Integrating Trust Notation in XML Database Labelling

Norah Farooqi
Umm Al-Qura University
Makkah, Saudi Arabia
Email: nsfarooqi [AT] uqu.edu.sa

Abstract— **The majority of existing labelling approaches for Extensible Markup Language (XML) databases focus on improving storage techniques and retrieval speed. In contrast, the integrated security aspects in XML database labelling need to be further developed. The labelling syntax in XML databases normally consists of structural relation notation without consideration for security notation. In this paper, a trust labelling for XML databases is proposed so that trust concepts may be included in the labelling. The proposed approach uses labelling to reflect node sensitivity and show the level of protection needed for an object. It can delineate labelling topics with permissions in access control systems for XML databases. The experimental evaluation results show the efficiency, flexibility, and scalability of applying this XML database trust labelling approach.**

Keywords - Labelling, Security, Trust, XML Databases

I. INTRODUCTION

Extensible Markup Language (XML) databases have been intensively developed and widely used for many different applications. Labelling XML databases is a research area that is of continuing interest. Labelling defines and assigns a unique identity to each node in an XML database [1–4]. The label is used to index the XML nodes and show the relationships in tree structures [5–8]. A majority of studies and proposed research in labelling has focused on improving label structures by including extra node relationships, decreasing size, and solving relabelling problems [1–9].

Labelling can be categorised into interval-based labelling, prefix-based labelling, and the recently proposed mathematics-based labelling. Interval-based labelling is also called range scheme or region-based labelling. In this type of labelling, the applied schemes traverse the XML tree from different directions, considering order, size, and level. Interval schemes are widely used with static XML databases [5–7]. These schemes depend on a static sequential numbering system that leads to major issues in the relabelling processes for dynamic XML databases. There are many existing labelling schemes that can be classified as this type, such as pre/post order labelling and containment labelling [7,10]. Figure 1 shows pre/post order labelling in an XML tree.

Prefix-based labelling focuses on the XML tree depth to generate the node label. In this type of scheme, the majority of applied label schemes consist of a parent section and node section [3,11,12]. These labelling schemes work well with both static and dynamic XML databases. Many of these schemes suggest different solutions to solve relabelling issues. This scheme type may face some limited storage issues due to tree width that increase label sizes. There are common models that use this technique in labelling, such as the Dewey scheme, Labelling Scheme for Dynamic XML (LSDX), and ORDPATH. Figure 2 illustrates the Dewey labelling scheme, and Figure 3 shows the LSDX model.

There are some limited studies related to labelling with security concerns [2,4,6,8,9,13]. The prime number labelling scheme is designed to integrate access control for a group of users into the labelling notation. It is structured as "lL1.L2.L3," where "l" reflects the level, "L1" indicates the parent label, "L2" points to a specific node, and "L3" reflects the role-based access control using a prime number [2,4]. Some access control models for XML databases benefit from labelling to speed up the retrieval query process [3,6].

Access control is one of the most common ways to grant secure access in systems. Trust-based access control is a new approach developed to prevent misuse from internal and external users. It monitors user behaviour over time and detects insecure transactions. The system automatically updates access permissions and privileges based on trust values. Trust-based access control is applied to XML databases as controlling system [14].

Labelling could be further developed to cover more security aspects and topics. Until now, there has been no direct work related to trust concepts in XML database labelling. In this research, we integrate the trust topic into XML database labelling fields. This proposed approach aims to extend the label structure by adding trust values that can be used for many security purposes. The new approach can be integrated with access control systems and shows the protection level needed for the data.

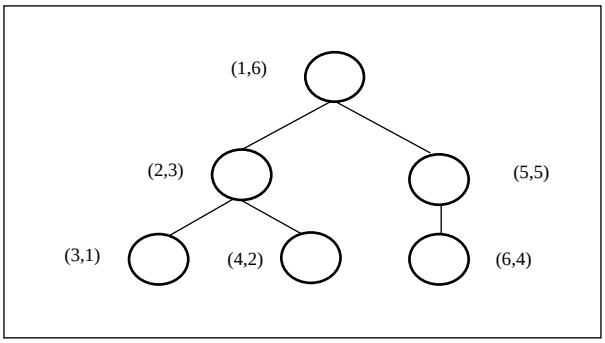

Figure 1. Pre/post labelling scheme.

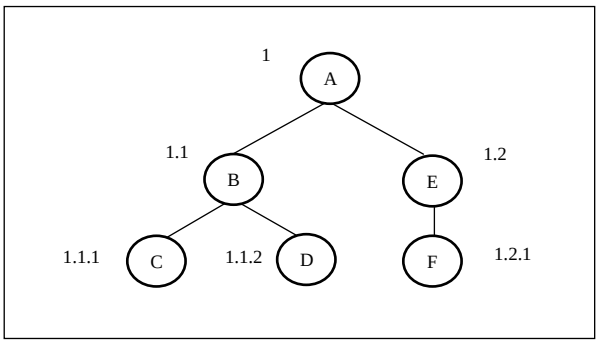

Figure 2. Dewey labelling scheme

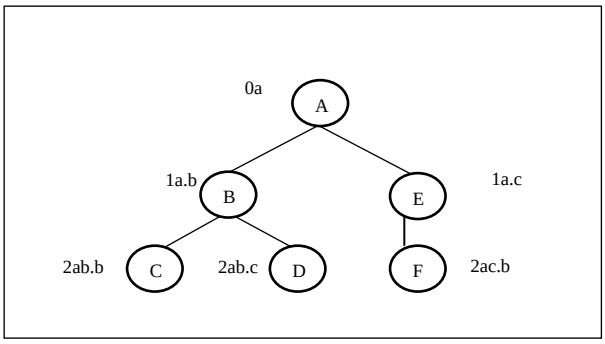

Figure 3. LSDX labelling scheme.

II. PROPOSED TRUST LABELLING APPROACH

This section explains the proposed approach to secure labelling for XML databases based on trust. As mentioned in the previous section, most traditional and proposed labelling schemes focus on node structure and reflect document order. The main goal of this study is to extend the labelling structure by adding an extra section for security purposes. It applies this approach to existing label types and evaluates the results.

The proposed trust label structure consists of the normal basic parts in addition to an extra new part that includes the trust values for the data. A trust value in labelling reflects the sensitivity and importance of the data and can be used for many security tasks. The most important tasks is trust-based access control, which primarily depends on trust values to assign accessibility permissions and allow or deny access processes. The proposed structure for trust labelling adds an extra section at the end that shows the trust value for the data as follows:

Normal label structure. Trust value.

The trust access system is dynamic and assigns trust values for subjects, i.e. system users, and for objects, i.e. nodes in XML databases. This paper focuses only on object trust values, as subject trust values are out of scope. The administrators define the trust values for nodes in the system. Each node has its own trust value that reflects the sensitivity of its data.

To make trust values work appropriately with labelling schemes, the values range from 0 to 100. The value of 0 reflects no trust at all, and 100 reflects full trust. When a node has a low trust value, its data content is general and not sensitive. In contrast, when a node has a high trust value, this reflects the importance and sensitivity of the node and the need for more protection.

This proposed approach can be easily integrated with the majority of existing labelling schemes because adding trust values for labelling structures is direct and covers security concerns. Table I shows some example applications of trust values to existing labelling schemes.

TABLE I. SOME TRADITIONAL LABELLING SCHEMES WITH AND WITHOUT TRUST VALUES FOR NODE C.

Labelling Scheme	Normal Labelling Scheme	Labelling Scheme with Trust Values
Dewey	1.1.1	1.1.1.75
LSDX	2ab.b	2ab.b.75
Pre/post	(3,1)	(3, 1, 75)
Containment	(3,4,3)	(3, 4, 3, 75)

Applying this process to the most common practical types of labelling, Dewey and LSDX, is explained in detail. Dewey labelling (Figure 2) is considered to be a basic type of labelling, and the integration of trust values with Dewey labelling is illustrated in Figure 4.

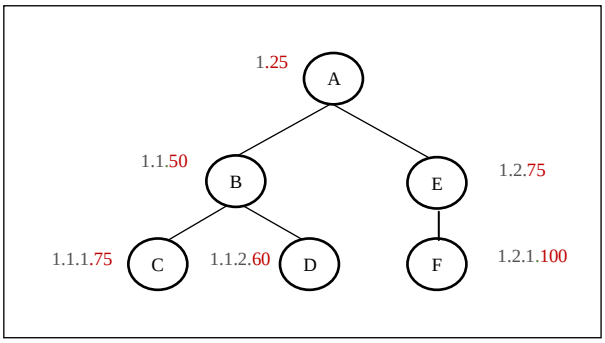

Figure 4. Dewey labelling scheme with trust notation.

LSDX labelling is widely used in practical evaluations [3,6,8]. The normal structure of the LSDX labelling is illustrated in Figure 3 and its integration with the trust value is depicted in Figure 5.

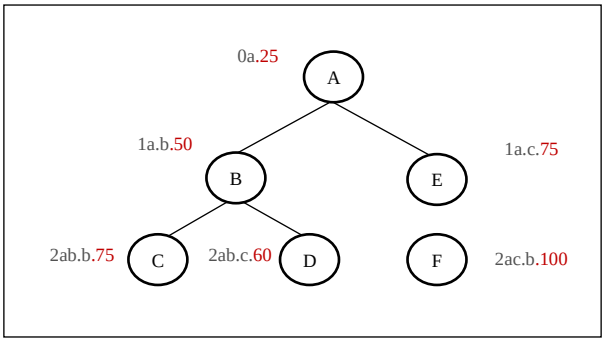

Figure 5. LSDX labelling scheme with trust notation.

III. Experimental Evaluations

We conducted practical evaluations of the proposed trust labelling approach with respect to performance, scalability and storage, and security perspectives. Because Dewey is considered a basic traditional and common labelling type for XML databases, it is used as the main labelling structure in our experiments. Two experiments were performed: a performance evaluation and storage evaluation. In both experiments, the results of the normal Dewey labelling system were compared with those of the trust Dewey labelling system.

A. Performance evaluation experiment

The main goal of the performance evaluation was to evaluate the additional time needed to apply the trust labelling approach to different XML file sizes to discover whether the results are acceptable. This experiment tested the speed of the labelling process and scalability of XML databases by measuring the real time consumed by the trust labelling approach on different sizes of XML files. In addition, it detected the real time for normal Dewey labelling on several XML files. The results of both methods were then compared to determine the difference in consumed time and investigate the time correlation among XML file sizes.

B. Storage evaluation experiment

The storage experiment aimed to determine the storage capacity that is needed to apply the trust labelling approach. Because we use Java, the heap space was considered when running the program. This experiment measured the consumed storage space in the heap memory for the trust labelling approach with different XML file sizes. It also measured the consumed size of the heap memory for the normal Dewey labelling approach. The results for both Dewey labelling approaches were then compared. In addition, the linked list that is used to store the labelling was calculated to determine the general storage.

C. Experimental environment and data

The evaluations were performed on a computer with 2.7 GHz Intel Core i7, 4 GB of main memory, and an OS X EI operating system. The proposed trust labelling approach was implemented using Java (JDK 7u80) on the Net Beans IDE 8.0.2 platform.

There are many real XML datasets and XML benchmarks that can be used in practical experiments. An XML dataset includes data in one specific file, but XML benchmarks generate different file sizes. Some XML datasets are useful for evaluating certain areas, such as the DBLP [15], Treebank [16], and NASA databases [17]. The XML benchmarks used for system and performance evaluation include XMark [18], and XOO7 [19]. In these experiments, XMark was used as a tool because of its popularity and usability in XML applications. It generates many XML databases of different sizes but with the same tree structure. Four XML file datasets were used in the evaluation, and their features are shown in Table II.

TABLE II. Features of the XMark datasets used in the experiments.

File Name	Scaling Factor (F)	File Size (MB)
XDB1	0.001	0.115 MB
XDB2	0.01	1.2 MB
XDB3	0.02	2.4 MB
XDB4	0.05	5.9 MB

IV. Results

The results of the performance evaluation experiment are shown in Figure 6. We first consider the real time consumed when the trust approach is applied to Dewey labelling. The time consumed is 236 ms when the XML database size is 0.115 MB and increases gradually until it reaches 21,504 ms when the size of the XML database is around 5.9 MB. Clearly, the time consumed for trust Dewey labelling is directly affected by the XML database size. With respect to the consumed time for normal Dewey labelling, the required time is 233 ms when the XML database size is 0.115 MB and 19,489 ms for a XML database size of 5.9 MB. Finally, we compare the consumed time for the normal Dewey labelling and trust Dewey labelling systems. Table III shows the time differences between the two approaches. When the XML database size is 0.115 MB, the time difference is just 3 ms, and when the XML database size is around 5.9 MB, the time difference is 2,015 ms. The time differences are quite small. Clearly, adding the trust part into the labels does not add excessive processing time.

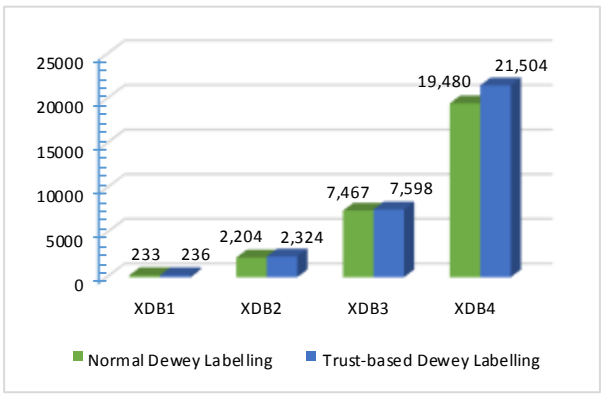

Figure 6. Comparison of the consumed real time for Dewey labelling with and without the trust notation.

TABLE III. TIME DIFFERENCES BETWEEN NORMAL DEWEY AND TRUST DEWEY LABELLING.

File Name	Time Differences (Millisecond)
XDB1	3
XDB2	120
XDB3	131
XDB4	2015

The results of the storage evaluation experiments are shown in Figure 7. We first consider the heap size in memory when the proposed trust labelling approach is applied to Dewey labelling. When the XML database size is 0.115 MB, the space consumed is 6.3 MB, and the space for the heap increases markedly to 54.3 MB when the XML databases size is around 5.9 MB. The heap size is clearly directly affected by the XML database size. We next consider the heap size consumed by normal Dewey labelling. The heap size results are 4.7 MB when the XML database is 0.115 MB and 52.4 MB when the XML database size is around 5.9 MB. Comparing the heap size consumed by the two labelling approaches for different XML database sizes indicates that the differences are acceptable when an extra part is added to the labelling. Table IV shows the differences in heap size for the Dewey labelling approaches with and without trust notation. When the XML database size is 0.115 MB, the difference is 1.6 MB, and when the XML database is 5.9 MB, the difference is 1.9 MB. The average difference is 1.325 MB.

In addition, this experiment measured the linked list size to further evaluate the consumed storage space. The linked list sizes for both Dewey labelling approaches are listed in TableV.

Figure 7. Consumed heap size for both normal and trust-based Dewey labelling.

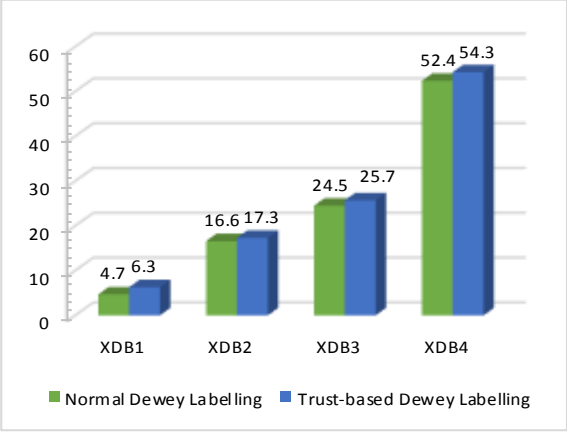

TABLE IV. CONSUMED HEAP SIZE DIFFERENCE FOR NORMAL AND TRUST-BASED DEWEY LABELLING.

File Name	Heap Size Differences (MB)
XDB1	1.6
XDB2	0.6
XDB3	1.2
XDB4	1.9

TABLE V. LINKED LIST SIZE FOR DIFFERENT XML DATABASES.

File Name	Linked List Size (MB)
XDB1	0.039
XDB2	0.40
XDB3	0.77
XDB4	1.96

V. CONCLUSION

This paper proposed a new approach for labelling XML databases via integrating trust notation into the label structure. The trust labelling approach extends the labelling with a trust value that reflects the importance of each node and shows the needed level of security for each node depending on its content. Using trust notation in labelling can provide connections and relations between labelling and access control in XML databases. The trust labelling can also be used to support trust-based access control for XML databases by storing the trust values of database objects in the labelling.

The experiments evaluated the performance and storage aspects of the proposed scheme. The performance evaluation showed that the consumed real time for trust labelling is acceptable when we compare the results with the consumed real time for normal labelling. The experiments tested the consumed storage for different XML databases when the trust labelling approach was applied by measuring the size of the resulting heap and linked list. All the experimental results illustrate the scalability of the proposed approach and suggest that it is worth proceeding with the integration of trust concepts into labelling.

VI. REFERENCES

[1] Ghaleb T, Mohammed S. A dynamic labeling scheme based on logical operators: A support for order-sensitive XML updates. In 3rd International Conference on Recent Trends in Computing, 2015, pp. 1211–1218.

[2] An D, Park S. New access control for secure query processing over XML data stream. International Conference on Convergence Information Technology, 2007, pp. 1764–1763.

[3] Duong M, Zhang Y. An integrated access control for securely querying and updating XML data. 19th Conference on Australasian Databases, Australia, 2008, pp. 75–84.

[4] An D, Park S. Efficient access control labeling scheme for secure XML query processing. Computer Standards Interfaces, 2011, 33(5): 439–447.

[5] Al-Shaikh R, Hashim G, Binhuraib A, Mohammed S. A modulo based labeling scheme for dynamically ordered XML trees. In 5th International Conference on Digital Information Management (ICDIM), 2010, pp. 213–221.

[6] Duong M. Access Control Model and Labelling Scheme for Efficient Querying and Updating XML Data. PhD, Victoria University, 2010.

[7] Xu L, Ling T, Wu H. Labeling dynamic XML documents: An order centric approach. IEEE Transactions on Knowledge and Data Engineering, 2010, 24(1):100-113.

[8] Duong M, Zhang Y. LSDX: A new labeling schema for dynamically updating XML data. In 16th Australasian Database Conference, 2005, pp. 185–193.

[9] Khaing A, Thein N. A persistent labeling scheme for dynamic ordered XML trees. IEEE/WIC/ACM International Conference on Web Intelligence, 2006, pp. 498–501.

[10] Sans V, Laurent D. Prefix based numbering schemes for XML: Techniques, applications and performances. In Proceeding VLDB Endow, 2008, pp. 1564–1573.

[11] Xu L, Ling T, Wu H,Bao Z. DDE: From Dewey to a fully dynamic XML labeling scheme. In 35th SIGMOD International Conference on Management of Data, USA, 2009, pp. 719–730.

[12] O'Neil P, O'Neil E, Pal S, Cseri I, Schaller G, Westbury N. ORDPATHs: Insert-Friendly XML node labels. ACM SIGMOD International Conference on Management of Data, France, 2004, pp. 903–908.

[13] Gabillon A, Fansi M. A persistent labelling scheme for XML and tree databases. In 1st International Conference on Signal Image Technology and Internet-Based Systems, Cameroon, 2005, pp. 110–114.

[14] Farooqi N, North S. Trust-Based Access Control for XML Databases. In 6th International Conference for Internet Technology and Secured Transactions, UAE, 2011, pp. 764–765.

[15] Dblp, The Dblp Computer Science Bibliography [Online], 2013, Available: http://dblp.uni-trier.de/db/ [Accessed 8-2-2017].

[16] Treebank, The Penn Treebank Project [Online],1999 ,Available :http://www.cis.upenn.edu/~treebank/ [Accessed 8-2-2017].

[17] Nasa, Gsfc Open Source Software [Online],2001,Available :http://opensource.gsfc.nasa.gov/ [Accessed 8-2-2017].

[18] Schmidt A, Xmark: an Xml Benchmark Project [Online], 2003, Available : http://www.xml-benchmark.org/ [Accessed 8-2-2017].

[19] Li Y, The Xoo7 Benchmark [Online], 2003,Available :http://www.comp.nus.edu.sg/~ebh/XOO7.html [Accessed 8-2-2017].