International Journal of Advance Foundation and Research in Computer (IJAFRC)
Volume 1, Issue 10, October 2014. ISSN 2348 - 4853

Performance Evaluation of RSA-CRT Rebalanced Variants.

Noureldien A. Noureldien, Wafa M. Mustafa.
Department of Computer Science, University of Science and Technology,
Omdurman, Sudan
noureldien@hotmail.com,wa2009fa@gmail.com

ABSTRACT

The wide range of applications today raises the issue of what is the adequate security option that satisfies the application's security requirements. For applications applying encryption the question may be tailored to what RSA variant is more appropriate. The security of RSA was based on the number of bits in private key, where large number of bits provides higher security, but this in turn, makes the decryption slow. To satisfy applications needs of fast decryption or encryption many RSA variants were designed. Quisquater and Couvreur proposed an RSA variant, RSA-CRT, to speed up RSA decryption. Then, Wiener suggested another RSA variant, Rebalanced RSA-CRT, to further accelerate RSA-CRT decryption by shifting decryption cost to encryption. To adjust the rebalanced variant encryption time, Hung-Min Sun and Mu-En Wu suggest two schemas of RSA-CRT Rebalanced, namely Schema A and Schema B, to accelerate the rebalanced encryption time. This paper compares the RSA variants in terms of encryption and decryption time with a focus on RSA-CRT Rebalanced variants and gives a sight to applicability of each variant. The empirical tests are carried out using a laptop with 1.60 GHz CPU and 2 GB RAM, and the algorithms are implemented using Java 7. The results show that; Schema A attained the best encryption performance using a key length 1024 and it is 4.4 times faster that RSA-CRT Rebalanced, white Schema B is only 1.62 times faster. For decryption, RSA-CRT Rebalanced achieved the best performance; it is 7.27 and 1.97 times faster than standard RSA and RSA-CRT respectively, while Schema A and Schema B are 2.13 and 1.97 times slower than RSA-CRT Rebalanced. Accordingly, RSA-CRT Rebalanced Schema A is the best choice for applications desires a security of 1024 bit key and a balanced fast encryption and decryption time.

I. INTRODUCTION

RSA is a public-key algorithm that based on mathematical functions rather than on substitution and permutation operations. RSA was developed in 1977 by Ron Rivest, Adi Shamir, and Len Adleman and first published in 1978. The Rivest-Shamir-Adleman (RSA) scheme has since that time reigned supreme as the most widely accepted and implemented general-purpose approach to public-key encryption.

The security of RSA was based on the number of bits in private-key [2].The encryption and decryption in RSA requires taking heavy exponential multiplications modulus of a large integer *N* which is the product of two large primes' p and q [2].
The encryption and decryption time in RSA are roughly proportional to the number of bits in public and private exponents respectively. To reduce the encryption time, one may wish to use a small public exponent e, and to reduce the decryption time, a short secret exponent d can be used. In this context many RSA variants are designed [6].

The wide range of applications today raises the issue of what is the adequate security option that satisfies the application's security requirements. For applications applying encryption the question may be tailored to what RSA variant is more appropriate.

This paper compares the RSA variants in terms of encryption and decryption time with a focus on RSA-CRT Rebalanced variants and gives a sight to applicability of each variant.

The remainder of this paper is organized as follows. Section 2 presents an overview of RSA and its variants. Section 3 describes the RSA-CRT Rebalanced variants; Scheme A and Scheme B, which were developed for speeding up encryption. Section 4 reviews some related work. Section 5 reports a series of experiments carried out to evaluate the performance of different RSA-CRT variants. Finally, Section 6 concludes this paper.

II. OVERVIEW OF RSA VARIANTS

This section describes the Standard RSA and the RSA variants that had been designed to decrease the decryption time.

A. RSA standard

The original RSA cryptosystem was proposed in 1978 by Rivest, Shamir and Adelman and consists of three parts [1], described in the following subsections.

1. RSA Key generation

1- Generate two different primes p and q of $(n/2)$-bit each.
2- Compute $N = pq$ and $\emptyset(N) = (p\text{-}1)(q\text{-}1)$.
3- Choose a random integer $1 < e < \emptyset(N)$ such that $\gcd(e,\emptyset(N)) = 1$.
4- Next, compute the uniquely defined integer $1 < d < \emptyset(N)$ satisfying $ed \equiv 1 \pmod{\emptyset(N)}$.
The public key is $<N, e>$ and the private key $<N, d>$.

2. RSA Encryption

To encrypt a message X with the public key $<N, e>$, transform the message X to an integer M in $\{0,\ldots,N\text{-}1\}$ and compute the ciphertext $C = M^e \bmod N$.

3. RSA Decryption

To decrypt the ciphertext C with the private key $<N, d>$, compute $M = C^d \bmod N$ and employ the reverse transformation to obtain the message X from M [3][1].

4. Complexity of RSA algorithm

The calculate the complexity of RSA algorithm, the exponentiations of the form $C^d \bmod N$ take time $O(dM(n))$,where $M(n)$ is the cost of multiplying two n-bit integers and can be taken as $O(n^2)$. And $|d| \approx |n|$, recall that the number of binary operations to compute $C^d \bmod N$ is $1.5n.n^2 = 1.5n^3$, thus these algorithms take time $O(n^3)$.

On another hand $Me \pmod N$ take time $O(eM(n))$, $|e|$ is small, thus the encryption algorithms take time $O(n^2)$. This means that encryption is much faster than decryption in RSA Standard.

B. Standard RSA with CRT

Based on the Chinese Remainder Theorem (CRT), Quisquater and Couvreur [5] proposed an RSA variant, RSA-CRT, to speed up RSA decryption.

Here we review the basic steps constituting the RSA, together with the frequently used technique CRT. Suppose the message is M, the two large primes are p and q, the modulus is $N = pq$, and public key is $<N, e>$, private key is $<N, d>$ and to decrease decryption time private key is $<dp, dq, p, q>$ such that $dp \equiv d \bmod (p-1)$ and $dq \equiv d \bmod (q-1)$.

The encryption/decryption steps are as follows:

1. RSA-CRT Encryption

E (M): $C = M^e \bmod N$

2. RSA-CRT Decryption

Instead of directly computing $M = C^d \bmod N$, the decryption algorithm evaluates $Mp = C^{dp} \bmod p$ and $Mq = C^{dq} \bmod q$ where $dp = d \bmod p-1$ and $dq = d \bmod q-1$. It is then possible to recover the plaintext M using the Chinese remainder theorem. This method is faster because it computes two exponentiations of n/2-bit integers instead of one exponentiation of n-bit integers [1].

Algorithm: RSA decryption using CRT
Input: C, p, q, N;
Output: M;

---

1. $dp \leftarrow d \bmod (p - 1)$;
2. $dq \leftarrow d \bmod (q - 1)$;
3. $Mp \leftarrow C^{dp} \bmod p$;
4. $Mq \leftarrow C^{dq} \bmod q$;
5. $pinv \leftarrow p^{-1} \bmod q$;
6. $M \leftarrow CRT(Mp, Mq, p, q, (pinv), N)$;
7. *return(M);* [1]

RSA-CRT decryption Algorithm shows an optimization of RSA using the Chinese remainder theorem and the fast exponentiation algorithm. The two inverses that are normally needed for the Chinese remainder theorem are reduced in one inversion, which is pre-computed.

3. Complexity of RSA-CRT algorithm

The private key in RSA-CRT is $<dp, dq>$, the key length of dp and dq are $(n/2)$. So RSA-CRT require two times $O((n)/2)^3)$ compared to the $O(n^3)$ decryption in RSA standard.
Thus, the theoretical speedup of RSA-CRT is

$$\text{RSA-CRT} = n^3/2(n/2)^3 = 4$$

Then, the decryption procedure of RSA-CRT is about 4 times faster than that of RSA standard [4].

C. Rebalanced RSA-CRT

Based on the Chinese Remainder Theorem (CRT), Wiener suggested another RSA variant, Rebalanced RSA-CRT, to make the decryption time is much faster by carefully choosing *d* in the key generation phase [6] such that $dp \equiv d \pmod{p-1}$ and $dq \equiv d \pmod{q-1}$ are small.

One first selects two small CRT-exponents *dp* and *dq*, and then these two CRT-exponents are combined to get the secret exponent. At last, computes the corresponding public exponent ***e*** satisfying $ed \equiv 1 \pmod{\varphi(N)}$ [5]. This variant of RSA enables rebalances the costs of encryption and decryption. In other words, it speeds up the RSA decryption by shifting the decryption cost to the encryption cost.

In Rebalanced RSA-CRT, both *d* and *e* will be of the same order of magnitude as $\varphi(N)$. The decryption time depends on the bit-size of *dp* and *dq*, while not on the bit-size of *d*, the decryption time is minimized. But the encryption time depends on the bit-size of ***e***, this will make the encryption for the original Rebalanced RSA-CRT very time-consuming.

The algorithm takes two security parameters as input: n (typically 1024) and k (typically 160) where k < n/2. It next picks two primes *p* and *q* verifying $gcd(p-1, q-1) = 2$ and whose bit length is n/2. The modulus *N* is $N = pq$. It then randomly picks two k-bit values *dp* and *dq* such that $gcd(dp, p-1) = 1$, $gcd(dq, q-1) = 1$ and $dp = dq \bmod 2$. The secret exponent *d* must verify: $d = dp \pmod{p-1}$ and $d = dq \pmod{q-1}$. Cannot directly compute *d* with the Chinese remainder theorem because $p-1$ and $q-1$ are not relatively prime (they are both even). But we chose them such that $gcd(p-1, q-1) = 2$, therefore:

$$gcd((p-1)/2, (q-1)/2) = 1$$

We also know that $dp = dq \pmod 2$. To compute the public exponent e, we just have to compute the inverse of *d* modulo Ø (N) = (p − 1) (q − 1). This is allowed because gcd (dp, p − 1) = gcd ($dq, q-1$) = 1. Therefore, gcd (d, p −1) = gcd (d, q−1) = 1. And finally gcd (d, (p −1)(q−1)) = 1. We have no control over ***e*** which is of the order of *N*. The encryption won't be as fast as in standard RSA but we manage to increase the speed of the decryption stage.

Now we describe three constituent algorithms: key generation, encryption, and decryption.

1. Rebalanced RSA-CRT Key generation:

The key generation of the Rebalanced RSA is as follows:

Algorithm: Key Generation in Rebalanced RSA-CRT

Input: N, k;
Output: *p, q, dp, dq, e, d;*

---

1. Randomly select two 512-bit primes $p = 2p1+1$ and $q = 2q1+1$ such that $gcd(p1, q1) = 1$.
2. Compute $p1inv = p1^{-1} \pmod{q1}$.

3. Randomly select two distinct odd numbers *dp* and *dq* of *160* bits such that *gcd (dp, p−1) =1* and *gcd (dq, q − 1) = 1*.
4. Compute $d = CRT(dp, dq, p1, q1, p1\ inv, N)$.
5. Compute $e \equiv d^{-1} \bmod (p-1)(q-1)$.
6. Return *(p, q, dp, dq, e, d).* [5]

The public key is given by the pair *(e, N)* and the private key is given by the tuple *(dp, dq, p, q)*. Note that *e* will be roughly the same order of magnitude as *Ø(N).* Thus, in order to reduce the decryption time even further than RSA-CRT, the encryption time is essentially maximized.

2. Rebalanced RSA-CRT Encryption

This is exactly the same as in standard RSA, that is $C \equiv \ (mod\ N)$, except that e is much larger. The public key is (N, e). The complexity of this algorithm Take time $O(e\ M(n))$,where $M(n)$ is the cost of multiplying two n-bit integers and can be taken as $O(n^2)$.And $|e| \approx |n|$, thus these algorithms take time $O(n^3)$,compare to $O(n^2)$in RSA Standard

3. Rebalanced RSA-CRT Decryption

The decryption is the same as the CRT decryption. The main difference is that in Rebalanced RSA-CRT, the CRT-exponents *dp* and *dq* are only of *160* bits which are much shorter than the CRT exponents of *512* bits in RSA-CRT. Thus, the decryption in Rebalanced RSA-CRT is about *512/160 = 3.2* times faster than that in RSA-CRT.

The private key is *(p, q, dp, dq)*. Can decrypt a cipher text *C* by Using the Chinese remainder theorem, we are then able to recover the plaintext M, verifying *M = Mp mod P* and *M = Mq mod q*. as follows:

Algorithm: Rebalanced RSA-CRT Decryption

Input: *N, k;*
Output: *C, N, dp, dq, p, q;*

---

1. $Mp \leftarrow C^{dp} \bmod p;$
2. $Mq \leftarrow C^{dq} \bmod q;$
3. $pinv \leftarrow P^{-1} \bmod p;$
4. $M \leftarrow CRT(Mp, Mq, p, q, (pinv), N);$
5. *Return(M);* [9]

---

4. Complexity of Rebalanced RSA-CRT algorithm

Recall that *dp, dq* in Rebalanced RSA are *k-bits* each. The cost of modular multiplications is the same as that of RSA-CRT; the main difference is the number of multiplications computed during each modular exponentiation.

The decryption in Rebalanced RSA require tow times $O(k(n/2)^2)$. Compared to the $O(n^3)$ decryption of the RSA standard, then a theoretical speedup of Rebalanced RSA is

Rebalanced RSA-CRT = $n^3/2k(n/2)^2 = 2n/k$

So, for modulo of 1024 bits with k = 160, Rebalanced RSA is theoretically 12.8 Faster than RSA standard and 3.2 faster than RSA-CRT.

III. IMPROVING REBALANCED RSA-CRT ENCRYPTION TIME

According to the key generation in the original Rebalanced RSA-CRT, if we first select the small CRT-exponents *dp* and *dq*, the public exponent *e* will be of the same bit-size as modulus *Ø(N)*. This causes heavy encryption cost. If we can make the public exponent *e* much shorter than *Ø(N),* it will be more convenient and practical in many applications.

Two variants of Rebalanced RSA-CRT, Scheme A and Scheme B have been design to achieve the above goal [5]. The two key generation algorithms for Rebalanced RSA-CRT which generates public exponents much smaller than *Ø(N).* Each key generation algorithm is based on the following fundamental theorem from number theory.

Theorem 3.1: If a and b are relatively prime, i.e. *gcd(a, b) = 1*, then we can find a unique pair *(u , v)* satisfying *au – bv = 1*, for any integer *u,v € z* [7] .

A. Rebalanced RSA-CRT Scheme A

The first key generation algorithm, scheme A, produces a *568-bit* public exponent, e.g., $e = 2^{567} + 1$, and two 160-bit CRT-exponents *dp, dq*. The key generation of the algorithm is as follows:

Algorithm: Key Generation in Scheme A
Input: *n, k*;

Output *dp, dq, p, q, e;*

---

1. Randomly select an odd number ***e*** of *568* bits.
2. Randomly select a number *kp1* of *160* bits, such that *gcd (kp1, e) = 1*.
3. Based on Theorem 3.1, we can uniquely determine two numbers *dp, kp1 < dp < 2 kp1*, and P, *e < P < 2e*, satisfying *e dp = kp1 P + 1.*
4. Factor *P* as *P = kp2 · ṗ* such that *kp2* is a number of *56* bits and p= (ṗ +1) is a prime number. If this is infeasible, then go to Step 2.
5. Randomly select a number *kq1* of *160* bits, such that gcd (*kq1*, e) = 1.
6. Based on Theorem 3.1, we can uniquely determine two numbers *dq,kq1 < dq <2 kq1*, and q, *e < q < 2e*, satisfying *e dq = kq1 q + 1.*
7. Factor *q* as *q = kq2 · q̇* such that *kq2* is a number of *56* bits and q= (q´ +1) is a prime number. If this is infeasible, then go to Step 5.
8. Return *(dp, dq, p, q, e).* [5]

---

1. Complexity of Rebalanced RSA-CRT Scheme A

The complexity of encryption in Rebalanced is the same as the complexity of decryption in RSA, because the public exponent *e* will be of the same bit-size as modulus *Ø(N)*.then *C* = mod *N* take $1.5\ n^3 = O(n^3)$. Scheme A, use a *512*-bit public exponent, thus Scheme A take $k.n^2 = O(kn^2)$, where k is bit length of public exponent. The theoretical speedup of Scheme A is

Schema A = $1.5\ n^3 / kn^2 = 1.5\ n/k$

So, for modulo of 1024 bits with k = 568, Scheme A is theoretically 2.7 Faster than Rebalanced RSA-CRT.

B. Rebalanced RSA-CRT Scheme B

The second key generation algorithm, called Scheme B. In Scheme B, The Key Generation produces a *512*-bit public exponent, e.g., e = 2^{511} +1, and two *198-bit* CRT-exponents *dp, dq*. The encryption is about *3* times faster than that of Rebalanced RSA-CRT [5].

Algorithm: Key Generation in Scheme B

Input: *n, k;*

Output *dp, dq, p, q, e;*

---

1. Randomly select an odd number *e* of *512* bits.
2. Randomly select an odd number *kp* of *198* bits, such that *gcd (kp, e) = 1.*
3. Based on Theorem 1, we can uniquely determine two numbers dp, kp < dp < 2kp, and p´, e < p´ < 2e, satisfying *e.dp – kp.p´ = 1.*
4. If *p = p´+1* is not a prime number, then go to Step 2.
5. Randomly select an odd number *kq* of *198* bits, such that *gcd (kq, e) = 1.*
6. Based on Theorem 3.1, we can uniquely determine two numbers dq, *kq < dq < 2kq*, and *q´, e < q´< 2e*, satisfying *e dq – kq q´ = 1.*
7. If *q = q´ + 1* is not a prime number, then go to Step 5.
8. The public key is *(N, e);* the secret key is *(dp, dq, p, q).* [5]

The key generation algorithm for this scheme is much more efficient than the one in Scheme A. The runtime of the algorithm is dominated by two loops, like Scheme A, but each iteration requires much less computational effort as there is no factoring required [7].

1. Complexity of RSA-CRT Rebalanced Scheme B

The Complexity of Scheme B the same as in Scheme A, the main different is bit length of *e = 512*

$$Scheme\ B = \frac{1.5n^3}{kn^2} = 1.5\ n/k$$

So, for modulo of 1024 bits with k = 512, Scheme B is theoretically 3 times faster than Rebalanced RSA-CRT. But the decryptions in two Schemes are a little slower than that of Rebalanced RSA-CRT.

IV. RELATED WORKS

Klaus Hansen, Troels Larsen and Kim Olsen [9], compares the efficiency of the RSA variants RSA-CRT, Rebalanced, Mprime, Mpower and Rprime in modern mobile phone in terms of encryption and decryption time. Cesar Alison and Gazzoni Filho compare the RSA variants Standard RSA, RSA-CRT and RSA-CRT Rebalanced in terms of encryption and decryption using 768, 1024 and 2048 key lengths [10]. Al-Mamun, et al. [11], compare the RSA variants RSA, Mprime, Mpower, Rebalanced, RPrime and Batch using 768, 1024 and 2048 key lengths. The developers of RSA-CRT Rebalanced Schema A and Scheme B, compare the basic RSA, RSA-CRT, RSA-CRT Rebalanced, RSA-CRT Rebalanced Schema A and Scheme B in terms of encryption and decryption time [5].

V. EXPERIMENTS AND RESULTS

We implement the algorithms and measure the average running time. The machine used for testing is a laptop with 1.60 GHz CPU and 1.98GB RAM. The programming language we used is java version 7 on Windows system.

To compare the encryption time (En-time) and decryption time (De-time) of RSA-CRT Rebalanced variants, we carry out two sets of comparisons.

First we compare RSA standard, RSA-CRT and RSA-CRT Rebalanced variants using 128, 256, 512, 1024 and 2048 key lengths respectively. To justify the output each experiment is carried out five times using different keys and average result is calculated.

Secondly, in a similar way, we compare RSA-CRT Rebalanced, RSA-CRT Rebalanced scheme A and RSA-CRT Rebalanced scheme B using 1024-bits modules. Table 1 shows the output of the tests as average time speed.

Table 1: Summary of experimental results

Set 1				Set 2		
Key length	**En&De Time**	**RSA Std**	**RSA-CRT**	**Rebalanced RSA**	**Scheme A**	**Scheme B**
128	En-Time	0 ms	0 ms	0 ms	_	_
	De-time	0 ms	0 ms	0 ms	_	_
256	En-Time	0 ms	3.75 ms	34 ms	_	_
	De-time	31 ms	3.75 ms	13 ms	_	_
512	En-Time	0 ms	3 ms	37.6 ms	_	_
	De-time	18.8 ms	6.2 ms	5.2 ms	_	_
1024	En-Time	34.2 ms	37.8 ms	137.4 ms	31.2 ms	84.6 ms
	De-time	138.2 ms	37.4 ms	19 ms	40.4ms	37.4 ms
2048	En-Time	225 ms	281.4 ms	649.8 ms	_	_
	De-time	672 ms	227.8 ms	31 ms	_	_

A. Results

For key length 1024, as expected, the best encryption time is achieved by standard RSA, since RSA-CRT and RSA-CRT Rebalanced are developed to accelerate decryption. For RSA-CRT Rebalanced variants, Schema A achieved the best performance and it is 4.4 times faster that RSA-CRT Rebalanced, while Schema B is only 1.62 times faster. For decryption, as expected, Rebalanced achieved the best performance, 7.27 and 1.97 times faster than standard RSA and RSA-CRT respectively, while Schema A and Schema B are 2.13 and 1.97 times slower than rebalanced.

For key length 2048 the encryption time of RSA-CRT and RSA-CRT rebalanced is 1.25 and 2.89 times slower than standard RSA respectively, while for decryption they are 2.95 and 21.68 times faster respectively. Fig. 1 and 2 shows the output of the tests and their average time speed of decryption time.

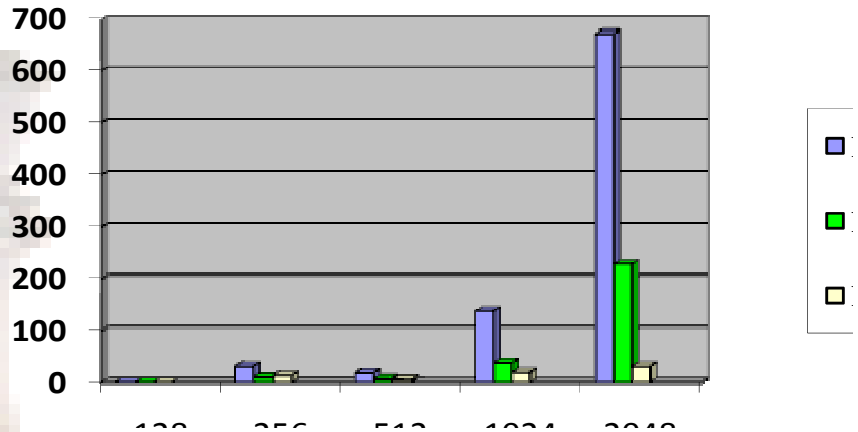

Figure 1. Comparison of the Decryption Time in Millisecond

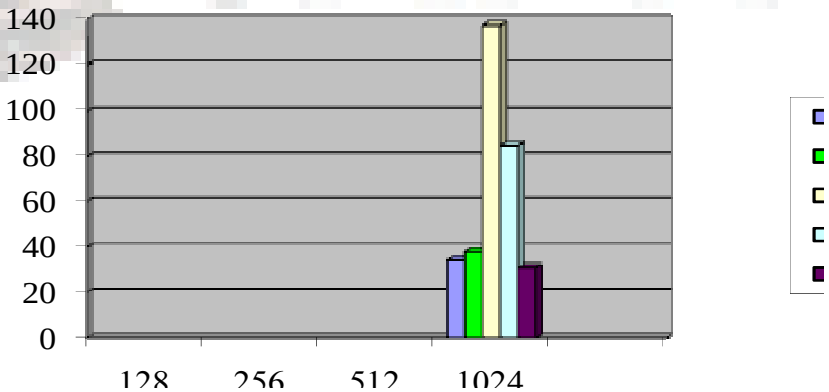

Figure 2. Comparison of the Encryption Time in Millisecond

VI. CONCLUSIONS

This paper presents a comparison between RSA-CRT Rebalanced which was designed originally to reduce the decryption time of RSA Standard, but unfortunately it slower the encryption time more than

that in RSA Standard, and the Rebalanced RSA-CRT Scheme A and Rebalanced RSA-CRT Scheme B, which have been designed to further reduce the encryption cost in the original RSA-CRT Rebalanced.
The experimental tests show that the encryption time of RSA-CRT Rebalanced Schema A is 4.4 times faster that RSA-CRT Rebalanced, while Schema B is only 1.62 times faster. For decryption, Schema A and Schema B are 2.13 and 1.97 times slower than RSA-CRT Rebalanced.

VII. REFERENCES

[1] Tsuyoshi Takagi, "Efficiency Comparison of Several RSA Variants", Fachbereich Informatik der TUDarmstadt, March 2003

[2] William Stallings, "Cryptography and Network Security Principles and Practices", Fourth Edition, prentice hall, November 2005-2006.

[3] L. M. Adelman, R. L. Rivest and A. Shamir, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems", Communications of the ACM, Vol.21, No.2, pp, 120-126, 1978.

[4] A. Menezes, P. Van Oorschot, and S. Vanstone. "Handbook for applied cryptography". CRC Press, 1997.

[5] H.-M. Sun and Mu-En. Wu, "Design of Rebalanced RSA-CRT for Fast Encryption, " Information Security Confer ence 2005. pp. 16-27.

[6] M. Wiener. "Cryptanalysis of Short RSA Secret Exponents". IEEE Trans. Information Theory 36(3):553–558. May 1990.

[7] Niven, H. S. Zuckerman, "an Introduction to the Theory of Number", John Wiley and SonsInc, 1991.

[8] Hung-Min Sun , Mu-En Wub, M. Jason Hinek , Cheng-Ta Yang , Vincent S. Tseng , "Trading decryption for speeding encryption in Rebalanced-RSA", 14 April 2009.

[9] Klaus Hansen, Troels Larsen and Kim Olsen "On the Efficiency of Fast RSA Variants in Modern Mobile Phone", vol. 6, no. 3, 2009.

[10] Cesar Alison Monteiro Paix˜ao, D´ecio Luiz Gazzoni Filho , "An efficient variant of the RSA cryptosystem".2005

[11] Md. Ali-Al-Mamun, Mohammad Motaharul Islam, S.M. Mashihure Romman and A.H. Salah Uddin Ahmad, "Performance Evaluation of Several Efficient RSA Variants", IJCSNS International Journal of Computer Science and Network Security, VOL.8 No.7, July 2008

