PERFORMANCE EVALUATION OF MODIFIED MODULAR EXPONENTIATION FOR RSA ALGORITHM

S. Satheeshkumar[1], S. Nithin Padmakumar[2], S. Sivakumar[3]

[1]*Research Scholar, Department of Computer Science, C.P.A.College, Bodinayakanur,* ***satheeshvasan007@gmail.com***

[2] *Software Engineer, Accenture, Pune,* ***nithinpadmakumar.s@gmail.com***

[3]*Associate Professor and Head, Department of Computer Science, C.P.A.College, Bodinayakanur,* ***sivaku2002@yahoo.com***

Abstract

Authentication is a very important application of public-key cryptography. Cryptographic algorithms make use of secret keys known to send and receive information. When the keys are known the encryption / decryption process is an easy task, however decryption will be impossible without knowing the correct key. The shared public key is managed by the sender, to produce a message authentication code (MAC) for every transmitted message. There are many algorithms to enable security for message authentication (secret key). RSA is one such best algorithm for public key based message authentication approaches. But it takes more time for encryption and/or decryption process, when it has large key length. This research work evaluates the performance of RSA algorithm with modified modular exponentiation technique for message authentication. As a result modified modular exponent based RSA algorithm reduces execution time for encryption and decryption process.

Key Words: *Cryptography, Message authentication, RSA, Modular Exponentiation.*

---***---

1. INTRODUCTION

Cryptography is frequently utilized information security techniques to protect sensitive information from unauthorized person during data transmission. Many authentication schemes offer message authenticity and integrity verification for distributed or cloud computing system. Message authentication divided into broad two categories: asymmetric-key (public key) cryptosystem and symmetric-key cryptosystem. The asymmetric-key based cryptosystem necessitates composite key management and does not have scalability. It is not flexible between peers of network due to hacker's attack, since the message sender send a secret key to and the receiver. The shared key is managed by the sender, to secure a message authentication code (MAC) for every transmitted message. The essential of such method is security of sending the secret key to the receiver, which should be protected from intrusion of cryptanalyst.

Public key cryptosystem provided a mean to avoid sending secret keys over communication channels, i.e. reducing the risk of key compromise [1]. Many algorithms are introduced based on the public key cipher; the most widely known and used of them is the RSA crypto-algorithm is suitable for privacy and authentication [2-6]. RSA crypto-algorithm is based on performing exponentiations in modular arithmetic both for encryption and decryption [2]. Encrypt Assistant Multi-Power RSA was proposed to enhance RSA decryption performance [10].

RSA makes use of the modular exponent (ME) or modular multiplication (MM) algorithm. Many research works in order to speed up the performance of MM or ME algorithm [7-9]. ME technique has made RSA method very attractive for data security and authentication. However, the main drawback of the RSA cryptosystem lies in the slow computation of encryption / decryption operations and high time complexity [11-13]. The strength of RSA algorithm and a survey of fast exponentiation method are explored [14-16]. The efficiency of RSA encryption and decryption is primarily depends on the efficiency of the ME algorithm [17]. An efficient implementation of RSA and the RSA algorithms and other related cryptography issues are reviewed in [18].

The main emphasis of this paper is to improve the efficiency of encryption and decryption process of RSA algorithm. This is achieved by having modified modular exponent based RSA algorithm; it enables encryption and decryption process faster than the original RSA. Rest of this paper is organized as follows: section 2, MAC is described and section 3 describes ME. Section 4 describes MME. The modified RSA (MRSA) technique is presented in section 5. At last conclusion is given in section 6.

2. MESSAGE AUTHENTICATION CODE

In cryptography, a MAC is a small piece of information that is helpful to authenticate a valuable message and it provide authenticity assurances on the message. The MAC function avoid from plaintext- attacks. The MAC can be developed from other cryptosystem primitives. RSA is a well known first Asymmetric cryptography and it very suitable for MAC.

RSA is made of the initial letters of the surnames of Ron Rivest, Adi Shamir, and Leonard Adleman, who first publicly described the algorithm in 1977. RSA is generally used as a public-key cryptosystem which is based on modular exponentiation. In RSA algorithm there are two keys used to encryption (public key) and decryption (private key). The public key is advertised to the receiver by sender and the private key should be kept mystery. In this manner an unknown person can't decrypt the encrypted message without private key. The working technique of RSA algorithm discussed in below:

Key Generation:

The RSA (encryption and decryption) keys are generated by the following steps

i. Choose two random prime numbers(p, q)
ii. Calculate $n = p \times q$
iii. Calculate $\varphi(n) = (p-1) \times (q-1)$
iv. Select an integer 'e' such that $1 < e < \varphi(n)$ and $GCD(e, \varphi(n)) = 1$.
v. Calculate $d = e^{-1} \bmod \varphi(n)$

The public key is 'e' and the private key is 'd'. Message authentication is performed by using these keys.

Process of Encryption and Decryption

Suppose user 'A' want to share secret message (M) to user 'B'

- User 'B' generate public and private keys
- User 'A' got public key from 'B' through any public source
- User 'A' encrypt a message and send it to user 'B'
- User 'B' decrypt a message using private key
- Finally user 'B' got a original message(M).

The security of RSA based on the length of the key, longer the key-length more safer for the data. When the key length is longer, then the key generation, encryption and decryption takes long. RSA includes Euclidean technique to compute 'd' value and ME technique is used to perform encryption and decryption work, ME leads to consume much time while execute RSA. In this proposed system asymmetric key RSA algorithm use to encrypt/decrypt a message and enhance it.

3. MODULAR EXPONENTIATION(ME)

ME is a general function used in many public key cryptosystem. ME works based on the modular multiplication. The ME method is used in many crypto algorithms. For instance, the Diffe-Hellman key exchange algorithm need ME operation [19]. Moreover, Digital Signature Standard (DSS) of the National Institute for Standards and Technology (NIST) [21] and the ElGamal signature algorithm[20] also need for the computation of ME. In RSA algorithm, encryption and decryption is set up by modular exponentiation.

ME calculates the remainder when a positive integer 'm' (the base) rose to the e^{th} power (the exponent) m^e, is divided by a positive integer 'n', called the modulus. The ME 'c' is: c= m^e (mod n), the pseudo code for the computation of ME is,

```
read e, m, n
  for k = 1 To e
    calculate c=(c*m) mod  n;
  end
print c;
```

4. MODIFIED MODULAR EXPONENTIATION

Fast ME algorithms were usually considered in public key cryptosystem. Recording the common parts of modular exponent in the folded sub strings could enhance the efficiency of the binary modulo algorithm. Here in this paper we are comparing the ME results of RSA and modified modular exponentiation (MME) results of MRSA work as effectively and it reduces the computational time.

The working procedure of MME technique is given in the flow chart shown in Fig.1.

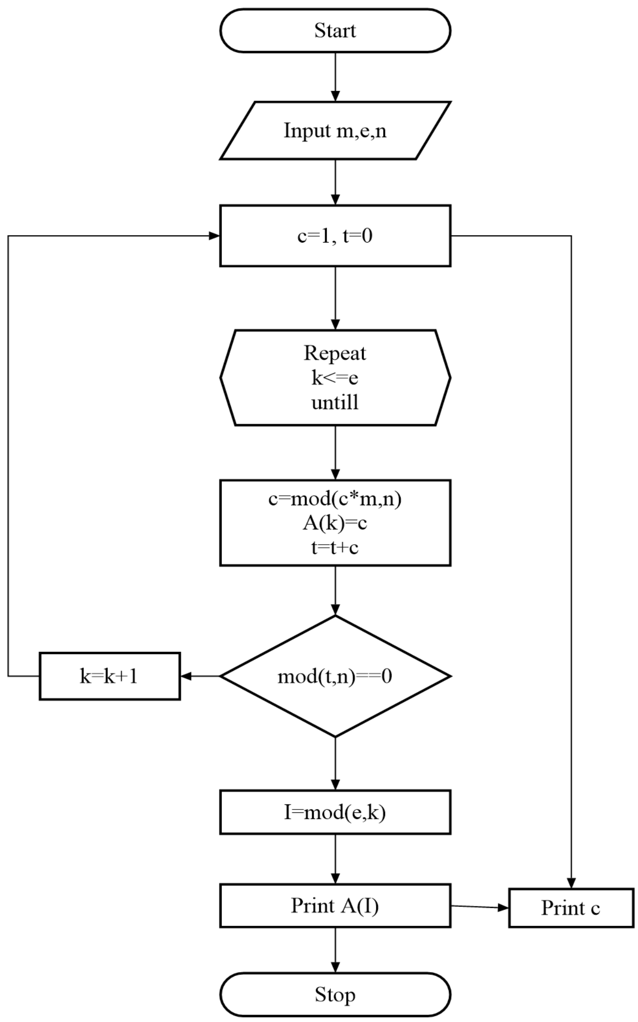

Figure 1: Flow Chart for MME

The Pseudo code for MME technique is

```
Read e,m,n
Initialize c=1, t=0, I=0 and A[] as array
For k=1 to e
Calculate c = (c*m)%n , A[k] = c, t = t+c
If (t%n) equal to zero then
Calculate I = ( e % k )
Print A[I]
Jump from condition and loop
Else if k equal to e then
Print c
End if
End of loop
```

Normally ME computation are based on residuum. To compute and analyze ME, it may have repeated answers. The computed values for 4^e(mod 7) with different exponents 'e' are tabulated in Tab. 1.

Table 1: Repetition in 4^e (mod 7) computation

e	1	2	3	4	5	6	7	8	9
4^e(mod 7)	4	2	1	4	2	1	4	2	1

In Tab.1 the result repetition (4 2 1, 4 2 1, 4 2 1) for ME occurs after 3^{rd} 'e' value computation similarly for next 3^{rd} 'e' value it continues.

From Tab.1 it is found that the same computation for (4^e(mod 7)) with 'e' value increases from 1 to 9. When we analyze this calculation, repeated result comes again and again. Hence we stop computing ME when 'e' value is at 3 and avoid other repeated calculation (e value 4 to 9). This modification with the ME is most useful to increase the execution speed of encryption and decryption process of RSA, thus reduces the time complexity. The advantage of MME 4^3(mod 7) and $4^{(>3)}$ (mod 7) both calculation take same execution time or little difference time, not high difference execution time. Commonly ME repeated results are linear not static. In Tab. 1 repetition start, when e value is 3, these repetitions differ from one problem to other problem.

The MME method is used for RSA with the following steps.

Step 1:

Sum all residuums and check whether it is divisible by 'n'.

Table 2: Computation step to find repetition

Iteration	Remainder	Computation
1	c_1	$\left[\left(\sum_{e=1}^{1} c_e\right)\%n\right] \neq 0$
2	c_2	$\left[\left(\sum_{e=1}^{2} c_e\right)\%n\right] \neq 0$
3	c_3	$\left[\left(\sum_{e=1}^{3} c_e\right)\%n\right] \neq 0$
…	…	…
k	c_k	$\left[\left(\sum_{e=1}^{k} c_e\right)\%n\right] = 0$

Sum of residuum $\left(\sum_{e=1}^{k} c_e\right)$ is divided by 'n', then stop this process and let final iteration value is k.

Step 2:

Next, we know that a value of k (Final Iteration value) and e (exponent value). And get an answer from table, using following condition

$$if\ (e\%k)\ value\ is \begin{cases} \neq 0, & (e\%k)^{th}\ Iteration\ value\ is\ answer \\ = 0, & k^{th}\ Iteration\ value\ is\ answer \end{cases}$$

A Working Example

From Tab. 1, we observed that the repetition (4 2 1, 4 2 1, 4 2 1) in ME residuum. Similar to that we analyze the problem 4^9 (mod 7) for computation with e=9, m=4 and n=7.

Step 1:

Check the sum of residuums, it is divisible by 7

Table-3: Computational results for MME

Iteration	Remainder (c_k)	Computation
1	4	$4 \% 7 \neq 0$
2	2	$6 \% 7 \neq 0$
3	1	$7 \% 7 = 0$

In 3^{rd} iteration, the sum of residuum (4 + 2 + 1) is divisible by 7. So, halt this process and let k=3.

Step 2:

k=3, e=9, (9%3 = 0). So 'c_3'' iteration value 1 is an answer. Suppose we take e value is 5,

(5%3 = 2) then 'c_2'' iteration value 2 is an answer.

Compute ME values of these two algorithms with different values of 'e' for a value for n=1321 and m=237. The computed results of ME and MME techniques are tabulated in Tab. 4.

Table 4: Comparison of ME and MME result values

'e' in sec.	4096	8192	16384	32768	65536
ME	0.000903	0.001784	0.003247	0.007709	0.014683
MME	0.000756	0.000292	0.000307	0.000330	0.000382

The 'e' value increases dynamically whereas 237^e (mod 1321) is static. From Tab. 4 we observe that MME got a predominant improvement. When 'e' value is very high, like 8192, 16384, 32768, 65536, MME performs better than ME. Because Modular algorithms work recursively 1 to 32768(till). When 'e' value is larger, the execution time is high. The MME technique stop repetition at its initial level. MME results almost a very close execution time for higher 'e' values. So we avoid the additional recursive repetition, for higher 'e' values. Hence MME technique for RSA is suitable to perform cryptographic operation like message authentication. Visualization of computation results of ME and MME techniques execution time is shown in Fig. 2.

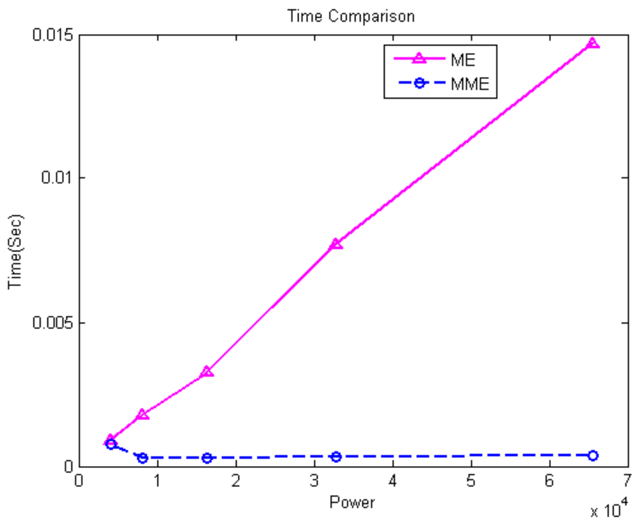

Figure 2: Comparative execution time of ME and MME

5. MODIFIED RSA

Basically RSA encryption and decryption based on ME algorithm, when we modify ME then, RSA encryption and decryption time also modified. MRSA algorithm will reduce execution time of encryption and decryption. The working procedure of MRSA is shown in flow diagram Fig.3.

The MRSA algorithm provides more security and speed in encryption and decryption process. Hence more security for message authentication is possible with MRSA method. This MME algorithm was developed in MATLAB.

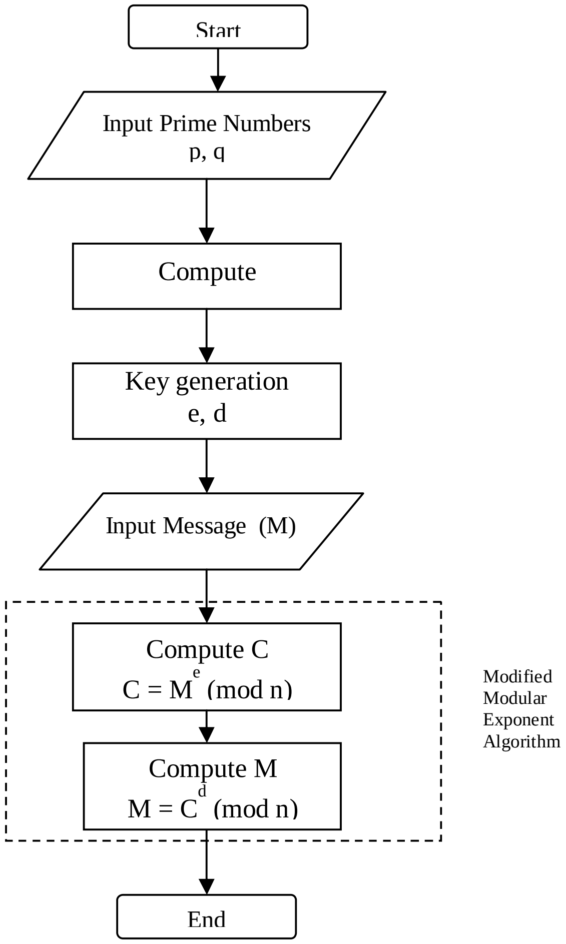

Figure-3: Flowchart for modified RSA Algorithm

A Working Example

Encryption:

p = 11; q = 31; n = 341; e = 13; d = 277; **m = 128**

Step 1:

Check the sum of residuums divisible by 341

Table-5: Computational Procedure for encryption using MRSA

Iteration(k)	Remainder (c_k)	$\left[\left(\sum_{e=1}^{k} c_e\right)\%n\right]=0$
1	128	No
2	16	No
3	**2**	No
4	256	No
5	32	No
6	4	No
7	171	No
8	64	No
9	8	No
10	1	**Yes**

Step 2:

From Tab.5, we know the value for ‘k’, k=10, e=13, (13%10 = 3). So iteration ‘c_3‘ value 2 is cipher text value (c).

Experimental studies were done on RSA and MRSA algorithms for various MAC sizes 8, 12, 16 and 32 bits and the execution time is given in Tab. 6.

Table-6: Execution time of RSA and MRSA for various - MAC

MAC Code Size (bits)	Algorithm	Encryption Time(sec)	Decryption Time(sec)
8	RSA	0.004290	0.013258
	MRSA	0.003058	0.003022
12	RSA	0.044046	0.083677
	MRSA	0.022548	0.022509
16	RSA	0.185550	0.450847
	MRSA	0.103741	0.264084
24	RSA	15.734496	19.214687
	MRSA	5.417496	10.666174
32	RSA	34.125403	39.201578
	MRSA	23.458021	27.501731

Visualization of the comparative performance of RSA and MRSA algorithms with execution of time of encryption and decryption process are indicated in Fig. 4 and Fig. 5. For 8 bits, the execution time variation is narrow for encryption and linear in decryption. The computation time of MRSA is better than RSA for 8, 16, 24 and 32 bits of MAC. The same input was applied to check the speed of both (RSA & MRSA) algorithm. In RSA algorithm, the execution time based on the MAC size. Suppose MAC size increases, the execution time was automatically increased based on MAC. In MRSA algorithm it reduces computation time, when MAC size increases.

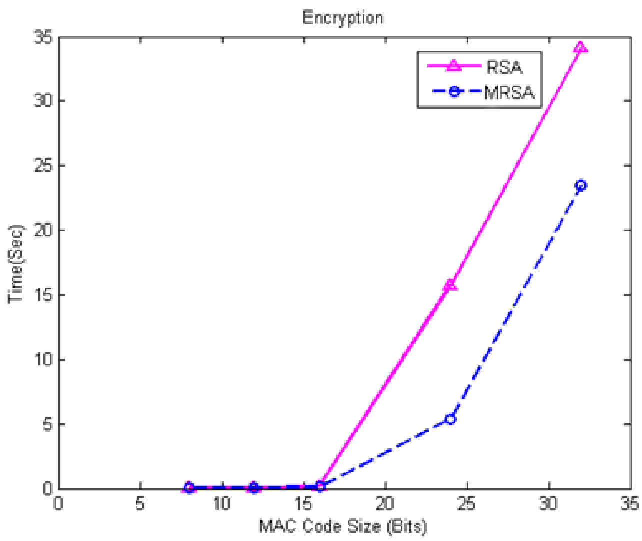

Figure 4: Execution Time for Encryption

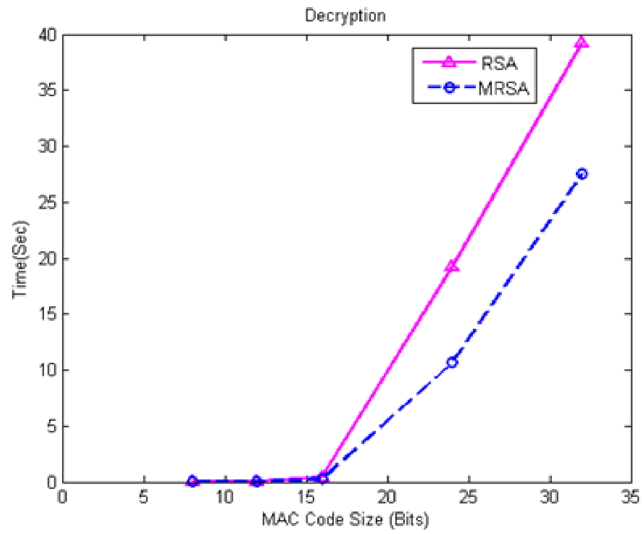

Figure 5: Execution Time for Decryption

6. CONCLUSION

Results of ME and MME techniques from working example using various MAC sizes are presented. The encryption and decryption speed of MRSA algorithm is fast as compared to RSA algorithm. The encryption and decryption execution time consumed by MRSA algorithm is least as compared to RSA algorithm. Security is essential for successful data transmission. The MRSA algorithm is of great use for secure data transmission. Investigation of the effect of MRSA can be used with many techniques like MRSA & DES, MRSA & AES and MRSA & Diffie Hellman by combining cryptography algorithms to improve security in progress.

REFERENCES

[1] Diffie W. and Hellman M.E., “New Direction in Cryptography,” *IEEE Trans. Information Theory*, vol. IT-22, no. 6, pp. 644-654, 1976.

[2] Rivest, R., Shamir A., and Adleman L., “Method for obtaining Digital Signature and Public Key Cryptosystem,” *Comm. of ACM*, vol. 21, no. 2, 1978.

[3] Mohan S. B., “Fast Algorithm for Implementing RSA Public-Key Cryptosystem,” *Electronic Letters*, vol. 21, no. 17, 1985.

[4] Selby A., “Algorithms for Software Implementation of RSA,” vol. 136, no. 3, 1989.

[5] Davis D. W., *Security for Computer Networks*, Wiley, UK, 1989.

[6] Saloma A., *Public-Key Cryptography*, Springer, 1996.

[7] C.K.Koc and C.Y.Hung, “Adaptive m-ary segmentation and canonical recoding algorithms for multiplication of large binary numbers,” Computer mathematic application, vol.24, no.3, pp.3-12, 1992.

[8] A.F.Tenca and C.K.Koc, “A scalable architecture for modular multiplication based on Montgomery’s algorithm,”IEEE Trans. On computer, vol.52, no.9, pp. 1215-1221, 2003.

[9] P. Keshavarzi and C. Harrison, “A new modular multiplication algorithm for VLSI implementation of

public-key cryptography," Proceedings of First International Symposium on Communication Systems and Digital Signal Processin, pp.516-519, 1998

[10] Qing Liu, Yunfei Li, Lin Hao, "On the Design and implementation of an Efficient RSA Variant", Advanced Computer Theory and Engineering (ICACTE), 2010, pp.533-536.

[11] K. Sakiyama, L. Batina, B. Preneel and I. Verbauwhede, " High-performance publik-key cryptoprocessor for wireless mobile applications," Mobile networks and applications, vol. 99, pp. 245-258, 2007.

[12] A. Rezai and P. Keshavarzi, "Improvement of highspeed modular exponentiation algorithm by optimum using smart methods, " Proceedings of 18th Iranian Conference on Electrical Engineering, Iran, pp.2104-2109, May 2010.

[13] A. Rezai and P. Keshavarzi, "Speed Improvement in elliptic curve cryptosystem scalar multiplication algorithm," proceedings of 7th International ISC Conference on Information Security and Cryptology2010, Iran, pp.181-188, September 2010.

[14] Daniel M. Gordon, *A survey of fast exponentiation methods*, Journal of algorithms, 27, 1998, 126-146.

[15] Giuseppe Ateniese, Michael Steiner, and Gene Tsudik, *New multiparty authentication services and key agreement protocols*, IEEE Journal of Selected Areas in Communication, 18(4), 2000.

[16] Cetin Kaya Koc, *High speed RSA implementation*, RSA Laboratories, CA, 1994.

[17] Anand Krishnamurthy, Yiyan Tang, Cathy Xu and Yuke Wang, *An efficient implementation of multi-prime RSA on DSP processor*, University of Texas, Texas, USA,2002.

[18] A. Menezes, P. Van Oorschot, S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996 (www.cacr.math.uwaterloo.ca/hac)

[19] W. Di_e and M. E. Hellman. New directions in cryptography. IEEE Transactions on Information Theory, 22:644-654, November 1976.

[20] T. ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. IEEE Transactions on Information Theory, 31(4):469-472, July 1985.

[21] National Institute for Standards and Technology. Digital signature standard (DSS). Federal Register, 56:169, August 1991.

BIOGRAPHIES

Mr. S. Satheeshkumar received his B.Sc. in Computer Science and M.Sc. in Computer Science & Information Technology and M.Phil. in Computer Science from Cardamom Planters' Association College, Bodinayakanur. He actively involves in research activities which is related to Network Security.

Mr. S. Nithin Padmakumar, completed B.E. Computer Science Engineering in 2014 from Mepco Schlenk Engineering college, Sivakasi, Tamilnadu, India. He is working as Software Engineer in Accenture Services Ltd., Pune and his role is Liferay Portal Developer- Java Liferay. He is an active young researcher presented many research papers.

Dr. S. Sivakumar, completed B.E Computer Engineering degree in 1988 from the National Engineering College of the Madurai Kamaraj University, Madurai, India, received M.S Software systems post graduate degree in 1995 from Birla Institute of Technology and Science, Pilani, Rajasthan, India, awarded Ph.D Computer Science in 2010 from Madurai Kamaraj University, Madurai, India. Since 1989, he has been working as Associate Professor and Head of the department of Computer Science in Cardamom Planters' Association College, Bodinayakanur, India. He teaches undergraduate and postgraduate level classes in computer science and information technology. His research interests include bio medical image processing, data mining and cloud computing.