Modified RSA Algorithm with CRT & OAEP

Bhumi J. Patel¹, Nitin J. Janwe²

¹²Department of Computer Science & Engg., RCERT, Chandrapur, India 442401

Abstract— The most active subjects in the security related communities are the necessary protection against the data thieves. This gives an importance and the value of exchanged data over the Internet or other media types. In many RSA cryptosystems, we 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 requires more computational time and therefore it costs more because of the larger decryption exponent d. The alternative way that can be taken to overcome this problem is to implement this operation is based on the Chinese Remainder theorem (CRT). This paper deals with implementation of RSA algorithm using Chinese remainder Theorem. To provide extra layer of encryption we will use Optimal Asymmetric Encryption Padding (OAEP) and Variable Radix Number System.

Keywords— Encryption, Decryption, RSA Algorithm, CRT, Optimal Asymmetric Encryption Padding, Hash Algorithm.

I. INTRODUCTION

Cryptography is the science of information security. In today’s computer-centric world, cryptography is most often associated with scrambling plaintext (ordinary text, sometimes referred to as clear text) into cipher text (a process called encryption), then back again (known as decryption).

As the technology is growing, security is considerable in many applications such as private networks, e-commerce, secure internet access, securing confidential reports at army level etc. E-business security is an overarching business issues that, based of an analyzed risks, and establishes the threat acceptance and reduction parameters for the safe use of technology. As an overarching issue, e-business security can be thought of as being absolutely fundamental to the effective and efficient use of Information Technology in support of e-business. There are many algorithms that are used to provide the encryption. RSA algorithm is the well known of all the algorithms. It provides a well defined security.

The RSA algorithm consists of three different steps i.e., key generation, encryption and decryption. All these three steps depend very much on each other with regards to efficiency and computational costs. RSA algorithm is asymmetric type of cryptosystems also called as public key systems.

It has two keys i.e., public key and private key. For encryption e is used as a public key and d is used as a private key also called as secret key for decryption process. The private key is very important as only the key holder can decrypt the cipher texts to the original plaintexts. 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. In order to include RSA cryptosystem efficiently in many protocols, it is desired to devise faster encryption and decryption operations. The alternative way that can be taken to overcome this problem is to implement this operation is based on the Chinese Remainder theorem (CRT).

II. EXISTING TECHNIQUE

A. RSA Algorithm

RSA is an asymmetric cryptograph algorithm. Asymmetric means that there are two different keys. This is also called public key cryptography, because one of them can be given to everyone. The other key must be kept private. It is based on the fact that finding the factors of a integer is hard (the factoring problem). RSA stands for Ron Rivest, Shamir and Adleman, who first publicly described it in 1978. A user of RSA creates and then publishes the product of two large prime numbers, along with an auxiliary value, as their public key. The prime factors must be kept secret. Anyone can use the public key to encrypt a message, but with currently published methods, if the public key is large enough, only someone with knowledge of the prime factors can feasibly decode the message.

The RSA scheme is a block cipher in which the plaintext and cipher text are integers between 0 and n-1 for some n. A typical size for n is 1024 bits. That is, n is less than $2^{1024}$. The scheme makes use of an expression with exponentials. Plaintext is encrypted in blocks, with each block having a binary value less than some number n. That is, the block size must be less than or equal to $\log_2(n)$; in practice, the block size is i bits, where $2^i < n < 2i+1$. Encryption and decryption are of the following form, for some plaintext block M and cipher text block C:
The Chinese Remainder Theorem (CRT) is one of the main theorems of mathematics. This can be used in the field of cryptography. CRT generates key pool and key chain for key pre distribution. One of the applications of this theorem is encryption and decryption of data sequences. The Chinese Remainder Theorem (CRT) allows for an efficient implementation of the RSA algorithm. As we know, the CRT is an algorithm with so many applications in mathematics, computing is the main area of its application and moreover, recently it is being used in cryptography also. But in the field of cryptosystem, the algorithm is used for functionality for modular computation. Random number generators have application in gambling, statistical sampling, computer simulation, cryptography, and other areas where a random number is useful in producing an unpredictable result. The random numbers also is useful for the prevention of reply attack also for counter measures.

Theorem 1: Let \( m_1, m_2, \ldots, m_n \) be a pairwise relatively prime, i.e. \( \gcd(m_i, m_j) = 1 \) for all \( i \neq j \). Then, the system of congruences

\[
\begin{align*}
    x &\equiv a_1 \pmod{m_1} \\
    x &\equiv a_2 \pmod{m_2} \\
    & \vdots \\
    x &\equiv a_n \pmod{m_n}
\end{align*}
\]

has a solution which is unique modulo the integer \( m_1 m_2 \ldots m_n \).

C. RSA with Chinese Remainder Theorem

The Chinese Remainder Theorem (CRT) allows for an efficient implementation of the RSA algorithm. This approach is often used for implementing RSA in embedded systems. Given input, \( m \), raise it to the \( e \)-th (or \( d \)-th) power modulo \( p \) and modulo \( q \).

The intermediate results are then combined through multiplication and addition with some predefined constants to compute the final result (the modular exponentiation to \( n \)). The complexity of the RSA decryption \( M = C^d \mod n \) depends directly on the size of \( d \) and \( n \). The decryption exponent \( d \) specifies the numbers of modular multiplications necessary to perform the exponentiation, and the modulus \( n \) determines the size of the intermediate results. A way of reducing the size of both \( d \) and \( n \) is to take advantage of properties stated by 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.

**RSA-CRT key generation:**

1. Let \( p \) and \( q \) be very be two large primes of nearly the same size such that \( \gcd(p-1,q-1) = 2 \).
2. Compute \( N = p \times q \).
3. Pick two random integers \( dp \) and \( dq \) such that \( \gcd(dp, p-1) = 1 \), \( \gcd(dq, q-1) = 1 \) and \( \gcd(dp, dq) = 2 \) mod 2.
4. Find \( d \) such that \( d \equiv dp \mod (p-1) \) and \( d \equiv dq \mod (q-1) \).
5. Compute \( e = d-1 \pmod{(N)} \).

The public key is \( <N, e> \) and the private key is \( <p, q, dp, dq> \). Since \( \gcd(dp, p-1) = 1 \) and \( \gcd(dp, p-1) = 1 \), we have \( \gcd(d, N) = 1 \). Similarly, \( \gcd(d, q-1) = 1 \). Hence \( \gcd(d, N) = 1 \) and by step 5, \( e \) can be computed.

To apply the Chinese Remainder Theorem in step 4, the respective moduli have to be relatively prime in pairs for a solution to necessarily exist. We observe that \( (p-1) \) and \( (q-1) \) are even and that we cannot directly apply the Chinese Remainder Theorem. However, \( \gcd((p-1)/2, (q-1)/2) = 1 \). Since \( \gcd(dp, p-1) = 1 \) and \( \gcd(dp, q-1) = 1 \), essentially \( dp, dq \) are odd integers and \( dp \equiv dq \mod (p-1), dq \equiv dq \) are even integers. We have \( \gcd(d, p-1) = 1 \) which implies that \( d \) is odd and \( (d-1) \) is even.

To find a solution to \( d \),

\[
d = dp \mod (p-1), \text{ and } d = dq \mod (q-1)
\]

We find a solution to

\[
d-1 \equiv (dp-1) \mod (p-1), \text{ and } d-1 \equiv (dq-1) \mod (q-1).
\]
By applying the cancellation law and taking the common factor 2 out, we have
\[
x = d' = \frac{(d-1)/2 \cdot (dp-1)/2}{(p-1)/2},
\]
x = d' = \frac{(d-1)/2 \cdot (dq-1)/2}{(q-1)/2} \mod \frac{(p-1)/2}{2},

Using Chinese Remainder Theorem we find d such that \(d = (2 \cdot d') + 1\).

### III. PROPOSED TECHNIQUE

RSACRT-OAEP is a public-key encryption scheme combining the modified RSA algorithm with the Optimal Asymmetric Encryption Padding (OAEP) method. An encoding method for encryption consists of an encoding operation and a decoding operation. Assuming that the mask generation function in OAEP has appropriate properties, and the key size is sufficiently large. RSAES-OAEP is semantically secure against adaptive chosen cipher text attacks. OAEP is parameterized by the choice of hash function Hash and mask generation function MGF. We can use hash algorithms such as SHA-1, SHA-256, and SHA-512.

#### A. OAEP Encoding Operation

**OAEP-Encode (M, P, \text{emLen})**

**Options:**

- **Hash**: Hash function (hLen denotes the length in octets of the hash function Output).
- **MGF**: Mask generation function.

**Input:**

- **M**: Message to be encoded, an octet string of length at most \(\text{emLen} - 1 - 2\cdot\text{hLen}\).
- **P**: Encoding parameters, an octet string.
- **\text{emLen}**: Intended length in octets of the encoded message, at least \(2\cdot\text{hLen} + 1\).

**Output:**

- **\text{EM}**: encoded message

**Steps:**

1. If the length of \(P\) is greater than the input limitation for the hash function (261 – 1 octets for SHA-1) then output ‘parameter string too long’ and stop.
2. If mLen > \text{emLen} – 2\cdot\text{hLen} – 1, output ‘message too long’ and stop.
3. Generate an octet string PS consisting of \text{emLen} – mLen – 2\cdot\text{hLen} – 1 zero octets. The length of PS may be 0.
4. Let pHash = Hash (P), an octet string of length \text{hLen}.
5. Concatenate pHash, PS, the message M, and other padding to form a data block DB as DB = pHash||PS||01||M.
6. Generate a random octet string seed of length \text{hLen}.
7. Let dbMask = MGF (seed, \text{emLen} – \text{hLen}).
8. Let maskedDB = DB XOR dbMask.
9. Let seedMask = MGF (maskedDB, \text{hLen}).
10. Let maskedSeed = seed XOR seedMask.
11. Let EM = maskedSeed\cdot\text{maskedDB}.
12. Output EM.

The reverse operation is to be done for decoding.

#### B. RSACRT OAEP Encryption Scheme

**Encryption Operation:**

**RSACRT-OAEP-Encrypt ((n, e), M, P)**

**Input:**

- **(n, e)** recipient’s RSA public key.
- **M**: Message to be encrypted, an octet string of length at most \(k – 2 – 2\cdot\text{hLen}\), where \(k\) is the length in octets of the modulus \(n\) and \text{hLen} is the length in octets of the hash function output.
- **P**: Encoding parameters, an octet string that may be empty.

**Output:**

- **C**: Cipher text, an octet string of length \(k\).

**Steps:**

1. Apply the OAEP encoding operation to the message \(M\) and the encoding parameters \(P\) to produce an encoded message \(EM\) of length \(k – 1\) octet:
   \(EM = OAEP-Encode (M, P, k – 1)\).
2. If the encoding operation outputs ‘message too long’, then output ‘message too long’ and stop.
3. Convert the encoded message \(EM\) to an integer message representative \(m\):
   \(m = OS2IP (EM)\).
4. Apply the RSAEP encryption primitive to the public key \((n, e)\) and the message representative \(m\) to produce an integer cipher text representative \(c\):
   \(c = RSACRT ((n, e), m)\).
5. Convert the cipher text representative \(c\) to a cipher text \(C\) of length \(k\) octets:
   \(C = I2OSP(c, k)\).
6. Output the cipher text \(C\).

**Decryption Operation:**

**RSACRT-OAEP-Decrypt (K, C, P)**

**Input:**

- **K**: Recipient’s RSA private key.
- **C**: Cipher text to be decrypted, an octet string of length \(k\), where \(k\) is the length in octets of the modulus \(n\).
- **P**: Encoding parameters, an octet string that may be empty.
Output: M - Message, an octet string of length at most k − 2 − 2hLen, where hLen is the length in octets of the hash function output.

Steps:
1. If the length of the cipher text C is not k octets, output "decryption error" and stop.
2. Convert the cipher text C to an integer cipher text representative c:
   \[ c = \text{OS2IP}(C) \]
3. Apply the RSADP decryption primitive to the private key K and the cipher text representative c to produce an integer message representative m:
   \[ m = \text{RSACRT}(K, c) \]
   If RSADP outputs "cipher text representative out of range," then output "decryption error" and stop.
4. Convert the message representative m to an encoded message EM of length k − 1 octets P(m, k − 1).
   If I2OSP outputs "integer too large," then output "decryption error" and stop.
5. Apply the EME-OAEP decoding operation to the encoded message EM and the encoding parameters P to recover a message M:
   \[ M = \text{OAEP-Decode}(EM, P) \]
   If the decoding operation outputs "decoding error," then output "decryption error" and stop.
6. Output the message M.

IV. PERFORMANCE

In this paper, we propose an efficient method to implement RSA decryption algorithm. This efficient decryption method can enhance the performance of the RSA algorithm. The proposed method reduces the computational cost. Thus algorithm using CRT and strong prime has speed up the decryption process as well as cost saving.

TABLE I

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Iterations</th>
<th>OAEP</th>
<th>Hash Algorithm</th>
<th>Computational Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA</td>
<td>50</td>
<td>-</td>
<td>-</td>
<td>52</td>
</tr>
<tr>
<td>RSA CRT</td>
<td>50</td>
<td>-</td>
<td>-</td>
<td>16</td>
</tr>
<tr>
<td>RSA CRT</td>
<td>50</td>
<td>Used</td>
<td>SHA-1</td>
<td>16</td>
</tr>
<tr>
<td>RSA CRT</td>
<td>50</td>
<td>Used</td>
<td>SHA-256</td>
<td>18</td>
</tr>
<tr>
<td>RSA CRT</td>
<td>50</td>
<td>Used</td>
<td>SHA-512</td>
<td>17</td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>Algorithms</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td>RSA</td>
<td>Secure</td>
</tr>
<tr>
<td>RSA CRT</td>
<td>More Secure</td>
</tr>
<tr>
<td>RSA CRT with OAEP and SHA-1</td>
<td>More Secure</td>
</tr>
<tr>
<td>RSA CRT with OAEP and SHA-256</td>
<td>Most Secure</td>
</tr>
<tr>
<td>RSA CRT with OAEP and SHA-512</td>
<td>Most Secure</td>
</tr>
</tbody>
</table>

V. CONCLUSION

In this paper better and faster algorithm requiring less cost is developed. The computational cost is reduced and speed of encryption and decryption is increased. Optimal Asymmetric Encryption Padding provides extra layer of encryption. The level of security is increased by using OAEP.

Acknowledgment

One of author BJP thankful to Prof. Nitin J. Janwe project supervisor, RCERT Chandrapur and Prof.Santosh Hedau NCET Gadchiroli for their kind support during this research work.

REFERENCES


[8] RSA Laboratories,” RSAES-Encryption Scheme”.
