International Journal of Information, Control and Computer Sciences
ISSN: 2517-9942
Vol:8, No:6, 2014

Enhance Security in XML Databases: XLog File for Severity-Aware Trust-Based Access Control

Asmawi A., Affendey L. S., Udzir N. I., Mahmod R.

***Abstract*—**The topic of enhancing security in XML databases is important as it includes protecting sensitive data and providing a secure environment to users. In order to improve security and provide dynamic access control for XML databases, we presented XLog file to calculate user trust values by recording users' bad transaction, errors and query severities. Severity-aware trust-based access control for XML databases manages the access policy depending on users' trust values and prevents unauthorized processes, malicious transactions and insider threats. Privileges are automatically modified and adjusted over time depending on user behaviour and query severity. Logging in database is an important process and is used for recovery and security purposes. In this paper, the Xlog file is presented as a dynamic and temporary log file for XML databases to enhance the level of security.

***Keywords*—**XML database, trust-based access control, severity-aware, trust values, log file.

I. Introduction

XML (Extensible Markup Language) is widely used in many applications as it has the ability to store, exchange and transfer data. Much of the research on XML focuses on storage strategies and query performance. Although data storage and retrieval techniques are important, so is security and in comparison this is a neglected area. XML databases are multi-user systems, meaning they can be accessed by millions of users and can provide a huge amount of data. Much of this data is sensitive and personal. Confidential data need to be protected and saved in a secure environment. Security research for XML databases is crucial in protecting data from unauthorized processes and misuse.

Most traditional access control models protect data from malicious activities of outside users but cannot protect the data from insiders [1]. Research has suggested that damage caused by insiders is more harmful than that of outsiders [2]. The insider threat is a huge topic in data security and many methods have been proposed to identify misuse behaviour, yet there has been no work on dynamic updates to access privileges in relation to trust for XML databases. Trust-based access control has become an established technique in many areas, such as networks and virtual organisations. It depends on a trust management system, which automatically calculates and updates the trust values of users. Trust values rely on users' behaviours, users' histories, users' credit and users' operations. Users can access resources through trust values and levels [3]-[7].

Trust based access control for XML databases tracks user operations and behaviour over time. Therefore, there is a clear need for logging in XML databases to record users' bad transaction, errors and query severities. In this paper, we proposed the implementation of Xlog file in order to improve XML database security.

Asmawi A. is with Universiti Putra Malaysia, Malaysia (phone: 012-2931997; e-mail: a_aziah@upm.edu.my).

Affendey L. S. (Assoc. Dr.) and Mahmod R. (Prof. Dr.) are with the Department of Computer Science, Universiti Putra Malaysia, Malaysia (e-mail: lilly@upm.edu.my, ramlan@upm.edu.my, respectively).

Udzir N. I. (Assoc. Dr.) is a Head of Department Computer Science, Universiti Putra Malaysia, Malaysia (e-mail: izura@upm.edu.my).

II. System Architecture

In this section, a practical trust-based access control module for XML databases is described. This system is dynamic and responsive to users' history of errors, bad transactions and also queries severity.

The module consists of two main parts: the Trust Module and the Access Control Module. The Trust Module is responsible for recording errors, bad transactions and query severity, evaluating them, and calculating the new trust value. The Access Control Module is responsible for the access permission policy and access decisions. System architecture for XTrust is depicted in Fig. 1.

Based on the Fig. 1, there are two main modules in XTrust. The first module is Trust Module and second module is Access Control Module. The Trust Module is the main part of the Trust-based access control system for XML databases. It receives XML queries from users through the user interface, evaluates their queries and calculates their trust values. The evaluation process depends on the users existing trust values, new bad transactions, new errors and queries severity. After calculating the new Trust Value for the user, it will send the Trust Value to the Access Control Module to update the user's privileges. This module consists of many parts: the Operation Evaluator, the Error Detector, the Severity Indicator, the Operation Recorder, and the Trust Calculator. Each part has its functions and works in the light of the related policy rules. All these parts are connected together to achieve the main goal of calculating trust values for users. This paper will discussed more details on XLog File which records all users' transaction.

III. XLog File

Taking into account the need for logging in the XML databases, we introduce the Xlog file for XML databases which unlike conventional log files is focused on security rather than recovery. Thus the Xlog file will support a secure environment for access control of XML databases, track user operations and behavior by recording and organizing their

actions and also produce a log file that can be used to calculate a trust value that directly affects user access privileges in a trust based access control model for XML databases.

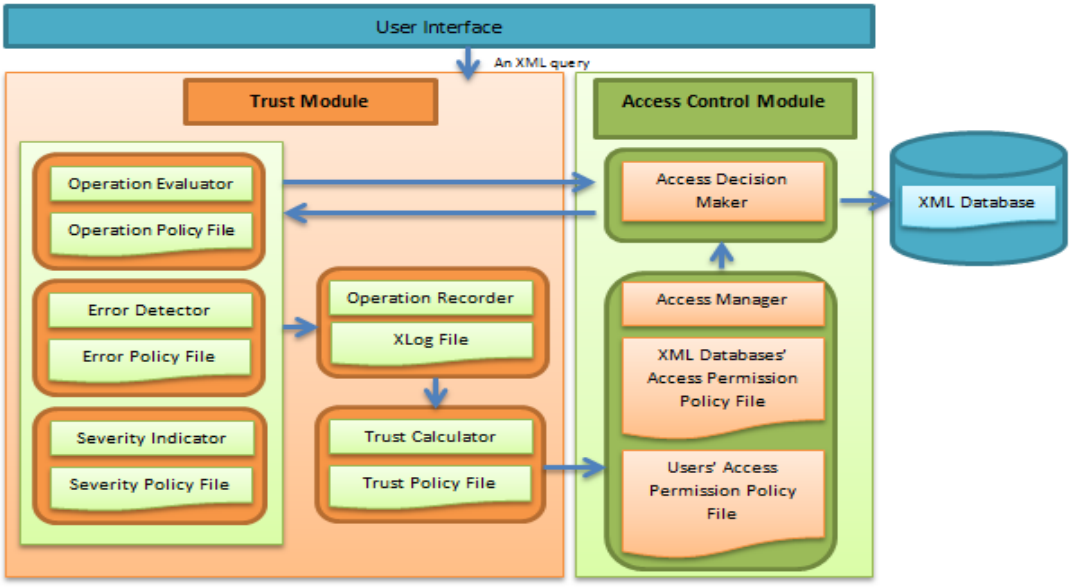

Fig. 1 XTrust Architecture

The structure of the Xlog file for XML databases is shown in Fig. 2. It is dynamic and temporary as it is retained only for a certain time period depending on the organization's policy such as a session, a day, or a week. The Xlog file is written in XML and is processed as a normal XML file. Its structure differs from a normal log file, since it depends on the user identifier instead of time. Using this structure makes capturing user behaviour fast and easy. It does not need to record time for each transaction because it is retained for defined period.

The Xlog file records specific kinds of transactions, errors and severities. Bad transactions are identified by rules defined in the operation policy file that is shown in Fig. 3. These rules cover accessing unauthorized nodes or deleting root and parent nodes. Each bad transaction is categorized by its identifier and type. At present, only five basic types of transactions are defined but the rules can be easily extended to consider other transaction types depending on the system needs.

```
<Users>
<User>
<ID> 30 </ID>
<Bad Transaction>1</Bad Transaction>
<Bad Transaction>5</Bad Transaction>
<Error>1</Error>
<Error>2</Error>
<Severity>2</Severity>
</User>
</Users>
```

Fig. 2 XLog file for XML database

```
<Bad Transactions>
  <Transaction >
    <ID> 1 </ID>
    <Type> Read unauthorized node </Type >
  </Transaction>
  <Transaction >
    <ID> 2 </ID>
    <Type> Write unauthorized node </Type >
  </Transaction>
  <Transaction >
    <ID> 3 </ID>
    <Type> Delete unauthorized node </Type >
  </Transaction>
  <Transaction >
    <ID> 4 </ID>
    <Type> Delete root node </Type >
  </Transaction>
  <Transaction >
    <ID> 5 </ID>
    <Type> Delete parent node with existing
           Children
    </Type >
  </Transaction>
</Bad Transactions>
```

Fig. 3 Operation policy file

Likewise, errors rules are defined in the error policy file shown in Fig. 4. They focus on accessing nonexistent nodes.

```
<Errors>
  <Error >
    <ID> 1 </ID>
    <Type> Read nonexistent node</Type >
  </Error>
  <Error >
    <ID> 2 </ID>
    <Type> Write nonexistent node </Type >
  </Error>
    <Error >
    <ID> 3 </ID>
    <Type> Delete nonexistent node</Type >
  </Error>
</Errors >
```

Fig. 4 Error policy file

All errors are classified by their identifier and type. These rules do not depend on the existence of a schema, a fixed structure for the XML document. They could easily be extended to cover other problems that affect the XML file structure when there is a schema. Severities are identified by rules defined in severity policy file depicted in Fig. 5. There are five kind of severities stated in the policy file.

```
<XPath Injection Severity>
  <Severity>
    <ID>1</ID>
    <Type>No injection query</Type>
  </Severity>
  <Severity>
    <ID>2</ID>
    <Type>View injection query</Type>
  </Severity>
  <Severity >
    <ID>3</ID>
    <Type>Insert injection query</Type>
  </Severity>
  <Severity>
    <ID>4</ID>
    <Type>Update injection query</Type>
  </Severity >
  <Severity >
    <ID>5</ID>
    <Type>Delete injection query</Type>
  </Severity >
</XPath Injection Severity>
```

Fig. 5 Severity policy file

The Xlog file is a useful tool to calculate trust value for the users by assigning weights for bad transactions and errors. These weights are subsequently used to adjust the user's trust values.

IV. Related Works

The main purpose of logging in normal databases is to record transaction information that is used for recovery when the system crashes and sometimes for concurrency control [8, 9]; it also can be useful for security purposes to track malicious transactions in databases [10]. The main classifications in logging are: undo logging, redo logging, and undo/redo logging, all of which are used mainly to restore data [8], [9]. Logs can be represented as tables in databases or files. Log files can be written in different syntaxes, formats, and languages. Reference [11] suggests that using XML language to create the log file saves both time and space compared with tables.

V. Conclusion

In this paper, we have presented the XLog file as adynamic and temporary log file for XML databases in order to enhance the XML database security level. This approach focuses on using logging for security issues related to access control. The XML log file is a part of trust based access control for XML databases. It is used to evaluate user behavior by recording user transactions, errors and query severities. The rules for bad transactions, errors and severities are defined and can be extended according to need.

References

[1] M. Chagarlamudi, B. Panda and Y. Hu, "Insider Threat in Database Systems: Preventing Malicious Users' Activities in Databases," in 2009 *Sixth International Conference on InformationTechnology: New Generations, ITNG '09*, 2009, pp. 1616-1620.

[2] J. S. Park and J. Giordano, "Role-based profile analysis for scalable and accurate insider-anomaly detection," in *25th IEEE International Performance, Computing, and Communications Conference, IPCCC 2006*, 2006, pp. 463-470.

[3] A. Lin, E. Vullings and J. Dalziel, "A Trust-based Access Control Model for Virtual Organizations," in *Fifth International Conference on Grid and Cooperative Computing Workshops, GCCW '06*, 2006, pp. 557-564.

[4] F. Almenarez, A. Marin, D. Diaz and J. Sanchez, "Developing a model for trust management in pervasive devices," in *Pervasive Computing and Communications Workshops*, 2006. *PerCom Workshops 2006. Fourth Annual IEEE International Conference on*, 2006, pp. 267-271.

[5] X. Ma, Z. Feng, C. Xu and J. Wang, "A Trust-Based Access Control with Feedback," in *International Symposiums in Information Processing (ISIP)*, 2008, pp. 510-514.

[6] X. Han-fa, C. Bing-liang and X. Li-lin, "A mixed access control method based on trust and role," in *2010 Second IITA International Conference on Geoscience and Remote Sensing (IITA-GRS)*, 2010, pp. 552-555.

[7] S. Singh, "Trust Based Authorization Framework for Grid Services," *Journal of Emerging Trends in Computing and Information Sciences*, vol. 2, pp. 136-144, 2011.

[8] H. Molina, J. Ullman and J. Widom, Database Systems The Complete Book, 2nd ed, USA: Pearson International Edition, 2009.

[9] R. Elmasri and S. Navathe, Fundamentals of Database Systems, 5th ed, USA: Pearson International Edition, 2007.

[10] F. Etho, K. Takahashi, Y. Hori and K. Sakurai, "Study of Log File Dispersion Management Method", 10th IEEE/IPSJ International Symposium on Applications and the Internet (SAINT), IEEE Computer Society, Seoul, Korea,2010. pp. 371-374.

[11] F. Wang, X. Zhou and C. Zaniolo, "Using XML to Build Efficient Transaction-Time Temporal Database Systems onRelational Databases", The 22nd International Conference on Data Engineering (ICDE), IEEE Computer Society, Atlanta, Georgia, 2006, pp.131-134.