Modified RSA Encryption Algorithm (MREA)

Ravi Shankar Dhakar
Sr. Lecturer, GIET,
Kota, Rajasthan, India
ravi.dhakar83@gmail.com

Amit Kumar Gupta
Asst. Prof. , SBCET,
Jaipur, Rajasthan, India
cseprof_amit@*rediffmail.com*

Prashant Sharma
Lecturer, MIT,
Kota, Rajasthan, India
Prashant_harmony@*rediffmail.com*

Abstract
In asymmetric key cryptography, also called Public Key cryptography, two different keys (which forms a key pair) are used. One key is used for encryption & only the other corresponding key must be used for decryption. No other key can decrypt the message, not even the original (i.e. the first) key used for encryption. The beauty of this scheme is that every communicating party needs just a key pair for communicating with any number of other communicating parties. Once some one obtains a key pair, he /she can communicate with any one else. RSA is a well known public-key cryptography algorithm. It is the first algorithm known to be suitable for signing as well as encryption, and was one of the first great advances in public key cryptography. The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring large numbers know mathematical attack and the problem of trying all possible private keys know brute force attack. So to improve the security, this scheme presents a new cryptography algorithm based on additive homomorphic properties called *Modified RSA Encryption Algorithm* (MREA). MREA is secure as compared to RSA as it is based on the factoring problem as well as decisional composite residuosity assumptions which is the intractability hypothesis. The scheme is an additive homomorphic cryptosystem; this means that, given only the public-key and the encryption of m_1 and m_2, one can compute the encryption of $m_1 + m_2$. This scheme also presents comparison between RSA and MREA cryptosystems in terms of security and performance.

Index Terms: Encryption, Public key, Private key, Security, RSA, Homomorphic

I. INTRODUCTION

To solve the problem of secure key management of Symmetric key cryptography, Diffie and Hellman introduced a new approach to cryptography and, in effect, challenged cryptologists to come up with a cryptographic algorithm that met the requirements for public-key systems. Public key cryptography uses a pair of related keys, one for encryption and other for decryption. One key, which is called the private key, is kept secret and other one known as public key is disclosed and this eliminate the need for the sender and the receiver to share secret key. The only requirement is that public keys are associated with the users in a trusted (authenticated) [8]manner through a public key infrastructure (PKI). The public key cryptosystems are the most popular, due to both confidentiality and authentication facilities [1].The message is encrypted with public key and can only be decrypted by using the private key. So, the encrypted message cannot be decrypted by anyone who knows only the public key and thus secure communication is possible. In a public-key cryptosystem, the private key is always linked mathematically to the public key. Therefore, it is always possible to attack a public-key system by deriving the private key from the public key. The defense against this is to make the problem of deriving the private key from the public key as difficult as possible. Some public-key cryptosystems are designed such that deriving the private key from the public key requires the attacker to factor a large number. The *RSA* and *MREA* public key cryptosystems [3] are the best known examples of such a system. This paper presents a hybrid cryptography algorithm which is based on the additive homomorphic properties called a *Modified RSA Encryption Algorithm(MREA)*.

A. Homomorphic Properties

A notable feature of the MREA cryptosystem is its homomorphic properties. As the encryption function is additively homomorphic, the following identities can be described:

i. Homomorphic addition of plaintexts

The product of two ciphertexts will decrypt to the sum of their corresponding plaintexts,

$$D(E(m_1,r_1) \cdot E(m_2,r_2) \bmod n^2) = m_1 + m_2 \bmod n.$$

The product of a ciphertext with a plaintext raising g will decrypt to the sum of the corresponding plaintexts,

$$D(E(m_1,r_1) \cdot g^{m2} \bmod n^2) = m_1 + m_2 \bmod n.$$

ii. Homomorphic multiplication of plaintexts

An encrypted plaintext raised to the power of another plaintext will decrypt to the product of the two plaintexts,

$$D(E(m_1,r_1)^{m2} \bmod n^2) = m_1 m_2 \bmod n,$$
$$D(E(m_2,r_2)^{m1} \bmod n^2) = m_1 m_2 \bmod n.$$

More generally, an encrypted plaintext raised to a constant *k* will decrypt to the product of the plaintext and the constant,

$$D(E(m_1,r_1)^k \bmod n^2) = km_1 \bmod n$$

However, given the MREA encryptions of two messages there is no known way to compute an encryption of the product of these messages without knowing the private key.

978-0-7695-4640-7/12 $26.00 © 2012 IEEE
DOI 10.1109/ACCT.2012.74

II. RSA CRYPTOSYSTEM

RSA is based on the principle that some mathematical operations are easier to do in one direction but the inverse is very difficult without some additional information. In case of RSA, the idea is that it is relatively easy to multiply but much more difficult to factor. Multiplication can be computed in polynomial time where as factoring time can grow exponentially proportional to the size of the number.

Key Generation Process:

i. Select two prime numbers p and q.
ii. Find n=p*q, Where n is the modulus that is made public. The length of n is
iii. considered as the RSA key length.
iv. Choose a random number 'e' as a public key in the range 0<e<(p-1)(q-1) such that
v. gcd(e,(p-1)(q-1))=1.
vi. Find private key d such that ed≡1(mod (p-1)(q-1)).

Encryption

i. Consider the device A that needs to send a message to B securely.
ii. Let e be B's public key. Since e is public, A has access to e.
iii. To encrypt the message M, represent the message as an integer in the range 0<M<n.
iv. Cipher text C = M^e mod n, where n is the modulus.

Decryption

i. Let C be the cipher text received from A.
ii. Calculate Message M = C^e mod n, where d is B's private key and n is the modulus.

III. OUR SCHEME(MREA)

MREA is an asymmetric-key cryptosystem, meaning that for communication, two keys are required: a public key and a private key. Furthermore, unlike RSA, it is one-way, the public key is used only for encryption, and the private key is used only for decryption. Thus it is unusable for authentication by cryptographic signing. Following is a key generation algorithm for MREA cryptosystem.

A. Key Generation Algorithm:

i. Choose four large prime numbers p, q, r and s randomly and independently of each other. All primes should be of equivalent length.
ii. Compute n = p x q, m= r x s, φ =(p-1) x (q-1) and λ =(r-1) x (s-1).
iii. Choose an integer *e*, 1 < e < φ, such that gcd (e, φ) = 1.
iv. Compute the secret exponent *d*, 1 < d < φ, such that e x d mod φ=1.
v. Select an integer g=m+1.
vi. Compute the modular multiplicative inverse: μ = λ^{-1} mod m.
vii. The public (encryption) key is (n, m, g, e).
viii. The private (decryption) key is (d, λ, μ).

B. Encryption:

i. Let m be a message to be encrypted where 0< mesg < n.
ii. Select random r where r < m.
iii. Compute ciphertext as: $c=g^{mesg\hat{}e \bmod n} \times r^m \bmod m^2$.

C. Decryption

i. Compute message: m = $(((c^{\lambda} \bmod m^2 - 1)/ m) \times \mu \bmod m)^d \bmod n$.

IV. COMPARISON OF RSA AND MREA ALGORITHM

A. Simulation Results The simulation result of the algorithm MREA, implemented in JAVA [4], running on a 2.20 GHz Dual Core Processor and 1 GB RAM has used a1000 characters long message for encryption/decryption. The algorithms (RSA & MREA) have many important parameters affecting its level of security and speed [2]. By increasing the modulus length it is caused of increasing the complexity of decomposing it into its factors. This also increases the length of private key and hence difficulty to detect the key. Another parameter is modular multiplicative inverse μ. Where the modular multiplicative inverse μ is new factor of private key, so it will be more difficult to choose μ by trying all possible private keys (brute force attack) hence the security also increases as well as difficulty of detecting the private key. The RSA and MREA parameters are changed one parameter at a time and the others are kept fixed to study the relative importance. Table 4.1 shows the simulation results of both the algorithms.

Table 4.1: Effect of changing the modulus length m, n and chunk size on the size of two different Private keys, Key generation time, Encryption time and Decryption time MREA cryptosystem, while the size of Public key has kept constant (256 bits, 512 bits).

Size of n,m, d and random no. (bits)	Public key size e (bits)	Chunk Size (bits)	RSA			
			Key Generation time (ms)	Encryption time (ms)	Decryption time (ms)	Total Execution time(ms)
256	256	128	469	109	62	**640**
512	256	128	140	188	218	**546**
512	256	256	140	109	141	**390**
1024	256	128	469	484	1453	**2406**
1024	256	256	469	281	735	**1485**
1024	256	512	469	172	375	**1016**
2048	512	128	2453	2953	15203	**20609**
2048	512	256	2453	1547	5609	**9609**
2048	512	512	2453	812	2750	**6015**
2048	512	1024	2453	515	1375	**4343**

Table 4.2: Effect of changing the modulus length m, n and chunk size on the size of two different Private keys, Key generation time, Encryption time and Decryption time RSA cryptosystem, while the size of Public key has kept constant (256 bits, 512 bits).

Size of n,m, d and random no. (bits)	Public key size (bits)	Chunk Size (bits)	MREA			
			key generation time (ms)	Encryption time (ms)	Decryption time (ms)	Total Execution Time (ms)
256	256	128	484	329	156	**969**
512	256	128	172	1672	968	**2812**
512	256	256	172	890	485	**1547**
1024	256	128	625	11625	6938	**19188**
1024	256	256	625	5860	3515	**10000**
1024	256	512	625	2969	1797	**5391**
2048	512	128	8125	99891	53609	**161625**
2048	512	256	8125	47157	31797	**87079**
2048	512	512	8125	22094	13515	**43734**
2048	512	1024	8125	11219	7016	**26360**

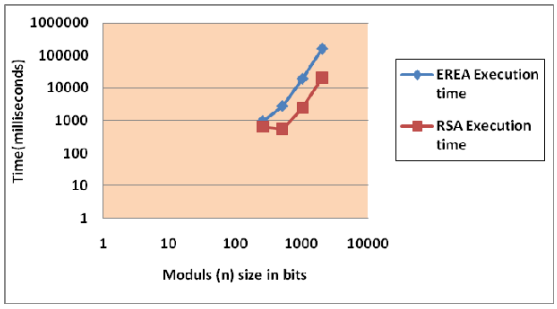

Fig.4.1: Modulus size v/s RSA & MREA algorithm's execution time.

B. Changing the modulus length: Changing the modulus affects the other parameters of the algorithms as shown in Table 4.1. It is clear here that increasing the modulus length (bits) increases the key generation time and encryption\decryption time so the time complexity is increased . Moreover, the length of the secret key (d) increases at the same rate modular multiplicative inverse increases. As a result, increasing the n-bit length provides more security. Hence increasing the n- bit length increases the security but decreases the speed of encryption, decryption and key generation process as illustrated by Figure 4.1 and 4.2.

C. Changing the chunk size: On the basis of simulation results of Table 4.1 & 4.2, following Figure 4.2 shows the effect of chunk size on encryption and decryption time of both the algorithms. Here key generation time of MREA algorithm depends on the size of chunk and as the size of chunk increases key generation time decreases , execution time of MREA algorithm also decreases. RSA algorithm's execution time doesn't depend on modular multiplicative inverse μ .

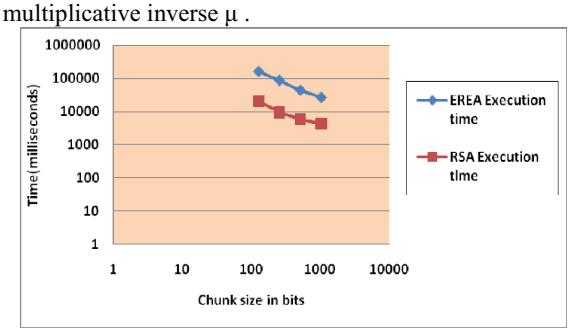

Fig.4.1: Chunk size v/s RSA & MREA algorithm's execution time, taking modulus size 2048 bits.

D. Complexity analysis of MREA cryptosystem

As regards the RSA cryptosystem, it is well-known that the computational complexities for both of the encryption and the decryption are on the order of k^3, where k is the number of bits of the modulus n [7]. The computational complexity of homomorphic problem is *O(k).* So in this way, computational complexity of MREA cryptosystem is on the order of $O(k + k^3)$ i.e. $O(k^3)$ or encryption and on the order of k^3 for decryption. So the computational complexity of MREA is equivalent to RSA cryptosystem. The simulation results of both the algorithms shows that the execution time of MREA is and about 6 times more than RSA.

E. Security analysis of MREA cryptosystem

As the MREA cryptosystem is based on additive homomorphic properties and RSA cryptosystem, additive homomorphic scheme required four prime numbers, it will be more difficult and take long time to factor dual modulus, so one have to factor the dual modulus into its four primes to break the MREA algorithm [7]. If RSA which is based on single modulus, is broken in time x and additive homomorphic based on dual modulus, is broken in time y then the time required to break MREA algorithm is $x*y$. So the security of MREA algorithm is increased as compare to RSA algorithm and it shows that the MREA algorithm is more secure for *Mathematical attacks* [8].

As in MREA double decryption is performed and unlike RSA that is not only based on private key but also based on the subset sum problem so one can't break MREA only guessing the private key only. So it shows that MREA algorithm is more secure as compare to RSA for *Brute force attack* [8].

CONCLUSION

In Public Key cryptography, two different keys are used. One key is used for encryption & the other corresponding key must be used for decryption. No other key can decrypt the message, not even the original (i.e. the first)

key used for encryption. The beauty of this scheme is that every communicating party needs just a key pair for communicating with any number of other communicating parties. It is the first algorithm known to be suitable for signing as well as encryption, The security of the RSA cryptosystem is based on two mathematical problems: the problem of factoring large numbers know mathematical attack and the problem of trying all possible private keys know brute force attack. So MREA improve the security,. MREA is secure as compared to RSA as it is based on the factoring problem as well as decisional composite residuosity assumptions which is the intractability hypothesis.

REFERENCES

[1] Adi Shamir, A Polynomial Time Algorithm for Breaking the Basic Merkle-Hellman Cryptosystem. *CRYPTO 1982*, pp279–288

[2] Allam Mousa , "Sensitivity of Changing the RSA Parameters on the Complexity and Performance of the Algorithm", *ISSN 1607 – 8926, Journal of Applied Science, Asian Network for Scientific Information*, pages 60-63,2005.

[3] Atul Kahate, "Cryptography and Network Security", *ISBN-10:0-07-064823-9*, Tata McGraw-Hill Publishing Company Limited, India, Second Edition, pages 38-62,152-165,205-240.

[4]. Neal R. Wagner, "The Laws of Cryptography with Java Code", *Technical Report, 2003*, pages 78-112.

[5] Ralph C. Merkle, Martin E. Hellman. "Hiding Information and Signatures in Trapdoor Knapsacks", *IEEE Transactions on Information Theory*, vol. IT-24, 1978, pp. 525-530.

[6] R. Rivest, A. Shamir and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Communications of the ACM*, February 1978, pages 120-126.

[7] RSA Laboratory (2009), "RSA algorithm time complexity", Retrieved from *http://www.rsa.com/rsalabs/node.asp?id=2215* (22 ct. 2009).

[8] William Stallings, "Cryptography and Network Security", *ISBN 81-7758-011-6*, Pearson Education, Third Edition, pages 42-62,121-144,253-297.