CHAPTER 5
PUBLIC-KEY ENCRYPTION IN XML DATA

5.1 INTRODUCTION

The flexible data format of XML supports the applications to exchange data across the internet. In order to use XML in e-commerce business applications, it must provide security and trust for the information exchanged. XML security requirements include confidentiality, authentication, and data integrity. The W3C fulfills these requirements with the help of XML encryption, and XML signature, for authenticating sellers, buyers and traders and for digitally signing and encrypting the XML documents. The existing security mechanism such as Hyper Text Transfer Protocol over Secure Socket Layer (HTTPS) is used to protect the data exchanged across the internet.

The transfer of XML over SOAP protocol leads to the challenging security problem and its processing. The name space substitution and entity reference replacement are carried out at the destination site. It shows that the XML data transit is not similar to XML data at the destination. This problem is addressed by the canonicalization of the XML document at the destination site after decryption of XML documents. The XML key management specification provides the trust and it is built on the XML signature and XML encryption. In this section the public-key encryption method is applied to XML data under different scenario such as partial encryption, full document
5.2 PERFORMANCE REQUIREMENTS

Performance is one of the key essential requirements of the web based application. For over a decade, the World Wide Web functioned well without any quality issues related to performance constraints. Earlier designers as well as users were not facing any performance problems associated with web and also initially web systems were designed only for providing information rather than a medium to transact business. Users also expected to seek only information from the web.

Nowadays millions of users are regularly interacting with online portal to complete their day to day activities. Web users are interested in inquiring about a subject and based on the replies to such queries, make decisions affect their careers, business and quality of their life. The development of e-commerce has further enhanced the user web interface as it has made the business transaction to be redefined to carry out between business to business (B2B) and business to consumers (B2C) organization. In future all the day to day businesses of all users may be supported by the web based system.

Web based transactions are existing in different forms. It includes surfing the news portal for latest events, e-buying a product in a retail showroom, reserving a railway ticket online, or participating in an e-auction event. In all these transactions users expect not only accuracy but also speed in getting response from the portal. The two key attributes accuracy and speed play an important role in the web based businesses. In order to retain the existing customer and add a new customer to it, the quality of performance of
the site must be ensured, in addition to the speed of response and consistency behavior.

All the users are always interested to access the website at any time of the day throughout the year. Users may follow any one of the strategies when unable to access the website.

Temporarily stop accessing the web site and try after a delay of time
Discard the site for some days
Never return to that site
Discourage the other users from accessing and use of the website.

The stakeholders of website such as owners, system developers, and sponsors expect responses in the form of

Web availability $24 \times 7$
Immediate response when a query is performed
High throughput when a user is involved in multiple transactions
Efficient memory usage of both client and server
Effective CPU usage of various systems user for the transactions
Enough bandwidth usage of networks used for the application
Maximum transaction per second
High return on investment

In addition to that, security and user friendly interface are considered as an important criteria for the stakeholders.

The performance requirement of a website varies according to the type of the websites such as
Sites to promote products

Sites with a provision of data and information

Sites processing business transactions

The degree of interactivity and complexity causes the performance variation. Based on this, the categories of websites identified are

- Static websites

  The website contains basic, plain HTML pages. The only interactivity offered to a user is to click the links to download the pages

- Static with form based interactivity

  Websites contain pages with forms, which are used for collecting information from the users. The information could be personal details, comments or requests

- Sites with dynamic data access

  The website provides a front end to access elements from the database. Users can search a catalog or perform queries on the content of a database. The results of a search or query is displayed through HTML pages

- Dynamically generated websites

  Websites display the customized pages for every user. The pages are created based on the execution of scripts

- Web-based software applications
Websites which are part of a business process that works in a highly interactive manner.

5.3 **ENCRYPTION CATEGORY**

Encryption is a process of hiding the meaning of a message, when the message is transmitted. As a result the reader or person viewing the data accidently or intentionally is unable to understand the significance of the information. This encryption process provides confidentiality. This process is represented in Figure 5.1. The input to this encryption process is plain text and output is called cipher text.

![Encryption Process Diagram](image)

**Figure 5.1 Encryption process**

The categories of encryption techniques used for XML data are

- Single-Key Cryptography
- Public-Key Cryptography

5.3.1 **Single-Key Cryptography**

In this method of encryption, a single-key called as a secret-key is used for both encryption and decryption. It is also called symmetric-key encryption. This method is the basis for classic encryption. The major challenging problem with this method is that the secret-key must be known to the receiver of the information for decoding. It is very much essential to securely transmit the symmetric-key to the decryption site. This symmetric-key encryption / decryption process is represented in Figure 5.2.
Figure 5.2 Single-key cryptography

In this method, let $m$ be the plain text message that person X wants to secretly transmit to person Y. Let $E_k$ be the encryption cipher, where $k$ is a secret-key. X must first transform the plain text into cipher text, $c$, in order to securely send the message to Y.

$$c = E_k(m)$$  \hspace{1cm} (5.1)

Both X and Y must know the choice of key, $k$, or else the cipher text is useless. Once the message is encrypted as cipher text, X can safely transmit it to Y. In order to read X message, Y must decrypt the cipher text using $E_k^{-1}$ which is known as the decryption cipher, $D_k$.

$$D_k(c) = D_k(E_k(m)) = m$$  \hspace{1cm} (5.2)

This single-key system is more suitable for e-commerce business applications for secure communication between fixed devices such as ATM machine and the server system, because encryption keys are computed in advance and stored in both the server system and the ATM machine. It is not
suitable for web based internet applications. The solution to this problem is public-key encryption.

5.3.2 Public-key Cryptography

The public-key encryption supports the secure communication between the sender and the receiver without transmitting the key information. The basic methodology of this approach is to generate two separate but related keys using a complex mathematical formula. One key is called a public-key known to all and the other one is called private-key only known to the individual. This approach provides confidentiality, authentication, integrity and non-repudiation which are the basic requirements for e-commerce business applications. This public-key encryption process is shown in Figure 5.3. For XML data transmission this encryption method is more suitable to fulfill all the requirements of secure communication. The RSA algorithm is based on the public-key cryptography.

The key required for the encryption / decryption process is generated by using the following steps:

I. Choose two distinct prime numbers $p$ and $q$.

   For security purposes, the integers $p$ and $q$ should be chosen at random, and should be of similar bit-length. Prime integers can be efficiently found using a primality test.

II. Compute $n = pq$.

   $n$ is used as the modulus for both the public and private-keys. Its length, usually expressed in bits, is the key length.

III. Compute $\varphi(n) = \varphi(p)\varphi(q) = (p - 1)(q - 1)$, where $\varphi$ is Euler's totient function.
IV. Choose an integer $e$ such that $1 < e < \varphi(n)$ and $\gcd(e, \varphi(n)) = 1$; i.e. $e$ and $\varphi(n)$ are coprime.

$e$ is released as the public-key exponent.

$e$ having a short bit-length and small Hamming weight results in more efficient encryption – most commonly $2^{16} + 1 = 65,537$. However, much smaller values of $e$ (such as 3) have been shown to be less secure in some settings.

V. Determine $d$ as $d^{-1} = e \pmod{\varphi(n)}$, i.e., $d$ is the multiplicative inverse of $e$ (modulo $\varphi(n)$).

This is more clearly stated as solve for $d$ given $de \equiv 1 \pmod{\varphi(n)}$

This is often computed using the extended Euclidean algorithm.

$d$ is kept as the private-key exponent.

The encryption process of RSA algorithm is represented as

The person X transmits his public key $(n, e)$ to Y and keeps the private key secret. Y then wishes to send message $M$ to X. Y first turns $M$ into an integer $m$, such that $0 \leq m < n$ by using an agreed-upon reversible protocol known as a padding scheme. Y then computes the cipher text $c$ corresponding to

$$c = m^e \pmod{n}$$

(5.3)

This can be done quickly using the method of exponentiation by squaring. Y then transmits $c$ to X.
The Decryption process of RSA algorithm is represented

X can recover \( m \) from \( c \) by using his private-key exponent \( d \) via computing

\[
m \equiv c^d \pmod{n}
\]  \hspace{1cm} (5.4)

Given \( m \), X can recover the original message \( M \) by reversing the padding scheme.

By using the key generation process, the two prime numbers 7 and 17 are selected. The computed value of an integer number \( e \) is 5. The computed public key is \((5,119)\) and the corresponding private key is \((77,119)\). These keys are used for the encryption / decryption process of the experimental setup implemented.

\[\text{Figure 5.3 Public-key cryptography}\]
5.4 XML SECURITY

The rules of XML support some special cases that make it difficult to easily encrypt or digitally sign the XML document. Such syntax rules are

- Missing attributes declared to have default values are provided to the application as if they are present with the default values.

- Character references are replaced with the corresponding character representations

- Entity references are replaced with the corresponding declared entity

- Attribute values are normalized by replacing the character and entity references.

- In attribute normalization, all leading and trailing spaces are stripped and all interior runs of space are replaced with a single space.

XML processing raises other issues when digitally signing the XML documents using the SOAP protocol.

SOAP security must describe how data can flow through an application and network layout to fulfill the requirements set by the rules of the commercial business without exposing the data to undue risk.

SOAP technology should not insist specific technology or resource, but it must provide portability, flexibility, interoperability and heterogeneity.
5.4.1 XML Security Scenarios

The public-key encryption is more suitable for XML data exchange over the web. It fulfills the essential security requirements such as confidentiality, authentication, integrity and non-repudiation. XML encryption allows the encryption of all or part of an XML document. It is flexible enough to allow the encrypting of any of the following:

- The entire XML document
- An element and all its sub elements
- The content of an XML element
- A reference to a source outside the document

Since the XML encryption is not locked into any specific encryption scheme, additional information is provided on

- Encrypted content
  The information itself

- Key information
  Information or links to information through a Uniform Resource Identifier about the keys involved in the encryption.

The steps for XML encryption include:

Selecting the XML to be encrypted (all or part of a document)
Converting to canonical form if using entities or namespaces with prefixes
Encrypting the resulting canonical form using public-key encryption
Sending the encrypted XML to the intended recipient
5.4.2 Encryption of Entire Document

In this type the entire content of the XML document is encrypted using the public-key. After encryption, the encrypted document is sent to the destination. At the receiver side, the received XML document is decrypted using the private key. The sample XML document with critical information such as credit card information and the corresponding cipher text is given as.

<enc>
<?xml version="1.0">
<bank>
  <creditcard>
    <name>Alex</name>
    <cardno>435576</cardno>
    <pin>7658</pin>
    <balance>25000</balance>
  </creditcard>
</bank>
</enc>

After encryption the cipher text of the document is represented as.

<enc>
93 105 1 79 75 2 118 33 88 47 56 76 94 108 -75 70 37 97 -75 27 40 93 98 20
94 116 27 40 93 29 88 33 53 56 114 29 20 88 53 27 40 93 94 20 79 33 27 46 75
33 1 93 115 94 20 79 33 27 40 93 29 20 88 53 94 76 27 103 102 100 100 55 73
93 29 20 88 53 94 76 27 40 93 91 56 94 27 55 73 100 14 93 115 91 56 94 27 40
93 98 20 75 20 94 29 33 27 50 100 97 97 97 93 115 98 20 75 20 94 29 33 27 40
93 115 29 88 33 53 56 114 29 20 88 53 27 40 93 115 98 20 94 116 27 40
</enc>
5.4.3 Encryption of Portion of Document

In this method of encryption, only a portion of the document is encrypted. The portion of the document to be encrypted is identified by a special tag inserted. The content between the beginning and the ending of this tag is encrypted. The remaining portion of the document is added to the cipher text file without any modification with the same order of information present in the original document. The sample document portion and the corresponding encrypted document is represented as

```xml
<?xml version="1.0">
<bank>
  <creditcard>
    <name>Alex</name>
    <enc>
      <cardno>435576</cardno>
      <pin>7658</pin>
      <balance>25000</balance>
    </enc>
  </creditcard>
</bank>
```

After encryption, cipher text of the portion of document with other information:

```xml
<?xml version="1.0">
<bank>
  <creditcard>
    <name>Alex</name>
    <enc>
```
5.4.4 Encryption of Content of the Element

In this category of encryption, the content portion of the element is encrypted. Here a user specifies the names of the element from which the content portion of the element is encrypted. If the user specifies the multiple elements, then all the specified element content is encrypted. For the sample document, user specifies the elements cardno, pin and balance as input. The content of these elements are only encrypted. Sample document encrypted under this category and the corresponding cipher text is represented as

```xml
<?xml version="1.0">
<bank>
  <creditcard>
    <name>Alex</name>
    <cardno><enc>435576</enc></cardno>
    <pin><enc>7658</enc></pin>
    <balance><enc>25000</enc></balance>
  </creditcard>
</bank>
</enc>
```
After encryption, cipher text of the content of element with other information:

```xml
<?xml version="1.0"?>
<bank>
  <creditcard>
    <name>Alex</name>
    <cardno><enc>103 102 100 100 55 73 40</enc></cardno>
    <pin><enc>55 73 100 14 40</enc></pin>
    <balance><enc>50 100 97 97 97 40</enc></balance>
  </creditcard>
</bank>
```

5.5 RESULT ANALYSIS

The sample XML data file with different sizes are being created. For example sample files with one thousand elements, two thousand elements, three thousand elements and etc. are created with the following fields. Credit card information such as card holder’s name, card number, pin number and balance. The RSA algorithm was implemented in C and the experiments were conducted both in Redhat Linux and windows XP platform using a data file as input. The experiment is repeated for the variable length of the data file. The input files created with different sizes and the number of elements accommodated in the corresponding file that is passed as input to the RSA encryption algorithm is shown in Table 5.1. Table 5.2 shows the size of the file that contains the only portion of the document to be encrypted and number of elements stored in that file.
The implemented public-key algorithm is tested for the different sizes of input data and the time taken for the encryption process is shown in Table 5.3 and 5.4 under different scenarios specified in sections 5.4.2 and 5.4.3, respectively.

Table 5.1 Input file size and number of elements

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Number of elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>144</td>
<td>1000</td>
</tr>
<tr>
<td>289</td>
<td>2000</td>
</tr>
<tr>
<td>434</td>
<td>3000</td>
</tr>
<tr>
<td>579</td>
<td>4000</td>
</tr>
<tr>
<td>724</td>
<td>5000</td>
</tr>
<tr>
<td>869</td>
<td>6000</td>
</tr>
<tr>
<td>1014</td>
<td>7000</td>
</tr>
<tr>
<td>1159</td>
<td>8000</td>
</tr>
<tr>
<td>1304</td>
<td>9000</td>
</tr>
<tr>
<td>1449</td>
<td>10000</td>
</tr>
</tbody>
</table>

Table 5.2 Input file size (portion of the document) and number of elements

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Number of elements</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>1000</td>
</tr>
<tr>
<td>140</td>
<td>2000</td>
</tr>
<tr>
<td>210</td>
<td>3000</td>
</tr>
<tr>
<td>280</td>
<td>4000</td>
</tr>
<tr>
<td>350</td>
<td>5000</td>
</tr>
<tr>
<td>420</td>
<td>6000</td>
</tr>
<tr>
<td>490</td>
<td>7000</td>
</tr>
<tr>
<td>560</td>
<td>8000</td>
</tr>
<tr>
<td>630</td>
<td>9000</td>
</tr>
<tr>
<td>700</td>
<td>10000</td>
</tr>
</tbody>
</table>
Table 5.3 Time required for encryption of data file (entire document)

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Time required in windows (seconds)</th>
<th>Time required in Red hat Linux (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>144</td>
<td>0.05495</td>
<td>0.02</td>
</tr>
<tr>
<td>289</td>
<td>0.11</td>
<td>0.05</td>
</tr>
<tr>
<td>434</td>
<td>0.1321</td>
<td>0.07</td>
</tr>
<tr>
<td>579</td>
<td>0.1648</td>
<td>0.10</td>
</tr>
<tr>
<td>724</td>
<td>0.2198</td>
<td>0.12</td>
</tr>
<tr>
<td>869</td>
<td>0.2747</td>
<td>0.14</td>
</tr>
<tr>
<td>1014</td>
<td>0.3162</td>
<td>0.16</td>
</tr>
<tr>
<td>1159</td>
<td>0.3462</td>
<td>0.19</td>
</tr>
<tr>
<td>1304</td>
<td>0.3846</td>
<td>0.21</td>
</tr>
<tr>
<td>1449</td>
<td>0.4396</td>
<td>0.23</td>
</tr>
</tbody>
</table>

Table 5.4 Time required for encryption of data file (specific elements)

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Time required in windows (seconds)</th>
<th>Time required in Redhat Linux (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>70</td>
<td>0.02323</td>
<td>0.01</td>
</tr>
<tr>
<td>140</td>
<td>0.55</td>
<td>0.02</td>
</tr>
<tr>
<td>210</td>
<td>0.0824</td>
<td>0.04</td>
</tr>
<tr>
<td>280</td>
<td>0.1091</td>
<td>0.05</td>
</tr>
<tr>
<td>350</td>
<td>0.1265</td>
<td>0.06</td>
</tr>
<tr>
<td>420</td>
<td>0.1512</td>
<td>0.07</td>
</tr>
<tr>
<td>490</td>
<td>0.1681</td>
<td>0.08</td>
</tr>
<tr>
<td>560</td>
<td>0.2091</td>
<td>0.09</td>
</tr>
<tr>
<td>630</td>
<td>0.2263</td>
<td>0.10</td>
</tr>
<tr>
<td>700</td>
<td>0.2483</td>
<td>0.11</td>
</tr>
</tbody>
</table>
Figure 5.4 Comparison graph encryption time

In Figure 5.4 the comparison graph shows the time required for the encryption process of input data under Redhat Linux and Windows platform. It clearly indicates that the encryption of specific elements provides better performance than the encryption of the entire elements.

Compared to the windows platform, unix takes much lesser time for the encryption operation of the same size of input data file. Time required for the decryption process for the entire XML elements and specific XML elements are shown in Table 5.5 and Table 5.6 respectively.

In figure 5.5 the comparison graph shows the time required for the decryption process of the cipher text generated during the encryption process. The experimental result proves that the decryption of the cipher text generated under the category of specific XML elements takes less time. The
Redhat Linux provides a better performance than the windows platform for the decryption process.

Table 5.5 Time required for the decryption of cipher file (entire document)

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Decryption time in windows (seconds)</th>
<th>Decryption Time in Redhat Linux (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>449</td>
<td>0.1648</td>
<td>0.13</td>
</tr>
<tr>
<td>898</td>
<td>0.3248</td>
<td>0.26</td>
</tr>
<tr>
<td>1347</td>
<td>0.5054</td>
<td>0.39</td>
</tr>
<tr>
<td>1797</td>
<td>0.6672</td>
<td>0.52</td>
</tr>
<tr>
<td>2247</td>
<td>0.989</td>
<td>0.65</td>
</tr>
<tr>
<td>2697</td>
<td>1.3186</td>
<td>0.79</td>
</tr>
<tr>
<td>3146</td>
<td>1.4835</td>
<td>0.92</td>
</tr>
<tr>
<td>3596</td>
<td>1.6483</td>
<td>1.06</td>
</tr>
<tr>
<td>4045</td>
<td>1.892</td>
<td>1.19</td>
</tr>
<tr>
<td>4494</td>
<td>2.033</td>
<td>1.33</td>
</tr>
</tbody>
</table>
Figure 5.5 Comparison graph decryption time

Table 5.6 Time required for the decryption of cipher file (specific elements)

<table>
<thead>
<tr>
<th>Input file size (KB)</th>
<th>Decryption time in windows (seconds)</th>
<th>Decryption Time in Redhat Linux (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>215</td>
<td>0.1099</td>
<td>0.07</td>
</tr>
<tr>
<td>431</td>
<td>0.1648</td>
<td>0.13</td>
</tr>
<tr>
<td>647</td>
<td>0.2747</td>
<td>0.19</td>
</tr>
<tr>
<td>863</td>
<td>0.4396</td>
<td>0.25</td>
</tr>
<tr>
<td>1080</td>
<td>0.495</td>
<td>0.31</td>
</tr>
<tr>
<td>1297</td>
<td>0.6044</td>
<td>0.38</td>
</tr>
<tr>
<td>1512</td>
<td>0.6593</td>
<td>0.44</td>
</tr>
<tr>
<td>1728</td>
<td>0.7692</td>
<td>0.51</td>
</tr>
<tr>
<td>1944</td>
<td>0.879</td>
<td>0.57</td>
</tr>
<tr>
<td>2160</td>
<td>0.9341</td>
<td>0.63</td>
</tr>
</tbody>
</table>
5.6 CONCLUSION

Security is very much important for the web based application, when critical information is exchanged over the internet. Performance is also another important criterion that must fulfill the user requirements. The public-key encryption algorithm provides the confidentiality, authentication, data integrity and non-repudiation for the XML data exchanged over the web application. The performance of the encryption is increased by encrypting the only portion of the XML document instead of an entire XML document. The existing public-key algorithm is used for encrypting the portion of the XML document under windows and Redhat Linux platform is implemented and the performance statistics are recorded. The experimental results proved that the specific elements of XML document encryption lead the performance certain extend.

The following chapter, which is the concluding chapter of the thesis, summarizes the results and findings of the proposed work. The scope for future work is also provided.