Towards a Lightweight Framework for Privacy Preserving P2P XML Databases

Angela Bonifati
ICAR Inst. – National Research Council, Italy
bonifati@icar.cnr.it

Alfredo Cuzzocrea
DEIS Dept. – University of Calabria, Italy
cuzzocrea@si.deis.unical.it

Abstract

The problem of securing XML databases is rapidly gaining interest for both academic and industrial research. It becomes even more challenging when XML data are managed and delivered according to the P2P paradigm, as malicious attacks could take advantage from the totally-decentralized and untrusted nature of P2P networks. Starting from these considerations, in this paper we propose the guidelines of a distributed framework for supporting (i) secure fragmentation of XML documents into P2P XML databases by means of lightweight XPath-based identifiers, and (ii) the creation of trusted groups of peers by means of "self-certifying" XPath links that exploit the benefits of well-known fingerprinting techniques.

Keywords *Privacy Preservation of XML Data, P2P XML Databases, XML Data Management over P2P Networks.*

1. Introduction

SECURING XML databases is a novel and leading research challenge. With similar attractiveness, very large publish-subscribe systems are rapidly gaining momentum as much as innovative knowledge processing and delivery paradigms like Web and Grid Services take place. In such scenarios, due to its well-understood features, XML is widely used as basic language for both representing and processing semi-structured data located in remote databases within distributed and heterogeneous settings. In our work, for the sake of simplicity, we model an XML database as a (large) collection of XML documents on top of which traditional DBMS-like indexing and query functionalities are implemented. Providing solutions for securing XML data and guaranteeing privacy preservation over sensitive XML documents has, thus, a critical role in next-generation distributed and pervasive applications, and, especially, in very large publish-subscribe systems, which in this paper are considered as a (significant) case study of distributed environments.

This privacy preservation issue is even more important when XML data are managed and delivered according to the popular P2P paradigm, as malicious attacks could take advantage from the totally-decentralized and untrusted nature of P2P networks. This scenario gets worse when these networks admit *mobile peers*, since mobile devices have limited power and resource capabilities thus they cannot process huge amounts of data. By contrary, in very large publish-subscribe systems, the information is often carried by *data providers* that tend to enclose it into very large XML databases, and *data replicators* maintaining *views* on top of those databases. Starting from these considerations, in this paper we address the issue of efficiently querying P2P XML databases in very large publish-subscribe systems while guaranteeing privacy preservation over sensitive XML documents, and propose the guidelines of a distributed framework allowing us to accomplish this task efficiently.

Specifically, such a framework is based on (*i*) secure fragmentation of XML documents in very large publish-subscribe systems by means of lightweight XPath-based identifiers, and (*ii*) trusted groups of peers by means of secure XPath links that exploit the benefits deriving from well-known *fingerprinting techniques* [11]. Due to the latter feature, we name as "self-certifying" our innovative XPath links. For what concerns the P2P layer, our framework relies on an *XML fragmentation model for P2P networks* presented in [2]. Therefore, properly, we assume to deal with P2P XML databases storing fragments of XML documents. For what concerns the query layer, our framework can efficiently support secure XML fragments querying in both *schema-aware* and *schema-less* mode, which is very useful for P2P networks.

2. (Secure) Query Scenarios

The peers of the distributed environment we investigate are usually not interested to the entire view published by the local replicator, but to point some data contained into it (i.e., extracting *fragments* of the view). To avoid excessive computational overheads, the peers prefer to access data replicators rather than data providers, while still wishing to access data providers whenever needed (e.g., due to load balancing issues). While the peers fetch their fragments of interest, they also keep links to the related fragments stored on their neighbors. A link is represented as an absolute XPath expression within the original document, which allows us to uniquely determine the related fragment via prefix-matching. Peers linked to each other form an *interest group* (also called *acquaintance* in [1]), i.e. a group of peers sharing semantically related fragments, i.e. fragments related to concepts having a certain relationship with respect to a specific application domain. As an example, fragments related to concepts like

10th International Database Engineering and Applications Symposium (IDEAS'06)
0-7695-2577-6/06 $20.00 © 2006 IEEE

"medical services", "train timetables", and "accommodations" could be shared by peers belonging to the same interest group. Without any loss of generality, a same peer can belong to multiple interest groups. In such a scenario, a pertinent problem is guaranteeing privacy preservation while accessing sensitive fragments, i.e. avoiding that untrusted peers can access secure fragments. It should be noted that while security issues concerning a given data provider/replicator can be handled in a centralized manner by adopting well-recognized solutions, devising privacy-conscious P2P XML data processing in environments like the one described above is still an open and, due the same nature of P2P networks, not-completely solved problem. As a consequence, in this paper we mainly focus on the latter research challenge.

A peer has two options when he/she is looking for further information. In case he/she needs up-to-date information, he/she has to access again the replicator while however consuming bandwidth and resources. In turn, when he/she needs static data, i.e. data with low variability over time, he/she is also offered to access his/her neighbor fragments via the above-mentioned links. Consider an example of this scenario in which a replicator publishes a view on public services available in a given urban area. Among the others, this view contains information about the local market stocks and the public transportation timetable. Notice that the former are subject to a great variability over time, whereas the latter remains unchanged for a long time (e.g., a season). Imagine that a (mobile) user Bob is seeking for information about both kinds of data. For stock trends, Bob will directly access the local replicator, whereas for the departure times of trains, he can still access (trusted) neighbor's data (e.g., Alan, who had previously downloaded the same fragment of interest). This strategy is possible thanks to our fragmentation model, which lets build links between related fragments. For instance, Alan's fragment is identified by the path expression `/cityservices/publictransport/timetables/trains`, and Bob's actual fragment is rooted in the path expression `/cityservices/events`, which prefix-matches the former.

A trusted peer can access the others' secure fragments by both directly browsing the links (as in the example above) or by formulating an arbitrary XQuery query on (schema-aware) documents (note that fragments of documents are, in turn, documents). A further capability of our framework lets exploit the set of path expressions of an interest group to aid query formulation in absence of a schema (i.e., against schema-less documents). It should be noted that the latter is a common situation in highly dynamic P2P networks, since, as shown in [1], a *global mediated schema* is not a reasonable assumption for such networks.

3. The P2P XML Fragmentation Model in a Nutshell

In [2], we focus on how to represent the fragments of a document split on several peers, in order to be able to (e.g., partially) re-unify those at wish. Hence, a fragment is a sub-document of the original document maintaining XPath links to the latter, thus retaining the convenient side effect of building a *decentralized catalog* over the P2P network. In our model, we do not expect the peers to agree on one schema and keep it updated with respect to every neighbor's changes to local data. On the contrary, every peer has one or more fragments and it may execute global queries without necessarily knowing the global schema. Thus, as we said above, we can handle both schema-less documents, which are very common in the Web, and schema-aware documents, more proper of distributed database systems. Our query mechanism only relies on links for the first kind of documents, while it also looks at the local version of the schema for the second kind. Notice that this local version may disagree with other versions present in the network, since, as we said previously, we are not assuming a common schema.

To enable fragmentation of XML documents, our model [2] exploits a set of *lightweight path expressions*. A fragment f is a valid sub-tree of the original document having a set of paths: (*i*) the *fragment identifier* f_I, i.e. the unique path expression identifying the root of the current fragment within the global document; (*ii*) the *super-fragment path expression* p_s, i.e. the unique path expression linking the parent of the current fragment; and (*iii*) one or more *child fragment path expressions* p_c, i.e. the path expressions linking the children of the current fragment. While f_I and p_s are stored separately from the fragment content, p_c paths are stored within special tags sub added as fragment leaves. Schema-aware documents also store on each peer the XML schema of the documents along with the local fragments and the above paths.

Each fragment comes with the p_c and p_s path expressions and with its fragment identifier f_I stored within the local peer. The fragment identifier is exposed to the outside under the form of a unique key. In such a way, any other peer that looks for a particular path expression will search it through the DHT. We have extended Chord DHT to support this behavior. However, in the original Chord, the hash function used is SHA-1, which is replaced in our model with the fingerprinting technique. Fingerprinting path expressions in a P2P network is similar to fingerprinting URLs [4], but different from an application point of view. In [2], we experimentally proved that hashing and fingerprinting guarantee the same load balancing. We preferred fingerprints to any arbitrary hash function because of their software efficiency, their well-understood probability of collision (discussed next) and their nice algebraic properties [2]. We next discuss how fingerprinting works. Let $A = \langle a_1 a_2 \ldots a_m \rangle$ be a binary string. We associate to A a polynomial $A(t)$ of degree $m - 1$ with coefficients in the algebraic field Z_2, $A(t) = a_1 \cdot t^{m-1} + a_2 \cdot t^{m-2} + \ldots + a_m$. Let $P(t)$ be an irreducible polynomial of degree k, over Z_2. Given $P(t)$, the fingerprint of A is the following: $f(A) = A(t) \ mod \ P(t)$. The irreducible polynomial can be easily found following the method in [11].

Thus, in order for a peer to compute the path expression fingerprint, it suffices to store the irreducible polynomial $P(t)$. The latter has a fixed degree equal to $NF + 2 \cdot DM + Q$, being (*i*) 2^{NF} the number of fragments in the network, (*ii*) 2^{DM} the

length of the longest path expression in the network, and (*iii*) 2^Q a threshold due to the probability of collision between two arbitrary distinct tokens [4]. Observe that such a polynomial is a quite small structure to be replicated on each participating peer if compared to *replicated global indexes* used in [3]. Moreover, our set of lightweight path expressions and the accompanying polynomial are not directly comparable to probabilistic approaches based on *bloom filters* as in [6,7], which can handle XML data of relatively small depth.

4. The Guidelines of a Distributed Framework for Privacy Preserving P2P XML Databases

In order to secure XML fragments (and, as a consequence, the database storing such fragments) over P2P networks in the context of very large publish-subscribe systems, we devise an efficient distributed framework according to which (XPath) links pointing to fragments are encrypted by using a trusted key founding on the fingerprinting technique, and shared by peers of the same (interest) group.

To this end, besides fingerprinting the path expressions as in [2], we also *fingerprint the actual XML content of the fragment*. This is novel with respect to our previous work [2] and not discussed at all in [4], where fingerprinting is in fact not used for authentication purposes. Indeed, since fingerprinting, like hashing, reduces any arbitrary string to a fixed length token, we can safely apply fingerprinting to the serialized content of an XML fragment. Since all we need to decode a fingerprinted item is the irreducible polynomial $P(t)$ (see Section 3), it is straightforward to create interest groups that share the same polynomial. Every peer within such groups can verify the authenticity of fragments in the community and contribute to any issued query, which would be blind to the others. Of course, there will be as many groups of peers as the number of polynomials we wish to allow in the network, ranging from the scenario with one distinct polynomial per peer or per groups of peers to the scenario with one unique polynomial for all peers. Notice that this approach also guarantees that peers that answer queries are trustworthy (we present our query algorithms in Section 4).

Our proposal resembles the use of cryptography (e.g., RSA and AES algorithms [8]) in [9], and *self-certifying path names* [5], which are used in SFS directories to encrypt the host symbolic names. However, these cryptographic functions yielding large pieces of data are very slow in an overlay P2P network.

Along with the basic idea of fingerprinting paths and fragments to support privacy preserving features, we provide two extensions to handle security among peers, which overcome more traditional schemes, and can be used to develop more specific security protocols for large and highly dynamic P2P networks. First, the public key in traditional schemes (e.g., SFS) is decided by a centralized server, which would not applicable in a P2P network. To decide the public key, we actually use an *authority group* policy as in [10]. More precisely, we require that a public key is jointly decided by all the group members. Secondly, traditional schemes focus on secure handling of host names, whereas in our solution we need a broader level of security, i.e. extended to any (XPath) links either embedded in a fragment or lying out of it.

Another interesting issue is the handling of new peers joining the network and wishing to be admitted in an interest group, in order to avoid that there are no malicious peers. To guide the admission procedures, according to our authority group policy, we require that a new peer entering the network can be admitted in a group *if and only if* all the members agree on its admission. In more detail, in our model, this only applies if the given peer holds fragments related to those stored by other members. If this is not the case, then the peer can only be assigned to a new public key and create its own group.

5. Conclusions and Future Work

In this paper, we have presented the guidelines of a framework for supporting secure XML fragmentation among peers, e.g. those embedded in very large publish-subscribe systems, where XML views are published via federated data providers and replicators. In such a framework, privacy preserving features across peers are supported by means of well-established fingerprinting techniques, and secure XML fragments are accessed and queried across peers via efficient distributed algorithms, which can act in both schema-less and schema-aware modes. Future work is mainly focused on testing the performances of our proposed framework against real-life settings, and on extending current query functionalities as to include more advanced IR capabilities.

References

[1] Bernstein, P.A., Giunchiglia, F., Kementsietsidis, A., Mylopoulos, J., Serafini, L., Zaihrayeu, I., "Data Management for Peer-to-Peer Computing: A Vision", *Proc. of ACM WebDB*, pp. 89-94, 2002.

[2] Bonifati, A., Cuzzocrea, A., "Storing and Retrieving XPath Fragments in Structured P2P Networks", *Data and Knowledge Engineering*, to appear, 2006.

[3] Bremer, J.-M., Gertz, M., "On Distributing XML Repositories", *Proc. of ACM WebDB*, pp. 73-78, 2003.

[4] Broder, A.Z., *Some Applications of Rabin's Fingerprinting Method, Springer-Verlag*, 1993.

[5] Fu, K., Kaashoek, M.F., Mazieres, D., "Fast and Secure Distributed Read-Only File System", *Computer Systems*, Vol. 20, No. 1, pp. 1-24, 2002.

[6] Gong, X., Yan, Y., Qian, W., Zhou, A., "Bloom Filter-based XML Packets Filtering for Millions of Path Queries", *Proc. of IEEE ICDE*, pp. 890-901, 2005.

[7] Koloniari, G., Pitoura, E., "Content-Based Routing of Path Queries in Peer-to-Peer Systems", *Proc. of EDBT*, pp. 29-47, 2004.

[8] Menezes, A.J., van Oorschot, P.C., Vanstone, S.A., *Handbook of Applied Cryptography, CRC Press*, 1996.

[9] Mostafa, M., Wigren, P., "Secure XML File Sharing in a JXTA P2P Network for Inter-Organizational Industrial Collaboration", *Research Report CS Department, Chalmers University of Technology*, 2004.

[10] Narasimha, M., Tsudik, G., Yi, J.H., "On the Utility of Distributed Cryptography in P2P and MANETs: The Case of Membership Control", *Proc. of IEEE ICNP*, pp. 336-345, 2003.

[11] Rabin, M.O., "Fingerprinting by Random Polynomials", *Technical Report CRCT TR-15-81, Harvard University*, 1981.