Two Efficient Variants of the RSA Cryptosystem

Qing Liu
School of Software
Yunnan University
Kunming, China
e-mail: liuqing@ynu.edu.cn

Yunfei Li,Lin Hao,Hua Peng
School of Information Science and Engineering
Yunnan University
Kunming, China
e-mail: liyunfei67@163.com

***Abstract*—Two variants are proposed to speed up RSA decryption in this paper. BMRSA (Batch Multi-Prime RSA) speeds up RSA decryption by combining the Multi-Prime RSA and Batch RSA. EAMRSA - Encrypt Assistant Multi-Prime RSA) improves RSA decryption performance based on the Multi-Prime RSA and RSA-S2 system[5]. The experimental results show that the speed of the two variants decryption has been substantially improved.**

Keywords-Multi-Prime RSA; RSA-S2 system; Batch RSA; acceleration

I. INTRODUCTION

The RSA cryptosystem [1] is one of the most widely used public key systems. But, when RSA decrypts the cipher text, more computation capacity and time will be required. Thus, some variants are proposed to speed up the RSA decryption, one of which is Multi-Prime RSA [2] [4]. It speeds up the RSA decryption by reducing the size of the moduli. Standard RSA decryption using Chinese Remainder Theorem (CRT) [9] requires two full exponentiations modulo *N/2*-bit numbers, where *N* is RSA moduli. In Multi-Prime RSA decryption requires *b* full exponentiations modulo *N/b* -bit numbers, where *b* is the number of the prime factors of *N*. This paper proposes two new variants of RSA based on the Multi-prime RSA to speed up Batch RSA [3][4] decryption and RSA-S2 system [5][10] decryption. These variants can not only speed up RSA decryption but also guarantee the security of RSA cryptosystem.

The experimental results show that the first variant for Batch RSA, for batch size is 4, can achieve average factor of 5.08 to standard RSA using CRT from 1024- to 3072-bit and another method can achieve the best factor of 7.06 speedup to standard RSA using CRT from 1795- to 3072-bit.

This paper is organized as follows. In section 2, two variants of RSA are proposed and security is also discussed in that section. In section 3 the experimental results are presented and the performance is analyzed. Section 4 concludes the paper.

II. THE PROPOSED VARIANTS

In this section, two variants of RSA that are respectively called BMRSA (Batch Multi-Prime RSA) and EAMRSA (Encrypt Assistant Multi-Prime RSA) are proposed. BMRSA effectively combines the Multi-Prime RSA and Batch RSA to speed up Batch RSA decryption. Multi-Prime RSA and RSA-S2 System in the EAMRSA are effectively combined to improve the performance of RSA decryption. New variants can obtain a higher speedup than the above original RSA variants. Firstly, the BMRSA will be described.

A. BMRSA

Batch RSA [3] [4] waits for more than one RSA decryption and performs one big computation for all decryption. It can spare a lot of running time capacity. Computation process is as a binary tree and performs computation in two phases, traversing the tree from leaf to root and then back again. The scheme only needs to perform a full exponentiation. In this subsection, the BMRSA for improving the full exponentiation is described with four phases: Setup, Percolate-Up, Exponentiation-Phase and Percolate-Down.

Setup: Given a security parameter n and two additional parameters b and l as input, b is the number of moduli N prime factors and l is the batch size.

- Compute b distinct primes $p_1,\cdots,p_b$ each one $\lfloor n/b \rfloor$ bits in length and generate $N=\prod_{i=1}^{b} p_i$, and $\varphi(N)=\prod_{i=1}^{b}(p_i-1)$.
- Taken l distinct and pairwise relatively prime public keys $e_1,\cdots,e_l$ as input. Each e_i computes, for $1\le i\le l$, $d_i=e_i^{-1} \bmod \varphi(N)$,at same time, compute $d=d_1\cdot d_2\cdots d_l \bmod \varphi(N)$.
- Compute $d_{pi}=d \bmod p_i-1$, for $1\le i\le b$. The private key is $<N,d_{p1},...,d_{pb}>$ and the public key is $<N,e_i>$ for each encryption, for $1\le i\le l$.
- l encrypted messages $v_1,...,v_l$ are obtained by $v_i=m_i^{e_i} \pmod N$,where m_i is the plaintext message.

Percolate-Up: During this phase, nodes in the binary tree deal with pairs (V,E) where the component called V is a value, and the component called E is an exponent:

- A leaf node provides inputs and the l-th such node sets: $V=C_i$ and $E=e_i$ .Pass (V, E) upward.
- An interior node combines results passed upward from children into a result. It can pass further upward. The left-hand and right-hand children yield the results$(V_L, E_L)$and(V_R, E_R).The node computes

$V = V_L^{E_R} \cdot V_R^{E_L} \pmod N$, $E = E_R \cdot E_L$ and passes (*V*,*E*) upward, storing (V_L, E_L) and (V_R, E_R) for use during Percolate-Down .

- The root node terminates the process .The result (*V*, *E*) of root is: $V = \prod_{i=1}^{l} V_i^{E/e_i} \pmod N$ and $E = \prod_{i=1}^{l} e_i \pmod N$. The Percolation-Up phase can be deployed in the encryption process.

Exponentiation-Phase: In the exponentiation phase, the *E* th root of *V* is the *d* that has been computed in the Setup phase. Compute $r = V^{1/E} \pmod N \equiv V^d \pmod N$ and store it for use during the downward percolation. In this paper, the performance of the Batch RSA decryption is enhanced by improving the speed of $V^d \pmod N$ that is the most time-consuming operation in the Batch RSA decryption. The CRT [10] and Multi-Prime RSA are used to compute the *r*. Improvement process is as follows:

- Calculate C_{pi} by the $C_{pi} = V \bmod (p_i)$, for $1 \le i \le b$.
- Compute M_{pi} by the $M_{pi} = C_{pi}^{d_{pi}} \bmod p_i$, for $1 \le i \le b$, where d_{pi} has been computed in the Setup phase.
- Compute $y_i = N / p_i = p_1 p_2 \cdots p_{i-1} p_{i+1} \cdots p_b$ and $n_i = y_i \times (y_i^{-1} \pmod{p_i})$, for each i, $1 \le i \le b$.
- Using the CRT to combine the M_i's to obtain $r = V^d = M_{p1} \times n_1 + \cdots + M_{pb} \times n_b \pmod N$.

The exponentiation yields $V^{1/E} = \prod_{i=1}^{l} V_i^{1/e_i}$, during which should be stored as *r* in the root node.

Percolate-Down: During the Percolation-Down phase, nodes in the tree deal with values called *r*.

- The root node passes the *r* that has been computed during Percolation-Up downward to the single child.
- An interior node has the value *r* passed downward and needs to break it up to pass down further and compute $r_R = r^t / (V_L^{t_R} \cdot V_R^{t_L}) \pmod N$ and $r_L = r / r_R \pmod N$, where *t* is a value obtained via CRT based on $t \equiv 0 \pmod{E_L}$ and $t \equiv 1 \pmod{E_R}$. $t_L = t / E_L$ and $t_R = (t-1) / E_R$. Pass r_L and r_R downward to the left-hand and right-hand children.
- The leaf node terminates the process and the node has $r = M_i = C_i^{1/e_i} \pmod N$ .All plaintext messages are obtained in the leaf nodes of binary tree.

BMRSA improves the performance of Batch RSA decryption by speeding up the exponentiation phase. BMRSA obtains higher decryption performance to standard RSA using CRT.

Using basic RSA algorithms computing $C^d \bmod N$ take time $O(\log d \log^2 N)$. When *d* is on the order of *N* the running time is $O(\log^3 N)$ .Therefore, using CRT RSA computing $C^d \bmod N$ takes $O(\log^3 N / 2)$ and the asymptotic speedup of BMRSA over Batch RSA is simply:

$$2(\log N / 2)^3 / b(\log N / b)^3 = b^2 / 4$$

For the BMRSA, *b* is 3, which gives a theoretical speed up of approximately 2.25 over Batch RSA.

B. EAMRSA

In this subsection, the new variant (EAMRSA) is described as cryptosystem with three algorithms: key generation, encryption and decryption.

Key generation: The key generation algorithm takes a security parameter and three additional parameters *b, k* and *c* as input. The key pairs (public and private) are generated according to the following steps:

- Compute *b* distinct primes $p_1, \cdots, p_b$ each one $\lfloor n / b \rfloor$ bits in length and generate $N = \prod_{i=1}^{b} p_i$.
- Compute $d = e^{-1} \bmod \varphi(N)$, where e picks the same e used in standard RSA [1] public keys, namely e=65537. The *e* is relatively prime to $\varphi(N) = \prod_{i=1}^{b} (p_i - 1)$.
- Compute $r_i = d \bmod p_i - 1$, for $1 \le i \le b$. Represent the r_i as follows:

$r_1 = d_{1,1} \cdot e_{1,1} + d_{1,2} \cdot e_{1,2} + \ldots + d_{1,k} \cdot e_{1,k} \pmod{p_1 - 1}$, ……,

$r_i = d_{i,1} \cdot e_{i,1} + d_{i,2} \cdot e_{i,2} + \ldots + d_{i,k} \cdot e_{i,k} \pmod{p_i - 1}$, ……,

$r_b = d_{b,1} \cdot e_{b,1} + d_{b,2} \cdot e_{b,2} + \ldots + d_{b,k} \cdot e_{b,k} \pmod{p_b - 1}$,

where the $d_{i,j}$'s and $e_{i,j}$'s, for $1 \le i \le b$ and $1 \le j \le k$, are random vector elements of *c* and $|n|$ bits, respectively. The choice and security of parameters: *b, k,* and *c* are discussed later. The public key is $< N, e, e_{1,1}, \ldots, e_{1,k}, \ldots, e_{b,1}, \ldots, e_{b,k} >$ and the private key is $< N, d_{1,1}, \ldots, d_{1,k}, \ldots, d_{b,1}, \ldots, d_{b,k} >$.

Encryption: The encryption of the new variant RSA includes two steps:

- Given plaintext message $M \in Z_N$, encrypt *M* as basic RSA. The ciphertext *C* is computed as $C \leftarrow M^e \bmod N$.
- Compute vector $Z = (z_{1,1}, \ldots, z_{1,k}, \ldots, z_{b,1}, \ldots, z_{b,k})$ .Take *C* as input, where $Z_{i,j} = C^{e_{i,j}} \bmod N$, for $1 \le i \le b, 1 \le j \le k$, and sends it to the decryption part. The ciphertext message is the vector *Z*.

Decryption: Decryption is done using the Chinese Remainder Theorem. Decrypt the vector *Z* and execute as the following steps:

- Calculate M_i by the $M_i = \prod_{j=1}^{k} Z_{i,j}^{d_{i,j}} \pmod{p_i}$, for $1 \le i \le b$, where p_i has been computed.

- According to the Chinese Remainder Theorem, Compute $y_i = N / p_i = p_1 p_2 \cdots p_{i-1} p_{i+1} \cdots p_b$ and $n_i = y_i \times (y_i^{-1} (\mathrm{mod}\ p_i))$, for each i, $1 \le i \le b$.
- Using the CRT to combine the M_i's to obtain $M = C^d = M_1 \times n_1 + \cdots + M_i \times n_i (\mathrm{mod}\ N)$, for each $1 \le i \le b$.

The variant improves the performance by evaluating a larger number of exponentiations with reduction modules and private exponents.

C. Security and Parameter Selection

In this subsection, the security of BMRSA and EAMRSA is analyzed. It is clear that the security of EAMRSA depends on the size of the used primes and on the security offered by the many private exponents $d_{i,j}$, for $1 \le i \le b$ and $1 \le j \le k$. The security of BMRSA depends on the size of the used primes.

The security of EAMRSA and BMRSA depends on the difficulty of factoring integers of the form $N = \prod_{i=1}^{b} p_i$ for b>2.The fastest known factoring algorithm cannot take advantage of this special structure of N [4].However, one has to make sure that the prime factors of N do not fall within the range of the Elliptic Curve Method [7], because 256-bit factors would be within the range of RSA-512 factoring project, using Elliptic Curve Method[4].Consequently, with a bit length of 1024 modules, it is not secure anymore to use a decomposition of more than three primes. When the value of b parameter is chosen in the new variant, one should make sure that the prime factors of N are bigger than 256-bit. In the following experiments, the key sizes of tested algorithms are from 1024- to 3072- bit, and b chooses 3 or 4 to make the prime factors of N bigger than 341-bit.

The security of BMRSA can be guaranteed based on the size of the used primes. To guarantee the security of EAMRSA, one also has to require for the variant to be infeasible to deduce values $r_i = d \bmod p_i - 1$, $1 \le i \le b$ via brute force, since an attacker with knowledge of some one would be able to factor modulus N and thereby break RSA. Given a vector, for each i, $1 \le i \le b$, $< N, e, e_{1,1}, ..., e_{b,k} >$,a search through all possible values of $d_{i,k}$ would reveal r_i .Because there are $b \times k$ c-bit vector elements, one has to make sure that the search space of $2^{b \times k \times c}$ values is large enough to prevent such an exhaustive search. When the values of the parameters c, k, and b are chosen, one has to make the difficulty of exhausting the resulting search space at least equivalent (or harder than) to breaking the underlying RSA cryptosystem. Exhaustive search of $2^{b \times k \times c}$ values is equivalent to searching for all possible keys in a symmetric-key cryptosystem. Thus, based on the RSA key size used, one has to make sure that the symmetric key size can provide equivalent security. Lenstra and Verheul give formulas for determining such keys [6]. RSA with 1024- and 1536-bit keys by the above formulas would be roughly equal in strength to a symmetric-key cryptosystem with 72- and 80-bit keys, respectively. The EAMRSA values c, b, and k were selected based on the key size formulas in [6][10], and $b \times k \times c$ corresponds to a symmetric key comparable in strength to the corresponding RSA key. In the following experiments, the key size of tested algorithms is from 1024- to 3072-bit, parameter b chooses 3 or 4, parameter c chooses 128- or 256-bit, and k=2, making the value of $b \times k \times c$ much larger than the value based on key size formulas in [6][10] .

III. PERFORMANCE AND EXPERIMANTAL RESULTS

A. Experiment set-up

The speedup is measured by the execution time of RSA decryption in the paper. These test algorithms were written by using the OpenSSL cryptographic library [8] (version 0.9.8 k). The hardware platform was a 1.5GHz Intel(R) Celeron(R)M processor with 512 MB RAM running Windows XP Professional. The RSA keys of 1024, 1536, 2048, 2304, 2560, 2816, and 3072 bits were used so as to test the variant performance with both current and future security.

B. EAMRSA Experimental Results

In EAMRSA experiments, the algorithms were called different names according to different parameters and values of the parameters. EA1M3RSA,EA1M4RSA,EA2M3RSA and EA2M4RSA in the paper respectively indicate the EAMRSA of the parameters b=3 and c=128-bit,the parameters b=4 and c=128-bit,the parameters b=3 and c=256-bit and the parameters b=4 and c=256-bit. M3RSA and M4RSA respectively indicate the Multi-Prime RSA of the parameter b=3 and the parameter b=4. The EA1RSA and EA2RSA respectively indicate the EARSA of the parameter b=3 and the parameter b=4.

TABLE I. SPEEDUP RELATED TO DECRYPTION FOR EAMRSA

Variant	Speedup					
	2048	*2304*	*2560*	*2816*	*3072*	*Average*
EA1RSA	3.86	4.88	3.41	4.38	5.38	4.14
EA2RSA	1.94	2.44	3.41	2.98	2.73	2.74
M3RSA	2.00	2.00	2.32	2.00	2.00	2.21
M4RSA	2.00	2.44	3.41	3.04	3.66	2.92
EA1M3RSA	4.13	5.20	7.27	5.83	5.55	5.19
EA2M3RSA	3.88	2.52	3.52	3.59	3.66	3.38
EA1M4RSA	4.13	5.20	6.81	8.75	10.75	7.06
EA2M4RSA	3.88	3.39	4.54	4.52	5.55	4.63

Table 1 shows the speedups for the five moduli with eight variants to the standard RSA using CRT. The EA1M4RSA got the highest speedup in all variants. The result shows the average EA1M4RSA speedup of 7.06 is from 2048- to 3072-bit.Table 2 lists the improved speedup among the EAMRSA, the EARSA and the Multi-Prime RSA. The results show that the speedup of every algorithm of the EAMRSA is improved to the relative EARSA and Multi-Prime RSA. The highest improved speedup is 4.14 for the EA1M4RSA to the M4RSA and the least improved speedup is 0.64 for the EA2M3RSA to the EA2RSA in table 2. Figure 1 shows the relationship between the varying size of EA1RSA, M4RSA, EA1M4RSA and the decryption speedup.

TABLE II. IMPROVED SPEEDUP RELATED TO DECRYPTION

Variant	Improved Speedup (Average)			
	EA1RSA	*EA2RSA*	*M3RSA*	*M4RSA*
EA1M3RSA	1.05		2.98	
EA2M3RSA		0.64	1.17	
EA1M4RSA	2.92			4.14
EA2M4RSA		1.89		1.71

The EAMRSA, for b=4, c=128, k=2 obtaining the highest speedup in four kinds of improved algorithms.

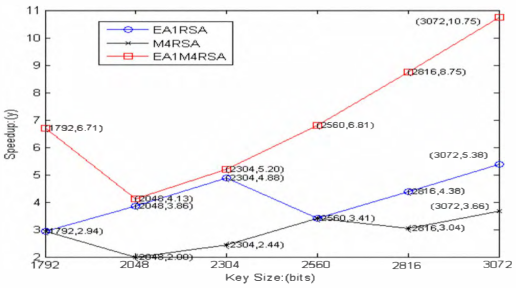

Figure 1. Decryption speedup when varying key size -For b=4,c=128,k=2

The above figure and tables show that when k is fixed, increasing b and decreasing c will lower system performance.

C. *BMRSA Experimental Results*

In this subsection, the performance and experimental results for BMRSA are discussed. The batch size l is 4 in the BMRSA experiments and the number b= 3.Four small public key exponents are $e_1 = 3$, $e_2 = 5$, $e_3 = 7$ and $e_4 = 11$ in the experiments.

TABLE III. DECRYPTION TIME FOR BATCH SIZE 4

Variant	Decryption Time (msec)				
	1024	*1536*	*2048*	*2560*	*3072*
BMRSA	10	31	47	78	125
Batch RSA	16	47	87	155	218
RSA	47	109	265	461	703

Table 3 lists the average decryption time (in msec) for the five moduli with three RSA variants, for l = 4 and b=3.

TABLE IV. SPEEDUP RELATED TO DECRYPTION FOR BATCH SZIE 4

Variant	Speedup					
	1024	*1536*	*2048*	*2560*	*3072*	*Average*
BMRSA	4.70	3.52	5.64	5.91	5.62	5.08
BatchRSA	2.93	2.32	3.05	2.97	3.22	2.90
Improved	1.77	1.20	2.59	2.94	2.40	2.18

Table 4 shows the speedups for the five moduli with two variants to the standard RSA using CRT. The BMRSA got the highest speedup among all variants. The average factor of speedup is 5.08 from 1024- to 3072-bit .Table 4 also lists the improved speedup between BMRSA and Batch RSA. Figure 2 shows the relationship between the varying size of BMRSA and Batch RSA, when the b is 3 and bath size l is 4. The experimental results show that the speeds of the Batch RSA decryption have been substantially improved.

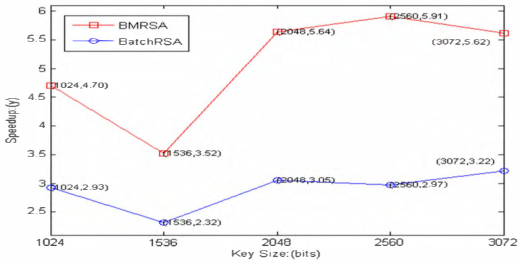

Figure 2. Decryption speedup when varying key size -For b=3,batch l=4

ACKNOWLEDGMENT

This work has been supported by the National Science Foundation of China under Grant No. 60963007 and by the Middle-aged Backbone Teacher Training Program of Yunnan University No.2113204.

IV. CONCLUSION

In this paper, two new RSA variants were proposed to improve the performance of the RSA decryption. BMRSA improved the performance of the Batch RSA decryption based on the Multi-Prime RSA. EAMRSA can obtain high performance by reducing the modulus and private exponents. The variants can get higher security and higher speedup. The next study will focus on: How to optimize the parameters of the new variant to make the variant get higher performance and security.

REFERENCES

[1] R. Rivest,A. Shamir, L. Aldeman, "A Methoed for Obtaining DigitalSignatures and Public-key Cryptosystems,"J. Communications of the ACM, 1978, 21(2): 120-126.

[2] T. Collins, D. Hopkins, and M. Sabin, "Public Key Cryptographic Apparatus and Method,"US Patent #5,848,159. Jan. 1997.

[3] D.Boneh,H.Shacham, "Fast Variants of RSA,"R.RSA Laboratories Cryptobytes,2002,5(1):1-8.

[4] A .Fiat , "Batch RSA,"C. Proc of Crypto '89, LNCS435,1989.Berlin:Springer-Verlag,1989:175-185.

[5] T.Matsumoto,K.Kato, "Speeding up secret computations with insecure auxiliary device,"C.Proc of the 8th Annual International Crypto Conference on Cryptology.London:Springer-Verlag,1988.

[6] A. K. Lenstra and E. R. Verheul, " Selecting cryptographic key sizes,"J. the journal of the International Association for Cryptologic Research, vol. 14, no. 4, pp. 255-293, 2001.

[7] R. Silverman and S. Wagstaff Jr, " A Practical Analysis of the Elliptic Curve Factoring Algorithm,"M. Comp. 61(203):445–462. Jul. 1993.

[8] J. Viega, M. Messier and P. Chandra, "Network Security with OpenSSL,",M.O'Reilly,2002.

[9] J-J. Quisquater and C. Couvreur, "Fast decipherment algorithm for RSA public-key cryptosystem,"J. Eletronic Letters, vol 18:905–907, 1982.

[10] C .Castelluccia, E. Mykletun, and G. Tsudik. "Improving secure server performance by re-balancing SSL/TLS handshakes,"C. Proc of the 2006 ACM Symposium on Information, computer and communications security.New York: ACM, 2006: pages 26-34.