

Secure Third Party Distribution of XML Data *

B. Carminati, E. Ferrari
University of Insubria at Como, Italy
barbara.carminati@uninsubria.it
elena.ferrari@uninsubria.it

E. Bertino
CERIAS Purdue University, Lafayette, USA
bertino@cerias.purdue.edu

1 Introduction

Web-based third-party architectures for data publishing are today receiving growing attention, due to their scalability and the ability of efficiently managing large numbers of users and great amounts of data. A third-party architecture relies on a distinction between the data Owner, i.e., the producer of information, and the Publisher that provides data management services and query processing functions for (a portion of) the Owner's information. Clearly, in such an architecture security is a primary challenge. Main security properties that should be considered are: *confidentiality*, *integrity*, and *authenticity*. Additionally to these traditional security requirements, we are interested in a further security property, that is, *completeness*. By completeness we mean that the user receiving a portion of data can verify whether he/she has received all the information is allowed to see according to the specified access control policies. Requiring Publishers to be trusted wrt these security properties is not an appropriate solution in that large web-based systems cannot be easily verified to be trusted and can be easily penetrated. Thus, our goal is to ensure security properties even in the presence of an untrusted Publisher. In this paper, we propose a comprehensive framework for a secure third party distribution of XML data. In particular, the framework is able to enforce all the above-mentioned properties, by exploiting encryption and non-conventional signature techniques.

Our work has been greatly inspired by the work by Hacigumus et al. [3], which develop an interesting method for querying encrypted data stored in relational databases thus ensuring confidentiality requirements. From such work we borrow the method for querying encrypted data, which is based on partitioning the domains of the relation attributes. However, such work only considers confidentiality wrt the

Publisher, whereas they do not consider confidentiality wrt user, in that they do not ensure the Owner that its data are only disclosed to users authorized according to the access control policies it specifies, nor it considers authenticity/integrity and completeness. Related work is the one by Miklau and Suciu [4] which proposed a method for a controlled sharing of XML data, dealing only with confidentiality issues. As in our approach, confidentiality is ensured by the use of cryptographic techniques. However, the main difference with our proposal is that they do not rely on a Publisher for managing data, rather the data are simply published on the web in an encrypted form and each user can access the authorized portions, using the keys he/she receives from the data Owner. However, we strongly believe that relying on a data Publisher has many benefits, in terms of efficiency and optimization of resource usage. Simply publishing the data over the web would make them the target of a huge number of attacks, with many users trying to perform queries over them, and thus consuming a huge amount of computational resources. By contrast, the Publisher can be equipped with sophisticated anti-intrusion tools and techniques avoiding queries floods. Additionally, not relying on a Publisher requires each user be equipped with a query engine able to process queries over encrypted contents.

2 Overview of the proposed framework

In our framework, before sending a document to a Publisher the Owner encrypts it. In particular, all the portions of the document that are protected by the same policies are encrypted with the same symmetric key. To make the Publisher able to answer user queries according to the specified access control policies and without the need of decrypting the Owner data, the Owner sends it some additional information. First of all, it sends the Publisher information on which access control policies apply on the managed documents. Additionally, the Owner sends the Publisher some information that makes it able to query encrypted data. The basic idea is that the Owner divides the domain of each doc-

*The work reported in this paper has been partially supported by the Italian MIUR under the project 'Web-based management and representation of spatial and geographical data' and by the Information Society Technologies programme of the European Commission under the IST-2002-01945 TrustCoM project (2004-2006).

ument node (i.e., attribute and element) into distinguished partitions, to which it assigns a unique id. Then, the Owner sends the Publisher together with the encrypted node also the id of the partition corresponding to its value. The Publisher is thus able to perform queries directly on the encrypted documents, by exploiting the partitioning ids. According to this approach, the Publisher is able to evaluate user queries on encrypted data, and to return to the user the corresponding encrypted view. Then, in order to make a user able to access the encrypted result, the Owner supplies him/her during a mandatory subscription phase with the appropriate encryption keys. More precisely, during this phase, the Owner assigns to a user one or more credentials.¹ Thus, as a result of the subscription phase, the Owner returns the user the keys corresponding to the portions of the source he/she is entitled to access and a user policy configuration, containing information on the access control policies the user satisfies. The user policy configuration is signed with the private key of the Owner, to prevent a user from altering its content, and it is submitted by the user to the Publisher along with a query. In this way, the Publisher knows which portions of the requested documents the user is allowed to access. Authenticity and integrity are assured by the use of *Merkle Signatures* [1], that is, signatures generated by the Owner using a bottom-up computation on the whole documents, which exploit Merkle hash trees. Such signature is generated before encrypting a document and it is thus sent to the Publisher along with the corresponding document. The Publisher will then forward it to a user querying the document to which it refers to. The problem here is that, since the Publisher answer may not contain all the document portions over which the signature is computed, because of the access control policies specified by the Owner, a user may not be able to validate the signature. To avoid this shortcoming, the Owner sends the Publisher a set of additional hash values, one for each node, which represent the information needed to validate the signature if the corresponding node is not inserted into the Publisher answer. Thus, when a user queries a certain document, the Publisher sends him/her, besides the corresponding Merkle signature, also some additional hash values referring to the document portions not contained in the query answer. This makes the user able to locally perform the computation of the Merkle signature and comparing it with the one generated by the Owner.

All additional information required by the Publisher for confidentiality and authenticity/integrity enforcement are encoded in XML and attached to the encrypted document, forming the so called *security enhanced encryption* of the original document. Similarly, all information needed by a user to prove the satisfaction of the considered security

¹Alternatively, credentials and/or keys can be managed by a trusted third-party Authority.

properties are encoded by the Publisher in XML and attached to the query answer, resulting in what we call the *reply document*. Moreover, to make a user able to verify the completeness of a query result, the first time the user submits a query to a new document, the Publisher returns him/her an additional XML document, called *query template*, for the requested document. The query template is generated directly by the Owner, and basically contains only the structure of the original document, encrypted by the Owner using the same strategy employed for XML documents. The query template has the twofold goal of making a user able to verify the completeness of the received answers, and to make easier the task of query submission, in that by inspecting the query template a user can obtain information on the structure of the documents (or portions) he/she is allowed to access. Note that the approach we have devised requires that some additional information are transmitted by the Owner to both the Publishers and the users. In particular, the Owner should deliver to users all the necessary keys. To reduce the overhead of these operations require, we assume that all such information are stored by the Owner into a directory (which can also be shared among different Owners belonging to the same domain or to federated ones). Each user/Publisher receives by the Owner only one key, to be used to access its entry in the directory.²

In the extend version of the paper [2] besides providing a formal foundation for the proposed infrastructure and the details of security property enforcement we prove its soundness.

References

- [1] E. Bertino, B. Carminati, E. Ferrari, B. Thuraisingham, A. Gupta. Selective and Authentic Third-Party Distribution of XML Documents. *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, 16(10):1263–1278, 2004.
- [2] B. Carminati, E. Ferrari, E. Bertino. Assuring Security Properties in Third-party Architectures. Technical Report DICO, University of Milano. Available at <http://scienze-como.uninsubria.it/carminati/SE-ENC.pdf>
- [3] H. Hacigumus, B. Iyer, C. Li, and S. Mehrotra. Executing SQL over Encrypted Data in the Database Service Provider Model. *In Proceedings of the SIGMOD Conference*, 2002.
- [4] G. Miklau and D. Suciu. Controlling Access to Published Data Using Cryptography, In Proc. of the *29th VLDB Conference*, Berlin, Germany, 2003.

²Access to such directories would be greatly simplified by recent developments in the framework for single-sign and federated identity management.