2013; 6(12):5607–11.
3. Jose JJR, Raj GDP. PACMATS: An Adaptive Symmetric Block Cipher for Parallel Computing Environments. Indian Journal of Science and Technology. 2014; 7(S4):99–105.
4. Kulkarni G, Gujar P, Joshi M, Jadhav S. Message security using Armstrong numbers and authentication using colors. IJARCSSE. 2014; 4(1).
5. Nag A, Jain VK. The hardware implementation of improved RSA algorithm. IJSCE. 2014; 4(4).
6. Javheri S, Kulkarni R. Secure data communication and cryptography based on DNA based Message Encoding. IJCA. 2014; 98(16):35–40.
7. Amin E, Abd Elbadea A. Building cryptosystem based on genetic algorithm and pattern recognition concepts for academic institution. International Journal of Advanced Research in Computer Science and Software Engineering. 2014; 4(10).
8. Goshwe NY. Data encryption and decryption using RSA Algorithm in a Network Environment. International Journal of Computer Science and Network Security. 2013; 13(7).
9. Hingmire A. Data Encryption/Decryption process using PSZ methodology and performance Analysis with RSA. International Journal of Engineering Research and Applications (IJERA); National Conference on Emerging Trends in Engineering & Technology (VNCET-30 Mar'12).
10. Gupta V, Singh G, Gupta R. Advance cryptography algorithm for improving data security. International Journal of Advanced Research in Computer Science and Software Engineering. 2012; 2(1).
11. Khatri M, Shukla N. Block cipher principles on cryptography. IJCTEE. 2013; 2(5).
12. Sharma S, Yadav JS, Sharma P. Modified RSA public key cryptosystem using short range natural number algorithm. International Journal of Advanced Research in Computer Science and Software Engineering. 2012; 2(8).