WmXML: A System for Watermarking XML Data

Xuan ZHOU[1] HweeHwa PANG[2] Kian-Lee TAN[3] Dhruv MANGLA[3]

[1]L3S Research Center
Expo Plaza 1
Hanover, Germany 30539
zhou@l3s.de

[2]Institute for Infocomm Research
21 Heng Mui Keng Terrace
Singapore 119613
hhpang@i2r.a-star.edu.sg

[3]Department of Computer Science
National University of Singapore
3 Science Dr 2, Singapore 117543
{tankl, dhruvman}@comp.nus.edu.sg

Abstract

As increasing amount of data is published in the form of XML, copyright protection of XML data is becoming an important requirement for many applications. While digital watermarking is a widely used measure to protect digital data from copyright offences, the complex and flexible construction of XML data poses a number of challenges to digital watermarking, such as re-organization and alteration attacks. To overcome these challenges, the watermarking scheme has to be based on the usability of data and the underlying semantics like key attributes and functional dependencies. In this paper, we describe WmXML, a system for watermarking XML documents. It generates queries from essential semantics to identify the available watermarking bandwidth in XML documents, and integrates query rewriting technique to overcome the threats from data re-organization and alteration. In the demonstration, we will showcase the use of WmXML and its effectiveness in countering various attacks.

1 Introduction

XML is emerging as a new standard for information representation and exchange over the internet. As increasing amount of commercial data is exchanged or published in this format, unauthorized duplication and distribution of XML data become a mandatory concern for many internet applications. An example is a job agent's web site, who would like to prevent his job advertisements from being stolen and posted on other web sites. A commercial digital library also would need to safeguard its copyright over its collection of knowledge information.

Digital watermarking is one of the most widely used measure to protect digital information from copyright offences. By introducing indiscernible perturbations into the data, it marks the data with copyright information, through which the publisher can prove his ownership or trace any reproduction of the data. Traditional techniques on digital watermarking have focused on multimedia data such as image [7], audio [2] and video [4]). Although some recent studies have investigated the watermarking of non-multimedia data like software [3] and relational data [1, 6], no satisfactory technique has been proposed for watermarking XML data. In fact, this semi-structured data offers a number of technical challenges to watermarking. They include:

(A) Identifying data elements and structures for watermarking: XML data is composed of a number of data elements, and structures that link the data elements together based on their relationships. Both of them could contain bandwidth for watermarking. However, to identify each data element or structure unit, it is inadequate to treat it in isolation; rather, we should consider its relationships with other data elements and structures. As illustrated by the XML data in figure 1(a), db1.xml contains a set of publication records. If we identify each <year> element by its value (i.e., 1998), we lose the distinction between the two <year> elements under the two different books. This significantly reduces the amount of watermark bandwidth that can be used. Instead, a better identifier of the <year> element would be the value of its sibling element <title>.

(B) Resilience to data reorganization and alteration: When data elements are identified through relationships and structures, adversaries could reorganize the data to prevent the elements from being correctly identified. The flexible format of XML data in particular enables it to be reorganized easily. As shown in figure 1, an adversary could redesign the schema

Proceedings of the 31st VLDB Conference, Trondheim, Norway, 2005

```
<db>
  <book publisher="mkp">
    <title>Readings in Database Systems</title>
    <author>Stonebraker</author>
    <author>Hellerstein</author>
    <editor>Harrypotter</editor>
    <year>1998</year>
  </book>
  <book publisher="acm">
    <title>Database Design</title>
    <writer>Berstein</writer>
    <writer>Newcomer</writer>
    <editor>Gamer</editor>
    <year>1998</year>
  </book>
  ...
</db>
```

(a) db1.xml

```
<db>
  <publisher name="mkp">
    <author name="Stonebraker">
      <book>Readings in Database Systems</book>
      <book>XML Query Processing</book>
    </author>
    <author name="Hellerstein">
      <book>Readings in Database Systems</book>
      <book>Relational Data Integration</book>
    </author>
    ...
  </publisher>
  <publisher name="acm">
    ...
  </publisher>
  ...
</db>
```

(b) db2.xml

Figure 1: Structure Reorganization

of db1.xml into db2.xml, without losing any information. In addition, he can also alter some parts of the structure (delete or add some edges or data elements) to hinder the detection of any embedded watermarks. Thus, the identifiers of data elements must be persistent enough to survive any form of reorganizations and alterations.

(C) Identifying data redundancy: Innate redundancies within the XML data could severely degrade the watermark quality. For example, db1.xml contains the semantic that an editor only works for one publisher. The semantic produces many duplicated *publisher* entries that correspond to the same editor. If these duplicates are selected to embed different bits of a watermark, the watermark can be erased easily by making all the duplicates identical. In contrast to challenge (A) which requires different data elements to be differentiated, this problem requires duplicates of the same data element to be identified and treated accordingly during watermarking.

In [5], the only work on watermarking semi-structured data that we are aware of, utilizes a graph labeling scheme to overcome these problems. However, without taking into account the semantics within the data, that scheme is still vulnerable to data reorganization. It also ignores the redundancy problem. In contrast, WmXML handles the above challenges by utilizing semantics and queries: (i) It uses query templates to naturally represent data usability. (ii) It generates queries from essential semantics to identify data elements and structure units for watermarking, so as it is difficult for reorganization or alteration to disable the identifiers without destroying data usability. (iii) When creating the identity queries, it considers the internal semantics to avoid vulnerabilities caused by redundancy, while taking advantage of the available watermark capacity.

In the remainder of this paper, we introduce the techniques of WmXML in more details, and outline the demonstration which is intended to show its effectiveness in countering various attacks.

2 Main Techniques

The techniques used by WmXML has been reported in [9]. These techniques include measuring data usability by the correctness of query results, using queries to identify data elements, and constructing identity queries through essential semantics.

2.1 Data Usability

An effective watermarking system should satisfy two basic requirements. First, it should be able to insert watermark imperceptibly, i.e. without degrading the usability of the data. Second, the embedded watermark should be sufficiently robust, so that it is difficult for adversaries to remove it without destroying the usability of the data. Usability is an important metric for a watermarking technique.

The usability of XML data is a measure of whether the data can provide useful and correct information to users. For example, a user would like to know "*Who is the author of the book titled DB Design?*", and expects to get the answer from either db1.xml or db2.xml in figure 1. The user's query could then be written into an XPath expression "*db/book[title='DB Design']/author*" to be conducted on db1.xml, or another XPath expression "*db/publisher/author[book='DB Design']@name*" to be conducted on db2.xml. The two queries would return the same results. Thus, db1.xml and db2.xml would be equally usable to the user. If one of them can no longer return correct result after some modification, its usability decreases. Based on this, WmXML uses the correctness of query results to measure the usability of XML data. A set of query templates, e.g. "*db/book[title]/author*", are specified

by user to depict data usability. After watermarking or attacks, if a certain fraction of the results to these query templates are destroyed, the usability of the XML data is regarded destroyed.

2.2 Identity Queries

Both the data elements and structure units in an XML document could be used to embed watermarks. But as attackers could reorganize the XML structure to prevent the watermarked data elements and structure units from being correctly identified, the identifiers created by WmXML for the data elements or structure units needs to be independent of the physical organization of data. Query is such a kind of identifier, as it could be adapted easily to different organizations of the same data through query rewriting techniques. For instance, though attacker could reorganize db1.xml into db2.xml, the query "*db/book[title='DB Design']/author*" on db1.xml can be rewritten as '*db/publisher/author[book='DB Design']@name*" on db2.xml and retrieves the same data element. Inspired by this, WmXML uses queries as the identifiers of the data elements in XML data. The watermarking scheme works as follows:

1. Initialization: Specify a schema and validate the XML data according to the schema. Specify a set of query templates to represent data usability. A secret key is used to select a number of data elements or structure units to embed watermark bits. Create queries as identifiers of these data elements or structure units, and safeguard the set of queries (denoted by Q) along with the secret key.

2. Watermark Insertion: Execute the queries in Q on the original data to retrieve the data elements or structure units. Next, the watermark bits are embedded into these elements or units through selected watermark embedding algorithms.

3. Watermark Detection: Execute the same set of queries to retrieve the data elements or structure units embedded with watermark bits, and reconstruct the watermark from them. As the schema and the XML data could be reorganized by attackers, these queries may have to be rewritten for the reorganized data (figure 2). The query rewriting could be conducted according to the mappings between the original schema and the new schema. While research into XML query rewriting is still on-going, there are already some practical schemes, such as [8].

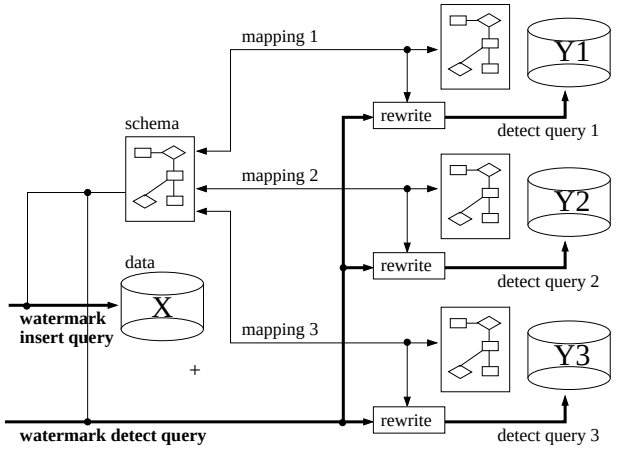

Figure 2: Watermark Insertion and Location

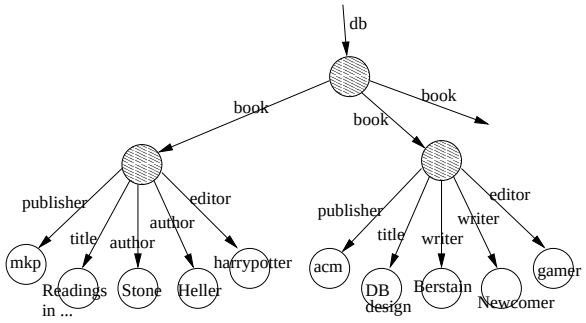

Figure 3: db1.xml As A Tree

2.3 Identifier Creation

The queries to identify data element should satisfy three criteria. First, they should be able to differentiate different data elements, in order to economize scarce watermarking bandwidths. Second, they should be able to identify data redundancies to overcome the threat of removal attacks. Finally, they should be closely related to the usability of the data, so as to survive reorganization and alteration attacks.

An XML document can usually be modeled as a tree structure (figure 3), in which two major forms of semantics could be found – keys and functional dependencies. In db1.xml of figure 3, attribute *title* could work as the key of element *book*, as the title of each publication is usually unique. If each editor only works for one publisher, there also exists functional dependency "*editor* → *publisher*". The keys and functional dependencies compose the crucial data relationships and are also responsible for data redundancies. WmXML constructs identifiers from these keys and functional dependencies, so that the identifiers can differentiate different data elements and be independent from data redundancies. The query templates used to represent data usability are also considered in the procedure of identifier creation, so that the constructed identifiers are closely related to the data usability and are difficult to be tampered. The detailed methods for constructing identifiers are presented in [9].

3 System Construction

WmXML was implemented in C++, following the architecture in figure 4. The system contains three components – an XML query engine, an encoder and a decoder. The XML query engine provides an access

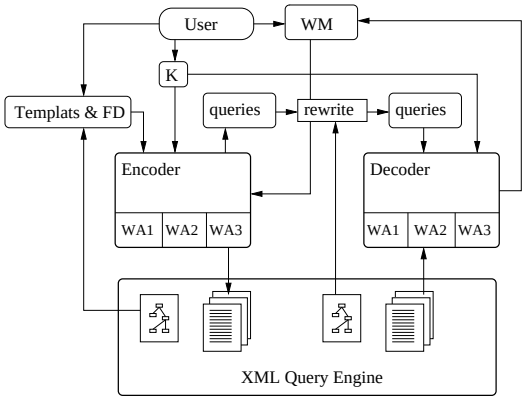

Figure 4: System Architecture

interface to XML data, and the encoder and decoder are responsible for watermark insertion and detection respectively. When embedding watermarks, the user inputs a watermark, a secret key, a set of query templates to depict data usability, along with the keys and FDs that he discovered from the schema of the copyrighted semi-structured data. Next, the encoder embeds the watermark into the data and generates a set of identifying queries to be kept by the user. In watermark detection, the user provides the same secret key and query set for the decoder to rewrite the queries and retrieve the watermark from the data. As XML could contain various types of data, the system prepares various plug-in watermarking algorithms for different data types, represented by WAi in the figure. The data types currently supported by the system include numeric data and images. Due to limitations of current technology, the query rewriter still needs human intervention.

4 Demonstration Overview

In this demonstration, we will primarily show (i) how WmXML embeds and retrieves watermark in/from XML documents and (ii) the effectiveness of WmXML in countering various attacks.

In the first part, we will apply the watermarking system to a few sets of real world semi-structured data to demonstrate how easily the system can be used. When embedding watermarks, a user needs to represent data usability through a set query templates, identify the important keys and FDs from the data schema and specify the data elements with watermark capacity. Then the watermark will be automatically embedded. We will show that the watermark capacity is fully utilized by WmXML, and the usability of XML document would not be seriously degraded. In watermarking detection, the user only need to provide the correct secret key and a set of queries, the watermark will be automatically reconstructed.

In the second part, we will perform a number of attacks on a watermarked XML document. The attacks include (A) data alteration: modify the elements or the structures of the semi-structured data to destroy the embedded watermark; (B) data reduction: selectively use a subset of the semi-structured data and discard the rest; (C) data re-organization: reorganize the data according to a new schema and reorder the data elements; (D) redundancy removal: identify and remove redundancies within the data. Then, we will show that (i) the watermark can still be successfully reconstructed if these attacks have not destroyed the data usability or (ii) once the attacks manage to destroy the watermark, the data usability will also be destroyed. Here, we measure the usability through the correctness of the results to the predefined query templates.

References

[1] R. Agrawal and J. Kiernan. Watermarking relational databases. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB)*, 2002.

[2] Laurence Boney, Ahmed H. Tewfik, and Khaled N. Hamdy. Digital watermarks for audio signals. In *Proceedings of International Conference on Multimedia Computing and Systems*, pages 473–480, 1996.

[3] Christian Collberg and Clark Thomborson. Software watermarking: Models and dynamic embeddings. In *Proceedings of the Principles of Programming Languages 1999, POPL'99*, pages 311–324, 1999.

[4] Frank Hartung and Bernd Girod. Watermarking of uncompressed and compressed video. *Signal Processing*, 66(3):283–301, 1998.

[5] Radu Sion, Mikhail Atallah, and Sunil Prabhakar. Resilient information hiding for abstract semi-structures. In *Proceedings of Workshop on Digital Watermarking IWDW 2003*, volume 2939/2004, pages 141–153.

[6] Radu Sion, Mikhail Atallah, and Sunil Prabhakar. Rights protection for relational data. In *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*, pages 98–109. ACM Press, 2003.

[7] Mitchell D. Swanson, Bin Zhu, and Ahmed H. Tewfik. Transparent robust image watermarking. In *Proceedings of the 1996 SPIE Conf. on Visual Communications and Image Proc.*, volume III, pages 211–214, 1996.

[8] Cong Yu and Lucian Popa. Constraint-based xml query rewriting for data integration. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, pages 371–382. ACM Press, 2004.

[9] Xuan Zhou, HweeHwa Pang, and Kian-Lee Tan. Right protection for semi-structured data through digital watermarking. Technique Report, Group of Secure DBMS, National University of Singapore, 2004.