CHAPTER 6

A TIME-STAMP BASED ALGORITHM FOR DECRYPTING
THE ENCRYPTED XML DOCUMENTS PARTIALLY

When a SOAP web service receives an encrypted XML document, it decrypts the key and uses it to decrypt all the parts of the document containing the encrypted data. Finally, the data are parsed with an XML parser, and the whole document is forwarded to the next module. In this case, the XML security processing module typically does not know which parts of the decrypted data are later processed by the business logic, due to the absence of a time-stamp during the decryption process. Thus, it could also happen that, the encrypted data are decrypted, parsed, and not processed further for security analysis. This chapter introduces a new time-stamp based algorithm, for decrypting the encrypted XML documents partially. This proposed algorithm utilizes a new temporal model for XML representation, based on a combination of valid time and transaction time, through the tuple time-stamping. In addition to the time-stamp, one attribute of the key components of the encrypted elements is selected to distinguish among them. The time-stamp and the selected attribute guide the receiver to decrypt the required parts of the encrypted XML documents, instead of decrypting all the parts of them. Moreover, they can be used to retrieve the parts that must be decrypted, to contribute to the query results on the encrypted XML documents. The proposed algorithm not only reduces the decryption time, but also protects the security of the data.
6.1 INTRODUCTION

Although XML encryption achieves the confidentiality of the transmitted data, the receiver must decrypt the entire document to get the required data, because there is no indication that the receiver can use it to determine the required data. In this chapter, a new time-stamp based algorithm for decrypting the encrypted XML documents partially is proposed. This proposed algorithm utilizes a new temporal model for XML representation, based on a combination of valid time and transaction time as the tuple time-stamping, and one selected attribute of the key components of the encrypted elements to distinguish among them. The time-stamp and the selected attribute help the receiver to get the required encrypted data of the encrypted XML documents to be decrypted. Hence, unnecessary decryption of the other parts of the encrypted data can be prevented. Moreover, the receiver can use the time-stamp and the selected attribute to decrypt the parts needed to answer queries on the encrypted XML documents. The proposed algorithm not only reduces the decryption time, but also protects the security of the data.

6.2 TIME REPRESENTATION IN XML DATABASES

Wang & Zaniolo (2004) represented the valid time and the transaction time in a hierarchical XML view (BH-document), as depicted in Figure 6.1. Each element in the BH-document is time-stamped (date-time, date, or time) using four XML attributes vstart, vend, tstart, and tend, which are called attributes time-stamp. The attributes vstart and vend represent the inclusive valid time interval, and tstart and tend represent the inclusive transaction time interval. The format of the time-stamp may be the date-time, time, or date, because the attribute value in XML is a string value. The value of tend can be assigned to UC (Until Changed), and vend can be assigned to now, to denote the ever-increasing current date. The parent element (e.g., employee Omar in Figure 6.1) always has a longer or equal lifespan than its children. Hence, the valid and transaction time intervals of a parent node always cover those of its child nodes.
<employees vstart = "01-01-2012" vend = "now" tstart = "01-01-2012" tend = "UC">
    <employee vstart = "01-01-2012" vend = "now" tstart = "01-01-2012" tend = "UC">
        <name vstart = "01-01-2012" vend = "now" tstart = "01-01-2012" tend = "UC">
            Omar
        </name>
        <title vstart = "01-01-2012" vend = "now" tstart = "01-01-2012" tend = "31-05-2012">
            employee
        </title>
        <title vstart = "01-01-2012" vend = "31-06-2012" tstart = "01-06-2012" tend = "UC">
            employee
        </title>
        <title vstart = "01-07-2012" vend = "now" tstart = "01-06-2012" tend = "UC">
            manager
        </title>
        <salary vstart = "01-01-2012" vend = "now" tstart = "01-01-2012" tend = "31-05-2012">
            5000
        </salary>
        <salary vstart = "01-01-2012" vend = "31-06-2012" tstart = "01-06-2012" tend = "UC">
            5000
        </salary>
        <salary vstart = "01-07-2012" vend = "now" tstart = "01-06-2012" tend = "UC">
            7000
        </salary>
    </employee>
</employees>

Figure 6.1 Example of the Time Representation in XML Databases
The XML time representation in Figure 6.1 is called attribute timestamping. In the proposed algorithm, the time representation is changed to tuple representation, instead of attribute representation; i.e., using the timestamp attributes only in the parent element, instead of using them in every element as in Wang & Zaniolo (2004), and a key attribute is added to each element. Moreover, the current and history data are separated in two XML documents. The proposed representation is depicted as in Figures 6.2 and 6.3.

```
<employees>
  <employee Id = 1  vstart = "01-01-2012"  vend = "now"
           tstart = "01-01-2012"  tend = "UC">
    <name> Omar </name>
  </employee>
  <employee Id = 1  vstart = "01-07-2012"  vend = "now"
           tstart = "01-06-2012"  tend = "UC">
    <title> manager </title>
    <salary> 9000 </salary>
  </employee>
</employees>
```

**Figure 6.2 Example of the Tuple XML Representation of the Current Data**

```
<employees>
  <employee Id = 1  vstart = "01-01-2012"  vend = "31-06-2012"
           tstart = "01-01-2012"  tend = "31-05-2012">
    <title> employee </title>
    <salary> 5000 </salary>
  </employee>
</employees>
```

**Figure 6.3 Example of the Tuple XML Representation of the History Data**
When a new employee is inserted, the new employee element with its children element is appended in the current BH-document; the vstart is set to the valid starting time-stamp, the vend is set to now, the tstart is set to the transaction starting time-stamp, and the tend is set to UC. When an employee is deleted, his details are moved to the history BH-document and the vend is set to the end time of the deletion, and the tend is set to the last updating time of the vend.

### 6.3 PROPOSED TIME-STAMP BASED ALGORITHM

In this section, a new time-stamp based algorithm for decrypting the encrypted XML documents partially is proposed. This proposed algorithm utilizes a new temporal model for XML representation, based on a combination of valid time and transaction time as the tuple time-stamp. In addition to the time-stamp, one selected attribute of the key components of the encrypted elements is used to distinguish among the encrypted elements, to decrypt the required parts, instead of decrypting all the parts of the encrypted XML documents. The proposed algorithm is depicted in Figure 6.4:

<table>
<thead>
<tr>
<th>Input</th>
<th>An XML document.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Output</td>
<td>Encrypted Current and History documents.</td>
</tr>
<tr>
<td>BEGIN:</td>
<td></td>
</tr>
<tr>
<td><strong>Step 1:</strong></td>
<td>Choose a time-stamp format (date-time, time, or date) that is suitable for the organization, and select the attribute that can distinguish among the elements.</td>
</tr>
<tr>
<td><strong>Step 2:</strong></td>
<td>Add the selected attribute as an attribute of each XML element.</td>
</tr>
<tr>
<td><strong>Step 3:</strong></td>
<td>Apply the tuple XML time representation, and create a current BH-document and a history BH-document.</td>
</tr>
<tr>
<td><strong>Step 4:</strong></td>
<td>Encrypt the current and history XML documents and send them to the receiver.</td>
</tr>
<tr>
<td><strong>Step 5:</strong></td>
<td>The receiver uses the chosen time-stamp and the selected attribute, to make queries using the XQuery language on the encrypted current and encrypted history XML documents, to get the results that should be decrypted.</td>
</tr>
<tr>
<td>END.</td>
<td></td>
</tr>
</tbody>
</table>

*Figure 6.4 Proposed Time-Stamp Based Algorithm*
The proposed algorithm will be explained by the following example:

**Example:** Let us have the following XML document that represents employees:

```xml
<employees>
  <employee>
    <name> Omar </name>
    <title> manager </title>
    <salary> 5000 </salary>
  </employee>
  <employee>
    <name> Ahmed </name>
    <title> employee </title>
    <salary> 3000 </salary>
  </employee>
  <employee>
    <name> zizo </name>
    <title> manager </title>
    <salary> 4000 </salary>
  </employee>
</employees>
```

**Figure 6.5 Employees' XML Document**

After applying the XML encryption, the employees' XML document will be as follows:
In the encrypted employees' XML document that is depicted in Figure 6.6, the receiver cannot determine which parts must be decrypted. Therefore, the entire encrypted employees' XML document must be decrypted to get the required parts. To solve this problem, the proposed algorithm is used as follows: First, choosing the time-stamp format (as the date in our example) and the rank attribute that can distinguish among the elements. Second, adding the selected rank attribute as an attribute of each XML element. Third, applying the tuple XML time representation of the time-stamp (valid time and transaction time) as shown in section 6.2, by using four XML attributes only in the parent element, instead of using them in every element as in Wang & Zaniolo (2004) and creating a current BH-document and a history BH-document. Therefore, the current and history BH-documents are depicted as in Figures 6.7 and 6.8 respectively:

```
<employees>
  <employee rank = 1 vstart = "01-06-2012" vend = "now"
             tstart = "01-06-2012" tend = "UC">
    <name> Omar </name>
    <title> manager </title>
    <salary> 5000 </salary>
  </employee>
  <employee rank = 2 vstart = "01-01-2012" vend = "now"
             tstart = "02-01-2012" tend = "UC">
    <name> Ahmed </name>
    <title> employee </title>
    <salary> 3000 </salary>
  </employee>
</employees>
```

**Figure 6.7 Current BH-Document**
The valid time represents the time of designation in reality, and the rank attribute represents the rank of the employees (i.e., rank =1 means a manager and rank =2 means an employee).

Fourth, the flexibility of XML encryption allows encrypting only the content of each XML element in the current and history BH-documents, and sending the encrypted documents to the receiver. The encrypted version of the current BH-document is depicted in Figure 6.9:

```xml
<employees>
    <employee rank = 1 vstart = "01-06-2012" vend = "now"
              tstart = "01-06-2012" tend = "UC">
        <EncryptedData
            Type = "http://www.w3.org/2001/04/xmlenc#Content"
            xmlns = "http://www.w3.org/2001/04/xmlenc#">
            <EncryptionMethod
                Algorithm = "http://www.w3.org/2001/04/xmlenc#aes128-cbc"/>
    </employee>
</employees>
```

**Figure 6.9 (Continued)**
Figure 6.9 Encrypted Current BH-Document

Finally, the receiver uses the timestamp and the rank attribute, to make queries by the XQuery language on the encrypted current BH-document and encrypted history BH-document, to retrieve the required parts and decrypt them, instead of decrypting the entire encrypted current XML document and entire encrypted history XML document as follows:
• Query 1: retrieve the current managers:
  
  for $s$ in doc ("ECEmployees.xml")/employees/employee [  
  @ rank = 1 and @ vend = "now"]  
  return <managers> {$s} </managers>.
  
  The above query is executed on the encrypted current BH-document (ECEmployees.xml), and the result is the first encrypted element. Hence, the receiver can decrypt the result and get the required information.

• Query 2: retrieve the previous managers who resigned on 31-12-2012:
  
  for $s$ in doc ("EHEmployees.xml")/employees/employee [  
  @ rank = 1 and @ vend = "31-12-2012"]  
  return <managers> {$s} </managers>.
  
  This query is executed on the encrypted history BH-document (HEmployees.xml).

• Query 3: retrieve all the previous employees:
  
  for $s$ in doc ("EHEmployees.xml")/employees/employee [  
  @ rank = 2]  
  return <employees> {$s} </employees>.

• Query 4: retrieve all the current employees:
  
  for $s$ in doc ("ECEmployees.xml")/employees/employee [  
  @ rank = 2 and @ vend = "now"]  
  return <employees> {$s} </employees>.

  From all these queries, it can be observed that, the proposed algorithm provides facilities to send queries to the XML databases, based on a combination of valid time and transaction time for time-stamping, and hence, can maintain the history data. Moreover, it is capable for decrypting the required parts of the encrypted databases, which are necessary to provide the
answer to the queries. This partial decryption helps in reducing the decryption time, and avoids the decryption of other parts thus enhancing the security.

6.4 RESULTS AND DISCUSSION

To encrypt and decrypt the XML databases, netbeans IDE 7 and the Apache XML Security have been used to encrypt the XML documents, encrypt the key used to encrypt the XML documents, and decrypt the XML documents. The parsing of the XML documents has been done, by using the DOM parser. To make XQueries, the XML documents have been stored in the eXist database. It is an open source native XML database management system, built by the XML technology. It is based on the XML data model to store XML data, and it has a highly efficient index-based XQuery processing. The Java application using the Apache XML security, and the eXist database run on the Windows 7 platform. The CPU is 2.20 GHz Pentium (R) Dual Core and 3GB RAM. The four XQueries mentioned in section 6.3 have been executed on the encrypted employees' XML documents stored in the eXist database, and a comparison between the times required for the partial and full decryption of the encrypted XML documents has been made. The results are depicted in Figure 6.10:

![Figure 6.10 Comparison between the Partial and Full Decryption of the Encrypted Employees’ XML Documents](image)
In Figure 6.10, the time consumption is reduced, when the proposed algorithm is applied to the encrypted XML databases, to decrypt them partially.

Moreover, the proposed algorithm is applied on three XML Databases of Information System and Technology Department of Anna University:

1) IST Research Scholars (RSs).
2) IST UnderGraduates (UGs).
3) IST Employees (Emps).

For each encrypted XML database, after the execution of a set of XQueries on it, the percentage of the partial decrypted results of each XQuery is computed in relation to the full decryption. Hence, the average of the set of XQueries for each encrypted XML database is computed.

For the IST Research Scholars' (RSs) XML database, after applying the proposed algorithm, the current BH-document (CRSs.xml) will be as depicted in Figure 6.11:

```xml
<RSs>
  <RS Rollno = "1116719192" vstart = "01-01-2010" vend = "now" 
tstart = "01- 01-2010" tend = "UC">
    <Name> Mohamed </Name>
    <Area> XML security </Area>
    <Guid> A. Kannan </Guid>
    <Phone> 3321 </Phone>
    <Salary> 9000 </Salary>
  </RS>
  ...
</RSs>
```

Figure 6.11 Proposed Current IST RSs ' XML Database
The following XQueries are executed on the encrypted temporal IST RSs' XML database (current and history BH-documents), and then the results are decrypted. Moreover, the same XQueries are executed on the encrypted conventional IST RSs' XML database after the decryption of the full document:

• **Query 1**: retrieve the RSs who joined on 01-01-2011:

  for $s in doc("ECRSs.xml")/RSs/RS [@ vstart = "01-01-2011 "]
  return <RSs> {$s} </RSs>.

• **Query 2**: retrieve the RS with roll no = "1116719192 ":

  for $s in doc("ECRSs.xml")/RSs/RS [@ Rollno = "1116719192 "]
  return <RS> {$s} </RS>.

• **Query 3**: retrieve the RSs who finished their Research:

  for $s in doc("EHRSs.xml")/RSs/RS [@ vend != "now "]
  return <RSs> {$s} </RSs>.

• **Query 4**: retrieve the RSs who joined between 01-01-2009 and 01-01-2010, both days inclusive:

  for $s in doc("ECRSs.xml")/RSs/RS [@ vstart <= "01-01-2010 
  and @ vstart >= "01-01-2009 "]
  return <RSs> {$s} </RSs>.

Figure 6.12 shows the difference between the partial decryption of the encrypted temporal IST RSs' XML database, and the full decryption of the encrypted conventional IST RSs' XML database, in the total time (compile time + execution time + decryption time) required to run the XQueries and get the results:
By applying the proposed algorithm, the running of the set of XQueries saves approximately 81% of the data from the decryption.

For the IST UnderGraduates' (UGs) XML database, after applying the proposed algorithm, the current BH-document (CUGs.xml) will be as depicted in Figure 6.13:

```xml
<UGs>
  <UG Rollno = "1115619188" vstart = "01-01-2010" vend = "now"
tstart = "01-01-2010" tend = "UC">
    <Name> Abdou </Name>
    <Semester> IV </Semester>
    <Courses>
      <Course> XML </Course>
      <Course> DB </Course>
      <Course> DM </Course>
      <Course> DS </Course>
      <Course> Cryptography </Course>
      <Course> NS </Course>
    </Courses>
    <GPA> 3.5 </GPA>
  </UG>
  ...
</UGs>
```

Figure 6.12 Comparison between the Partial and Full Decryption of the Encrypted IST RSs' XML Databases

Figure 6.13 Proposed Current IST UGs' XML Database
The following XQueries are executed on the encrypted temporal IST UGs' XML database (current and history BH-documents) and on the encrypted conventional IST UGs' XML database, as done on IST RSs' XML database:

- **Query 1**: retrieve the UGs who joined on 01-01-2011:

  ```xml
  for $s in doc("ECUGs.xml")/UGs/UG [@ vstart = "01-01-2011 "]
  return <UGs> {$s} </UGs>.
  ```

- **Query 2**: retrieve the UG with roll no= "1116519188 ":

  ```xml
  for $s in doc("ECUGs.xml")/UGs/UG[@Rollno = "1115619188 "]
  return <UG> {$s} </UG>.
  ```

- **Query 3**: retrieve the UGs who finished before 01-01-2010, including 01-01-2010:

  ```xml
  for $s in doc("EHUGs.xml")/UGs/UG[@ vend <= "01-01-2010 "]
  return <UGs> {$s} </UGs>.
  ```

- **Query 4**: retrieve the UGs who joined between 01-01-2009 and 01-01-2010, both days inclusive:

  ```xml
  for $s in doc("ECUGs.xml")/UGs/UG [@ vstart <= "01-01-2010"
  and @ vstart >= "01-01-2009 "]
  return <UGs> {$s} </UGs>.
  ```

Figure 6.14 shows the difference between the partial decryption of the encrypted temporal IST UGs' XML database, and the full decryption of the encrypted conventional IST UGs' XML database, in the total time (compile time + execution time + decryption time) required to run the XQueries and get the results:
Figure 6.14  Comparison between the Partial and Full Decryption of the Encrypted IST UGs' XML Databases

By applying the proposed algorithm, the running of the set of XQueries saves approximately 73% of the data from the decryption.

For the IST Employees' (Emps) XML database, after applying the proposed algorithm, the current BH-document (CEmps.xml) will be as depicted in Figure 6.15:

```
<Emps>
  <Emp ID = "11167"  vstart = "01-01-2010"  vend = "now"
       tstart = "01-01-2010"  tend = "UC">
    <Name> Ali </Name>
    <Des> Director </Des>
    <Phone> 2341 </Phone>
  </Emp>
  ...
</Emps>
```

Figure 6.15  Proposed Current IST Emps' XML Database
The same set of XQueries, as done on RSs' and UGs' XML databases, are executed for employees on the encrypted temporal IST Emps' XML database and on the encrypted conventional IST Emps' XML database. Figure 6.16 shows the difference between the partial decryption of the encrypted temporal IST Emps' XML database, and the full decryption of the encrypted conventional IST Emps' XML database, in the total time (compile time + execution time + decryption time) required to run the XQueries and get the results:

**Figure 6.16  Comparison between the Partial and Full Decryption of the Encrypted IST Emps' XML Databases**

By applying the proposed algorithm, the running of the set of XQueries saves approximately 52% of the data from the decryption. Therefore, the proposed algorithm saves approximately 69% of the data from the decryption for all the three encrypted temporal XML databases. Figure 6.17 shows a comparison of the percentage of the decrypted data between the partial and full decryption of the three encrypted XML databases:
6.5 SUMMARY

In this chapter, a new time-stamp based algorithm for decrypting the encrypted XML documents partially is proposed. This proposed algorithm utilizes a new temporal model for XML representation, based on a combination of valid time and transaction time, through the tuple time-stamping. In addition to the time-stamp, one attribute of the key components of the encrypted elements is selected to distinguish among them. The time-stamp and the selected attribute guide the receiver to decrypt the required parts of the encrypted XML documents, instead of decrypting all the parts of them.

The new temporal model is proposed not only for the effective representation of the temporal XML, but also for providing facilities for encryption and decryption; i.e., the receiver uses the time-stamp and the selected attribute to retrieve the parts that must be decrypted, to contribute to the query results on the encrypted XML documents. Moreover, this model uses the valid time and transaction time for time-stamping, and hence, can maintain the history data efficiently. This model can be used to develop secure temporal applications.