International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

Hybrid Cryptography Algorithm Based on Prime Factorization

Indiver Purohit[1], Raj Kumar Somani[2]

[1]*Research Scholar, Institute of Technology & Management, Bhilwara, Rajasthan, India*
[2]*Associate Professor, Institute of Technology & Management, Bhilwara, Rajasthan, India*

***Abstract*— The security of RSA public key cryptosystem is based on the assumption that factoring of a large number (modulus) is difficult. In RSA if one can factor modulus into its prime numbers then the private key is also detected and hence the security of the cryptosystem is broken. The Rabin cryptosystem is an asymmetric cryptographic technique, whose security, like that of RSA, is related to the difficulty of factorization. So in this paper a Rabin over RSA public key cryptosystem called *Hybrid cryptography Algorithm based on Prime Factorization* (HCAPF) is presented. This paper also presents comparison among Rabin, RSA and HCAPF cryptosystems in respect of security and performance. [3,4]**

***Keywords*—GCD: Greatest Common Divisor, HCAPF: Hybrid Cryptography Algorithm based on Prime Factorization, MRR: Modified Rabin over RSA, PKC: Public Key Cryptosystem, PKI: Public Key Infrastructure, RSA: *Rivest-Shamir-Adleman*.**

I. INTRODUCTION

Symmetric-key cryptography is based on the sender and receiver of messages knowing and using the same secret key. The sender uses the secret key to encrypt the message and the receiver uses the same secret key to decrypt it. The main problem of symmetric key cryptography is getting the sender and receiver to agree on the same secret key without anyone else knowing it. As all the keys in a symmetric key cryptosystem must remain secret, symmetric key cryptography often has difficulty providing secure key management, especially in open systems with a large number of users.

To solve this problem, 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 [5].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) 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 *Rivest-Shamir-Adleman (RSA)* and *Rabin* public key cryptosystems [12] are the best known examples of such a system [2, 6].

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.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

- Key Generation Process:

i. Generate two large random primes, *p* and *q*, of approximately equal size such that their product n = p x q is of the required bit length, e.g. 1024 bits.
ii. Compute n = p x q and φ = (p-1) x (q-1).
iii. Choose an integer *e*, satisfying 1 < e < φ, such that gcd (e, φ) = 1.
iv. Compute the secret exponent *d*, 1 < d < φ, such that e x d ≡ 1 (mod φ).
v. The public key is (n, e) and the private key is (n, d). Keep all the values d, p, q and φ secret.
   - n is known as the *modulus*.
   - e is known as the *public exponent* or *encryption exponent* or just the *exponent*.
   - d is known as the *secret exponent* or *decryption exponent*.

Public key (n, e) is published for everyone and private key (p, q, d) must be kept secret. Then by using these keys encryption, decryption, digital signing and signature verification are performed.

- Encryption Process:
  Sender A does the following: -
  - Obtains the recipient B's public key (n, e).
  - Represents the plaintext message as a positive integer *m*.
  - Computes the cipher text $c = m^e \bmod n$.
  - Sends the cipher text *c* to B.
- Decryption Process:
  Recipient B does the following: -
  - Uses private key (n, d) to compute $m = c^d \bmod n$.

Extracts the plaintext from the message representative *m*.

III. Rabin Key Cryptosystem

As the Rabin public cryptosystem [12] is an asymmetrical cryptosystem, it uses a pair of keys, one of which is used to encrypt the data in such a way that it can only be decrypted with the other key. A common process generates the keys, but they cannot be feasibly generated from each other. Following is a key generation algorithm for Rabin cryptosystem.

- Key Generation Algorithm:

The precise key-generation process is follows:
Choose two large distinct primes *r* and *s*, each equal to 3 modulo 4, to simplify the computation of square roots modulo *r* and *s* (see below). But the scheme works with any primes, and forms the product m = r*s.

- *Public key*: the number m.
- *Private key*: the numbers r and s.

To encrypt a message only the public key m is needed. To decrypt a ciphertext the factors r and s of m are necessary.

- Encryption:

Sender A does the following: -
- Obtains the recipient B's public key (m).
- Represents the plaintext message as a positive integer *mesg*.
- Computes the ciphertext to encrypt a message *mesg*, $c = mesg^2 \bmod m$.
- Sends the ciphertext *c* to B.

- **Decryption:**

Recipient B does the following: -

Uses private key (r, s) to compute *mesg* from given ciphertext c. for getting message *mesg* from c, use the formulas below to calculate the four square roots modulo m of c: m_1, m_2, m_3, and m_4. One of the four is the original message mesg, a second square root is m - mesg, and the other two roots are negatives of one another, but otherwise random-looking. Somehow one needs to determine the original message from the other three roots (see below). Thus the square roots $m_r = \sqrt{c} \bmod r$ and $m_s = \sqrt{c} \bmod s$ must be calculated (see section below).

By applying the extended Euclidean algorithm, a and b, with a*p + b*q = 1 are calculated. Now, by invocation of the Chinese remainder theorem, the four square roots

- $m_1 = x = (a*m_r*s + b*m_s*r) \bmod m$.
- $m_2 = y = (a*m_r*s - b*m_s*r) \bmod m$.
- $m_3 = n-x$.
- $m_4 = n-y$.

In the special case in which both primes when divided by 4 give remainder 3, there are simple formulas for the four roots:
Formulas for the four square roots of a square c are as follows:

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

Calculate a and b satisfying a*r + b*s = 1, using the extended GCD algorithm, computed once when the keys are generated.

- $m_r = c^{(p+1)/4} \bmod r$.
- $m_s = c^{(q+1)/4} \bmod s$.
- $x = (a*m_r*s + b*m_s*r) \bmod n$.
- $y = (a*m_r*s - b*m_s*r) \bmod n$.

Now the four square roots are

- $m_1 = x$,
- $m_2 = n-x$,
- $m_3 = y$, and
- $m_4 = n-y$.

In this way, four square roots are received and one of the square-root is the original message. The effectiveness of this algorithm has been questioned mainly because of the fact that it produces 4 results, one correct and 3 false. In order to distinguish the true message from the other three square roots returned, it is necessary to put redundant information into the message, so that it can be identified.

IV. HCAPF Cryptosystem

HCAPF is a public key cryptosystem. As with all public key cryptosystem techniques one key is needed for encryption and a different but related key is needed for decryption. It is very difficult to determine the decryption key if one knows the algorithm and the encryption key. Following are the processes of HCAPF cryptosystem.

- Key Generation Algorithm:
  1. Generate four large random primes p, q, r and s of approximately equal size.
  2. Compute n = p x q and m = r x s.
  3. Calculate: a and b satisfying a*r + b*s = 1, using the extended GCD algorithm [1].
  4. Compute φ = (p-1) x (q-1).
  5. Choose an integer e where $1 < e < \varphi$, such that GCD (e, φ) = 1.
  6. Compute the secret exponent d, such that e x d mod φ = 1.
  7. The public key is (n, m, e) and the private key is (r, s, d). Keep all the values d, p, q, r, s and φ secret.
     - n and m are known as the *modulus*.
     - e is known as the *public exponent* or *encryption exponent* or just the *exponent*.
     - d is known as the *secret exponent* or *decryption exponent*.
- Encryption:
  Sender A does the following:-
  - Obtains the recipient B's public key (e, n, m).
  - Represents the plaintext message as a positive integer *mesg*.
  - Computes the cipher text $c = ((mesg^2 \bmod m)^e \bmod n)$.
  - Sends the cipher text c to B.
- Decryption:
  Recipient B does the following:-
  - Uses A's private key d to first compute $mesg_1 = c^d \bmod n$.
  - Compute the square root of mesg1 as follows
  - $m_r = \sqrt{mesg_1} \bmod r$ and $m_s = \sqrt{mesg_1} \bmod s$.
  - Four messages is calculated by using following formulas:
    - $m_1 = (a \times r \times m_s + b \times s \times m_r) \bmod m$.
    - $m_2 = m - m_1$.
    - $m_3 = (a \times r \times m_s - b \times s \times m_r) \bmod m$.
    - $m_4 = m - m_3$.

One of these square roots (m_1, m_2, m_3, m_4) is the original plaintext *mesg*. There are many techniques of getting the original message from all received messages such as Hashing, replicating of message bits and adding specific padding scheme.

V. Comparison of RSA, Rabin and HCAPF Cryptosystem

The work in this dissertation is focused primarily on the security enhancement of public key cryptosystem, implementation of *Hybrid cryptography Algorithm based on Prime Factorization* (HCAPF), performance analysis of this cryptosystem and comparison of this cryptosystem with RSA and Rabin's cryptosystem. This GUI application is implemented using JAVA library functions [10]. In this application, one can either enter the prime numbers or can specify the bit length of the prime numbers to be generated automatically. Then by using prime numbers modulus are calculated and further public and private keys of specific bit length are generated.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

There is a text box in which a message (plain text) which to be encrypted is inserted and by using encryption, decryption button, the message will be encrypted and decrypted. The bottom of this application contains a text box to display all the time analysis results of the algorithms (RSA, RABIN & HCAPF). Any practical implementation of the RSA, RABIN & HCAPF cryptosystem would involve working with large integers (512 bits or more in size). This application developed in JAVA uses the BigInteger library. The BigInteger library provides operations for modular arithmetic, GCD calculation, primality testing, prime generation, bit manipulation etc.

Size of n (bit)	Size of m (bit)	Size of d (bit)	Chunk size (bit)	RSA Cryptosystem			
				RSA key gen time (ms)	RSA Encryption time (ms)	RSA Decryption time (ms)	Total execution time (ms)
256	256	256	128	78	47	63	188
256	256	256	256	78	31	42	151
512	512	512	128	95	110	375	580
512	512	512	256	95	63	187	345
512	512	512	512	95	31	94	220
1024	1024	1024	128	844	359	2453	3656
1024	1024	1024	256	844	187	1235	2266
1024	1024	1024	512	844	94	640	1578
1024	1024	1024	1024	844	47	312	1203
2048	2048	2048	128	2688	1390	18063	22141
2048	2048	2048	256	2688	671	9141	12500
2048	2048	2048	512	2688	328	4547	7563
2048	2048	2048	1024	2688	156	2297	5141
2048	2048	2048	2048	2688	78	1140	3906

Table 4.1(a)

Size of n (bit)	Size of m (bit)	Size of d (bit)	Chunk size (bit)	Rabin Cryptosystem			
				Rabin key gen time (ms)	Rabin Encryption time (ms)	Rabin Decryption time (ms)	Total execution time (ms)
256	256	256	128	78	47	50	175
256	256	256	256	78	16	15	109
512	512	512	128	406	47	156	609
512	512	512	256	406	32	78	516
512	512	512	512	406	15	31	452
1024	1024	1024	128	1578	32	953	2563
1024	1024	1024	256	1578	18	422	2018
1024	1024	1024	512	1578	16	219	1813
1024	1024	1024	1024	1578	16	93	1687
2048	2048	2048	128	8109	31	6125	14265
2048	2048	2048	256	8109	22	2766	10897
2048	2048	2048	512	8109	16	1406	9531
2048	2048	2048	1024	8109	15	640	8764
2048	2048	2048	2048	8109	16	328	8453

Table 4.1(b)

Size of n (bit)	Size of m (bit)	Size of d (bit)	Chunk size (bit)	HCAPF Cryptosystem			
				HCAPF key gen time (ms)	HCAPF Encryption time (ms)	HCAPF Decryption time (ms)	Total execution time (ms)
256	256	256	128	156	94	219	469
256	256	256	256	156	31	94	281
512	512	512	128	501	172	1078	1751
512	512	512	256	501	78	500	1079
512	512	512	512	501	31	235	767
1024	1024	1024	128	2422	500	6907	9829
1024	1024	1024	256	2422	203	3140	5765
1024	1024	1024	512	2422	94	1531	4047
1024	1024	1024	1024	2422	47	734	3203
2048	2048	2048	128	10797	1875	50797	63469
2048	2048	2048	256	10797	719	22687	34203
2048	2048	2048	512	10797	359	11000	22156
2048	2048	2048	1024	10797	172	5516	16485
2048	2048	2048	2048	10797	80	2609	13486

Table 4.1(c)

Table 4.1(a,b,c): Effect of changing the modulus length and chunk size on the size of Private key, Key generation time, Encryption time and Decryption time, while the size of Public key has kept constant (128 bit).

VI. Simulation Results

The simulation of the algorithm, implemented in JAVA, running on a 2 GHz P-IV Processor and 1.256 GB RAM has used a 1000 characters long message for encryption/decryption. The PKC algorithms (RSA, RABIN & HCAPF) have some important parameters affecting its level of security and speed [7]. By increasing the modulus length plays an important role in 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 the length of message to be encrypted at a time (chunk size).

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

When the length of message is changed then the length of encrypted message will proportionally change, hence larger chunks are selected to obtained larger encrypted message to increase the security of the data in use [4]. The RSA, RABIN and HCAPF parameters are changed one parameter at a time and the others are kept fixed to study the relative importance

- 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 bit length of their factors and so the difficulty of factoring them into their prime factors. Moreover, the length of the secret key (d) increases at the same rate as n-bit increases. As a result, increasing the n-bit length provides more security. On other hand by increasing the n-bit length increases the values of key generation time, encryption time and decryption time. 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, 4.2, and 4.3.

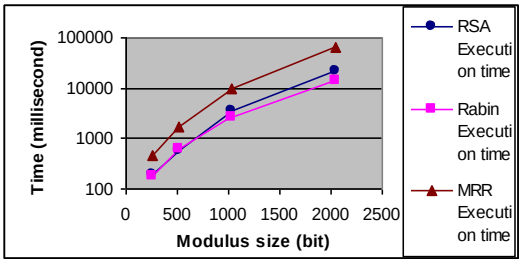

Fig.4.1: Modulus size v/s RSA, Rabin and MRR algorithm's execution time, taking size of Chunk 128 bit and size of Public Key 128 bit.

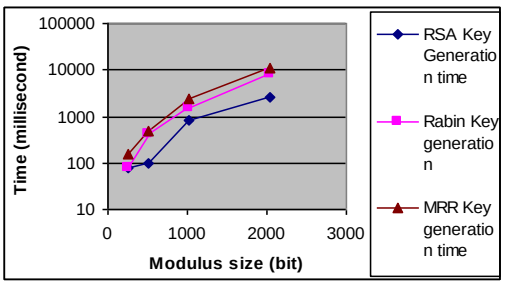

Fig. 4.2: Modulus size v/s Key generation time, taking size of Public key 128 bit and size of chunk 128 bit

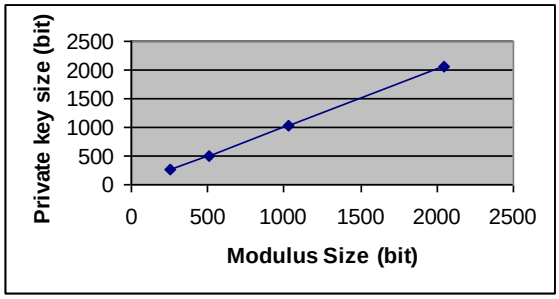

Fig. 4.3: Modulus size v/s Private Key size.

- Changing the Length of the message to be processed(chunk size) :

The chunk size is the number of characters to be processed at a time, either in encryption or decryption. Here the message is divided into sub blocks each of length equal to block size or chunk size. To illustrate the importance of this parameter, the message is taken long enough and the cut length is allowed to vary in both the encryption/decryption process. On the basis of simulation results of Table 4.1, following Figure 4.4 shows the effect of chunk size on encryption and decryption time of both the algorithms. Here key generation time doesn't dependent on chunk size as there is no role of chunk size in key generation process.

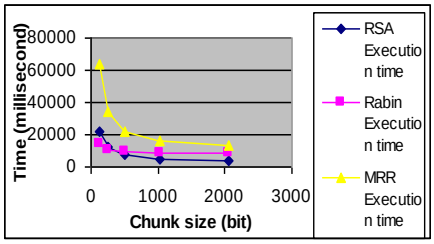

Fig.4.4: Chunk size v/s RSA, Rabin & HCAPF algorithm's execution time, taking size of modulus 2048 bit and size of private 128 bit.

VII. Complexity Analysis of HCAPF

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 [11]. In the Rabin cryptosystem, the computational complexity is on the order k^2 for encryption and on the order of k^3 for decryption [12]. In this case, too, k represents the number of bits of the public key m.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 2, Issue 2, February 2014)

So in this way, computational complexity of HCAPF cryptosystem is on the order of $O(k^2 + k^3)$ i.e. $O(k^3)$ or encryption and on the order of k^3 for decryption. Here k represents the number of bits of m and n (assume size of m and n is equal). So the computational complexity of HCAPF is equivalent to RSA and Rabin's cryptosystem.

VIII. Security Analysis of HCAPF

Two possible approaches to attacking the HCAPF / RSA/ Rabin's algorithms are as follows [1]:

- Brute force: This involves trying all possible private keys.
- Mathematical attacks: These are based on factoring the product of large primes such as factor n and m into its prime factors p, q and r, s respectively then calculating φ, which, in turn, enables determination of $d = e^{-1} \bmod \varphi$.

Estimated resources needed to factor a number within one year are as follows [11].

Size of number (bits)	PCs	Memory
430	1	128 MB
760	215,000	4 GB
1020	342×10^6	170 GB
1620	1.6×10^{15}	120 TB

Table 4.2: Recourses needed to factor a number within one year.

As the HCAPF cryptosystem uses dual modulus m and n, so for breaking this one have to factor both the modulus. If RSA and Rabin's algorithm, which is based on single modulus, is broken in time x then the time required to break HCAPF algorithm is $x^{2.}$ So the security of HCAPF algorithm is increased exponential as compare to RSA and Rabin's algorithm and it shows that the HCAPF algorithm is more secure for *Mathematical attacks*.

As shown in Table 4.2, it is very difficult to factor a large number in a specific time using limited resources so using dual modulus is sufficient difficult to factor two large number consecutively as the complexity is increased exponentially.

As in HCAPF double decryption is performed and unlike RSA that is not only based on private key but also based on factors of modulus m so one can't break HCAPF only guessing the private key only. So it shows that HCAPF algorithm is more secure as compare to RSA for *Brute force attack*.

IX. Conclusion

This paper presents a modified version of Rabin's and RSA algorithm called *Modified Rabin's over RSA Public Key Cryptosystem or Hybrid cryptography Algorithm based on Prime Factorization (HCAPF)*. It relies on facts of mathematics that indicates that given a very large number it is quite impossible in today's aspect to find two prime numbers whose product is the given number. The size of number increases, the possibility for factoring the number decreases. In RSA and Rabin, if one can factor modulus into its prime numbers then he can get the private key too. To improve the security, in HCAPF cryptosystem dual modulus (n & m) are used and the complexity of factoring the modulus is increased exponentially although the speed of encryption and decryption process is slowed down.

References

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

[2] Atul Kahate, "Cryptography and Network Security", *ISBN-10:0-07-064823-9*, Tata McGraw-Hill Publishing Company Limited, India, Second Edition, pages 38-62,152-165,205-240,340-370.

[3] R. Rivest, A. Shamir and L. Adleman. "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", *Communications of the ACM*, 21 (2), February 1978, pages 120-126.

[4] Bryan Poe, "Factoring the RSA Algorithm**",** Mat / CSC 494, April 27, 2005, pages 1-6.

[5] Alfred Menezes, "Evaluation of Security Level of Cryptography: RSA-OAEP, RSA-PSS, RSA Signature", University of Waterloo, 2001, pages 4-13.

[6] Diffie, M Hellman, "New Directions in Cryptography", *IEEE Transactions on Information Theory*, Vol 22, 1976.

[7] DI Management Services Pvt Limited, "RSA Algorithm", *ABN 78 083 210 584* Sydney, Australia, pages 1-20.

[8] 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.

[9] Rajorshi Biswas, Shibdas Bandyopadhyay, Anirban Banerjee, "A fast implementation of the RSA algorithm using the GNU MP library", IIIT – Calcutta, National workshop on cryptography, 2003,pages 1-15.

[10] CHUK, Chinese university (2009), "RSA Algorithm security and Complexity", Retrieved from http://www.cse.cuhk.edu.hk/~phwl/mt/public/archives/old/ceg5010/rsa.pdf (26 Oct. 2009)

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

[12] Christpher M. king, Curtis E. Dalton, & T. Ertem Osmanolu, "Security Architecture Design, Deployment & Operations", *ISBN-0-07-047272-6, Tata McGraw-Hill Publishing Company Limited*, New Delhi, pages 91-93, 347-370.

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

[14] Rabin, Michael. *Digitalized Signatures and Public-Key Functions as Intractable as Factorization* MIT Laboratory for Computer Science, January 1979.

[15] Mykola Karpinskyy , Yaroslav Kinakh, "Reliability of RSA Algorithm and its Computational Complexity" , *IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications,* 8-10 September 2003.

[16] Orhan K AKYILDIZ, "The RSA Cryptosystem", *COMP5703 - Advanced Algorithms*, October 21, 2008, pages 1-6.

[17] Cormen, Thomas H., Charles E. Leiserson; Ronald L. Rivest; Clifford Stein (2001). "Introduction to Algorithms", Second Edition, 2001, *ISBN 0-262-03293-7*, MIT Press and McGraw-Hill. Pages 881–887.

[18] P.L. Montgomery." A survey of modern integer factorization algorithms", CWI Quarterly, 7(4):337-365, 1994.

[19] Qing Liu Yunfei,Lin Hua pen," Two efficient variants of the RSA Cryptosystem".

[20] T. ElGamal. "A public key cryptosystem and a signature scheme based on discrete logarithms". Information Theory, IEEE Transactions on, 31(4):469-472, 2002.

[21] B. De Weger. Cryptanalysis of RSA with small prime di_erence. Applicable

[22] Algebra in Engineering, Communication and Computing, 13(1):17-28, 2002.

[23] D. Boneh and H. Shacham. "Fast variants of RSA. CryptoBytes (RSA Laboratories)", 5:1-9, 2002.

[24] Rania Salah EL-Sayed, Prof. Moustafa ABD EL-Aziem, Dr. Mohammad Aligomaa "An Efficient Signature System Using Optimized RSA Algorithm".

[25] Wuling Ren; Coll. of Comput. & Inf. Eng., Zhejiang Gongshang Univ., Hangzhou, China ; Zhiqian Miao "A Hybrid Encryption Algorithm Based on DES and RSA in Bluetooth Communication", 2010

[26] Al-Hamami, A.H. ,Aldariseh, I.A. "Enhanced Method for RSA Cryptosystem Algorithm", 2012

[27] Sharma, S. ,Sharma, P. ; "RSA algorithm using modified subset sum cryptosystem", 2011

[28] Zhang Qing ,Hu Zhihua : "The Large Prime Numbers Generation of RSA Algorithm Based on Genetic Algorithm", 2011

[29] Harn, L,Kiesler, T. : "Two new efficient cryptosystems based on Rabin's scheme: alternatives to RSA cryptosystem", 1989