British Journal of Mathematics & Computer Science
14(1): 1-11, 2016, Article no.BJMCS.22196
ISSN: 2231-0851

SCIENCEDOMAIN *international*
www.sciencedomain.org

RAM-RSA and RBM-RSA: A New Fast Hybrid RSA Variants

Yossria M. Elhassan[1] and Noureldien A. Noureldien[1*]

[1]Department of Computer Science, University of Science and Technology, Omdurman, Sudan.

Authors' contributions

This work was carried out in collaboration between both authors. Both authors read and approved the final manuscript.

Article Information

DOI: 10.9734/BJMCS/2016/22196
Editor(s):
(1) Sun-Yuan Hsieh, Department of Computer Science and Information Engineering, National Cheng Kung University, Taiwan.
Reviewers:
(1) Hammad Khalil, University of Malakand, Pakistan.
(2) S. K. Rososhek, Tomsk State University, Russia.
Complete Peer review History: http://sciencedomain.org/review-history/12991

Original Research Article

Received: 22nd September 2015
Accepted: 17th November 2015
Published: 16th January 2016

Abstract

Since the development of RSA in 1977, many RSA variants are developed. The objectives behind these variants are either to improve RSA decryption time, to accelerate RSA encryption time or to rebalance RSA encryption and decryption time. The Mprime RSA variant is a well known variant that improve standard RSA decryption time, while Rebalanced RSA-CRT variant lowered decryption time. Later, the Rebalanced RSA-CRT Scheme A and Scheme B variants are developed to improve encryption cost in Rebalanced RSA-CRT variant.

The Rprime RSA variant was proposed in 2002, as a hybrid variant that use the key generation algorithm of Rebalanced RSA (modified for k primes) together with the decryption algorithm of Mprime RSA in order to further improve decryption time.

In this paper, we combine the achievements of Mprime variant and RSA Rebalanced Scheme A and Scheme B variants to develop two RSA variants that improve both the encryption and decryption time. We call these variants RAM-RSA (Rebalanced Scheme A with Mprime) and RBM-RSA (Rebalanced Scheme B with Mprime).

The experimental tests show that; for decryption RAM-RSA and RBM-RSA are respectively 3.4 and 2.8 times faster than Rprime RSA, and for encryption they are respectively 4.5 and 5.8 times faster than Rprime RSA.

Keywords: RSA; RSA variant; encryption time; decryption time; rebalanced RSA; Mprime RSA.

**Corresponding author: E-mail: yossria@gmail.com, noureldien@hotmail.com;*

1 Introduction

The most popular public key cryptosystem in use today is the RSA cryptosystem, introduced by Rivest, Shamir, and Adleman [1]. Its security is based on the intractability of the integer factorization problem.

The modulus N of a RSA cryptosystem is the product of two large primes p and q. The public exponent e and the secret exponent d are related by $ed = 1\ (mod\ \emptyset(N))$. To encrypt a message M with the public key <N, e>, the message M is transformed to an integer in {0,…,N-1} and the cipher text is $C = M^e \bmod N$. To decrypt the cipher text C with the private key <N, d>, compute $M = C^d\ mod\ N$ [1].

The main drawback of using RSA, however, is the relatively costly encryption and decryption operations due to the large values of e and d. To overcome this limitation, RSA-CRT was proposed to improve decryption time [2]. The encryption is done as the same as the Standard RSA, but instead of directly computing $M = C^d\ mod\ N$, the decryption algorithm evaluates $M_p = C^{d_p}\ mod\ p$ and $M_q = C^{d_q}\ mod\ q$, where $d_p = d\ mod\ (p-1)$ and $d_q = d\ mod\ (q-1)$.

It is then possible to recover the plaintext M using the Chinese remainder theorem. This method is faster than Standard RSA in decryption, because it computes two exponentiations of N/2-bit integers instead of one exponentiation of N-bit integers. Unfortunately this improvement places extra cost on encryption time. This situation leads to the concept of rebalanced RSA where both encryption and decryption has to have relatively equal times. Accordingly, RSA-RCT Rebalanced variant was proposed in 1990 [3].

In Rebalanced RSA-CRT, both d and e have the same order of magnitude as φ (N). The decryption time will depends on the bit-size of d_p and d_q, but not on the bit-size of d, while the encryption time depends on the bit-size of e. This makes the encryption for the Rebalanced RSA-CRT very time-consuming. One solution to this problem is provided by two variants of Rebalanced RSA-CRT, namely Scheme A and Scheme B [4].

The Mprime RSA variant was introduced by Collins [5] to improve the decryption time. Here the RSA modulus was modified so that it consists of k primes $N=p_1, p_2, p_3, \ldots p_k$ instead of the standard two prime's p and q. Encryption of a message M is carried out exactly as in the original RSA, i,e $C = M^e\ mod\ N$. To decrypt a cipher text C, $M_i = C^{d_i}\ mod\ p_i$ is calculated for each i, i≤1≤ k. Next, the algorithm applies the CRT to the M_i's to get $= C^d\ mod\ N$.

The Rprime RSA variant was proposed by Cesar [6]. The idea was to combine two RSA variants; Rebalanced RSA and Mprime RSA to further enhance the decryption speed. The general idea of this scheme is to use the key generation algorithm of Rebalanced RSA (modified for k primes) together with the decryption algorithm of Mprime RSA. Accordingly, Rprime RSA lowered decryption costs, but it does not change the cost of encryption time.

Our proposed variants RAM-RSA and RBM-RSA aim to improve both encryption and decryption times simultaneously by combining Rebalanced RSA Scheme A and Scheme B variants with Mprime variant.

The rest of this paper is organized as follows. In section 2 we review RSA variants relative to the developed variants, in section 3 we present the proposed new variants. Section 4 presents experiment and results, and conclusions are drawn in section 5.

2 RSA Variants

In this section we presenta precise description for RSA variants that are relevant to our proposed variants.

2.1 RSA-CRT variant

Based on the Chinese Remainder Theorem (CRT), an RSA-CRT variant was proposed speed up RSA decryption [2].

The key generation and encryption in RSA-CRT is similar to standard RSA, but for decryption instead of directly computing $M = C^d \bmod N$, the decryption algorithm evaluates $M_p = C_p^{d_p} \bmod p$ and $M_q = C_q^{d_q} \bmod q$, where $d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$. It is then possible to recover the plaintext M from M_p and M_q using the Chinese remainder theorem. This method is faster, approximately four times as fast as evaluating $M = C^d \bmod N$ directly.

2.2 Rebalanced RSA-CRT variant

Basically, in the original RSA, encryption is more efficient than decryption because *e* is small and *d* is large. A straightforward way to optimize decryption would be to "switch" the exponents, i.e. make *e* large and *d* small. However, this is not recommended because small values of *d* open up RSA for what is known as small *d* exponent attacks, such as Wiener attack [3], and Boneh and Durfee attack [7].

One solution is proposed by Wiener named Rebalanced RSA [3]. The proposed variant retains the size of *d* but makes d_p and d_q ($d_p = d \bmod (p-1)$ and $d_q = d \bmod (q-1)$) each of length *w* bit such that *w* is much smaller than n/2, the length of d_p and d_q in RSA-CRT variant.

Key generation in Rebalanced RSA works as follows:

1. Given integers n and *w*, generate two different primes p and q each is (n/2)-bit long such that gcd ((p-1),(q-1))=1.
2. Set N = pq and φ(N) = (p-1)(q-1).
3. Compute two *w*-bit integers d_p and d_q satisfying gcd(d_p,p-1) = gcd(d_q,q-1) = 1 and $d_p \equiv d_q \pmod 2$.
4. Find a *d* such that $d = d_p \bmod (p-1)$and $d = d_q \bmod (q-1)$.
5. Compute$e = d^{-1} \bmod Ø(N)$. The public key is <N, *e*> and the private key <p, q, d_p, d_q>.

Encryption is the same as for the original RSA, $C = M^e \bmod N$, but with a much larger *e* (on the order of N), while decryption is the same as for RSA-CRT but with smaller d_p and d_q, each is *w*-bit long versus (n/2)-bit in CRT RSA, usually $w \geq 160$ and $n/2 \geq 512$.

The speedup of Rebalanced RSA-CRT over standard RSA-CRT is: $S_{CRT}=n/2s$, for s= 160, the theoretical speedup is 6.4 times than RSA-CRT [9].

2.3 Rebalance RSA-CRT scheme A

In Rebalanced RSA-CRT, both *d* and *e* will be of the same order of magnitude as φ(N), and the decryption time will depends on the bit-size of d_p and d_q, not on the bit-size of *d*, while the encryption time depends on the bit-size of *e*. This makes the encryption cost for the Rebalanced RSA-CRT very high.

To improve the encryption time in Rebalanced RSA-CRT, two variants of Rebalanced RSA-CRT are proposed, namely, Rebalanced RSA-CRT Scheme A and Rebalanced RSA-CRT Scheme B [4].

The key generation algorithms for Rebalanced RSA-CRT scheme A and B, generates a public exponent much smaller than Ø(N). Each key generation algorithm is based on the following fundamental theorem from number theory.

Theorem 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 [8].

The key generation algorithm, for scheme A, produces a 568-bit public exponent, and two 160-bit CRT-exponents d_p, d_q. The encryption time is therefore reduced to about 2.6 of the time required by the original Rebalanced RSA-CRT.

The key generation of the algorithm is as follows [4]:

1. Randomly select an odd number e of 568 bits.
2. Randomly select a number kp_1 of 160 bits, such that gcd (k_{p1}, e) = 1.
3. Based on Theorem 1, determine two numbers d_p, $k_{p1}<d_p< 2\ k_{p1}$, and P, e < P < 2e, satisfying e *d_p = k_{p1}*P + 1.
4. Factor P as P = k_{p2} · ṗ such that $k_{p2}$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 k_{q1} of 160 bits, such that gcd (k_{q1}, e) = 1.
6. Based on Theorem 1, we can uniquely determine two numbers d_q, $k_{q1}<d_q< 2\ k_{q1}$, and q, e < q < 2e, satisfying e* d_q = k_{q1}* q + 1.
7. Factor q as q = k_{q2}· q̇ such that k_{q2} 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 (d_p, d_q, p, q, e).

The complexity of encryption in Rebalanced scheme 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 $C = M^e \bmod N$ take $1.5\ n^3 = O(n^3)$.

Scheme A, use a *568*-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 = $1.5\ n^3/kn^2 = 1.5n/k$.

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

2.4 Rebalance RSA-CRT scheme B

The Key Generation Algorithm for scheme B produces a 512-bit public exponent, and two 198-bit CRT-exponents dp, dq. The encryption is about 3 times faster than that of Rebalanced RSA-CRT, but the decryption is a little slower than that [4].

The key generation in scheme B is as follows:

1. Randomly select an odd number e of 512 bits.
2. Randomly select an odd number k_p of 198 bits, such that gcd (k_p, e) = 1.
3. Based on Theorem 1, we can uniquely determine two numbers d_p, $k_p<d_p< 2k_p$, and p´, e < p´ < 2e, satisfying e.d_p–k_p .p´ = 1.
4. If p = p´+1 is not a prime number, then go to Step 2.
5. Randomly select an odd number k_q of 198 bits, such that gcd (k_q, e) = 1.
6. Based on Theorem 1, we can uniquely determine two numbers d_q, $k_q<dq< 2k_q$, and q´, e < q´< 2e, satisfying e d_q–k_qq´ = 1.
7. If q = q´ + 1is not a prime number, then go to Step 5.

The public key is (N, e); the secret key is (d_p, d_q, p, q). The complexity of scheme B is the same as in Scheme A, the main different is bit length of *e = 512.*

So, for modulo of 1024 bits with k = 512, scheme B is theoretically 3 times faster than Rebalanced RSA-CRT. But the decryptions in schemes A and B is a little slower than that of Rebalanced RSA-CRT.

2.5 Mprime RSA variant

Mprime RSA was introduced by Collins [5], who modified the RSA modulus so that it consists of k primes $N=p_1 \cdot p_2 \cdot p_3 \cdot \ldots \cdot p_k$ instead of the traditional two primes p and q. The key generation, encryption and decryption algorithms are as follows:

1. Compute k distinct prime $p_1, p_2, \ldots p_k$, each of log n/k bits in length and $N = \prod_{i=1}^{k} p_i$
2. Compute e and d such that $d = e^{-1} \, mod \; Ø\,(N)$, where gcd(e, Ø (N))= 1, Ø (N)= $\prod_{i=1}^{k} p_i - 1$
3. For all i, 1≤ i≤ k, compute $d_i = d \, mod \, (p_i - 1)$

The Public key is <N, e>, and the private key = <N, $p_1, p_2, \ldots p_k, d_1, d_2 \ldots d_k$>

The encryption of M is exactly as in the original RSA, thus $C = M^e \bmod N$.

To decrypt a cipher text C, first calculate $M_i = C^{di} mod \, p_i$ for each i, i≤1≤ k. Next, apply the CRT to the M_i's to get $M = C^d \; mod \; N$.

Decryption in Multi-Prime RSA requires *r* times $O((n/r)^3)$compared to the $O(n^3)$ decryption of the original RSA, Mprime RSA improves decryption time with a factor $n3 / (r \cdot (n/r)^3) = r^2$.

2.6 Rprime RSA

This variant was proposed by Cesar [6]. The idea of this scheme is to use the key generation algorithm of Rebalanced RSA (modified for k primes) together with the decryption algorithm of Mprime RSA in order to further improve decryption time.

The key generation algorithm takes as inputs, k, n and s, where k denotes the number of primes to be generated, and s > 0 : s ≤ n/k bits, and works as follows:

1. Generate k distinct random primes of n/k bits $p_1, p_2 \ldots, p_k$, with gcd(p_1- 1, p_2– 1, …, p_k– 1) = 2; and calculate $N=p_1 p_2 \ldots p_k$.
2. Generate k random numbers of s-bits $d_{p1}, d_{p2} \ldots, d_{pk}$, such that gcd($dp_1$, p_1– 1) = 1, gcd(d_{p2} , p_2– 1) = 1, …,gcd(d_{pk} , p_k– 1) = 1 and $d_{p1} \bmod 2 = d_{p2} \bmod 2 = \ldots = d_{pk} \bmod 2$.
3. Find d such that $= d_{p1} \, mod \, (p_1 - 1)$, $d = d_{p2} \, mod \, (p_2 - 1) \ldots . \, d = d_{pk} \, mod \, (p_k - 1)$ [1].
4. Calculate d by using CRT.

Public key = <N, e> and Private key = <$p_1, p_2 \ldots p_k, d_{p1}, d_{p2} \ldots d_{pk}$> [2].

Encrypting with the public key <N, e> is identical to the original RSA. However, as in Rebalanced RSA, the public exponent e is much larger than the normally used e. The decryption is the same as in Mprime RSA.

Compared to the $O(n^3)$ decryption of the original RSA, Rprime RSA improves decryption time with a factor $n^3/(rw \cdot (n/r)^2) = nr/w$.

3 The Proposed RAM-RSA and RBM-RSA Variants

Our work is similar to Rprime RSA variant as it combines a Rebalanced RSA variant and Mprime variant, but instead of lowering the decryption cost only, our proposed variants aims to lower both encryption and decryption time. This is accomplished by combining each of Rebalanced RSA-CRT Schema A and B, which are proposed to improve encryption time, with Mprime which was proposed to improve decryption time.

We call the first variant RAM-RSA (Rebalanced Schema A and Mprime RSA variant). The idea of RAM-RSA is to use the key generation algorithm of Rebalanced RSA scheme A (modified for k primes) together with the decryption algorithm of Mprime RSA.

The second variant RBM-RSA (Rebalanced Schema B and Mprime RSA variant) is similar to RAM-RSA except that, the key generation algorithm of Rebalanced RSA scheme B (modified for k primes) is used.

The direct hybridization of the Rebalanced scheme and Mprime variants will result in a variable key length which will be determined by lengths of the *k* primes and a fixed length of the exponent *e*.

To give the user control on the size of *e* and on the number of multiple prime's *k*, we modify the direct hybrid variants by modifying key generation algorithms.

3.1 RAM-RSA variant

To realize this hybrid variant, we use the key generation algorithm of Rebalanced RSA Scheme A modified for k primes rather than two prime p and q as in Scheme A. The key generation, encryption and decryption algorithm of this hybrid schema are as follows.

3.1.1 Key generation algorithm in RAM- RSA

In Scheme A, keys are generated by selecting two numbers k_{p1} and k_{q1} of 160 bits, relatively prime to e, to calculate d_p and d_q and further d_p and d_q are used to generate the two primes p and q. To modify this key generation scheme to suite Mprime variant, we could repeat steps 2,3 and 4 in Schema A algorithm k times by selecting k numbers (k_{p1} , k_{p2}, …, k_{pk}) relatively prime to e, and to calculate d_{p1} , d_{p2}, d_{p3}, …, d_{pk} and to use these further to generate k primes p_1, p_2, …, p_k.

The key generation algorithm takes the following steps:

1. Randomly select an odd number e of 568 bits.
2. Set i = 1.
3. While (i ≤ k) do
4. Randomly select a number k_{pi} of 160 bits, such that gcd (k_{pi}, e) = 1.
5. Based on Theorem 1, determine two numbers d_{pi}, $k_{pi} < d_{pi} < 2\ k_{pi}$, and P, e < P < 2e, satisfying $e * d_{pi} = k_{pi} * p + 1$.
6. Factor P as $P = k_{pn} \cdot \grave{p}$ such that k_{pn} is a number of 56 bits and $p_i = (\grave{p} + 1)$ is a prime number. If this is infeasible, then go to Step 4, else i = i + 1.
7. Return (d_{p1} , d_{p2}, d_{p3}, …, d_{pk} , p_1, p_2, …, p_k , e)

The public key is (N, e); the secret key is (d_{p1} , d_{p2}, d_{p3}, …, d_{pk} , p_1, p_2, …, p_k)

3.1.2 Encryption and decryption of RAM- RSA

Encrypting with the public key <N, e> is identical to the original RSA, $C = M^e \bmod N$. However, as in Rebalanced RSA Schema A, the public exponent *e* is smaller than the normally used in Rebalanced RSA.

To decrypt a cipher text C, first we calculate $M_i = C^{di} mod\ p_i$ for each i, i≤1≤ k. Next, we apply the CRT to the M_i's to get $M = C^d\ mod\ \ N$. The difference between RAM-RSA and Mprime in the decryption algorithm is that, in Mprime d_i was used to decrypt the cipher text and it's calculated from d and p_i's, where each d_i length can be 512 or less, but in RAM- RSA we use dp_i's to decrypt cipher text as follows:

Step1. Calculate $M_i = C^{dp} mod\ p_i$ for each i, 1≤ i≤k.
Step2. Apply the CRT to the M_i's to get $M = C^d\ mod\ \ N$.

3.1.3 Improving RAM-RSA encryption

From the above theory the encryption of RAM-RSA is identical to that of RSA schema A, due to that fact that e and k_{pi} are selected with the same lengths in both algorithms. To improve encryption in RAM-RSA we modify the key generation as follows; so that user can control the size of *e* and the number of multiple prime's *k*, and accordingly a fast encryption can be achieved:

Step1: Set values of n, s, and k.
Step2. Randomly select an odd number e of s bits (s < 568).
Step3. Set i = 1.
Step4. While (i ≤ k) do
Step5. Randomly select numbers kp_i .of 160 bits, such that gcd (kp_i, e) = 1.
Step6. Based on Theorem 1, we can uniquely determine two numbers dp_i, $kp_i < dp_i < 2kp_i$, and pi, $e < p_i^{\setminus} < 2e$, satisfying $e * d_{pi} - k_{pi} * p_i^{\setminus} = 1$.
Step7. If $p_i = p_i^{\setminus} + 1$ is not a prime number, then go to Step4, else i = i + 1
Step8. Return (d_{p1}, …….. …d_{pk}, p_1, …p_k ,e)

The public key is (N, e); the secret key is (d_{p1} , d_{p2}, d_{p3}, …, d_{pk} , p_1, p_2, …, p_k).

3.2 RBM-RSA variant

RBM-RSA is similar to RAM-RSA, except that the Rebalanced RSA Scheme B is used instead of Scheme A. The motivation behind this variant is the fact that scheme B was shown by its developers to be a bit faster than scheme A in encryption time.

3.2.1 Key generation algorithm of RBM-RSA

The key generation algorithm in RBM-RSA is similar to that of Rebalanced RSA-CRT Scheme B, but instead of using two prime number p and q and use k_p and k_q and use d_p and d_q , we will use k primes numbers and therefore multi k_p's.

The key generation algorithm takes the following steps:

Step 1. Randomly select an odd number e of 512 bits.
Step 2. Set i = 1.
Step3. While (i ≤ k) do
Step4. Randomly select numbers k_{pi} of 198 bits, such that gcd (k_{pi}, e) = 1.
Step5. Based on Theorem 1, we can uniquely determine two numbers d_{pi}, $k_{pi} < d_{pi} < 2k_{pi}$, and p_i , $e < p_i^{\setminus} < 2e$, satisfying $e * d_{pi} - k_{pi} * p_i^{\setminus} = 1$.
Step6. If $p_i = p_i^{\setminus} + 1$ is not a prime number, , If this is infeasible, then go to Step 4, else i = i + 1
Step7. Return (d_{p1}, …d_{pk}, p_1, …p_k ,e).

The public key is (N, e); the secret key is (d_{p1},d_{pk}, p_1, …p_k).

3.2.2 Encryption and decryption in RBM-RSA

Encrypting and decryption steps are the same as RAM-RSA variant.

3.2.3 Improving RBM-RSA encryption

As in RAM-RSA, encryption of RBM-RSA is identical to that of RSA schema B, due to that fact that e and k_{pi} are selected with the same lengths in both algorithms. To improve encryption in RBM-RSA we modify the key generation as follows; so that user can control the size of *e* and the number of multiple prime's *k*, and accordingly a fast encryption can be achieved:

Step1: Set values of n, s, and k.
Step2. Randomly select an odd number e of s bits.
Step3. Set i = 1.
Step4. While ($i \leq k$) do
Step5. Randomly select numbers kp_i of 198 bits, such that $\gcd(kp_i, e) = 1$.
Step6. Based on Theorem 1, we can uniquely determine two numbers d_{pi}, $k_{pi} < d_{pi} < 2k_{pi}$, and p_i, $e < p_i^{\setminus} < 2e$, satisfying $e * d_{pi} - k_{pi} * p_i^{\setminus} = 1$.
Step . If $p_i = p_i^{\setminus} + 1$ is not a prime number, , If this is infeasible, then go to Step 4, else i = i + 1
Step8. Return (d_{p1}, …d_{pk}, p_1, …p_k ,e)

The public key is (N, e); the secret key is (d_{p1}, ...d_{pk}, p_1, …p_k).

4 Experiments and Results

To evaluate the performance of the developed RSA variants, we compare the encryption and decryption time of the direct hybrid variants, the modified variants and RPrime RSA variant.

The codes of all variants are implemented in Java 7, running on top a laptop with processor Intel (R) Core (TM) i3 -2350M CPU @2.30 GHz and 4.00 GB RAM.

A data block of size 64 bits is used in testing, and each algorithm is executed 10 times and the average running time calculated and compared. Tables 1 -5 shows the average running time of each variant, and Table 6 shows the comparison result.

Table 1. The results of testing the Rprime RSA variant, with key =3072

Ex-no	Encryption time	Decryption time
1	748 ms	16 ms
2	733 ms	32 ms
3	717 ms	15 ms
4	765 ms	31 ms
5	733 ms	32 ms
6	748 ms	32 ms
7	733 ms	16 ms
8	748 ms	32 ms
9	734 ms	31 ms
10	733 ms	31 ms
Average	739.2 ms	26.8 ms

Table 2. The results of testing the RAM-RSA variant, with key =3408

Ex-no	Encryption time	Decryption time
1	219 ms	0 ms
2	203 ms	15 ms
3	187 ms	15 ms
4	202 ms	0 ms
5	203 ms	15 ms
6	280 ms	16 ms
7	219 ms	15 ms
8	218 ms	16 ms
9	125 ms	15 ms
10	156 ms	16 ms
Average	201.2 ms	12.3 ms

Table 3. The results of testing the modified RAM- RSA with k=6, s= 512, n=3072

Ex-no	Encryption time	Decryption time
1	125 ms	0 ms
2	172 ms	0 ms
3	156 ms	0 ms
4	188 ms	15 ms
5	218 ms	16 ms
6	187 ms	16 ms
7	141 ms	0 ms
8	171 ms	16 ms
9	156 ms	0 ms
10	141 ms	15 ms
Average	165.5 ms	7.8 ms

Table 4. The results of testing the RBM-RSA variant, with key =3072

Ex-no	Encryption time	Decryption time
1	124 ms	0 ms
2	125 ms	0 ms
3	124 ms	16 ms
4	140 ms	16 ms
5	141 ms	0 ms
6	124 ms	16 ms
7	141 ms	15 ms
8	125 ms	15 ms
9	125 ms	0 ms
10	140 ms	16 ms
Average	130.9 ms	9.4 ms

Table 5. The results of testing the modified RBM-RSA with k=6, s=512, n =3072

Ex-no	Encryption time	Decryption time
1	156 ms	16 ms
2	125 ms	16 ms
3	140 ms	16 ms
4	141 ms	15 ms
5	94 ms	16 ms
6	109 ms	0 ms
7	109 ms	16 ms
8	140 ms	0 ms
9	140 ms	0 ms
10	124 ms	0 ms
Average	127.8 ms	9.5 ms

Table 6. The comparison results

Variant	Key size (bits)	Encryption time	Decryption time
RPrime RSA	3072	739.2 ms	26.8 ms
RAM-RSA	3408	201.2 ms	12.3 ms
Modified RAM-RSA	3072	165.5 ms	7.8 ms
RBM-RSA	3072	130.9 ms	9.4 ms
Modified RBM-RSA	3072	127.8 ms	9.5 ms

4.1 Results

From Table 6 we deduce the following:

1. All developed variant have faster encryption time than RPrime RSA.
2. RAM-RSA and RBM-RSA are 3.4 and 2.8 times faster in decryption than Rprime RSA respectively.
3. RAM-RSA and RBM-RSA are 4.5 and 5.8 times faster in encryption than Rprime RSA respectively
4. These results are concise with the theoretical expected ones, that is, RBM-RSA must be faster than RAM-RSA in encryption and slower in decryption.

5 Conclusion

This paper proposed a new fast RSA variant that combines features of previously developed RSA variants. The new variants improve both encryption time and decryption time.

The new variants represent good choices for applications that need high encryption and decryption speeds.

Competing Interests

Authors have declared that no competing interests exist.

References

[1] Rivest R, Shamir A, Adleman L. A method for obtaining digital signatures and public key cryptosystems. Communications of the ACM. 1978;21(2):120-126.

[2] Quisquater J, Couvreur C. Fast decipherment algorithm for RSA public key cryptosystem. Electronic Letter Philips Research Laboratory, Brussels, Belgium, Electronics Letters. 02/1982; DOI: 10.1049/el:19820617, Source: IEEE Xplore. 1982.

[3] Wiener M. Cryptanalysis of short RSA secret exponents. IEEE Transactions on Information Theory. 1990;36(3):553–558.

[4] Sun H, Mu W. Design of rebalanced RSA-CRT for fast encryption. Proceedings of Information Security Conference. 2005;16-27.

[5] Collins T, Hopkins D. Langford S, Sabin M. Public key cryptographic apparatus and method. US Patent #5,848,159; 1997.

[6] Cesar A. An efficient variant of the RSA cryptosystem. Preprints; 2002.

[7] Boneh D, Durfee G. Cryptanalysis of RSA with private key d less than $n^{0.292}$. IEEE Transactions on Information Theory. 2000;46(4):1399-1349. Inc.

[8] Niven I, Zuckerman H. An Introduction to the theory of number. John Wiley and Sons; 1991.

[9] Mamun A, Islam M, Romman S, Ahmad A. Performance evaluation of several efficient RSA variant. IJCSNS. 2008;8(7):7-11.

Peer-review history:
The peer review history for this paper can be accessed here (Please copy paste the total link in your browser address bar)
http://sciencedomain.org/review-history/12991