

RSA Decryption Techniques and the Underlying Mathematical Concepts

Hailiza Kamarul Haili and Norfadhilah Basir

School of Mathematical Sciences,

Universiti Sains Malaysia,

Minden 11800 Penang, Malaysia

E-mail: hailiza@cs.usm.my, dila_dax@yahoo.com

ABSTRACT

Number theory is always known for its practicality in many fields. One of the most interesting applications of it is in the study of RSA cryptosystem and this is in fact one of the famous and widely used public key cryptosystem. The implementation of RSA cryptosystem based its security on concepts in number theory. Even though RSA cryptosystem has developed more than two decades, but it is still now a hot research topic. In this paper we discuss decryption techniques in RSA cryptosystem that based its operation on the Chinese Remainder Theorem. We will show that this approach can speed up the decryption process and it can reduce the computational cost compared to the traditional method. Comparisons were done on three different types of operations, the traditional method, the Chinese Remainder Theorem method and the Chinese Remainder Theorem method with added strong prime criterion. We will also discuss a few fast decryption methods as well as an efficient decryption method.

INTRODUCTION

In RSA cryptosystem, basically, implementation involved three different stages namely key generation, encryption and decryption processes. These stages depend very much on each other in order to optimize efficiency as well as computation costs. In the encryption process the key e is made publicly where as in decryption process the key d is a secret key. This secret key is very important as only the key holder can decrypt the ciphertexts to the original plaintexts. To obtain a secure and intractable key d a few algorithms have been developed. There are three well known methods for the key d generation, where these methods are used to obtain the secret key d . They are Euclid's extended GCD (greatest common divisor) algorithm [4], Derome's algorithm [6] and Chin-Chen Chang and Shing-Jia Hwang algorithm [3]. All these methods had used modular arithmetic operations and in fact the whole process of the RSA cryptosystem uses modular arithmetic and the correctness of the RSA algorithm has been proven mathematically [3]. In many RSA cryptosystems, they usually select a small value for the public key e . This kind of choice can only speed up the encryption operation but do not forget that by this way, the corresponding decryption operation costs more computational time because of the larger decryption exponent d .

The alternative way that can be taken to overcome this problem is to implement the decryption operation based on the Chinese Remainder theorem (CRT). This paper is organized as follows: In section 2, we will show some fundamental theorems which include the Chinese Remainder Theorem (CRT). Section 3 discussed the decryption operation using the CRT method in replacement of the normal decryption method. Four types of fast decryption methods are also discussed and mathematically analysed. Evaluation of computational cost and comparison on three different methods, traditional, CRT method and CRT plus strong prime criterion method are shown in a great detail in section 4. Finally section 5 concluded.

THE CHINESE REMAINDER THEOREM (CRT)

The size of the decryption exponent, d and the modulus, n is very important because the complexity of the RSA decryption depends directly on it. The decryption exponent specifies the numbers of modular multiplication necessary to perform the exponentiation. The modulus, n play a role in determined the size of the intermediate results. A way to reduce the size of both d and n is by using the Chinese Remainder theorem [1].

Theorem 2.1 :

Let m_1, m_2, \dots, m_n be a pairwise relatively prime, i.e $\gcd(m_i, m_j) = 1$, for all i and j less than or equal to n where $i \neq j$. Then, the system of congruences

$$\begin{aligned} x &\equiv a_1 \pmod{m_1} \\ x &\equiv a_2 \pmod{m_2} \\ &\bullet \\ &\bullet \\ x &\equiv a_n \pmod{m_n} \end{aligned}$$

has a solution which is unique modulo the integer m_1, m_2, \dots, m_n . Further, if

$$M_j = \frac{\prod_{i=1}^n m_i}{m_j}$$

and z_j is a solution of $M_j z_j \equiv a_j \pmod{m_j}$ for each j , then the solution is given by

$$x = \sum_{j=1}^n M_j z_j \quad .$$

Now we will show the special case of the Chinese Remainder theorem with only two factors. This theorem is often used in implementation of the RSA decryption operation.

Theorem 2.2 (Davis, 2003):

Let p and q be two numbers (co-prime positive integer) such that $\gcd(p, q) = 1$. If $a \equiv b \pmod{p}$ and $a \equiv b \pmod{q}$, then we have $a \equiv b \pmod{pq}$.

RSA DECRYPTION OPERATION

As usual, in order to obtain the original plaintext, one should do the decryption operation

$$\tilde{M} = C^d \pmod{n} \quad (3.1)$$

Unlike the normal implementation of RSA decryption operation, this time we will show that the decryption operation can be speed up using the Chinese Remainder theorem. Using this theorem, the decryption speed of RSA can be increased by approximately a factor of four. The decryption operation based on the Chinese Remainder theorem is implemented as follows [10]:

Since the recipient knows the secret primes p and q , he can compute the following modular components:

$$\begin{aligned} d_p &\equiv d \pmod{p-1} \text{ and } d_q \equiv d \pmod{q-1} \\ C_p &\equiv C \pmod{p} \text{ and } C_q \equiv C \pmod{q} \\ M_p &\equiv C_p^{d_p} \pmod{p} \text{ and } M_q \equiv C_q^{d_q} \pmod{q} \end{aligned} \quad (3.2)$$

There are many ways that we can get the original plaintext, \tilde{M} using the Chinese Remainder theorem. From the proof of Chinese Remainder theorem, we know that the unique solution x of the congruences can be written as

$$x = \left(\sum_{i=1}^k x_i r_i s_i \right) \pmod{n} \quad (3.3)$$

where $r_i = n/n_i$ and $s_i = r_i^{-1} \pmod{n_i}$ for $i = 1, 2, \dots, k$.

Let $n_1 = p, n_2 = q$ and $n = pq$, we get $r_1 = q$ and $r_2 = p$. We can get \tilde{M} by simplify the equation (3.3) using Fermat's Little theorem:

$$\begin{aligned} \tilde{M} &= (M_p q(q^{-1} \bmod p) + M_q p(p^{-1} \bmod q)) \bmod n \\ &= (M_p q(q^{p-2} \bmod p) + M_q p(p^{q-2} \bmod q)) \bmod n \\ &= (M_p (q^{p-1} \bmod n) + M_q (p^{q-1} \bmod n)) \bmod n \end{aligned}$$

Besides that, Penzhorn (2004) suggested that the decrypted message can be computed as follows:

$$\tilde{M} = [((M_q + q - M_p) \cdot A) \bmod q] \cdot p + M_p \tag{3.4}$$

with the given p and q , $p < q$. Let A be a constant integer such that $0 < A < q - 1$ and $A \cdot p \equiv 1 \pmod q$. A is known as the multiplicative inverse of q and can be determined by the Euclid's Extended algorithm [4]. Note that p and q have about half the bit size of n . So, we can say that all exponentiations modulo p and q can be performed roughly four times faster than exponentiation modulo n [12].

In 2004, W T Penzhorn has developed four fast decryption methods to increase the decryption speed of the RSA cryptosystem. In his methods, the decryption exponent, d is obtain by means of special constructions, and the two secret primes p and q are chosen such that the difference between them has low Hamming weight. Hamming weight is defined as the number of "1" bits in the binary sequence. These methods were implemented to shorten the length of d . By using a small value of d , users can reduce decryption time. Small d can improve performance by at least a factor of 10 (for a 1024 bits modulus)[12]. Unfortunately, this method can lead to an attack and results in a total break of the cryptosystem. This is supported by the following result of Boneh[2].

Theorem 3.1 :

Let $n = pq$ with $q < p < 2q$. Let $d < \frac{1}{3}n^{1/4}$. Given $\langle n, e \rangle$ with $ed = 1 \pmod{\phi(n)}$, the eavesdropper can efficiently recover d .

Since typically n is 1024 bits, it follows that d must be at least 256 bits long in order to avoid this attack. So, it is safer if we use the methods that do not shorten the length of d . Compared to Penzhorn[12], Hwang, *et al.* [9] has proposed an efficient decryption method that do not only based on the Chinese Remainder theorem but also the strong prime of RSA criterion.

An Efficient Decryption Method

In 2005, Ren-Junn, Feng-Fu, Yi-Shiung and Chia-Yao[9] have proposed an efficient decryption method based on the Chinese Remainder theorem and the strong prime of RSA criterion. They included the strong prime criteria because the security of the RSA cryptosystem is based on the difficulty of factoring problem. So, the prime of modulus of the RSA algorithm must be strong primes.

By the security consideration, the prime factors p and q of modulus n in the RSA cryptosystem must be strong primes. It is reasonable that the private key (decryption key) holder knows the prime factors of $p-1, p+1, q-1$ and $q+1$. The private key holder performs the decryption procedure $C^d \pmod{n}$ by the following steps [6].

Step 1: Factor $p-1, p+1, q-1$ and $q+1$ to get their prime factors. Assume that moduli $p-1, p+1, q-1$ and $q+1$ be expressed as follows:

$$\begin{aligned} p-1 &= \prod_{i=1}^h r_i^{\alpha_i}, \\ p+1 &= \prod_{i=1}^j s_i^{\beta_i}, \\ q-1 &= \prod_{i=1}^k t_i^{\chi_i}, \\ q+1 &= \prod_{i=1}^l u_i^{\delta_i}. \end{aligned}$$

Step 2: Compute the modular exponentiations with prime factors of $p-1$ as the modulus. Then, apply the CRT to generate $y_1 = C^d \pmod{p-1}$ based on these results. By the same way, the private key holder generates $y_2 = C^d \pmod{p+1}$, $y_3 = C^d \pmod{q-1}$ and $y_4 = C^d \pmod{q+1}$ individually. The detail steps are as follows:

- 2.1 Compute $C_{(p-1),i} = C \pmod{r_i^{\alpha_i}}, i = 1, \dots, h$
- 2.2 Compute $d_{(p-1),i} = d \pmod{\phi(r_i^{\alpha_i})}, i = 1, \dots, h$
- 2.3 Compute the modular exponentiation

$$M_{(p-1),i} = C_{(p-1),i}^{d_{(p-1),i}} \pmod{r_i^{\alpha_i}}, i = 1, \dots, h$$
- 2.4 Apply the CRT to generate $y_1 = C^d \pmod{p-1}$ based on $M_{(p-1),i}, i = 1, \dots, h$

Step 3: Compute $X_1 = 2^{-1}(y_1 + y_2 - z) \pmod{p}$ where $z = 0$ if $y_1 \geq y_2$; otherwise $z = 1$ and $X_2 = 2^{-1}(y_3 + y_4 - z) \pmod{q}$ where $z = 0$ if $y_3 \geq y_4$; otherwise $z = 1$.

Step 4: Apply the CRT to generate the plaintext $M = C^d \pmod{n}$ based on X_1 and X_2 .

The numbers $p-1, p+1, q-1$ and $q+1$ can be individually decomposed into three prime factors at least at step 1. The bit lengths of $r_i^{\alpha_i} (i = 1, \dots, h)$ are smaller than $p-1$. The bit lengths of $d_{(p-1),i} (i = 1, \dots, h)$ are smaller than d . In general, the complexity of modular exponentiation depends on the bit length of exponent and modulus. The totally computation time of step 2 to get $y_1 = C^d \pmod{p-1}$ is smaller than to compute y_1 by $C^d \pmod{p-1}$ directly. The efficient results are similar to $y_2 = C^d \pmod{p+1}$, $y_3 = C^d \pmod{q-1}$ and $y_4 = C^d \pmod{q+1}$. The proposed decryption method is more efficient than the traditional method. The following theorem demonstrates that X_1 and X_2 generated in step 3 are correct.

Theorem 3.1.1 (Hwang *et al.*, 2005):

Given $y_1 = X \pmod{p-1}$, $y_2 = X \pmod{p+1}$ such that $0 \leq X \leq \frac{(p^2-1)}{2}$ and p is a prime. Then $X = 2^{-1}(y_1 + y_2 - z) \pmod{p}$, where $z = 0$ if $y_1 \geq y_2$; otherwise $z = 1$.

EVALUATION OF COMPUTATIONAL COST

In this section, we will show that the decryption method that has been proposed in section 3.1 is more efficient than the traditional decryption method and the decryption method that based on the Chinese Remainder theorem only. Nowadays, the bit length of modulus should be at least 1024 bits in order to make the operations secure. We will use this value in our calculation below. Before that, let's define some notations as follows:

- $MOD_E(y, z)$ denotes an operation of modular exponentiation, $x^y \bmod z$.
- $M(w)$, $A(w)$ and $Mod(w)$ denote operations of multiplication, addition and modulus with the bit length of operand is w .
- $l(w)$ denotes the bit length of w .
- S denotes the shift operator.

In the previous section, we implement the decryption operation by computing $C^d \pmod{n}$, where d is the decryption exponent. By the repeated modular exponentiation operation, we can recover the encrypted message. But, in this section, the modulo operation $C^d \pmod{n}$ can be replace with other equation. According to Hwang, *et al.* (2005), the modulo operation $C^d \pmod{n}$ can be expressed as follows:

$$MOD_E(d, n) = 1.5 \times l(d) [M(l(n)) + 2Mod(l(n)) + 1]$$

Before we compute the total number of operations for the three methods that we want to compare, three equations concerning the multiplication operation, the addition operation and also the modulus operation will be introduce. These equations are very important in order to do the calculation. The multiplication operations can be expressed as follows (Hwang *et al.*, 2005):

$$M(w) = 3M(w/2) + 5A(w) + 2S \quad (4.1)$$

The addition operation can be expressed as follows:

$$A(w) = w/32 \quad (4.2)$$

The modulus operation can be expressed as:

$$Mod(w) = Mod(w/2) + 4M(w/2) + 1.5A(w) + 3S \quad (4.3)$$

In the first place, assume that all of $Mod(32)$, $M(32)$, $A(32)$ and S take one clock cycle. For convenient, we find all the values of $M(w)$ and $Mod(w)$ first. Utilizing equations (4.1) and (4.2), we get:

$$\begin{aligned}
 M(1024) &= 3M(512) + 5A(1024) + 2S \\
 &= 3M(512) + 162 \\
 &= 3[3M(256) + 5A(512) + 2S] + 162 \\
 &= 9M(256) + 408 \\
 &= 9[3M(128) + 5A(256) + 2S] + 408 \\
 &= 27M(128) + 786 \\
 &= 27[3M(64) + 5A(128) + 2S] + 786 \\
 &= 81M(64) + 1380 \\
 &= 81[3M(32) + 5A(64) + 2S] + 1380 \\
 &= 243M(32) + 2352 \\
 &= 2595 \text{ clock cycles}
 \end{aligned}$$

$$\begin{aligned}
 M(512) &= 3M(256) + 5A(512) + 2S \\
 &= 3M(256) + 82 \\
 &= 3[3M(128) + 5A(256) + 2S] + 82 \\
 &= 9M(128) + 208 \\
 &= 9[3M(64) + 5A(128) + 2S] + 208 \\
 &= 27M(64) + 406 \\
 &= 27[3M(32) + 5A(64) + 2S] + 406 \\
 &= 81M(32) + 730 \\
 &= 811 \text{ clock cycles}
 \end{aligned}$$

$$\begin{aligned}
 M(256) &= 3M(128) + 5A(256) + 2S \\
 &= 3M(128) + 42 \\
 &= 3[3M(64) + 5A(128) + 2S] + 42 \\
 &= 9M(64) + 108 \\
 &= 9[3M(32) + 5A(64) + 2S] + 108 \\
 &= 27M(32) + 216 \\
 &= 243 \text{ clock cycles}
 \end{aligned}$$

$$\begin{aligned}
 M(128) &= 3M(64) + 5A(128) + 2S \\
 &= 3M(64) + 22 \\
 &= 3[3M(32) + 5A(64) + 2S] + 22 \\
 &= 9M(32) + 58 \\
 &= 67 \text{ clock cycles}
 \end{aligned}$$

$$\begin{aligned}
 M(64) &= 3M(32) + 5A(64) + 2S \\
 &= 3M(32) + 12 \\
 &= 15 \text{ clock cycles}
 \end{aligned}$$

Utilizing equation (4.1), (4.2) and (4.3), we get:

$$\begin{aligned}
 Mod(64) &= Mod(32) + 4M(32) + 1.5A(64) + 3S \\
 &= 11 \text{ clock cycles} \\
 Mod(128) &= Mod(64) + 4M(64) + 1.5A(128) + 3S \\
 &= 80 \text{ clock cycles} \\
 Mod(256) &= Mod(128) + 4M(128) + 1.5A(256) + 3S \\
 &= 363 \text{ clock cycles} \\
 Mod(512) &= Mod(256) + 4M(256) + 1.5A(512) + 3S \\
 &= 1362 \text{ clock cycles} \\
 Mod(1024) &= Mod(512) + 4M(512) + 1.5A(1024) + 3S \\
 &= 4657 \text{ clock cycles}
 \end{aligned}$$

We compiled those values above into the following table:

TABLE 4.1: The values for $M(w)$, $Mod(w)$ and $A(w)$

w (bits)	1024	512	256	128	64	32
$M(w)$	2595	811	243	67	15	1
$Mod(w)$	4657	1362	363	80	11	1
$A(w)$	32	16	8	4	2	1

Now we come to the purpose of this section. We will calculate the total number of operation for each method.

1. Traditional method

The total number of decryption operation for traditional method can be repressed as:

$$\begin{aligned} MOD_E(d, n) &= 1.5 \times l(d)[M(l(n)) + 2Mod(l(n)) + 1] \\ &= 1.5 \times 1024[M(1024) + 2Mod(1024) + 1] \\ &= 1.5 \times 1024[2595 + 2(4657) + 1] \\ &= 18293760 \text{ clock cycles.} \end{aligned}$$

So, the traditional decryption method should take 18293760 clock cycles.

2. Method that based on the Chinese Remainder theorem (CRT)

In this method, the bit length of two distinct primes is equal. So, the total number of decryption operation for this method can be repressed as:

$$\begin{aligned} \text{Total number} &= 2MOD_E(d/2, n/2) + A(512) \\ &\quad + 2M(512) + Mod(512) \end{aligned}$$

$$\begin{aligned} \text{Here, } MOD_E(d/2, n/2) &= 1.5 \times l(d/2)[M(l(n/2)) + 2Mod(l(n/2)) + 1] \\ &= 1.5 \times 512[M(512) + 2Mod(512) + 1] \\ &= 1.5 \times 512[811 + 2(1362) + 1] \\ &= 2715648 \text{ clock cycles.} \end{aligned}$$

$$\begin{aligned} \therefore \text{Total number} &= 2MOD_E(d/2, n/2) + A(512) \\ &\quad + 2M(512) + Mod(512) \\ &= 2(2715648) + 16 + 2(811) + 1362 \\ &= 5434296 \text{ clock cycles.} \end{aligned}$$

The decryption method that based on the Chinese Remainder theorem should take 5434296 clock cycles.

3. Method that based on the CRT and strong primes criterion

In this method, $p-1, p+1, q-1$ and $q+1$ can be factored into at least three numbers. Without loss of generality, assume that the bit length of the largest prime factor is about $l(n)/4$ and others are about $l(n)/8$.

The total number of decryption operation for this method can be repressed as:

$$\begin{aligned} \text{Total number} &= 4MOD_E\left(\frac{d}{4}, \frac{n}{4}\right) + 8MOD_E\left(\frac{d}{8}, \frac{n}{8}\right) \\ &\quad + 4[A(256) + 2M(256) + Mod(256)] \\ &\quad + 4[A(128) + 2M(128) + Mod(128)] \\ &\quad + 2[A(512) + M(512) + Mod(512)] \\ &\quad + A(512) + 2M(512) + Mod(512) \end{aligned}$$

$$\begin{aligned} \text{Here, } MOD_E\left(\frac{d}{4}, \frac{n}{4}\right) &= 1.5 \times l\left(\frac{d}{4}\right) \left[M\left(l\left(\frac{n}{4}\right)\right) + 2Mod\left(l\left(\frac{n}{4}\right)\right) + 1 \right] \\ &= 1.5 \times 256 [M(256) + 2Mod(256) + 1] \\ &= 372480 \text{ clock cycles.} \end{aligned}$$

$$\begin{aligned} MOD_E\left(\frac{d}{8}, \frac{n}{8}\right) &= 1.5 \times l\left(\frac{d}{8}\right) \left[M\left(l\left(\frac{n}{8}\right)\right) + 2Mod\left(l\left(\frac{n}{8}\right)\right) + 1 \right] \\ &= 1.5 \times 128 [M(128) + 2Mod(128) + 1] \\ &= 43776 \text{ clock cycles.} \end{aligned}$$

$$\begin{aligned} \therefore \text{Total number} &= 4(372480) + 8(43776) + 4(857) \\ &\quad + 4(218) + 2(2189) + 3000 \\ &= 1851806 \text{ clock cycles.} \end{aligned}$$

So, this method should take 1851806 clock cycles.

From the calculation above, we can see that the traditional decryption method takes 18293760 clock cycles totally. The decryption method that based on Chinese Remainder theorem takes 5434296 clock cycles while the method that based on CRT and strong prime of RSA criterion takes only 1851806 clock cycles. The CRT based method should take about 30% computational cost of the traditional method. It means that 70% of computational cost can be reduced if we use the decryption method based on CRT. More interesting, if we choose the method that based on CRT and strong prime of RSA criterion, we can reduce the computational cost about 90% from the traditional method. It means that the method that based on CRT and strong primes takes only 10% of computational cost.

While compared to the CRT based method, the method that based on CRT and strong primes takes about 34% of computational cost, meaning that almost 2.9 times faster than the CRT based method. So, from the observation, we can conclude that the method that based on CRT and strong primes of RSA criterion is more efficient than the traditional method and method that based on CRT only. Users will have an alternative way to implement the decryption operation.

CONCLUSION

Number theory was considered the purest branch in pure mathematics, with no direct applications to the real world. But we have shown that number theory could provide unexpected answers to real-world problems such as RSA cryptosystem. As we know that the private decryption key, d is very important in RSA algorithm. The ciphertext can only be converted to plaintext if we know the key d . As we have seen in our discussion in this paper, time spent to decrypt the plaintext rely very much on the efficiency of the given algorithms. We based the performance on the number of operation cycles. Three methods were compared and we have shown that algorithm using the CRT is much better then the traditional method and the method using CRT and strong prime is better than using CRT alone. Thus algorithm using CRT and strong prime has speed up the decryption process as well as cost saving.

REFERENCES

- [1] Anderson, J. A. and Bell, J. M. 1997. *Number Theory with Applications*. New Jersey: Prentice-Hall.
- [2] Boneh, D. 2000. *Twenty Years of Attacks on the RSA Cryptosystem* [Online]. [Accessed 27th March 2006]. Available from World Wide Web: <http://crypto.stanford.edu/~dabo/papers/RSA-survey.pdf>
- [3] Chang, C. C. and Hwang, S. J. 1997. A Simple Approach for Generating RSA Keys. *Information Processing Letters*. **63** (1), 19-21.
- [4] Coutinho, S. C. 1999. *The Mathematics of Ciphers – Number Theory and RSA Cryptography*. Massachusetts: A K Peters.

- [5] Dence, T. P. and Dence, J. B. 1999. *Elements of The Theory of Numbers*. San Diego: Academic Press.
- [6] Derome, M. F. A. 1993. Generating RSA Keys without the Euclid Algorithm. *Electronics Letters*. **29** (1), 19-21.
- [7] Davis, T. 2003. *RSA Encryption* [Online]. [Accessed 27th March 2006]. Available from World Wide Web: <http://www.geometer.org/mathcircles>
- [8] Eynden, C. V. 2001. *Elementary Number Theory*. New York: McGraw-Hill.
- [9] Hwang, R. J., Su, F. F., Yeh, Y. S. and Chen, C. Y. 2005. An Efficient Decryption Method for RSA Cryptosystem. *AINA'05*. **1**, 585-590.
- [10] Mollin, R. A. 1998. *Fundamental Number Theory with Applications*. Florida: CRC Press.
- [11] Niven, I. M. & Zuckerman, H. S. 1980. *An Introduction to the Theory of Numbers*, 4th ed. Canada: John Wiley & Sons.
- [12] Penzhorn, W. T. 2004. Fast Decryption Algorithms for the RSA Cryptosystem. *IEEE AFRICON 2004*. **1**, 361-364.
- [13] Rivest, R. L., Shamir, A. and Adleman, L. M. 1978. A Method for Obtaining Digital Signatures and Public-Key Cryptosystem. *Comm of the ACM*. **21**(2), 120- 126.
- [14] Rivest, R. L. and Silverman, R. D. 1999. *Are 'Strong' Primes Needed for RSA?* [Online]. [Accessed 27th March 2006]. Available from World Wide Web: <http://theory.lcs.mit.edu/~rivest>
- [15] Strayer, J. K. 1993. *Elementary Number Theory*. Boston: PWS Publishing Company.
- [16] Song, Y. Y. 2000. *Number Theory For Computing*. New York: Springer-Verlag.