Improvement in Rebalanced CRT RSA

Seema Verma and Deepak Garg
Department of Computer Science and Engineering, Thapar University, India

Abstract: *Many improvements have been made since the RSA origin in terms of encryption/decryption speed and memory saving. This paper concentrates on the performance improvement. Rebalanced RSA is designed to improve the decryption speed at the cost of encryption speed. Further work was done to improve its encryption speed in terms of rebalanced Chinese Remainder Theorem (CRT) variants. Rebalanced CRT variants improved the encryption speed at the cost of decryption speed. This paper also improves the performance of the encryption side in rebalanced RSA, while still maintaining the same decryption speed as in rebalanced RSA by adding the multiprime RSA feature to the rebalanced CRT variant. Proposed scheme gains the same advantage in encryption side as in rebalanced CRT variants, besides it is 2 times faster at decryption side than rebalanced CRT variants. Due to the use of multiprime feature, the key generation time is also decreased in this case. It is decreased approximately by a factor of 2.39 from rebalanced RSA CRT variant. Comparison of the RSA variants with the new scheme is shown in tabular and graphical way for better analysis.*

Keywords: *Cryptography, computational complexity, encryption, public key.*

Received June 9, 2012; accepted April 18, 2013; published online December 3, 2014

1. Introduction

Thirty five years after the origin, RSA cryptosystem [15] is still a de-facto standard in all branches of public key cryptography. Using the standard double and square method for modular exponentiation, which is commonly used, the cost of encryption and decryption is roughly proportional to the number of bits in e (public key) and d (private key), respectively.

Boneh and Shacham [2] gave a very nice comparison of the variants of RSA (batch RSA [5], Mprime RSA [18], Mpower RSA [3] and rebalanced RSA [20]). All these variants are improving the decryption/signature verification performance. Their work was further extended by Paixao and Gazzoni [11]. He combined the two variants of RSA, multiprime RSA and rebalanced RSA and gave the performance by improving the decryption/signature generation speed by 27 times to RSA and by 4.8 times to RSA with Chinese Remainder Theorem (CRT) for 2048bit modulo. Rebalanced RSA, speeds up RSA decryption by shifting decryption costs to encryption costs. Hence, encryption speed decreases. In [16] improvement was done in encryption speed, two variants of rebalanced RSA were introduced in which the public exponent e is much smaller than the modulus, thus reducing the encryption costs, while still maintaining low decryption costs. This paper was modified in [17]. Our work concentrates on the encryption and decryption speed. In our work, multiprime feature is added into the schemes of rebalanced RSA and the decryption speed enhances for the same gain of encryption speed in [17].

The paper is organized as follows: In section 2 RSA and its variants are reviewed. Section 3 describes the proposed algorithm. Algorithm comparison with other variants is shown theoretically in tabular and graphical manner in section 4. In section 5, the implementation aspects of the new algorithm are given and the comparison with the other variants is shown. The conclusion is given in the last section.

Throughout the complete paper, following parameters are used:

- n= No of bits in modulus.
- n_e= No of bits in public exponent (e).
- n_d= No of bits in private exponent (d).

2. RSA and its Variants

2.1. Basic RSA

Rivest *et al.* [15] described the RSA algorithm as shown in Figure 1. The keys are derived from the multiple of two large prime numbers. The private key can only be deduced from the public key by factoring the large multiple.

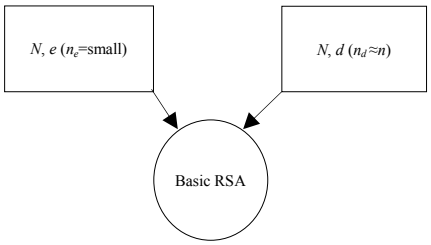

Figure 1. Basic RSA.

- Key Generation Method:
  1. Generate two large random primes of n/2 bits each, p and q of approximately equal size such that their product $N= p*q$ is of the required bit length, e.g., 1024 bits.
  2. Compute $N= p*q$ and $\varphi(N)= (p-1)(q-1)$.

3. Choose an integer e, $1<e<\varphi(N)$, such that $gcd(e, \varphi(N))=1$.
4. Compute the secret exponent d, $1<d<\varphi(N)$, such that: $ed\equiv 1 (mod\ \varphi(N))$, public key= (N, e), private key= (N, d).

- Encryption Method:
  1. Obtains the recipient B's public key (N, e).
  2. Represents the plaintext message as a positive integer M.
  3. Computes the cipher text $C= M^e\ mod\ N$.
  4. Sends the cipher text C to B.

Very elegant description of the complexity is given in [19].

$$Encryption\ Complexity = (3n_e\text{-}2)(n^2+n) \quad (1)$$

- Decryption Method:
  1. Uses his private key (N, d) to compute $M= C^d\ mod\ N$.
  2. Extracts the plaintext from the message representative M.

Here, the iteration is done n_d times (n_d= No of bits in d) and $n_d \approx n$ So:

$$Decryption\ Complexity = (3n - 2)(n^3+n) = 3n^3+n^2+o(n^2) \quad (2)$$

2.2. RSA with CRT

This method [14], Figure 2, is used to improve the computational cost at the decryption side of the RSA, by dividing d into two parts (d_p, d_q).

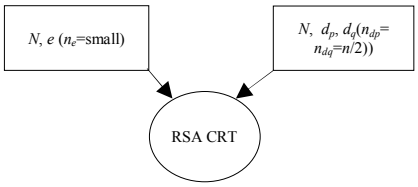

Figure 2. RSA CRT.

- Key Generation Method:
  Same as the basic RSA.
- Encryption Method:
  Same as the basic RSA.

  Complexity is same as for the basic RSA.
- Decryption Method:
  1. Calculate $d_p= d\ mod\ p$-1 and $d_q= d\ mod\ q$-1.
  2. Calculate $M_p= C^{dp}\ mod\ p$ and $M_q= C^{dq}\ mod\ q$.
  3. Calculate M from M_p and M_q using CRT.

$$Complexity\ of\ Decryption\ Algorithm = 3n^2/4+7n^2/2+o(n^2) \quad (3)$$

2.3. MultiPrime RSA

In this variant [18] Figure 3, the RSA modulus was modified so that it can further decrease the decryption time. It consists of k primes $p_1, p_2, \ldots, p_k$ instead of using only two.

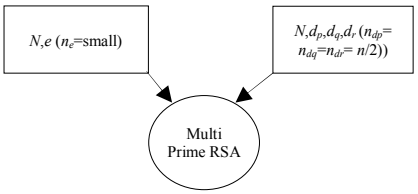

Figure 3. Multiprime RSA.

- Key Generation Method: k= No. of primes to be used.
  1. Compute k distinct primes $p_1, \ldots, p_k$ each one [$log\ N/k$] bits in length and $N=\Pi_{i-1}^{\ k} p_i$.
  2. Compute e and d such that $d=e^{-1}\ mod\ \varphi(N)$, where $gcd(e, \varphi(N))=1$, $\varphi(N)=\Pi_{i-1}^{\ k}(p_i\text{-}1)$.
  3. For $1\leq i\leq k$, compute $d_i= d\ mod\ (p_i\text{-}1)$.
     Public key= (N, e).
     Private key= $(d_1, d_2, ..., d_k)$.
- Decryption Method: Here, k=3.
  1. Calculate $d_p= d\ mod\ p$-1, $d_q= d\ mod\ q$-1 and $d_r=d\ mod\ r$-1.
  2. Calculate $M_p= c^{dp} mod\ p$, $M_q= c^{dq} mod\ q$, $M_r= c^{dr} mod\ r$.
  3. Calculate M from M_p, M_q and M_r using CRT

$$Decryption\ Complexity = (3*(n\text{-}n/3)(n/3+2))+(3*(3*(n/3)^3+(n/3)^2) \quad (4)$$
$$+16n^2/3+o(n^2) = n^3/3+19n^2/3+o(n^2)$$

2.4. Rebalanced RSA

This variant [20] Figure 4, also decreases the decryption time, but at the cost of increase in encryption time.

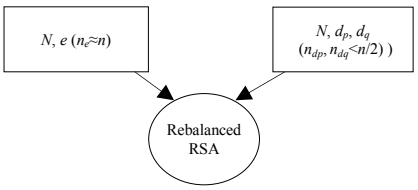

Figure 4. Rebalanced RSA.

- Key Generation Method: Take $s\leq n/2$ bits.
  1. Generate two distinct random $(n/2)$bit prime numbers p and q with $gcd(p\text{-}1, q\text{-}1)=2$ and calculate $N= p*q$.
  2. Generate two s-bits random numbers d_p and d_q, such that $gcd(d_p, p\text{-}1)=1$, $gcd(d_q, q\text{-}1)=1$ and $d_p= d_q\ mod\ p$-1.
  3. Calculate one d such that $d= d_p\ mod\ p$-1 and $d= d_q\ mod\ q$-1.
  4. Calculate $e=d^{-1}\ mod\ \varphi(N)$.
     Public key= (N, e).
     Private key= (p, q, d_p, d_q).
- Encryption Method:
  Same as the basic RSA.

$$Encryption\ Complexity = (3n_e - 2)(n^2+n) = 3n^3+n^2+o(n^2) \quad (5)$$

Where $n_e \approx n$

- Decryption Method: Same as RSA CRT.

$$\begin{aligned} Decryption\ complexity &= 2*((3*s-2)(n^2/4+n/2))+5n^2/2+o(n^2) \\ &= 483n^2/2+o(n^2) \end{aligned} \quad (6)$$

Here, s= 160 bits.

2.5. Rprime RSA

This variant [11] Figure 5, further improves the computational cost at the decryption side by combining rebalanced RSA and Mprime RSA.

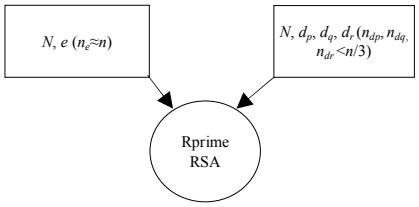

Figure 5. Rprime RSA.

- Key Generation Method: Take s, an integer, $s \leq n/k$ and executes the following steps:
  1. Generate k distinct random primes of n/k bits p_1, p_2, ..., p_k with $gcd(p_1\text{-}1, p_2\text{-}1, \ldots, p_k\text{-}1)$= 1 and calculate $N= p_1, p_2, \ldots, p_k$.
  2. Generate k random numbers of $s$bits d_{p1}, d_{p2}, ..., d_{pk}, such that: $gcd(dp_1, p_1\text{-}1)=1$, $gcd(dp_2, p_2\text{-}1)=1$, ..., $gcd(dp_k, p_k\text{-}1)=1$ and $dp_1= dp_2= \ldots, dp_k \ mod\ 2$.
  3. Find d such that $d= d_{p1}\ mod\ (p_1\text{-}1)$, $d= d_{p2}\ mod\ (p_2\text{-}1)$, ..., $d= d_{pk}\ mod\ (p_k\text{-}1)$.
  4. Calculate $e=d^{-1}\ mod\ \varphi(N)$.
     Public key= (N, e).
     Private key= $(p_1, p_2, \ldots p_k, d_{p1}, d_{p2}, \ldots d_{pk})$.
- Encryption Method:
  Same as the MultiPrime RSA.

$$Encryption\ \ Complexity = (3n_e-2)(n^2+n)=3n^3+n^2+o(n^2) \quad (7)$$

- Decryption Method:
  Same as the Multiprime RSA.

$$\begin{aligned} Decryption\ Complexity &= 3*((3*s-2)(n^2/9+n/3))+16n^2/3+o(n^2) \\ &= 494n^2/3+o(n^2) \end{aligned} \quad (8)$$

Here, s=160bits.

2.6. Rebalanced RSA Variants

This variant [17] Scheme *B*, Figure 6, is also improving the computational cost of RSA algorithm. It is a further improvement of rebalanced RSA, so that the increased encryption cost be lowered to balance both encryption and decryption cost.

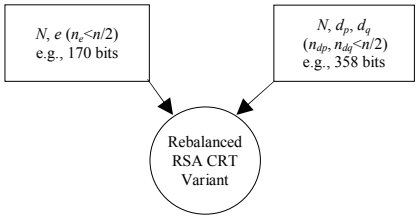

Figure 6. Rebalanced RSA CRT variant.

This algorithm do not only shorten the public exponent in Rebalanced RSA-CRT from 1024bits down to 512bits, but also make the public exponent to be of the special form of $2m+1$ where m=511. Scheme-A produces a 512bit public exponent, e.g., $e= 2^{511}+1$, two 198bit CRT exponents d_p, d_q and an RSA modulus $N=pq$, where p and q are about of 512bits. The encryption time is therefore reduced to about one-third of the time required by rebalanced RSA-CRT. This RSA variant can be widely applied to several systems and software.

The key generation algorithm is based on the following fundamental theorem from number theory [8]:

- ***Theorem A***: Let a and b be relatively prime integers unequal to 1 (i.e., $gcd(a, b)=1$ and $a, b \neq 1$). For every integer h there exists a unique n_d, efficiently computable pair of integers satisfying $au_h\text{-}bv_h=1$, where $(h\text{-}1)b<u_h<hb$ and $(h\text{-}1)a<v_h<ha$.

- Key Generation Method: Here, $e<n/2$.
  1. e= Random (n_e)bit odd integer.
  2. d_{p1}= Random $(n/2\text{-}n_e)$bit odd integer, $E_p= ed_{p1}$.
  3. K_p= Random $(n_e+n_d\text{-}n/2)$bit integer satisfying $gcd(k_p, E_p)=1$.
  4. With h=2, compute (d_{p2}, p_1) satisfying $E_pd_{p2}= k_pp1+1$, where $k_p<d_{p2}< 2k_p$, $E_p< p_1< 2E_p$ and let $d_p= d_{p1}d_{p2}$.
  5. $p= p_1+1$, If p is not prime, then go to step 3.
  6. $d_{q1}= Random(n/2\text{-}n_e)$bit odd integer, $E_q= ed_{q1}$.
  7. $K_q=Random(n_e+n_d\text{-}n/2)$bit integer satisfying $gcd(k_q, E_q)=1$.
  8. With h=2, compute (d_{q2}, q_1) satisfying $E_qd_{q2}= k_qq_1+1$, where $k_q< d_{q2}< 2k_q$ and $E_q< q_1< 2E_q$, $d_q= d_{q1}d_{q2}$.
  9. $q= q_1+1$, if q is not prime, then go to Step 7.
     Public key = (e, N).
     Private key = (d_p, d_q, p, q).

- Encryption Method: Same as rebalanced RSA.
- Decryption Method: Same as rebalanced RSA. For this scheme, n_e= 170bits and n_d= 358bits.

$$Encryption\ Complexity = (3n_e\text{-}2)(n^2+n)=508n^2+o(n^2) \quad (9)$$

$$\begin{aligned} Decryption\ Complexity &= 2*((3*n_d-2)(n^2/4+n/2))+5n^2/2+o(n^2) \\ &= 1077n^2/2+o(n^2) \end{aligned} \quad (10)$$

3. Proposed Scheme

3.1. Algorithm

In our scheme, Figure 7 multiple prime feature is added to rebalanced RSA to increase the decryption side of the rebalanced RSA CRT variant and due to the use of smaller primes it also reduces the key generation time.

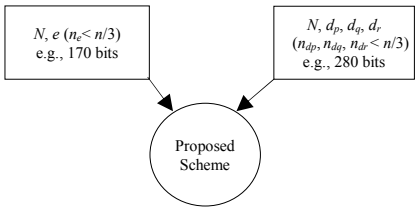

Figure 7. Proposed Scheme.

The key generation algorithm, takes (n, n_e, n_d) as input, with $n_e < n/3$ and outputs a valid public key $<e, N>$ and corresponding private key $<d_p, d_q, d_r, p, q, r>$, where $|N| = n|$, $|e| = n_e$ and $|d_p| = |d_q| = |d_r| = n_d$. The algorithm is as follows:

- Key Generation Method: Here, $e < n/3$.

1. e= Random (n_e)bit odd integer.
2. d_{p1}= Random ($n/2$-n_e)bit odd integer, $E_p = ed_{p1}$.
3. k_p= Random (n_e+n_d-$n/3$)bit integer satisfying $gcd(k_p, E_p)=1$.
4. With h=2, compute (d_{p2}, p_1) satisfying $E_p d_{p2} = k_p p1+1$, where $k_p < d_{p2} < 2k_p$, $E_p < p_1 < 2E_p$ and let $d_p = d_{p1}d_{p2}$.
5. $p = p_1+1$. If p is not prime, then go to Step 3.
6. d_{q1}= Random($n/3$-n_e)bit odd integer, $E_q = ed_{q1}$.
7. k_q= Random (n_e+n_d-$n/3$)bit integer satisfying $gcd(k_q, E_q)=1$.
8. With h=2, compute (d_{q2}, q_1) satisfying $E_q d_{q2} = k_q q1+1$, where $k_q < d_{q2} < 2k_q$ and $E_q < q_1 < 2E_q$, $d_q = d_{q1}d_{q2}$.
9. $q = q_1+1$. If q is not prime, then go to Step 7.
10. d_{r1} = Random ($n/2$-n_e)bit odd integer, $E_r = ed_{r1}$.
11. K_r= Random (n_e+n_d-$n/3$)bit integer satisfying $gcd(k_r, E_r)=1$.
12. With h=2, compute (d_{r2}, r_1) satisfying $E_r d_{r2} = k_r r_1+1$, where $k_r < d_{r2} < 2k_r$ and $E_r < r_1 < 2E_r$, $d_r = d_{r1}d_{r2}$.
13. $r = r_1+1$. If r is not prime, then go to Step 10.

Public key = (e, N).

Private key (d_p, d_q, d_r, p, q, r).

3.2. Verification of the Key Equation

The key equation of the basic RSA algorithm is: $ed = k\varphi(N)+1$. To show that the output (public and private) keys are valid, let $k_p = k_{p1}k_{p2}$, $k_q = k_{q1}k_{q2}$, $k_r = k_{r1}k_{r2}$. The equation defined for RSA CRT are:

$$ed_p = k_p(p-1)+1 \quad (11)$$

$$ed_q = k_q(q-1)+1 \quad (12)$$

$$ed_r = k_r(r-1)+1 \quad (13)$$

Multiplying these three equations together yields:

$$(ed_p\text{-}1)(ed_q\text{-}1)(ed_r\text{-}1) = k_p(p\text{-}1)k_q(q\text{-}1)k_r(r\text{-}1) \quad (14)$$

After rearrangement:

$$e(e^2d_pd_qd_r\text{-}ed_pd_r\text{-}ed_qd_r\text{-}ed_pd_q+d_p+d_q+d_r) = k_pk_qk_r(p\text{-}1)(q\text{-}1)(r\text{-}1)+1 \quad (15)$$

Here,

$$d_0 = e^2d_pd_qd_r\text{-}ed_pd_r\text{-}ed_pd_r\text{-}ed_qd_r\text{-}ed_pd_q+d_p+d_q+d_r$$

And

$$k = k_pk_qk_r$$

Hence, e and d_0 are the valid keys for the algorithm satisfying the equation $ed_0 = k_0\varphi(N)+1$.

Thus, e and d_0 are a valid public/ private exponent pair for the modulus $N = pqr$.

3.3. Security Analysis

Coppersmith gave a new direction in cryptanalysis of RSA and its variants by lattice reduction method. After Coppersmith *et al.* [4, 6, 7] have given their analysis of lattice attack.

The method basically find the roots of polynomial modular equation using lattice reduction. Multivariate linear polynomials with m variables can be solved by generating m algebraically independent equations and then finding the roots of these equations, proving of independence of polynomial is still an open problem.

- ***Theorem* 1.** *Four Variate Linear Modular Equations*: Jutla [7]. Let $f(x, y, z, w)$ be a linear polynomial with integer coefficients. For every $\varepsilon>0$, there exists a positive M_0 such that for every integer $M > M_0$ that is relatively prime to at least one non-constant coefficient of f we can find four linearly independent polynomials such that each root (x_0, y_0, z_0, w_0) of $f(x, y, z, w)$ $(mod\ M)$ is also a root of each the four polynomials modulo M, and if $|x_0|<X$, $|y_0|<Y$, $|z_0|<Z$, $|w_0|<W$ and $XYZW<M^{1-\varepsilon}$ for some bounds X, Y, Z and W, then (x_0, y_0, z_0, w_0) is also a root of each of the four polynomials over the integers. If these four polynomials are also algebraically independent then we can compute (x_0, y_0, z_0, w_0).
- ***Corollary* 1.1.** *Trivariate Linear Modular Equations*: Let $f(x, y, z)$ be a linear polynomial with integer coefficients. For every $\varepsilon>0$, there exists a positive M_0 such that for every integer $M > M_0$ that is relatively prime to at least one non-constant coefficient of f we can find three linearly independent polynomials such that each root (x_0, y_0, z_0) of $f(x, y, z)$ $(mod\ M)$ is also a root of each the three polynomials modulo M and if $|x_0|< X$, $|y_0|<Y$, $|z_0|<Z$ and $XYZ<M^{1-\varepsilon}$ for some bounds X, Y and Z, then (x_0, y_0, z_0) is also a root of each of the three polynomials over the integers. If these three polynomials are also algebraically independent then we can compute (x_0, y_0, z_0).
- ***Corollary* 1.2.** *Bivariate Linear Modular Equations*: Let $f(x, y)$ be a linear polynomial with integer coefficients. For every $\varepsilon>0$, there exists a positive M_0 such that for every integer $M > M_0$ that is relatively prime to at least one non-constant coefficient of f we can find two linearly independent polynomials such that each root(x_0, y_0) of $f(x, y)$ $(mod\ M)$ is also a root of each the two polynomials modulo M and if $|x_0|< X$, $|y_0|<Y$ and $XY<M^{1-\varepsilon}$ for some bounds X and Y, then (x_0, y_0) is also a root of each of the two polynomials

over the integers. If these two polynomials are also algebraically independent then we can compute (x_0, y_0).

- ***Howgrave Graham Theorem***: Let $h(x_1, ..., x_n) \in Z[x_1, ..., x_n]$ be the sum of at most w monomials and let m be a positive integer. Suppose that there exists $y_1, ..., y_n \in Z$ such that:

$$h(y_1, ..., y_n) \equiv 0 (mod\ M^m) \quad (16)$$

And

$$\|h(x_1X_1, ..., x_nX_n)\|^2 < w^{0.5}M^m \quad (17)$$

Where $|y_1| < x_1, ..., |y_n| < x_n$ for some positive bounds $x_1, ..., x_n$. Then, $h(y_1, ..., y_n)=0$ over the integers.

- ***Proof of Theorem* 1.** Let M be a positive integer with unknown factorization and consider the linear integer polynomial:

$$f(x, y, z, w) = Ax + By + Cz + Dw + E \in Z[x, y, z, w] \quad (18)$$

such that, at least one non-constant coefficient is relatively prime to M. We will assume, without loss of generality, that $gcd(D, M)=1$. Multiplying $f(x, y, z, w)$ by D^{-1} modulo M we obtain the new polynomial:

$$f_M(x, y, z, w) = f(x, y, z, w) = ax + by + cz + w + e \quad (19)$$

which has the same roots as $f(x, y, z, w)$ modulo M. We will look for all small roots of $f_M(x, y, z, w)$ modulo M. That is, given bounds X, Y, Z and W, we wish to find all $(x_0, y_0, z_0, w_0) \in Z^4$ such that: $|x_0| < X$, $|y_0| < Y$, $|z_0| < Z$, $|w_0| < W$ and $f_M(x_0, y_0, z_0, w_0) \equiv (mod\ M)$. In the following, let (x_0, y_0, z_0, w_0) be any such root. We will try to find the maximal bounds that lattice basis reduction techniques (based on Coppersmith's techniques) can allow.

For some positive integer m, to be determined later, we consider the set of polynomials $g_{i,j,k} = x^i y^j z^k M^{m-1} f^k M(x, y, z, w)$, for non-negative integer values of i, j, k, l. Notice that, for any values of $i, j \geq 0$ and $0 \leq k \leq m$, we have $g_{i, j, k, l}(x_0, y_0, z_0, w_0) \equiv 0 (mod\ M^m)$. We construct a lattice basis for a lattice L using the coefficients of $g_{i, j, k,l}(xX, yY, zZ, wW)$ for certain values of i, j, k and l. In particular, for each $0 \leq d \leq m$ we use all values $0 \leq l \leq d$ along with each combination of i, j and k. Table 1 shows the corresponding matrix. This lattice gives $w=m(m+1)(m+2)(m+3)/24$ linearly independent vectors which form the lattice basis for L. The structure of the basis matrix is shown in Table 1 (lower right-hand side of the table). Each row in the basis matrix is the coefficient vector of one of the $g_{i, j, k, l}(xX, yY, zZ, wW)$. The ordering of the columns ensures that the basis matrix is triangular and that the lattice is full dimensional (with dimension w). Since, the matrix is triangular, we can easily compute the determinant/ volume of the lattice (given by the absolute of the determinant of the basis matrix) yielding $det(L)=(XYZWM^4)^{m(m+1)(m+2)(m+3)(m+4)/120}=(XYZWM^4)^{w(m+4)/5}$, which simplifies to $det(L)<M^{m(w-2)}/(2^{(w-1(w-2)/4}w^{(w-2)/2})$. Applying the LLL-algorithm to the above lattice we can find four linearly independent vectors in L whose sizes are bounded by $2^{(w-1)/4}(det(L)^{1/(w-2)})$ where w is the lattice dimension defined above. Each of these vectors, being elements of L, correspond to a polynomial of the form $h(xX, yY, zZ, wW)$ such that determinant into this bound, and rearranging, for m and w large enough, $XYZW<M$. Whenever this equation is satisfied we can only guarantee that four linearly independent polynomials having the integer root (x_0, y_0, z_0, w_0) can be found. If the polynomials are further algebraically independent, we can use resultants to remove variables. A similar proof of corollary 3.2 and 3.3 can be given directly from Theorem 1.

Table 1. The matrix spanned by $g_{i, j, k, l}$ where "*" represents the non-zero entry.

	i, j, k, l	-	X	Y	Z	W	X^2	XY	Y^2	XZ	YZ	XW	YW...
D=0	0,0,0,0	M^m											
D=1	1,0,0,0		M^mX										
	0,1,0,0			M^mY									
	0,0,1,0				M^mZ								
	0,0,0,1	*	*	*	*	$M^{m-1}W$							
D=2	2,0,0,0						M^mX2						
	1,1,0,0							M^mXY					
	0,2,0,0								M^mY^2				
	1,0,1,0									M^mXZ			
	0,1,1,0										M^mYZ		
	1,0,0,1		*		*		*	*				$M^{m-1}XW$	
	0,1,0,1			*					*	*		*	$M^{m-1}YW$
	.....												

For a given linear trivariate polynomial $f(x, y, z)$ with desired solution (x_0, y_0, z_0) we can simply consider the linear trivariate polynomial $g(x, y, z, w)=f(x, y)+z+w$ and for bivariate polynomial $f(x, y)$ with desired solution (x_0, y_0), the equation can be considered as:

$$g(x, y, z, w) = f(x, y) + f(z, w) \quad (20)$$

That's why the proof is omitted here.

3.3.1. Security Parameters

From Equation 14:

$$e^3 d_p d_q d_r + e^2[d_p d_r(k_q - 1) + d_q d_r(k_p - 1) + d_p d_q(k_r - 1)] + e[d_p(k_q k_r - k_r - k_q + 1) + d_q(k_p k_r - k_r - k_p + 1) + d_r(k_p k_q - k_q - k_p + 1)] - k_p k_q k_r(N-1) + k_p k_q - k_q k_r - k_p k_r + k_p + k_q + k_r - 1 = 0 \quad (21)$$

This equation can be rewritten as:

$$e^3 d_p d_q d_r + e^2[d_p d_r(k_q - 1) + d_q d_r(k_p - 1) + d_p d_q(k_r - 1)] + e[d_p(k_q k_r - k_r - k_q + 1) + d_q(k_p k_r - k_r - k_p + 1) + d_r(k_p k_q - k_q - k_p + 1)] - k_p k_q k_r(N-1) + k_m + k_0 = 0 \quad (22)$$

In this equation d_p, d_q, d_r, k_p, k_q, k_r, k_0 are variables. K_m can be found by exhaustive search. Lattice attack can

be mounted on this equation by several ways. This equation can be viewed as a linear equation in four variable modulo e^3, as a linear equation in three variables *mod* e^2, as a linear equation in two variables mod e, as a linear equation in four variables modulo N. Considering the Equation 22 modulo e^3, the equation in four variables is obtained. The equation is: $f(x, y, z, w)= e^2x+ey-(N-1)z+w+K_m$ of which we can find the roots according to Theorem 1. Using the bounds:

$$X=d_pd_r(k_q-1)+d_qd_r(k_p-1)+d_pd_q(k_r-1)$$

$$Y=d_p(k_qk_r-k_r-k_q+1)+d_q(k_pk_r-k_r-k_p+1)+d_r(k_pk_q-k_q-k_p+1)$$

$$Z=k_pk_qk_r$$

$$W=k_0$$

We can obtain the following condition with which the roots of this equation can be found:

$$11n_d+5n_e<(8/3)n+m \tag{23}$$

Considering the Equation 22 modulo e^2, the equation in three variables is obtained, i.e., $f(x, y, z)= ex+(N-1)y+z+K_m$. Using corollary 3.2 we can find the roots of this equation. Using the bounds:

$$X=d_p(k_qk_r-k_r-k_q+1)+d_q(k_pk_r-k_r-k_p+1)+d_r(k_pk_q-k_q-k_p+1)$$

$$Y=k_pk_qk_r$$

$$Z=k_0$$

We can find the following condition:

$$8n_d+5n_e<(7/3)n+m \tag{24}$$

The Equation $f(x, y)=(N-1)x+y+k_m$ is obtained considering the Equation 22 modulo e. Using corollary 3.3 and bounds $X=k_pk_qk_r$, $Y=k_0$ we can find the following condition to solve the equation for the roots:

$$5n_d+4n_e<(5/3)n+m \tag{25}$$

lastly we consider Equation 22 modulo N and get the equation $f(x, y, z, w)= e^3x+e^2y+ez+w+K_m$, we can solve the equation by using bounds:

$$X=d_pd_qd_r$$

$$Y=d_pd_r(k_q-1)+d_qd_r(k_p-1)+d_pd_q(k_r-1)$$

$$Z=d_p(k_qk_r-k_r-k_q+1)+d_q(k_pk_r-k_r-k_p+1)+d_r(k_pk_q-k_q-k_p+1)$$

$$W=k_pk_qk_r+k_0$$

We find the following enabling condition:

$$12n_d+6n_e<3n+m \tag{26}$$

Small private exponent attack: Rearranging Equation 22:

$$e(e^2d_pd_qd_r-ed_pd_r-ed_qd_r-ed_pd_q+d_p+d_q+d_r)=1+k_pk_qk_r\varphi(N) \tag{27}$$

The small private exponent attacks on RSA present a more serious danger. We consider Boneh and Durfee's [1] lattice-based attack. Let $\Lambda=N-\varphi(N)=\Lambda_m+\Lambda_0$ and $M=N-\Lambda_m$, where Λ_m represents the m MSBs of Λ and Λ_0 represents the remaining bits. The Equation 27 can be rewritten as $ed_0=1+k_0(M-\Lambda_0)$. Assuming that M is known (i.e., Λ_m is known) and considering this equation as modulo e, the lattice based small private exponent attacks involve trying to find all integer pairs (x_0, y_0) such that $|x_0|<X$, $|y_0|<Y$ and $f(x_0, y_0)= x_0(M- y_0)\equiv 1 \ mod \ e$, where the bounds X and Y are defined so that, $X= |k_0|$, $Y= |\Lambda_0|$. Once we know Λ_0 we also know $\varphi(N)$ which then allows us to invert e modulo $\varphi(N)$; which breaks the system. Since, $log(e)=n_e$ the bounds become $log(X)=3n_e+3n_d-n$ and $log(Y)=(n/2-m)$. Using corollary 3.3, we find that a sufficient condition for the attack to succeed, is given by:

$$3n_d+2n_e<n/2+m \tag{28}$$

All the above attacks are not effective, because they results in number of equations less than the unknown parameters.

Baby-Step Giant-Step Attack [13] has the complexity $O(min\{d_p^{0.5}, d_q^{0.5}, d_r^{0.5}\})$ Due to this reason $n_d\geq 160$ bits:

$$n_d\geq 2m \tag{29}$$

3.3.2. Security Summary

Table 2 shows the summary of the various parameters we have discussed. Using these conditions, we have taken n_e=170 bits and n_d=280 bits.

As the inequalities from Equations 23 to 26 can be used to find the solution for X, Y and Z, but these values are not sufficient to retrieve any valuable information because of the three unknown variables in all the situation. So, all the inequalities except baby step giant step, in Table 2 causes a real attack on the system, one can use lower parameters than we have used here. Also, this scheme can be combined with the scheme [12] or [10], which will result in better offline encryption computations.

Table 2. Security Parameter.

Method	Inequality
Mod e^3	$11n_d+5n_e>(8/3)n+m$
Mod e^2	$8n_d+5n_e>(7/3)n+m$
Mod e	$5n_d+4n_e>(5/3)n+m$
Mod N	$12n_d+6n_e>3n+m$
Small d	$3n_d+2n_e>n/2+m$
Baby-step giant-step	$n_d>2m$

Encryption Complexity as shown in section 2:

$$(3n_e-2)(n^2+n)+o(n^2)=508n^2 \ (for 1024 bits)$$

Decryption Complexity as shown in section 2:

$$3*((3*s-2)n^2/9+n/3))+16n^2/3+o(n^2)=285n^2+o(n^2)$$

4. Comparison

Table 3 gives the encryption complexity comparison of the proposed scheme with the other variants of RSA cryptosystem. Here, n=1024bits.

Table 3. Encryption Complexity.

Variant	Encryption Complexity	
RSA	n_e=16 bits	46 n^2
RSA CRT	n_e=16 bits	46 n^2
MultiPrime RSA	n_e=16 bits	46 n^2
Rebalanced RSA	n_e≈1024 bits	3070n^2
RPrime RSA	n_e≈1024 bits	3070n^2
Rebalanced CRT Variant	n_e=170 bits	508 n^2
Proposed Scheme	n_e=170 bits	508 n^2

Table 4 gives the decryption complexity comparison of the proposed scheme with the other variants of RSA cryptosystem. Here, n=1024bits.

Table 4. Decryption Complexity.

RSA Variants	Decryption Complexity	
RSA	n_d≈ 1024 bits	3073 n^2
RSA with CRT	n_d≈ 512 bits	1543 n^2
MultiPrime RSA	n_d≈ 342 bits	347 n^2
Rebalanced RSA	n_d= 160 bits	242 n^2
RPrime RSA	n_d= 160 bits	165 n^2
Rebalanced RSA CRT Variant	n_d= 358 bits	536 n^2
Proposed Scheme	n_d= 280 bits	285 n^2

Figure 8 shows its corresponding values graphically. Horizontal rule is showing the different RSA variants and vertical rule is showing the theoretical complexities in n^2. Usage of three primes increase the computational speed at decryption side as well as it increases the security.

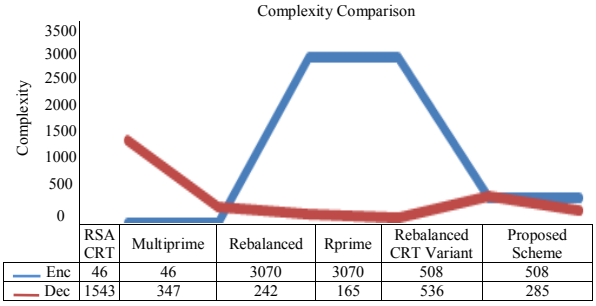

Figure 8. Comparison of Encryption and decryption complexities.

5. Implementation

To demonstrate the feasibility of the proposed scheme the key generation algorithm is implemented. A personal computer with 2.3GHz CPU and 5GB RAM was used. For the implementation NTL [9] with GMP using Cygwin tools on Windows operating system was used. The algorithm was run 100 times, it took 101ms. The average number of iteration for each loop of p, q, r was 275. The algorithm was implemented with (n_e, n_d)= (170, 280).

Other RSA variants (RSA CRT, Multiprime RSA, rebalanced RSA, Rprime, rebalanced RSA scheme B) were also, implemented with the same platform for comparison purpose. The Tabular comparison is shown in Table 5 and the graphical comparison as shown in Figure 9.

Table 5. Implementation Time in ms.

RSA Variant	Key Generation time (ms)	Encryption time (ms)	Decryption time (ms)
RSA CRT	212.7	0.1	1.25
MultiPrime RSA	77.3	0.1	0.62
Rebalanced RSA	334.4	3.7	0.46
RPrime	215.28	3.9	0.31
Rebalanced RSA Variant	256.6	0.78	0.84
Proposed Scheme	101.1	0.78	0.47

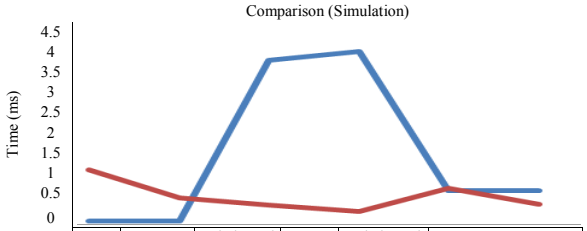

Figure 9. Run time comparison of the RSA variants.

It is clearly shown in the graph Figure 9 that the Encryption time is negligible in case of first two RSA variants but it rapidly increases with rebalanced RSA and Rprime RSA, rebalanced RSA variant further decreases the encryption value to balance both of these factors, i.e., encryption time and decryption time.

Our scheme further decreases this time maintaining the balancing of both the sides of encryption and decryption. Decryption time is decreased by a factor of 1.3, maintaining the balancing of encryption and decryption times. Due to the use of Multiprime features (p, q, r), the key generation time is also decreased in this case. It is decreased approximately by a factor of 2.39 from the previous scheme (i.e., rebalanced RSA variant).

In this implementation, n=1022, n_p=340, n_q=341, n_r=341, n_e=170, n_{dp}=280, n_{dq}=280, n_{dr}=280.

The values used were:

N=28096785309507025470676496057550409942275722500414352170637402215694143579378626792469275113805278547077173387182773954848795102969057760286880812455964585532367645292898337922242541578797335286669211590032930097028009372857216918554176634734125533736633219328433720339298580630453689250780403580054750078073.

p=22177690176171415709688735251990827103236995171916414287988382303552353294594475193737443467623806002777.

q=33775272282251677058964073241220693174810486097297442034676345944389669674929044533455698899980238252077.

r=37509516156424003319562727764590620273944722549154076055734496289448173920121908062477851043930051003077.

e=13794990386414784906476286377476812012954202981458 4.

6. Conclusions

As many applications have the requirement to balance both the encryption and decryption sides, e.g., both sender and receiver are having constrained resources. rebalanced RSA CRT variant balances both the encryption and decryption sides. Our scheme further improves the decryption side roughly by a factor of 2 still maintaining the same encryption speed.

Our scheme has the proportion of encryption cost and decryption cost is about 1.6. We can get a higher and lower proportion by adjusting the security parameter (n_e, n_d). As the Multiprime RSA variant has been already an active variant of RSA, so our scheme is applicable everywhere in the environments where this proportion 1.6 is needed. Our scheme is feasible in

any RSA embedded protocol that are used in the current environment, such as off-line generator, online proxy generators and so-on.

References

[1] Boneh D. and Durfee G., "Cryptanalysis of RSA with Private Key d Less Than $N^{0.292}$," *IEEE Transactions on Information Theory*, vol. 46, no. 4, pp. 1339-1349, 2000.

[2] Boneh D. and Shacham H., "Fast Variants of RSA," *CryptoBytes*, vol. 5, no. 1, pp. 1-9, 2002.

[3] Collins T., Hopkins D., Lanford S., and Sabin M., "Public Key Cryptographic Apparatus and Method," available at: http://www.google.com/patents/US5848159, last visited 2012.

[4] Coppersmith D., "Small Solutions to Polynomial Equations and Low Exponent RSA Vulnerabilities," *the Journal of Cryptology*, vol. 10, no. 4, pp. 233-260, 1997.

[5] Fiat A., "Batch RSA," *the Journal of Cryptology*, vol. 10, no. 2, pp. 75-88,1997.

[6] Howgrave-Graham N., "Finding Small Roots of Univariate Modular Equations Revisited," *in Proceedings of the 16th International Conference on Cryptography and Coding*, Cirencester, UK, pp. 131-142, 1997.

[7] Jutla C., "On Finding Small Solutions of Modular Multivariate Polynomial Equations," *in Proceedings of International Conference on the Theory and Application of Cryptographic Techniques*, Espoo, Finland, pp. 158-170, 1998.

[8] Niven I., Zuckerman H., and Montgomery H., *An Introduction the Theory of Numbers*, John Wiley and Sons Inc., 1991.

[9] NTL: A Library for Doing Number Theory., available at: http://shoup.net/ntl/, last visited 2008

[10] Padhye S., "On DRSA Public Key Cryptosystem," *the International Arab Journal of Information Technology*, vol. 3, no. 4, pp. 334-336, 2006.

[11] Paixao C. and Gazzoni Filho D., "An Efficient Variant of the RSA Cryptosystem," available at: http://www.lbd.dcc.ufmg.br/colecoes/sbseg/2005/0010.pdf, last visited 2013.

[12] Pointcheval D., "New Public Key Cryptosystem based on the Dependent-RSA Problem," *in Proceedings of International Conference on the Theory and Application of Cryptographic Techniques*, Prague, Czech Republic, pp. 239-254, 1999.

[13] Qiao G. and Lam Y., "RSA Signature Algorithm for Microcontroller Implementation," *in Proceedings of the 3rd International Conference on Smart Card Research and Applications*, Louvain-la-Neuve, Belgium, pp. 353-356, 2000.

[14] Quisquater J. and Couvreur C., "Fast Decipherment Algorithm for RSA Public-Key Cryptosystem," *Eletronic Letters*, vol. 18, no. 21, pp. 905-907, 1982.

[15] Rivest R., Shamir A., and Adleman L., "A Method for Obtaining Digital Signatures and Public Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, pp. 120-126, 1978.

[16] Sun H., Hinek M., and Wu M., "On the Design of Rebalanced RSA CRT," available at: http://www.cacr.math.uwaterloo.ca/techreports/2005/cacr2005-35.pdf, last visited 2013.

[17] Sun H., Hinek M., and Wu M., "Trading Decryption for Speeding Encryption in Rebalanced-RSA," *the Journal of Systems and Software*, vol. 82, no. 9, pp. 1503-1512, 2009.

[18] Takagi T., "Fast RSA-Type Cryptosystem Modulo $p^k q$," *in Proceedings of the 18th Annual International Cryptology Conference*, California, USA, pp. 318-326, 1998.

[19] Vuillaume C., "Efficiency Comparison of Several RSA Variants," *Master Thesis*, Darmstadt University, 2003.

[20] Wiener M., "Cryptanalysis of Short RSA Secret Exponents," *IEEE Transactions on Information Theory*, vol. 36, no. 3, pp. 553-558, 1990.

Seema Verma received her BTech. degree in computer science and engineering in 2001 and MTech degree in computer science and Engineering in 2007. Currently, she is pursuing her PhD degree in computer science and engineering and working as associate professor at Echelon Institute of Technology, Faridabad, affiliated to MDU India. She has more than 12 years of experience in academics. Her research interests include information security and cryptography.

Deepak Garg has done his PhD degree in the area of efficient algorithm design from Thapar University. He has more than 85 publications in International Journals and Conferences. He is chair, IEEE India Council Computer Society and Chair, IEEE India Council Education Society.