Secure Querying of Recursive XML Views: A Standard XPath-based Technique*

Houari Mahfoud
University of Nancy 2 & INRIA-LORIA Grand Est
Nancy, France
houari.mahfoud@loria.fr

Abdessamad Imine
University of Nancy 2 & INRIA-LORIA Grand Est
Nancy, France
abdessamad.imine@loria.fr

ABSTRACT

Most state-of-the art approaches for securing XML documents allow users to access data only through authorized views defined by annotating an XML grammar (e.g. DTD) with a collection of XPath expressions. To prevent improper disclosure of confidential information, user queries posed on these views need to be *rewritten* into equivalent queries on the underlying documents, which enables us to avoid the overhead of view materialization and maintenance. A major concern here is that XPath query rewriting for recursive XML views is still an *open* problem. To overcome this problem, some authors have proposed rewriting approaches based on the non-standard language, "Regular XPath", which is more expressive than XPath and makes rewriting possible under recursion. However, query rewriting under Regular XPath can be of exponential size as it relies on automaton model. Most importantly, Regular XPath remains a theoretical achievement. Indeed, it is not commonly used in practice as translation and evaluation tools are not available. In this work, we show that query rewriting is always possible for recursive XML views using only the expressive power of the standard XPath. We propose a general approach for securely querying of XML data under arbitrary security views (recursive or not) and for a significant fragment of XPath. We provide a linear rewriting algorithm that is efficient and scales well.

Categories and Subject Descriptors

H.2.7 [**Database Administration**]: Security, integrity and protection—*Access control*

General Terms

Algorithm, Security

Keywords

XML Access control, XML views, XPath

1. MOTIVATION

XML is rapidly emerging as the new standard for data representation and exchange across the web. With this emergence, a challenge is raised with regards to the securing of XML content. Specifically, an XML document may be queried simultaneously by different user groups. An *access policy* may be imposed for each group of users, specifying parts of the document these users are granted access to. The problem of *secure XML querying* is to enforce these access policies at query time. More formally, given an XML document T and a user query q, our goal is to ensure that the evaluation of q over T returns only information in T that the user is allowed to access. To fulfill this requirement, most of the access control models proposed during the last decade are based on the notion of *security views*.

We briefly review the main principle of the XML security view-based approaches. Given a DTD D, for each class of users a security view is defined by annotating D with some access conditions to specify the accessibility of XML nodes in instances of D. The annotated version of D is later sanitized to derive a view D_v which represents to users the schema of all and only accessible data. Next, the security view V is defined as a pair (D_v, σ) where σ is a function used to extract for each XML document T conforms to D, its virtual view[1] T_v, conforms to D_v, and which represents only information in T accessible to the users. Each query over the view T_v is translated into an equivalent one in order to be evaluated over the original data T.

Problem Statement. Most of the existing XML access control models deal only with non-recursive security views [1, 4]. A security view is recursive, if its view D_v is recursive (at least one of its element types is defined directly or indirectly in terms of itself). For each pair of element types A and B in the view D_v connected with parent-child relation (note that some element types may be hidden between A and B), $\sigma(A,B)$ returns the XPath expressions to reach B element from A element in the original DTD D. However, if D is recursive, then function σ cannot be defined since there may be an infinite set of paths which connect A to B in D.

Example 1. Consider the DTD D_1 depicted in Fig. 1(a) where dashed edges denote disjunction. Suppose elements C are inaccessible. Then the view D_v of D_1 can be derived as represented in Fig. 1(c) by eliminating the information relative to C. The function σ is defined as: $\sigma(root,A)=A$, $\sigma(A,D)=(D \cup C/D)$. The user query $/root/A/D$ over D_v is rewritten over D_1 into: $/root/\sigma(root,A)/\sigma(A,D)=/root/A/(D \cup C/D)$. Consider now the recursive DTD D_2 depicted in Fig. 1(b) and suppose elements C are inaccessible. Then the view of this

*Extended version of this paper can be found in [6].

[1]It should be noted that view instances are not materialized.

WWW 2012 Companion, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1230-1/12/04.

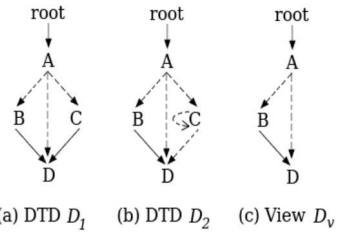

Figure 1: DTD and DTD view Example.

DTD is the same than depicted in Fig. 1(c). The user query $/root/A/D$ over D_v cannot be rewritten to an equivalent one over D_2 since $\sigma(A,D)$ leads to infinitely many paths. This query can be rewritten using the *Kleene star* into: $/root/A/(C)^*/D$. However, the Kleene star cannot be expressed in the standard XPath. □

Consider the downward fragment of XPath denoted by $\mathcal{X}$ and defined as follows:

$p := \alpha::lab \mid p[q] \mid p/p \mid p \cup p$
$q := p \mid p/text()='c' \mid q \text{ and } q \mid q \text{ or } q \mid not\ (q)$
$\alpha := self \mid child \mid descendant \mid descendant\text{-}or\text{-}self$

where p denotes an XPath query, *lab* refers to element type or $*$ (that matches all types), $\cup$ stands for union, c is a string constant, α is the XPath axis relations, and finally, the expression q enclosed in [.] is called a *qualifier*. Theoretically, this XPath fragment has some interesting decision results. Practically, it is commonly used and is essential to XQuery, XSLT and XML Schema [2].

It has been proved in [1] that query rewriting is not always possible in $\mathcal{X}$ in case of recursive security views. That is why some authors [2, 3] resort to the use of Regular XPath to avoid this problem. However, Regular XPath remains of theoretical use since no evaluation tools have been provided for practical use of this language.

To the best of our knowledge, no practical approach exists for answering queries under recursive XML security views.

2. CONTRIBUTION

We show that the expressive power of the standard XPath is sufficient to overcome the query rewriting problem over recursive views. We propose to extend the fragment $\mathcal{X}$ as follows:

$p := \alpha::lab \mid p[q] \mid p/p \mid p \cup p \mid p[n]$
$q := p \mid p/text()='c' \mid q \text{ and } q \mid q \text{ or } q \mid not\ (q) \mid p{=}\varepsilon::lab$
$\alpha := self \mid child \mid descendant \mid descendant\text{-}or\text{-}self \mid parent \mid ancestor \mid ancestor\text{-}or\text{-}self$

we enrich $\mathcal{X}$ by the upward-axes (*parent*, *ancestor*, and *ancestor-or-self*), the *position* and the *node comparison* predicates. The position predicate, defined with $[n](n \in \mathrm{N})$, is used to return the n^{th} node from an ordered set of nodes. The *node comparison* predicate $[p_1{=}p_2]$ is valid at a node n only if the evaluation of the right and left XPath queries at n result in exactly the same single node. We summarize the resulted fragment by $\mathcal{X}^{\Uparrow}_{[n,=]}$.

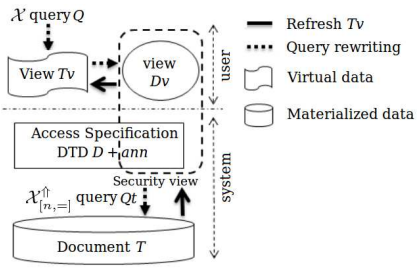

Figure 2: XML Access Control Framework.

Based on this extension, we present our access control framework in Fig. 2. For each class of users, the administrator defines an access policy over the DTD D. The view D_v is derived first w.r.t this policy and given to the users to formulate their queries. For each instance T of D, we compute a virtual view T_v of T to show all and only accessible data to the users. Next, each query Q defined in $\mathcal{X}$ over D_v is efficiently rewritten to another one Q_t defined in $\mathcal{X}^{\Uparrow}_{[n,=]}$ over D such that: the evaluation of Q over T_v gives the same result than evaluating Q_t over T.

We have developed a practical tool to improve effectiveness of our rewriting approach. A performance study is conducted using a real-life recursive DTD and a various forms of XPath queries. The experimental results showed the efficiency of our XPath query rewriting approach w.r.t the answering approach based on the materialization of the view.

3. CONCLUSION

The proposed approach yields the first practical solution to rewrite XPath queries over recursive XML security views, using only the expressive power of the standard XPath. The extension of the downward class of XPath queries with some axes and operators has been studied in order to make query rewriting possible under recursion. We have investigated the use of the same approach to securely updating XML. Our first results in this direction can be found in [5]. We plan in the future to provide an optimized version of our solution and to extend it to handle larger fragments of XPath.

4. REFERENCES

[1] W. Fan, C.-Y. Chan, and M. Garofalakis. Secure xml querying with security views. *In SIGMOD 2004, pp. 587-598.*

[2] W. Fan, F. Geerts, X. Jia, and A. Kementsietsidis. Rewriting regular xpath queries on xml views. *In ICDE 2007, pp.666-675.*

[3] B. Groz, S. Staworko, A.-C. Caron, Y. Roos, and S. Tison. Xml security views revisited. *In DBPL 2009, pp, 52-67.*

[4] G. Kuper, F. Massacci, and N. Rassadko. Generalized xml security views. *In SACMAT 2005, pp. 77-84.*

[5] H. Mahfoud and A. Imine. A general approach for securely querying and updating xml data. *INRIA Report.* `http://hal.inria.fr/hal-00664975/en`, *2012.*

[6] H. Mahfoud and A. Imine. Secure querying of recursive xml views: A standard xpath-based technique. *INRIA Report.* `http://hal.inria.fr/hal-00646135/en`, *2011.*