Study on Computer Software Encryption Based on AES and RSA Combinational Algorithm

Lili Wang1, Jianguo Wu3

1. Ronggehi College of Chongqing Technology and Business University, Chongqing, China
2. Department of Basic Subjects , Hunan University of Finance and Economics, Hunan, China

Abstract: There are two categories of cryptosystems applied in today's software encryption and decryption technology, one is symmetric cryptosystem like AES algorithm, and the other is asymmetric cryptosystem like RSA algorithm. This paper summarizes the mathematical theory of AES algorithm and RSA algorithm, puts forward optimized AES algorithm that conducts conversion in the unit of line and the optimized RSA algorithm that the changes modulus operation in traditional RSA algorithm into a series of remainder and operation of the power of 2 based on the above summary, and designs the realization process of a combination of cryptosystem by combining these two optimized algorithms, hoping to improve the processing efficiency of software encryption and decryption.

Keywords: AES algorithm; RSA algorithm; round transformation matrix; modulus operation; combination cryptogram

1. Introduction

Development of network technology and information technology plays an important role in today’s technology and economy, but the two technologies are facing security problems. Therefore, information security technology focusing on data encryption has attracted great attention from relevant personnel. According to the type of cipher code, data encryption technology can be divided into symmetric encryption and asymmetric encryption. Since both types of encryption technology have advantages and disadvantages, this paper optimizes and combines AES algorithm and RSA algorithm based on the analysis on AES algorithm and RSA algorithm, hoping to design an efficient computer software encryption mechanism by making best use of the advantages and bypassing the disadvantages.

AES is currently the main symmetric encryption algorithm, a new generation of data encryption standard following DES, and a new encryption standard collected by the announcement issued by American National Standards Institute of Technology (NIST) in 1997. Rijndael algorithm proposed by Daemen et al in Belgium in 2000 is announced as AES algorithm without modification. RSA is an asymmetric encryption algorithm most widely used, which is characterized by high security and easy implementing, and it can be used in both data encryption and identity verification. Aiming at the time consuming difference between the column mixed computing and inverse column mixed computing of the simplest form in the finite field GF (2^8), and make them consume same computing resources in the encryption and decryption process. Dong Junli et al (2012) designs mixed AES and RSA encryption policy for the problem of slow encryption speed and security of the encryption strategies in current hard disk partition and the problem that the encryption strategies can only achieve data encryption but not software encryption. Zhang Yuanfeng et al. (2013) describes the security issues of the test papers in online examination system, and proposes the use of AES and RSA mixed encryption algorithms to encrypt the test papers, to ensure the security of test papers. Chen Junbo(2012) introduces the encryption algorithms, AES algorithm and RSA algorithm, for private key and public key, combines the advantages of the two algorithms, uses AES symmetric encryption algorithm to encrypt message data, and uses RSA asymmetric encryption algorithm to produce digital signature. Bo Xiaoyan et al (2012) analyze the encryption technology of the available data, and propose an encryption system combining both AES and RSA. Above-mentioned scholars all adopt the new encryption system designed by the strategy combining both symmetric encryption technology and asymmetric encryption technology, and this paper optimizes the traditional symmetric encryption AES algorithm and the asymmetric encryption RSA algorithm, and designs a system of mixed cryptogram based on the optimization.

2 Mathematical basis of the algorithm

2.1 Mathematical basis of AES algorithm

Elements in the finite field (2^8) GF (2^8) can be expressed with concentrated method, suppose a byte \( h \) consists of \( b_7b_6b_5b_4b_3b_2b_1b_0 \), these \( h_j \) can be seen as the coefficients of the polynomial of seven times shown in Formula (1), and the values of these coefficients are either 0 or 1, that is, when a digit is expressed by binary system, it can be written to a polynomial that has this digit as the coefficient.

\[
P(x) = \sum_{i=0}^{7} b_ix^i
\]  

(1)

There are two kinds of operations, summation and multiplication operation, in the finite field (2^8) GF (2^8), wherein: the addition of two polynomials is defined as the coefficient sum of the same exponents, that is, difference or operation, and the actual computing example is shown in Formula (2):

\[
(x^6 + x^4 + x^3 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^3
\]  

(2)

It can be easily got from Formula (2) that the polynomial sets on the addition operation in this finite field can constitute a commutative group that meets closure, associativity, zero element (00), inverse element and commutativity.

In the multiplication in the finite field (2^8) GF (2^8), the phenomenon of overflow tends to appear after polynomial multiplication, the way to solve the problem is to reduce a solvable polynomial \( m(x) \) on the multiplication results, and the actual calculation is shown in Formula (3):
It can be easily got from Formula (3) that the result of mold is a polynomial lower than eight order, and different from the addition, multiplication is not a simple operation at a byte. Multiplication operation conducted according to the method shown in Formula (3) has closure, binding and zero element (01). In addition, for any polynomial \( b(x) \) (except 00) of binary system lower than eight order, there are polynomials \( a(x) \) and \( c(x) \), making Formula (4) established.

\[
b(x)a(x) + m(x)c(x) = 1
\]

(4)

\( a(x) \) and \( c(x) \) can be obtained from the Euclidean extended algorithm shown by Formula (5), thus getting the inverse of \( b(x) \).

\[
\begin{align*}
&b(x)a(x) \mod m(x) = 1 \\
b^{-1}(x) = a(x) \mod m(x)
\end{align*}
\]

(5)

If multiplication is conducted on \( b(x) \) and \( x \), Formula (6) can be obtained, and molding results of the multiplication results and \( m(x) \) have the following two principles:

1) If \( b_7 = 0 \), \( (b(x) \cdot x) \mod m(x) = b(x) \cdot x \cdot \Phi \text{mod} 1 \);
2) If \( b_7 = 1 \), \( (b(x) \cdot x) \mod m(x) = b(x) \cdot x \oplus m(x) \cdot \Phi \text{mod} 1 \).

\[
b_7x^8 + b_6x^7 + b_5x^6 + b_4x^5 + b_3x^4 + b_2x^3 + b_1x
\]

(6)

Therefore, \( (b(x) \cdot x) \mod m(x) \) can be considered as the left shift operation of the first byte, XOR operation is conducted on this byte and "IB"(\( m(x) \)) according to the shifting results, and the operation of this type is expressed as \( b = x \times \text{time}(a) \).
Theorem 3: (Fermat Theorem) If \( p \) is a prime number, \( \gcd(a, p) = 1 \), \( a^{p-1} \equiv 1 \pmod{p} \), then it can exit \( a^p \equiv a \pmod{p} \).

Theorem 4:
\[ a \equiv b \pmod{m_1}, a \equiv b \pmod{m_2}, \ldots, a \equiv b \pmod{m_k} \Rightarrow a \equiv b \pmod{m_1m_2\cdots m_k}. \]

Definition 5: \( F \) is a collection containing at least two elements, and the algebraic system \( \left< F, +, \cdot \right> \) defining two kinds of operation “+” and “*” and meeting one of the following conditions is a domain.

1) \( F \) element with respect to operation “+” constitutes Abelian group, and its unit is set as \( O \).
2) \( F \setminus \{O\} \) with respect to operation “*” constitutes Abelian group.

For \( a, b, c \in F \), distribution rate is established, that is, \( (a + b) \cdot c = a \cdot c + b \cdot c \) and \( c \cdot (a + b) = c \cdot a + c \cdot b \).

If the number of elements in domain \( F \) is limited, it is called finite field.

The theoretical basis of RSA is a special kind of reversible modular exponentiation operation. Its cryptosystems are RSA encryption algorithm and RSA digital signature algorithm.

3 Algorithm optimization

3.1 AES algorithm optimization

Figure 1 AES Algorithm Encryption and Interface Structure

The complete structure of the encryption and decryption of AES algorithm is shown in Figure 1, and wherein the input packet of decryption algorithm and the output packet of encryption algorithm are both 128bits.

Analysis on Figure 1 shows the main operation of AES algorithm occurs in the round transformation, and the round transformation should be performed and iterated for at least 10 times, so improving the implementation efficiency of round transformation in realizing the algorithm will help to improve the algorithm's computational speed, and achieve the optimized effect of the algorithm. This paper proposes the idea of optimizing round transformation process. During the round transformation process, line shift transformation only changes the
location of byte data in state matrix, but does not change the value of byte data, but byte substitution transformation only conducts one to one replacement according to S-box byte data, regardless of the position of byte data. Therefore, after the exchanged order of the two transformation layers, the byte replacement and transformation can easily be combined with column mixture, to replace matrix multiplication operation in the original algorithm by checking the table, and the results after conversion are obtained. The optimization method conducts transformation in the unit of column.

Suppose after the completion of line shift, the byte data in state matrix are represented as \(k_{ij}\), the byte in round key is represented as \(k_{ij}\), the byte data after column mixed transformation are represented as \(c_{ij}\), the byte data in state matrix after completing round transformation are represented as \(e_{ij}\), \(s(w)\) represents byte replacement and transformation (wherein \(i = 0, 1, 2, 3, j = 0, 1, \ldots, N_{h-1}\)), and then Formula (8) is established.

\[
\begin{bmatrix}
    e_{0j} \\
    e_{1j} \\
    e_{2j} \\
    e_{3j}
\end{bmatrix} = \begin{bmatrix}
    c_{0j} \\
    c_{1j} \\
    c_{2j} \\
    c_{3j}
\end{bmatrix} \oplus \begin{bmatrix}
    k_{0j} \\
    k_{1j} \\
    k_{2j} \\
    k_{3j}
\end{bmatrix} = \begin{bmatrix}
    02 & 03 & 01 & 01 \\
    01 & 02 & 03 & 01 \\
    01 & 02 & 03 & 03 \\
    03 & 01 & 01 & 02
\end{bmatrix} \oplus \begin{bmatrix}
    s(e_{0j}) \\
    s(e_{1j}) \\
    s(e_{2j}) \\
    s(e_{3j})
\end{bmatrix} \oplus \begin{bmatrix}
    k_{0j} \\
    k_{1j} \\
    k_{2j} \\
    k_{3j}
\end{bmatrix}
\]

(8)

Formula (9) can be rewritten as Formula (10)

\[
c_6x^6 + c_5x^5 + c_4x^4 + c_3x^3 + c_2x^2 + c_1x + c_0
\]

\[
c_4x^4 \mod (x^4 + 1) = (a_1b_1 \oplus a_2b_2 \oplus a_3b_3) x^4 \mod (x^4 + 1)
\]

\[
c_5x^5 \mod (x^4 + 1) = (a_1b_2 \oplus a_2b_3) x^5 \mod (x^4 + 1)
\]

\[
c_6x^6 \mod (x^4 + 1) = a_1b_3 x^2
\]

(9)

\[
\begin{bmatrix}
    e_{0j} \\
    e_{1j} \\
    e_{2j} \\
    e_{3j}
\end{bmatrix} = \begin{bmatrix}
    02 \\
    01 \\
    01 \\
    03
\end{bmatrix} \oplus \begin{bmatrix}
    03 \\
    02 \\
    03 \\
    01
\end{bmatrix} \oplus \begin{bmatrix}
    s(e_{0j}) \\
    s(e_{1j}) \\
    s(e_{2j}) \\
    s(e_{3j})
\end{bmatrix} \oplus \begin{bmatrix}
    k_{0j} \\
    k_{1j} \\
    k_{2j} \\
    k_{3j}
\end{bmatrix}
\]

(10)

If assume \(T_1(w), T_2(w), T_3(w), T_4(w)\) like the expression form of Formula (11), the state matrix after round transformation can be written in the form of Formula (12) shown below:

\[
\begin{bmatrix}
    T_1(w) \\
    T_2(w) \\
    T_3(w) \\
    T_4(w)
\end{bmatrix}^T = \begin{bmatrix}
    02 \cdot s(w) & 03 \cdot s(w) & s(w) & s(w) \\
    s(w) & 02 \cdot s(w) & 03 \cdot s(w) & s(w) \\
    s(w) & s(w) & 02 \cdot s(w) & 03 \cdot s(w) \\
    03 \cdot s(w) & s(w) & s(w) & 02 \cdot s(w)
\end{bmatrix}
\]

(11)

Before round algorithm, the value of \(T_1(w), T_2(w), T_3(w), T_4(w)\) (such as \(T_1(00) = C66363.45\)) can be calculated and stored in four \(16 \times 16\) two-dimensional arrays with elements of 4 bytes, and then the optimization algorithm of round transformation presented herein can be described as follows:
STEP1. Suppose row shift transformation, and a new state matrix $A_i$ is obtained.

STEP2. Suppose the four byte data of each row of $A_i$ are $r_{0j}, r_{1j}, r_{2j}$ and $r_{3j}$ ($j = 0, 1, \ldots, N_b - 1$), the values of $T_1(r_{0j}), T_2(r_{1j}), T_3(r_{2j}), T_4(r_{3j})$ are got from above-mentioned four two-dimensional arrays, recorded as $t_0, t_1, t_2, t_3$, XOR is successively conducted on the No. $i$ bytes of these four values, thus obtaining four new byte data of this row sequentially. State matrix $A_i$ has $N_b$ rows, and transformation is conducted on each row according to above steps, thus obtaining new state matrix $A_{i+1}$. The full completion of state matrix transformation only requires $N_b \times 4$ times of look-up table and $N_b \times 12$ magnetic XOR, which will greatly simplify the complexity of the operation and reduce the execution time of round transformation.

STEP3. XOR is conducted on state matrix $A_{i+1}$ and the bytes data on the corresponding location of round key, and additional layer transformation of keys is implemented, obtaining the final state matrix $A_{i+2}$ of this round transformation, and then completing the round transformation.

3.2 RSA algorithm optimization

The initialization process of RSA algorithm first requires the system to generate two large prime numbers (confidentiality), calculate $n = pq$ (public) and Euler function $\phi(n) = (p-1)(q-1)$, randomly select integer $e$ as a public key (encryption key), to meet $\gcd(e, \phi(n)) = 1$ (public), finally calculate the private key $d$ (decryption key), to meet $ed \equiv 1 (mod \phi(n))$, and destruct $p, q$ and $\phi(n)$.

RSA encryption and decryption transformation first needs to divide and digitalize plaintext block, length of each digitized plaintext block should be no less than $\lceil \log_2 n \rceil$, and then encryption and decryption transformation are sequentially conducted on each plaintext block $m(0 < m < n)$. Wherein, encryption transformation uses public key $e$ to encrypt plaintext $m$, namely, $c \equiv m^e (mod n)$; decryption transformation uses private key $d$ to decrypt the ciphertext $c$ and obtains plaintext $m$, namely, $m \equiv c^d (mod n)$.

Since $ed \equiv 1 (mod \phi(n))$, $ed = t\phi(n) + 1$, and for certain $t \geq 1$, suppose $x \in Z^*_n$, then: $x^{ed} = x^{t\phi(n)+1} (mod n) = x^{t\phi(n)}x^{(mod n)} \equiv x^{(mod n)} \equiv x^{(mod n)}$, so that encryption and decryption transformation process is reversible.

RSA algorithm flow is shown in Figure 2:
Improved RSA algorithm is to change the modulo operation in RSA traditional algorithm into a series of remainders and operations of power of 2, as is shown in Formula (13), wherein \( m_i \) represents the No. \( i \) binary bit of the binary representation form of \( m \). If suppose \( r_i = 2^i \mod n \). Formula (14) is established, and therefore the mode from \( m \) to \( n \) is equal to the sum of the corresponding non-zero remainders.

\[
m \mod n = \left( \sum_{i=0}^{j} m_i 2^i \right) \mod n = \left( \sum_{i=0}^{j} m_i \left( 2^i \mod n \right) \mod n \right)
\]

\[
m \mod n = \left( \sum_{i=0}^{j} m_i r_i \right) \mod n
\]

Formula (14) shows the algorithm of solving the No. \( i \) remainder, which can be obtained from the No. \( i - 1 \) remainder iteration shown by Formula (15):

\[
r_i = (2 * 2^{i-1}) \mod n = (2 * \left( 2^{i-1} \mod n \right) \mod n = (2 * r_{i-1}) \mod n
\]

If \( t \) is used to represent the digit of \( n \), \( J \) represents the least common multiple of Euler number of each element factor of \( n \), namely, \( p, q \) are prime numbers, \( n = p * q, \phi(p), \phi(q) \) respectively represent the Euler numbers of \( p, q \), \( J = [\phi(p), \phi(q)] \).

Since in the No. \( i \) iteration of the traditional BP algorithm, the maximum absolute value of the product does not exceed \((2t - 1)\) bit, only up to \( (2t - 1) \) remainders need to be calculated while solving \( 2^i \mod n \). As when \( i \leq t - 1 \), and \( 2^i \mod n = 2^i \), there is no need to solve the remainder, only \( 2^i \sim 2^{2^t} \) with respect to \( (t - 1) \) remainders of \( n \) need to be calculated.

When \( t \leq J \leq 2t - 1 \), only \( 2^i \sim 2^J \) with respect to \( J - t - 1 \) remainders of \( n \) need to be calculated, then recursive remainder and remainders generated by iteration before, and remainder table is established. While performing modulo operation, just to find out the corresponding non-zero remainders of the mod and get the sum, and if the sum exceeds \( n \), RSA algorithm can be reused, and the direct result falls completely into the complete residue system of \( n \).

4. Implementation flow design of combination cryptosystem

This study uses symmetric AES encryption algorithm to encrypt the plaintext data, and applies asymmetric RSA encryption algorithm to generate the encryption key used by symmetric encryption algorithm. The two algorithms are combined to form a new combination of cryptosystems, in order to improve the efficiency and transport security of software data.

The generation of secret key requires the identification of two prime numbers \( p, q \), then selects \( e \) or \( d \), finally calculate \( d \) or \( e \), satisfying Formula (16):
\[
\begin{align*}
\begin{cases}
  d &\equiv e^{-1}(\text{mod} \ \varphi(n)) \\
n &\equiv pq
\end{cases}
\end{align*}
\]

(16)

Taking the transmit mode and receive mode of network data files as an example, this paper designs the specific implementation process of combination cryptosystem with sender of A and recipient of B. Wherein: sender A conducts combination encryption on data; recipient B obtains the plaintext data by conducting decryption on the data of combination encryption.

The encryption process of sender A is shown in Figure 3:

![Figure 3 Encryption design process of sender A](image)

Figure 3 shows that in the encryption process, AES algorithm is first applied to encrypt the plaintext \(m\), ciphertext \(c\) and encryption key \(K_A\) are obtained, then RSA algorithm is used to encrypt \(K_A\), AES key block is obtained, and finally ciphertext \(c\), AES key block are passed to recipient B at the same time. The decryption process of recipient B is shown in Figure 4:

![Figure 4 Decryption design process of recipient B](image)

In Figure 4, RSA is applied to decrypt AES key block, \(K_A\) is obtained, \(K_A\) is used to decrypt the ciphertext, and plaintext message \(m\) is obtained.

The hardware environment for implementing combination cryptography system: memory more than 256M, CPU above P4, hard disk capacity more than 80G; software environment: Chinese Windows XP, programming language C# .net, database of SQL.

5. Conclusion

Software encryption is conducive to the security of technology and information transfer, and encryption algorithm is the key technology of software encryption that based on mathematical theory. RSA algorithm and AES algorithm have both pros and cons, and the advantages of the former lie in that it uses public key encryption and private key decryption and there is no need to transmit the secret private key in the encryption process, representing that RSA algorithm is better than AES algorithm in the aspect of key management; the advantages of the latter are that its operation does not require high processing power and large memory of computer, and the operation can easily withstand the attack of time and space and maintain good performance in different operating environments, making AES algorithm safe, efficient and flexible, while RSA algorithm is significantly weaker than AES algorithm in the aspect of data processing speed. Therefore, the combined application of the two optimized algorithms can effectively improve the corresponding performance of software encryption system.

Reference


