An Implementation of a XML Documents Secure Circulation System

Miao Liu and Ya Li

School of Computer Science and Educational Software, Guangzhou University, Guangzhou, 510006, China
E-mail: liumiao@gzhu.edu.cn

Abstract. XML is already one standard of data exchanging among applications. How to assure XML data security is an important problem. The paper designs and implements a flexible XML documents secure circulation system by using S2SH frameworks. The system assures XML documents confidentiality by providing XML documents encryption and decryption, assures XML documents integrity by implementing digital signature and verification. It allows owners of documents to define access control policies by XACML, and assures strict and flexible access control for XML documents. It also uses digital certificates to authenticate users, to create digital signatures using corresponding private keys, and assures non-repudiation for updating XML documents.

1. Introduction

The XML is the de facto standard for storing and exchanging information now. When developers agree with the XML schemas in advance, the applications which they developed could then exchange XML data automatically. The XML is also the basis of the semantic web, and so more and more XML data will be exchanged in the Internet. A lot of XML data deals with personal privacy and some other secrets, thus, it is critical to ensure XML data security[1-3].

There are many applications which process and exchange XML documents among them, such as electronic medical records, electronic official documents and transaction lists. These applications must provide mechanisms to prevent those XML documents from unauthorized accessing and modifying, to ensure non-repudiation and to allow owners to define flexible access control rules. We design and implement a flexible XML documents secure circulation system which is suggested as a solution to these goals. The system is a general XML documents secure management system, no matter how different XML documents schemas are.

This paper is organized as follows. In section 2, we show the composition of the system, and the design of four core modules. In section 3, we explain the implementation architecture of the system, the implementation of the access control function and the documents circulation function. Conclusions are discussed in section 4.

2. System design

2.1. The composition of the system

The system consists of four first class modules, which are IssuingDocuments, Outbox, Inbox, and PAP. The module IssuingDocuments is divided into four second class modules; they are the Create, Modify,
Delete and Designate. The Designate module designates receivers who will audit the issuing documents. The Outbox module is divided into two modules which are Unfinished and Finished. The Unfinished module manages documents which are not finished auditing yet, and the Finished module manages audited documents. The Inbox module is divided into three modules which are Auditing, AuditedUnfini and AuditedFini modules. The Auditing module manages documents which are being audited, the AuditedUnfini module manages those documents which is audited by the user, but not finished auditing by the other receivers, and the Auditedfini module manages those documents which are finished auditing by all receivers. The PAP module is the policy administration point, which is divided into two modules, and they are the UserAdmin and Policy modules. The UserAdmin module manages users and their certificates, and the Policy module edits and checks access control policies. When users log in the system, they could use the above modules to operate on their XML documents. The module structure diagram of the system is illustrated as Figure 1.

![Diagram](image)

**Figure 1.** The module structure diagram of the system

As illustrated in the Figure 1, the four bottom mouldes are XMLEncryption, XMLSignature, XMLDecryption and XMLSigVerication mouldes. They are the core modules of the system, because they implement XML Encryption and XML Signature standards, and many functional modules need to invoke them.

2.2. Design of XMLEncryption and XMLDecryption modules

XML Encryption[4] is not a new encryption algorithm, it defines how to use XML to represent encrypted data and cipher data, and it also specifies the standard encryption and decryption processes. In the XML encryption, the encrypted data may be all of a XML document, a XML element or its content, and non-XML data. The encryption data may be placed in the encrypted document, or placed in another file, and be referenced in the encrypted document. If the XML documents have many receivers, XML Encryption allows encrypting different document parts by using different keys, and thus we may send the same XML document to different receivers. XML Encryption also defines the standard decryption process according to the cipher data contained or referenced in the encrypted XML document.

The flowcharts of the XMLEncryption and XMLDecryption modules are illustrated in the Figure 2.
In the XMLEncryption module, the sender may select one or more receivers of his document to be sent. The session key is generated randomly and encrypted by using the receiver’s public key. The data to be encrypted is encrypted by using the session key. Each receiver’s encrypted data will be encrypted with different session key.

In the XMLDecryption module, the receiver must provide his private key to decrypt the session key first, if succeed, and then decrypt the encrypted data which sent to him.

2.3. **Design of XMLSignature and XMLSigVerification modules**

XML Signature[5] is not a new signature algorithm, it defines how to use XML to represent signature data and cipher data. In the XML Signature, the data to be signed may be all of a XML document, a XML element or its content, and non-XML data. The signature may be placed in the signed document, or placed in another file, and also could be detached from the signed data.

XML Signature also specifies the standard signing and verification processes. The process of signing includes two phases: generating Reference element and generating Signature element. One or more Reference elements identify the data that is digested via a URI. The Reference element contains an option Transforms element that in turn contains a list of one or more Transform elements describing a transformation algorithm used to transform the data before they are digested; the DigestMethod element specifies the method used to generate the digest value reported in the DigestValue element. Thus, the process of verifying also includes two steps: calculating the digest values of the referenced data and compares them with the values in the Reference element, if same, then proceed to verify the corresponding digital signature values. The flowcharts of the XMLSignature and XMLSigVerification modules are illustrated in the Figure 3.
3. The implementation of the system

3.1. The system architecture
The system adopts ExtJs + S2SH (Struts2 + Spring + Hibernate) frameworks to construct the system, it includes four tiers: the presentation tier, the controller tier, the business logic tier and the database tier. The system architecture is illustrated as the Figure 4.

3.2. The implementation of access control
XACML (XML access control markup language)[6-7] defines some XML elements to represent access control policies for XML documents, and also specifies the XML access control model. The model includes PEP (Policy Enforce Point), PDP (Policy Decision Point) and PAP (Policy Administration Point). The PEP intercepts access requests, and queries the PDP whether allows them or not. The PDP makes decisions according to the subject information, the access control policies and context parameters. The PAP is used to create access control policies and checks whether access control policies are conflict with each other.

The PEP is a struts2 filter which acts as AOP access proxyer; it intercepts all requests, and sends user’s information and the request to the PDP. The PDP decides whether allow or disallow the user to access the data. The owner of XML documents uses PAP to write access control policies for his
documents, and check whether they are consistent or not. The access control flow chart is illustrated in the Figure 5.

![Figure 5. The access control flow chart](image)

3.3. The implementation of documents circulation

The designate module in the Figure 1 is used to assign the documents receivers and specify the order of them. The XML documents will circulate among the receivers in the predefined order. Each receiver reads or updates the documents according to access control policies which the owner specified before.

The XML documents circulate according to predefined workflow. We design two tables in the database to help push the documents circulation: one is node table and the other is workflow table. Each row in the node table represents a receiver, and it contains userID, status, flowstep and active fields. The status field has four values to chose: 0 represents starting, 1 represents finished, 2 represents to be audited and 3 represents audited. The flowstep field records the receiving order, the default value is 0. The active field records if the document circulates to the current node. The value 0 represents not received yet, and 1 represents already received. One workflow record associates with many records in the node table.

When sending one XML document, the system sets the flowstep values of the receiver nodes according to the receiving order. The first node’s status value is set to 0, the last node’s status value is set to 1, and all immediate nodes’ status values are set to 2. In the meantime, the first node’s active value is set to 1, and the other nodes’ status values are set to 0.

When the first receiver logs in the system, the auditing in his Inbox has the XML document which is associated with the first node’s wfid field value. When finished auditing, the system sets the node’s status value to 3, then judges if the current node is the ending node or not. If yes, set the corresponding workflow’s status value to 1, meaning the workflow is finished. If no, find the next node, which has the same wfid field value, and the flowstep field value is current node’s flowstep field value plus 1, then sets the next node’s active value to 1. The flow chart of documents Circulation is illustrated in Figure 6.
4. Conclusions
In this paper, we design and implement a flexible XML documents secure circulation system. The system deals with all aspects of XML security. The system is consistent with the XML Encryption, XML Signature and XACML standards, thus, it is a general framework which circulates XML documents, and could be used in the circulation of any XML documents. The system allows owners of XML documents define and modify access control policies at any time, and implements dynamic access control for all XML documents.

Acknowledgments
The paper’s work is supported by Guangzhou Municipal Universities project 1201620342.

References
[1] Sookhak M, Gani A, Khan MK, Buyya R, Dynamic remote data auditing for securing big data storage in cloud computing. Inf Sci 380:101-116, 2017.
[2] Y. Li, K. Gai, Z. Ming, H. Zhao, and M. Qiu. Intercrossed access control for secure financial services on multimedia big data in cloud systems. ACM Transactions on Multimedia Computing Communications and Applications, 12(4s):67, 2016.
[3] Azhar Rauf, Waqas Ali, Maher Ahmed, Shah Khusro, Shaukat Ali, Efficient XQuery over Encrypted XML Documents, The 10th International Conference on Computer Science & Education (ICCSE 2015), July 22-24, 2015.
[4] Luigi Coppolino, Salvatore D'Antonio, Giovanni Mazzeo, and Luigi Romano, Cloud security: Emerging threats and current solutions. Computers and Electrical Engineering 59, Supplement C (2017), 126-140, 2017.
[5] C. Mainka, M. Jensen, L. Lo Iacono, J. Schwenk, I. Ivanov, M. Sinderen, F. Leymann, T. Shan, "Making XML Signatures Immune to XML Signature Wrapping Attacks", Cloud Computing and Services Science Springer International Publishing, vol. 367, pp. 151-167, 2013.
[6] S. Jan, C. D. Nguyen, L. Briand, Automated and Effective Testing of Web Services for XML Injection Attacks, Proceedings of the 2016 International Symposium on Software Testing and Analysis (ISSTA), Jul 2016.
[7] Panagiotis Ilia, Barbara Carminati, Elena Ferrari, Paraskevi Fragopoulou, SAMPAC: Socially-Aware collaborative Multi-Party Access Control, Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy, March 22-24, 2017.